Tree Adjoining Grammar and the Reluctant Paraphrasing of Text

Mark Dras

Department of Computing
Division of Information and Communication Sciences
Macquarie University
NSW 2109 Australia

Submitted in Partial Fulfilment of the Requirements of the Degree of Doctor of Philosophy

February 1999

Abstract

Paraphrase, as a general concept, is a very mutable thing; at one extreme, paraphrase is the relation between units of text with identical meanings, as under the transformational-generative view, and at the other extreme there is the view that no text is interchangeable with any other. Each point in the spectrum ranging between these two views is a valid one, and in order to focus on a specific point it is necessary to examine paraphrase under a particular context and application.

In this thesis, the context is the task of Reluctant Paraphrase (RP). Given an ideal starting text, which expresses exactly what its author intends, it is often necessary to alter the text in some way in order to fit it to some externally imposed constraints—length, readability as specified by a house style guide, and so on. In this context, paraphrase is a relation between interchangeable units of text; the task is then to choose the best set of such paraphrases that will fit the whole text to the constraints, with 'best' in this case being the set that changes the text the least. This differs from other tasks where paraphrase is central: in natural language generation, particularly revision-based generation, where positively correlated constraints have meant there is no need for advanced search strategies to find the best solution, and where the starting point is a knowledge representation rather than text, with the system consequently being omniscient regarding equivalence of concept realisations into text; and in style checkers and controlled language checkers, where the goal is to correct text, and consequently where the locality of alterations again means that advanced strategies for finding the best solution are unnecessary.

The thesis uses the application of paraphrase as a means of exploring properties of the Tree Adjoining Grammar (TAG) formalism. TAG is a mathematical formalism, within which linguistic theories can be expressed, that is constrained in expressive power, which means that formal results about the language can be established, and that a linguistic theory has less work to do in circumscribing the class of acceptable constructions, since the formalism imposes universal constraints. And, as one of a class of mildly context sensitive grammars, it is well suited to describing natural language. Moreover, it is possible to map between TAG grammars using the Synchronous TAG (S-TAG) formalism, which has been used in applications such as machine translation, syntax–semantics mapping and modelling of coordination. However, in these applications the S-TAG formalism is not able to model all the phenomena that occur. Paraphrase is another, different application that allows exploration of the formalism from a different perspective, one which is particularly structurally complex.

This application and the context of RP are closely interrelated. In performing the RP task, a representation of paraphrase is necessary, for which S-TAG provides a formalism. And, in using paraphrase to explore TAG, the RP context provides a focus narrowing down the paraphrases and consequently the requirements of the representation formalism.

The thesis then looks at the two related modelling tasks. Firstly, the modelling of the RP task. Integer Programming is shown to be an appropriate framework, allowing a text fitting the constraints to be determined, one minimally altered by paraphrasing. A range of model variants is possible, and the thesis investigates which is the best model in terms of search criteria. Secondly, the modelling of paraphrase in S-TAG. The relative structural

complexity gives insights into the cause of representational difficulties with S-TAG, and leads to generalisations of S-TAG that resolve existing problems in the formalism.

Acknowledgements

Unaccustomed as I am to public writing ...

For me, doing a PhD has felt somewhat like a biblical journey: starting off in the wilderness, a featureless expanse, praying that there'd be some revelation from the heavens which would tell me not only where the Topic-Path was, but also how I'd recognise a Topic-Path if I fell over it. And wandering, for what felt like 40 years, landscape features becoming more distinct, until I arrived in the land flowing with milk and honey and clear research directions and academic grants.

So, in getting there, there's a variety of people I'd like to acknowledge. First of all are the people who have, to varying extents, acted in a mentoring capacity for me. Vance Gledhill, who set up the Microsoft Research Institute in which I carried out my PhD, for which I and my bank balance are profoundly grateful. Robert Dale and Mike Johnson, who supervised me during the course of my wanderings, and whose startling resemblance to Plato and Aristotle never ceases to amaze me. Owen Rambow, who more resembles a hybrid Ogion the Silent-Kevin Kline, who guided me when I was lost in the desert of the untopic'ed and when burdened by cluttered thinking, and who has no idea the extent to which I am endebted to him.

Then there are the people who have been part of MRI, who made the environment such a stimulating one in which to work: Einat Amitay, Sarah Boyd, David Clarke, Brent Curtis, Victor Essers, Stephen Green, David Holmes, Michael Kirk, Mark Lauer, Alpha Luk, Josef Meyer, Maria Milosavljevic, James Noble, Geoff Outhred (for sundry sporting diversions), John Potter, Michael Richmond, Ryan Shelswell (also for thesis diagrams), Steven Sommer, Margaret Wasko, Sandra Williams.

Being part of MRI has meant that I've been able to talk to a lot of people who have also contributed to the shape of the PhD. In the beginning, before the Topic came into being, I was helped along by several extremely patient people who made me feel more certain about what I was doing, in particular William Gale, Don Hindle, David Yarowsky. After I found the road, there were many other people with whom I had helpful conversations, including Helen Bateman, Claire Craig, Judy Delin, Dominique Estival, Graeme Hirst, Aravind Joshi, Judith Klavans, Chris Manning, Keith Miller, Philip Miller, Kathy McKeown, John Oberlander, Mick O'Donnell, Cécile Paris, Graeme Ritchie, Jim Rogers, Donia Scott, David Weir, Ingrid Zukerman.

Near the end, a couple of people were particularly helpful, proofreading the thesis: Sarah Boyd (again) and Philippa Clymo.

And for the final stage, I appreciate that my examiners worked particularly hard in reading the thesis: Aravind Joshi, Chris Manning, and David Weir.

And all through it, with the patience of a Job afflicted by constant storms of moodiness, Tim Rowlands.

Preface

The research presented in this thesis is the original work of the author except where otherwise indicated. Some parts of the thesis include revised versions of published papers. This work has not been submitted for a higher degree to any other University or Institution.

Mark Dras

Amédée: ... L'esprit coupant se glisse, sournoisement, dans la conversation, avec ses pointes ... Êtes-vous géomètre?

Le Soldat Américain: I get it ... I get it ...

Amédée: Dans ce cas, prenez le parti des sphères ... remplacez l'angle par la calotte, le triangle par le cercle, le parallélipipède par la sphère ... les cylindres, rarement les cônes ... jamais les pyramides, comme le firent les Égyptiens, c'est ce qui les perdit ...

Le Soldat Américain: I get it ... I get it ...

Amédée: Et surtout, tournez les questions, parlez beaucoup en périphrases ... périphrases ... périphrases ... Périphrasez, périphrasons ... Ne pas rester immobile, on devient clou, on devient pointe ...

[Ionesco (1954): Amédée ou Comment s'en Débarrasser, Act III.]

Amédée: ... a cutting wit slips its barbs into conversation ... Are you a geometrician?

Soldier: I get it ... I get it ...

Amédée: In that case put yourself on the side of the spheres ... Choose a curve and not an angle, a circle not a triangle, an ellipse but never a parallelepiped ... cylinders, perhaps, but cones only now and then ... never pyramids as the Egyptians did, that's what caused their downfall ...

Soldier: I get it ... I get it ...

Amédée: And above all, evade the question ... always move in a circle and paraphrase and paraphrase ... I paraphrase ... you paraphrase ... we paraphrase ... keep going round and round or you'll have to stick to the point ...

Contents

Chapter 1

Introduction

1.1 The Framework

This thesis is a two-headed one: one focus is the building of a framework for modelling a particular text manipulation task, where text is paraphrased reluctantly; and the other focus is the use of this framework as a means of exploring a grammar formalism, Synchronous Tree Adjoining Grammar, and resolving some of its problematic aspects.

1.1.1 The Text Manipulation Task

The central theme of this thesis is the development of a particular text manipulation task—that of 'reluctantly' paraphrasing a text under externally imposed constraints—and then investigating what is necessary for modelling this task. The two major components investigated are a formalism for representing the possible paraphrases of a text, and a model for choosing from among these paraphrases a subset that will satisfy the task requirements. The task is not one that has been addressed in the literature, so the following is a broad characterisation, albeit one which will be progressively refined in the course of the thesis:

> Imagine you have just written an ideal document: it captures every idea you want to express, and says it in the way that you want it said. However, it is necessary to modify the document because of externally imposed constraints. For example, a conference requires that a paper submitted to it be no longer than seven pages; or a house style guide makes mandatory, for some software documentation, conformance to a range of readability formulae; or an editor insists on your writing being less dense, as measured by her rule-of-thumb metric. In general, there is some requirement to rearrange your text to conform to some combination of these or other surface constraints, although you will obviously want to keep the text as close to the original as possible.

In this thesis, this task framework will be referred to as Reluctant Paraphrase (RP). In practice, of course, the original text will not be exactly ideal, but this is a convenient theoretical starting point, from which deviation can occur later; and even in practice, it is often the case that the text is corrected first, and major flaws removed, and then a version generated which fits the external constraints.

Other, related, tasks occur in the areas of Natural Language Generation (NLG) and language checking (covering grammar checking, style checking and controlled language checking). In NLG, for example, text is generated from some underlying representation to conform to constraints imposed by the system. These constraints can be as minimal as ensuring grammaticality, but can be refined further so that certain stylistic or other criteria are met; the area of NLG most closely related to RP is that of revision-based NLG, where texts are successively generated until they conform to surface constraints similar to those mentioned for RP. In language checking, a text has paraphrases applied to it so that the text fits certain requirements, generally correctness according to some standard.

RP differs from both of these. NLG systems start with an underlying representation from which text is ultimately generated. This means that the system has knowledge of exactly which expressions are equivalent, and can generate these as alternatives. In RP, this is not known precisely, because the starting point is text, and it is often necessary, in order to conform to constraints, to alter the text in ways that mean that the resulting text may not be equivalent. This leads in this thesis to a looser notion of interchangeability for paraphrases, and a semantic model of the effects of any such interchanges. Secondly, the constraints used in existing NLG systems have not required any consideration of search issues: constraints are generally either singular or positively correlated, and the choice of each paraphrase is consequently straightforward. In RP, this is not the case. Constraints naturally conflict, and there is the additional objective of trying to keep the text as close to the original as possible. Thirdly, mapping from text to text requires some sort of formalism for expressing individual text-to-text mappings; this raises issues concerning the properties of such a formalism, and what can be known and guaranteed about the properties of a resulting text that is essentially a translation of the original.

RP differs from language checking in that the latter's concern with correction leads it to be both more general and more specific at the same time. It is more specific in the sense that each check of a particular construction for correctness is local: no more information is required to determine whether a paraphrase will occur other than the form of the construction. In RP, whether or not a particular paraphrase will occur depends not just on local knowledge, but also on the status of the entire text with respect to overall goal satisfaction. Language checking is, on the other hand, more general, in the sense that there is a very broad range of constructions that it is necessary to paraphrase for correctness. In RP, however, the paraphrases are only tools for modifying the texts to conformance with the constraints, so a conveniently characterised subset can be chosen such that it is possible to develop a formalism whose properties can be investigated. In this thesis, syntactic paraphrases are the ones used, as their general nature accords with the idea of paraphrases in RP being general purpose tools.

RP is not, it should be noted, tied to any particular set of constraints, or any particular type of paraphrase; rather, it is just a framework for the task of fitting a text to some arbitrary surface constraints, using some arbitrary collection of paraphrases, such that there is minimal change from the original. Of course, in order to illustrate the framework it is necessary to use some set of constraints, and some paraphrases. The constraints have been chosen such that their conflicting nature demonstrates the utility of RP; the paraphrases chosen because their syntactic nature makes them quite general.

Given a formal representation for the paraphrases, discussed below, it is possible to consider ways to handle the task of choosing from among the paraphrases that subset which

will satisfy the externally imposed constraints and yet minimally change the text. In this thesis, Integer Programming is explored as a suitable framework given the desiderata of RP: ability to handle conflicting surface constraints, capacity to represent the aim of minimising paraphrase effect, and the availability of methods for dealing with search space tractability issues. It is well known that Integer Programming allows much scope for differing formulations, and hence a large difference between good and bad formulations. Thus, in addition to investigating how RP can be modelled in the Integer Programming framework—what the decision variables are, how constraints can be represented, and so on—this thesis looks at using aspects of language and the idea of symmetry breaking from the field of Artificial Intelligence to produce a 'good' model, one where the search space of solutions is more easily navigable, and tests this on actual text.

1.1.2 The Grammar Formalism

In some sense, the 'central theme' described in the preceding section, although of significance in its own right, can be considered as only a setting for the exploration of a grammar formalism, and how it handles paraphrase. The second major component of the thesis involves taking a particular formalism, Synchronous Tree Adjoining Grammar, and investigating how representing paraphrases within it can lead to insights which resolve problematic aspects of that formalism.

A generative grammar is an obvious alternative for representing these paraphrases; for a computationally motivated task it is especially appealing because of the ability to precisely characterise phrase structure trees using the tools of formal grammar. Representing paraphrasing using generative grammar is not new: shortly after the birth of generative grammar in Chomsky's (1957) *Syntactic Structures*, the use of generative grammar as a paraphrase representation was proposed in Klein (1965a; 1965b). The work in this thesis differs from Klein's in several respects. In terms of the task, Klein was only interested in randomly permuting phrase structure trees given some style parameters, and he did so using the transformations of Transformational Generative Grammar. In RP, the task is different, in that it is a goal-driven rewriting of the text. A more significant difference concerns representational and formal issues. Klein was not interested in determining formal properties of his paraphrase system—what the characteristics of the resulting language were, whether the phrase structure representation for the resulting text was the same as for the original text (or indeed whether phrase structure notions could be applied at all), and so on. Moreover, his use of transformations meant that his system was formally unconstrained, equivalent to a Turing Machine, as demonstrated by Peters and Ritchie (1973).

In this thesis, Synchronous Tree Adjoining Grammar (S-TAG), an extension of the Tree Adjoining Grammar of Joshi *et al* (1975), is used as a paraphrase formalism. S-TAG has a number of features which make it useful for a paraphrase representation, including its tree-based nature, its bidirectionality, and the library of standard constructions in the XTAG grammar (XTAG, 1995). Having S-TAG as a paraphrase formalism has benefits in two directions: using S-TAG to represent paraphrases gives a description of paraphrase that is formally constrained in a way suited to natural language; and expressing paraphrases in S-TAG allows an exploration of issues in S-TAG that have previously been touched on only lightly.

In terms of using S-TAG to represent paraphrases, this has the general advantage of using

a mathematical formalism: it allows the paraphrase to be represented in a precise and computational way, and it is useful as a means of expression, or a common framework for discussing linguistic issues. S-TAG shares these properties with the formalism used by Klein; however, it has the additional advantage that it is formally restricted in power and expressiveness. In describing language it is desirable to have a formalism that is expressive enough to cover all features of natural language, but which is not so expressive that it is possible to describe any permutation of words over and above this; less work is then left for a specific linguistic theory realised in the formalism. Additionally, results relating to polynomial parsability and other computational factors can be shown in a restricted formalism.

In terms of using paraphrases to explore S-TAG, a structural complexity occurs with paraphrases which is only present to a limited extent in other applications of S-TAG, such as machine translation or syntax–semantics mapping. In these other areas, S-TAG, while possessing many useful properties, is not adequate in some ways: examples include the isomorphism requirement, which restricts to too great an extent the ability to map between parallel trees in machine translation, and the difficulty in handling coordination. Representing paraphrases in S-TAG, because of their structural complexity, gives an insight into issues in S-TAG such as derivation, coreference and isomorphism, and provokes generalisations which enable the representation in a natural way of these problematic constructions from related areas of linguistics and natural language processing. Additionally, in proving results in paraphrasing, it is necessary to give tight definitions and results for S-TAG which have not been given before.

1.2 Thesis Outline

This thesis can conceptually be divided into three parts. Part I contains the background and foundational material: looking at characteristics of related systems, discussing appropriate constraints, investigating paraphrases and what characterises them. As mentioned earlier, RP is not tied to the particular constraints or paraphrases given here; but by examining these constraints and paraphrases, it becomes clear what choosing them involves. Part II contains the more formal part: representing paraphrases using S-TAGs, and exploring the consequences for S-TAGs, leading to new results for the formalism. Part III contains the model of RP with the Integer Programming framework, and the investigations of good versus less good formulations.

More specifically, the chapter breakdown is as follows.

Chapter 2 presents previous research from the areas of NLG and language checking that is in some way related to RP. It discusses how they differ, and consequently highlights the characteristics of RP that need to be modelled.

Chapter 3 examines possible constraints that might be appropriate within RP, and provides a rationale for using four in particular: length, readability, lexical density and sentential variation.

Chapter 4 presents a shallow taxonomy of paraphrases which have been taken from a variety of sources, prescriptive, descriptive, linguistic and practical. From these, it develops a definition of paraphrase, and proposes a semantic model of paraphrase effect that fits within the RP framework.

Chapter 5 provides a background on TAG, giving definitions and concepts used in later chapters, and setting out the areas where the current definition of S-TAG is problematic.

Chapter 6 characterises the types of paraphrase described in Chapter 4 using S-TAGs, and introduces concepts of structural mapping pairs and generating functions for this characterisation. It shows that this S-TAG representation of paraphrase is well-formed, and that properties such as the weak language preservation property hold.

Chapter 7 presents a key aspect of the thesis, a generalisation of S-TAG that arises out of issues considered in Chapter 6. This generalisation allows a natural way of solving problems that have arisen with coreference and isomorphism in S-TAG as currently defined, as seen in applications both in paraphrase and elsewhere. It then demonstrates concrete examples of paraphrase representation in S-TAG; and then defines a mapping from the S-TAG paraphrase representation, including the semantic model of Chapter 4, to a form usable by the Integer Programming model presented in Chapter 8.

Chapter 8 gives a model of RP within an Integer Programming framework, discussing appropriate decision variables, and demonstrating the realisation of constraints as constraint equations and the semantic model as an objective function. It looks at using symmetry breaking and different choices of decision variables to give better formulations, and provides results of some experiments demonstrating that better formulations are possible through the use of the language related characteristics of RP.

Chapter 9 presents conclusions, with an overview of the main results of the thesis, and a section on possible future work following from these.

Part I

The RP Framework

Chapter 2

Related Work

This chapter looks at several models of text manipulation, both in Natural Language Generation (NLG) and in language checking systems, and examines ways in which their tasks overlap with the Reluctant Paraphrase (RP) task described in Chapter 1. RP has in common with NLG and language checking systems a number of characteristics, centering around the making of a choice between different expressions of some given content. In NLG a system starts with the content stored in some knowledge representation, and one of its functions is then to navigate among the possible realisations of that content to find the most appropriate one. Textual correction systems, on the other hand, are by their nature less concerned with navigating through a space of possible realisations, but have in common with RP that an existing text must be rewritten to achieve some aim, here correctness of expression. The following two sections look at several different systems in NLG and text correction, and explore the extent to which the RP task requires a different model for navigating the space of possible realisations to find the solution best achieving the aim of fitting the text to the imposed constraints.

2.1 Generating Text

Systems that generate text from an underlying representation of information can be seen as operating by applying constraints—grammatical, semantic, pragmatic, and so on—to a body of knowledge until a text that satisfies these constraints can be produced. This section looks at three categories of generation systems, and how they relate to the RP task. Section 2.1.1 discusses examples of early generation systems, and the way in which their use of default mechanisms means that the texts produced by the systems do not represent the best satisfaction of constraints, but one of many alternatives taken to be equally valid. And, in doing this, the systems in effect carry out an explicit enumeration of possible solutions, which is not very efficient. Section 2.1.2 discusses the addition of other directives to generation systems so that the use of defaults is decreased; specifically, the use of stylistic directives, so that more appropriate text is generated. The RP task is to solve a similar language problem, fitting a text to constraints, so that the best solution is found—best, in this case, meaning minimal change from the starting text—and that this occurs efficiently, which is an important consideration for systems that aim to be scalable in the number of constraints.

Section 2.1.3 discusses those systems most closely related to RP, revision-based generation

systems. As well as having the same similarities as the systems discussed in the first two subsections, these systems also carry out paraphrasing, mapping from text to text, to various extents. Moreover, some of these also do this based on surface constraints—that is, constraints on surface attributes of text such as word length—as happens under RP. Where RP differs from these is in the starting point of what is known about the text, and also in the way progress is made from this point, searching for an optimal solution as against finding just any solution.

2.1.1 Early Generation Systems

Early generation systems made extensive use of defaults, although that was partly because it was necessary at the beginning of the field of NLG to build every part of a system, whereas now there are already-existing, broad-coverage components such as the realiser FUF (Elhadad, 1992). Notwithstanding this, it is still useful to note two important, and related, characteristics of early models that are different from RP, with these characteristics related to the nature of the systems rather than to the fact that they were built early on.

Two examples of such early text production systems are McKeown's (1985) TEXT and Dale's (1988) EPICURE systems. Both of these used their own tactical or surface generation component, and both have the same two characteristics different from RP and other, later NLG systems: that of having only a restricted set of syntactic constructions as the focus of interest, and that of dealing with the alternatives (which can be treated as equivalence classes, in that all alternatives in a class are interchangeable) within this set of constructions through ad hoc defaults.

This restriction on constructions is not a fault of the system, but a deliberate focussing on what is relevant. For McKeown, whose system is designed to achieve fluent text by discourse-level organisation, the important thing was to choose syntactic constructions which expressed the connectedness of messages:

> The [functional] grammar [based on the formalism of Kay (1979)] was designed so that it can use the focus information provided in the message to select appropriate syntactic constructions. [McKeown, 1985: 11]

Hence, the emphasis was on constructions which embodied different focus alternatives:

> In particular, the tactical component uses focus information provided by the strategic component to select pronominalization, the passive construction over the active, and *there*-insertion. It uses information about rhetorical strategy to select various textual connectives. [McKeown, 1985: 133]

The focus of Dale's work was the generation of referring expressions within the domain of recipes. This led him to use a very simple grammar for sentence generation, restricting himself to imperative sentences; and the noun phrase grammar, more complex so as to allow scope for exploring issues in reference, concentrated on a restricted set of constructions which are found in recipes: for example, partitives (*six of the onions*), 'pseudo-partitives' (*a ring of onion*), and so on. It is possible to expand the grammar so as to include a variety of other constructions; however, the grammar used contains those constructions which are most central, in the recipe domain, in investigating reference.

In contrast, a central feature of the RP task is not to concentrate on a restricted class of constructions (although there will, of necessity, be only a finite number of them, this being a finite task), but instead to be a model where potentially all constructions can be included and treated equally: the purpose is to have a general method for evaluating the effects of particular constructions (or more precisely, the effects of choosing to change via a paraphrase relation—a specification for mapping from one unit of text to another—from one particular construction to another), and then to select from among the constructions on that basis, rather than to investigate one particular phenomenon embodied in a particular set of constructions.

The other main difference is the way in which a choice is made between alternative constructions. In these early systems (and, in fact, in systems generally), there is a default choice; so, for example, an unmarked choice (such as the simple active declarative) may be made the default over a marked choice (such as the clefted version of the same sentence). This is a reasonable approach, and may actually mirror human cognitive processes in choosing language constructions, but the method of choosing a default is somewhat ad hoc. In TEXT, the default is the alternative which occurs in the first position of the tactical generator's grammar:

> If the grammar value is an alternative, all options are unified and the first successful result taken. [McKeown, 1985: 141]

> The first success halts the unification of following alternatives. [McKeown, 1985: 144]

Similarly, in EPICURE, even though the sentence grammar is more restricted, the noun phrase grammar has choices which appear to be defaults (for example, *an onion ring* over *a ring of onion*), although it is not precisely clear. EPICURE has two intermediate levels of representation, a Recoverable Semantic (RS) Structure, and an Abstract Syntactic (AS) Structure, the latter embodying the form the construction will take when realised as text. There is a one-to-many mapping between RS and AS levels: for example, the one RS describing an onion ring maps to two AS structures corresponding to *an onion ring* and *a ring of onion*. The precise mechanism for choosing between these alternatives is not explicitly detailed, but is described as being parallel to the choice between mappings from the knowledge base (KB) to the RS structures, also one-to-many; and this process is described as just choosing the first alternative (making this the default) and then backtracking to generate other alternatives:

> Mapping rules are collected together into ordered sets, such that for any given input structure, only the first rule in each set which matches the input structure will be used. By forcing the system to backtrack, subsequent rules which match the input structure can be used, thus producing multiple output structures for a given input structure. [Dale, 1988: 150]

And:

> Just as there is a one-to-many mapping from KB structures to RS structures, there is a one-to-many mapping from RS structures to AS structures: the

> recoverable semantics of an utterance underspecify the particular realization of that utterance. [Dale, 1988: 153]

There is no guiding principle in the choosing of these defaults by the system; they are just the first alternative in the grammar. There may, of course, be some guiding principle in the construction of the grammar, but this is not mentioned in McKeown's description of TEXT or in Dale's description of EPICURE. Related to this is that there is no evaluation of which is the best alternative, unless a particular alternative is specifically chosen, such as in the choice of the passive construction based on focus information from the strategic generator. The selection of defaults in such a generation system is a complex task: in RP, however, it falls out fairly naturally from the framework itself, where the original construction is the default. All possible alternatives are evaluated against this, and the best (which may be the default) is chosen, based on the overall system constraints.

2.1.2 Stylistically Controlled Generation

In the work described earlier in this section, once the message to be generated was decided on by a text planning component, the actual realisation often led to a default choice. That is, there was in general no preference which proposed, for example, the use of an Adjective Phrase over a Relative Clause (*the new car* over *the car which is new*) in a specific set of circumstances, but instead just a sometimes arbitrary, all-purpose choice. Some attempts were made to generate texts appropriate to different circumstances—for example, Hovy's PAULINE system (1988), which made a choice from among sets of 'equivalent' grammatical constructions based on a user model—but these were generally hand-coded and lexically specific.

Scott and de Souza

A more general approach is to incorporate ideas of style into the generation process. Early proponents of this were Scott and de Souza (1990), who considered a cognitively-based (as opposed to aesthetically-based) notion of style. Their ideas were developed within the framework of Rhetorical Structure Theory (RST), and were realised as a number of heuristics based on psycholinguistic evidence. For example, two of the heuristics were:

- Rhetorical relations that are expressed within a single sentence are more easily understood than those expressed in more than one sentence.
- Syntactically simple expressions of embedding are to be preferred over more complex ones.

The second of these would in general prefer *the new car* over *the car which is new*, although Scott and de Souza suggest it should not apply if the syntactically simpler form is lexically unusual (this codicil leading to the preference of *with rancour* over *rancourously*).

The contrast with RP is that these heuristics all work at the level of rhetorical relations; the most surface-level constraint is a heuristic preferring concise expression over verbosity. This is appropriate for the internal workings of a generation system, but has the same two problems as Meteer's SPOKESMAN system (see Section 2.1.3) when it comes to text-to-text

revision under surface-level constraints: too great a level of abstraction and the absence of a natural language understanding system which can infer from a text the rhetorical relations it embodies. The level of rhetorical relations can really only deal with surface-level constraints in conjunction with a level closer to syntax; but the question is moot in any case since there is no system (or comprehensively detailed standard theory) which achieves the transformation of text to an RST representation.

STYLISTIQUE

Di Marco and Hirst (1993) advance the idea of using style in generating text, although they use an aesthetically-based notion of style. They are careful to emphasise that the 'style' they discuss is not just literary style, which is often seen as a superfluous addition when the aim is clear understanding or expression of a text; nor is it just a precriptivist set of rules adjuring, for example, the use of dangling participles. Instead, it is a characteristic of all writing, neither 'good' nor 'bad', but an expression of the writer's intended effect on the reader. Di Marco and Hirst express it this way:

> Style in language is not just surface appearance, a decorative veneer. Rather, it is an essential part of meaning, part of the author's communication to the reader. So to fully understand the nuances of a text, one must determine not only the propositional content, but also how its communicative effect is colored by the form, which reflects affective content. While propositional content provides the basic *tone*, the expressive form provides the *tonal quality*. Together, form and content create style, that which distinguishes both an individual text and a collective body of writing. [DiMarco and Hirst, 1993: 451–452]

In practical terms, Di Marco and Hirst position their work by noting the need for stylistic mapping in machine translation: in human translation, to achieve what is considered 'good' translation, stylistic factors are taken into account to produce a natural-sounding resultant text. So, for example, in a translation from English to French, a text will often need to become more abstract if it is to sound natural (Vinay and Darbelnet, 1958). Di Marco and Hirst take the standpoint, in common with Hovy, that this type of intention by the author can be captured by a notion of stylistic goal (Hovy's term being 'rhetorical goals of style'), such as clarity–obscurity, or abstraction–concreteness. Di Marco and Hirst differentiate their work from Hovy's by noting the essentially heuristic nature of his goals and pragmatic effects; in contrast, Di Marco and Hirst point to the quite complex stylistic grammar they have developed. This grammar has several layers. At the top are the goals already mentioned; these are realised by abstract elements of style, which characterise the features of balance, dominance and position, and also describe whether they are concordant or discordant; and these abstract elements are in turn realised by primitive stylistic elements. In the key paper describing the work (Di Marco and Hirst, 1993), only syntactic constructions are investigated in building a set of primitive elements, but later work (such as Di Marco *et al*, 1993) also looks particularly at lexical choice.

This goal-directed notion of style leads, in a process like machine translation, to a goal-driven process of text production. Instead of fairly arbitrary choices being made between 'equivalent' expressions (beyond the realisation choices necessary to ensure grammaticality and so on), the choice of syntactic construction is driven by goals which mean that there is no default generation.

Di Marco and Hirst's fine-grained approach to linguistic decision-making, the idea that each decision made is significant, is similar to one of the philosophical foundations of RP, in that the starting text being analysed under RP is assumed to have been precisely crafted by the author:

> These [lexical, syntactic and semantic] decisions are assumed by the audience not to have been made randomly, but rather in specific, deliberate ways that encode additional information, such as opinion, emotional affect, and interpersonal relationships ... Thus the speaker or writer is held *accountable* for his or her stylistic decisions, in the sense that that term is used in Ethnomethodology and Conversation Analysis. A person encountering friends on the street, for example, may choose to greet them or not, but is held accountable either way—failure to greet is a snub; it is not possible to opt out of the situation altogether (Heritage, 1984). Similarly, a speaker or writer is accountable for all the stylistic nuances of his or her utterances; it is not possible to utter a sentence in such a way that only the propositional content counts and not the form in which it is expressed. [DiMarco and Hirst, 1993: 455]

However, the purposes of Di Marco and Hirst are different from those of RP. In addition to the given application of preserving stylistic goals in translation, other possible uses could be, for example, to modify the style of a text, perhaps from concrete to abstract, in a way similar to PAULINE. However, these changes are 'deep' changes, very different in character from the surface constraints RP deals with. Also, Di Marco and Hirst's work does not look at choosing the best (in this case, stylistically most appropriate) text, another main aim under RP, although it is possible to envisage the RP technique for a scalable handling of goal interaction, described in Chapter 8, being applied to STYLISTIQUE: currently, there are only three goals in the STYLISTIQUE grammar, which is not problematic; but with more, the interaction would become more complex.

2.1.3 Revision-Based Generation

Although they start from some knowledge base, revision-based generation systems are quite similar—in some cases, very similar—to RP. They generate drafts, and review each draft to see if it fits the constraints imposed on the text; going from one draft to the next involves some kind of paraphrase, just as in RP. This is particularly appropriate for dealing with surface constraints, as argued in this section. This subsection looks at four frameworks for revision-based generation, and discusses how RP differs in its finding of an optimal solution, and its doing so in one, as opposed to more than one, draft.

YH

Gabriel's YH system (1988) is an early example of a revision-based generation system which is, in part, motivated by surface constraints, although, unlike WEIVER (described later in this section), the motivation comes mainly from the AI goal of attempting to model the process of text production in a human-like way. Gabriel argues for 'deliberate writing', which for people is a process involving writing, walking away, then looking again at the writing after enough time has elapsed to notice any textual infelicities.

YH generates text by consulting 'experts'. These experts are objects within an object-oriented programming framework, encapsulating knowledge about both domain and linguistic expression. That is, some experts encapsulate knowledge about syntactic constructions such as the passive voice version of a declarative sentence (referred to below as the 'passive voice expert'), while other experts embody knowledge about the Dutch National Flag program (the YH domain of generation) and about Lisp programming of it.

Each expert has a number of features, such as GOALS, CONSTRAINTS and INFLUENCES. When taken together, these decide which expert is the appropriate one for each situation. In terms of textual expression, situations which call for the intervention of an expert are flagged by 'critics'; an original text is generated, and then criticised:

> ... this program generates text from left-to-right, making locally good decisions at each step. As the generation proceeds, other parts of the program observe the generation process, and, because these parts of the program are able to make connections between distant parts of the text, they are able to criticise the result. [Gabriel, 1988: 10]

As an example, an original text generated by YH is:

> The one-dimensional, zero-based array of n elements, FLAG, represents the flag. There are three array markers, L, M, and R, standing for left, middle and right, respectively. L is initialised to 0. M is initialised to 0. R is initialised to $n - 1$.

After this is generated, the critics review it; they decide, in this case, that it is important that the concrete, real-world object (the flag) be made the topic of the paragraph. After consultation, the passive voice expert is selected (based on its goals, influences, and so on, a MEASURE OF IMPORTANCE, a numeric value, is calculated from these and weighed against that of other experts) as the most appropriate, and then it is applied to the text. The following text results.

> The flag is represented by a 1-dimensional, 0-based array of n elements, FLAG. There are three array markers, L, M, and R, standing for Left, Middle and Right, respectively. L and M are initialised to 0; R is initialised to $n - 1$.

RP's approach to rewriting is quite similar to this, although the YH perspective, with its experts, critics and so on, is more anthropomorphic. The first difference of substance is in the way that paraphrasing decisions are made (or, in the YH analogue, experts invoked). As indicated by the first quote above, YH decides on changes in a left-to-right manner, that is, sequentially rather than simultaneously. This has two main effects. The first main effect is that it leads to a non-optimal traversal of the space of possible rewritings. Taking the example described above, the passive voice expert is invoked to change the first sentence for the given reason of concreteness. However, if there are situations further along in the text where the only alternative is invocation of the passive voice, but where this is prevented by the passive voice expert's constraints on adjacency of passives (as there are not allowed to be too many passives located near each other), the consequent backtracking will lead to an unwinding of all the decisions made. Subsequently, all the decisions are remade;

this is equivalent to an explicit enumeration of the space of alternatives, notwithstanding any chart-style mechanism for storing blocks of choices. A central feature of the RP model of this thesis is the phrasing of the problem in such a way that it is possible to *implicitly* enumerate the alternatives (implicit enumeration being, in fact, the name of the optimisation technique involving binary variables that is introduced in Chapter 8).

The second main effect is that the making of locally good decisions does not guarantee a globally good result. Choosing the passive voice expert for the first sentence may, in fact, lead to a valid sequence of changes for the text, but there is no sense in which this can be assured of being the sequence which is the best overall. Taking a specific numeric example, the MEASURE OF IMPORTANCE (MoI), which estimates the relevance of an expert, may have been some value x for the passive voice expert discussed previously, and $x - 1$ for an alternative expert which consequently lost out (say, one which indicated topic by italicisation). Other things being equal, this may lead to, say, a choice of an expert other than the passive voice expert for the next topicalisation decision, with this other expert's MoI being $x-5$; the passive voice expert's alternative for this second topicalisation decision, whose MoI would otherwise be x, is disallowed by proximity considerations. This gives a total for the actual choices of $2x - 5 + K$, where K is the total relevance of the other choices, against $2x - 1 + K$ for the global optimum. In other words, because the changes are made sequentially, it is possible to overlook better long-range alternatives.

The second key difference between YH's framework and RP, which is even more significant, is in the type of rewritings. Robin (1994), in his comparison of YH with his own STREAK system, describes the YH rewritings as information-*preserving*, in contrast to his own information-*adding* ones. From Gabriel's description of his own system, this does not appear to be precisely accurate—YH does make decisions to add information to a basic framework. For example, it places on the agenda, which is apparently used for both domain and language experts, the idea of including information about initialisation of the marker values used in the Dutch National Flag program; it seems possible that the inclusion of this information is thus optional, depending on the measure of importance associated with its expert, and so on. In any case, whether these rewritings are genuinely information-adding (that is, bringing in extra information) or merely information-preserving (rearranging information that would be said under a fixed plan), they are different from RP's INFORMATION-*lossy* rewritings in that the importance of the changes is known in YH—because YH is an NLG system, with domain experts which have knowledge about important features of programs, about which phrasings are equivalent, and so on—while in RP, because the starting point is text, it is not known with certainty what alternatives are equivalent, or how to assess the effects of a paraphrase.

SPOKESMAN

Meteer's (1991) work in the area of revision centres around the development of her Text Structure formalism. This representation describes text at a level more abstract than syntax, and allows mappings to be defined between texts represented in this way. Meteer defined a set of such mappings after an analysis of a corpus in which the alternative versions were individual texts before and after revision by a professional editor. This led to paraphrases like the following example of Meteer's:

(1) a. Hoey thus makes a distinction between ...

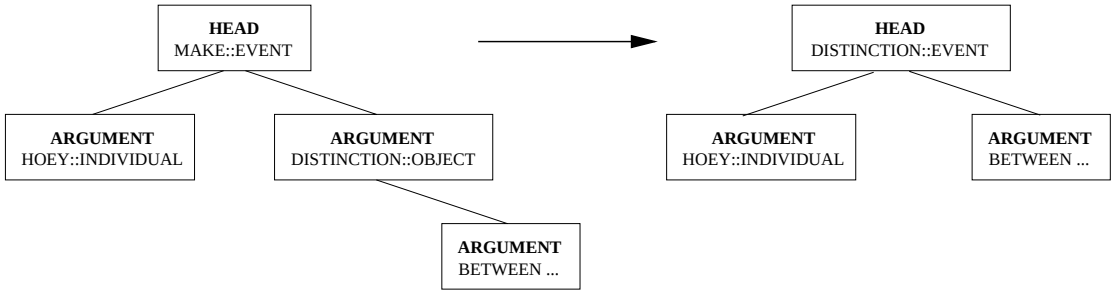


Figure 2.1: Mapping from light verb to full verb structure.

b. Hoey thus distinguishes between ...

In Meteer's representation, this gives the mapping of Figure 2.1.

Meteer notes that Text Structure captures three important generalisations of the sorts of paraphrases she describes, by the use of:

- constituency, as constituents often move while remaining largely intact (as, for example, in the *between ...* phrase of (1));
- functional relations, such as head or adjunct;
- semantic types, such as event or object, which can be used to neatly describe such patterns as the change from the light verb structure in (1a) to full verb structure of (1b).

The inclusion of this information makes Text Structure a neat representation for carrying out text-to-text paraphrasing, as was Meteer's original intention. However, approaching paraphrasing where text is the starting point is a very complex task: a natural language understanding system is needed to produce the above information on constituency, functional relations and semantic types (the latter two of which are not, in any case, standard, and would hence require a lot of hand-construction effort for a broad-coverage system), and for a broad-coverage system this is currently infeasible.

This is why Meteer herself used Text Structures within a generation system, rather than within a text-to-text system, producing a Text Structure representation from the knowledge base of her SPOKESMAN system, and revising that. Robin (1994) notes, in commenting on Meteer's work, that this inability to carry out a complex and thorough interpretation of natural language text is why he, and others (including, in fact, Meteer), have chosen to work on a representation that is, to varying extents, internal to the generation system used. Robin also notes in his critique that the Text Structure representation is too abstract to carry out revisions necessitated by surface constraints; the SPOKESMAN system would have to act upon both the Text Structure level and the Linguistic Specification Level to make the text conform to length and embeddedness constraints.

The RP model of this thesis differs from Meteer's work in two key ways. Firstly, in accepting that, at the current level of development in the field of natural language processing, it is not possible to have a broad-coverage natural language understanding system which

will transform natural language text into a Text Structure representation, it chooses to use a different, syntactic, formalism for paraphrase, rather than move from a paraphrase task to an NLG one. Secondly, Text Structure does not appear to be formally constrained in a way that allows any meaningful guarantees to be made about the formal properties of the resulting text. Instead, it works at the level of syntax, for which broad-coverage systems are available—for example, the IBM parser (Jelinek *et al*, 1994), the Alvey Natural Language Tools parser (Carroll, 1993), or the XTAG system (Doran *et al*, 1994)—and uses a representation which is adequate for describing syntactic paraphrases (see Chapters 6 and 7). This is not as much of a restriction as it might seem: the paraphrases in RP are only tools, rather than the corrective devices of SPOKESMAN. Meteer explicitly mentions this when describing her corpus analysis, stating that "[the] first phase of the analysis ... was to simply itemize the changes and look at how they functioned to *improve* the text" (Meteer, 1991: 156; emphasis added). As argued elsewhere in this thesis, particularly Section 2.2.1, the fact that in RP the paraphrases are only tools for achieving conformance of the text to surface constraints means that it is adequate to deal only with that subset of paraphrases which can be described syntactically. While Text Structure is a richer representation which allows a much wider range of paraphrases to be described (for example, covering the mapping *We have been in the process of developing RST ...* to *We have been developing RST ...* , a mapping which, in order to achieve any generality, requires a degree of semantic knowledge about the nature of processes), RP just does not attempt to cover everything that Text Structure describes. Chapter 8 will describe the limitations that this brings, in terms of the maximum conformance to surface constraints that this choice allows.

This is not to say that Text Structure and RP are incompatible: if, at some future time, natural language technology is sufficiently advanced to allow broad-coverage mapping of text to Text Structure, then this could be incorporated into an optimisation framework like that presented in Chapter 8 to allow a wider range of paraphrases and, likely, improved resultant texts.

The second key difference is in the effect of the mappings. Meteer's 1991 paper has a footnote:

> I am not claiming that the original and revised text have the exact same meaning—one can argue that any change to the text affects the meaning, however subtly—only that the change is minimal in this type of editing.

Thus she treats both alternatives in a mapping such as (1) as being equivalent; but (1b) is actually more general that (1a), also being a paraphrase for *Hoey makes <u>some</u> distinctions between ...* (see Section 4.3 for an expansion of this argument). RP does take the position that changes to the text do affect the meaning, and treats these differences as something to be minimised. This is due to the philosophical starting point taken: Meteer sees the revision process as starting from a text which is flawed, but understood by an editor (human or computational) who can fix it, while RP starts from the position that the text is the embodiment of the author's intentions, and that in changing it it is not possible to know for certain how much this change will disturb the author's intended meaning. While the truth is somewhere between the two positions—in the field of editing, the editor does have some confidence about what should be changed, but will often check with the author to make sure that no intended meaning has changed—for a computational system, where

having the skill of a human editor is a far-distant goal, the RP approach seems particularly applicable.

WEIVER

The WEIVER system by Inui *et al* (1992) is a revision-based generation architecture that incorporates paraphrasing as part of the generation process. The aim of the WEIVER system is to produce (Japanese) text that fits both rhetorical and surface constraints, in order to generate text that is both coherent and readable. Inui *et al* argue for the need for a revision-based architecture because of the difficulty of detecting certain kinds of problems before any text is generated; in particular, surface problems such as structural ambiguity, and surface complexity (average sentence length, or depth of embedding). For instance, when generating a description of an entity, there may be so many distinguishing characteristics to be enumerated that the resultant referring expression has too many levels of embedding and thereby violates the relevant surface constraint. The simple example given in the 1992 paper is the expression of the proposition *My car is new* contained within another proposition, *My car is French*. Two alternatives are *My new car is French* and *My car, which is new, is French*; the latter has an extra level of syntactic embedding which the former does not. So the level of embedding is known only after lexical and grammatical choices are made.

This led to WEIVER comprising three modules: a surface generator, an evaluator, and a revision planner. The surface generator produces a draft text, one that is not necessarily the final version. This draft is evaluated for surface problems (that is, violations of constraints) by the evaluator, and then the revision planner suggests a change to fix a particular problem, leading to a new draft produced by the surface generator.

In order to carry out an evaluation of surface features, the evaluator works on actual text, the draft produced by the surface generator. This is quite similar in concept to paraphrasing, although the actual regeneration of text is carried out by the surface generator mapping from the knowledge representation to text, rather than a mapping occurring from the draft text to a subsequent draft text.

An example of the process is as follows. The input to the system is a rhetorical structure tree, as in Figure 2.2 (taken from Inui *et al*). This can be split into sentences by the application of PC PAIRS, of which there may be a number of alternatives for one draft sentence, such as the two alternatives (PC pair #101 and PC pair #102) in Figure 2.2. The first choice of a PC pair is made by reference to a set of heuristics, drawn from Scott and de Souza (1990). For this example, where the heuristics lead to the choices of PC pair #101, this gives the sentence (translated into English):

(2) The document is kept in the library in the most inner part of the next building on the 4th floor. The library is open from 9 to 5 and is open to the public.

After generation of the draft, the evaluation shows that there is too much embedding in the draft, so the revision planner proposes PC pair #102 as an alternative, leading to:

(3) The document is kept in the library. The library is in the most inner part of the next building on the 4th floor. It is open from 9 to 5 and is open to the public.

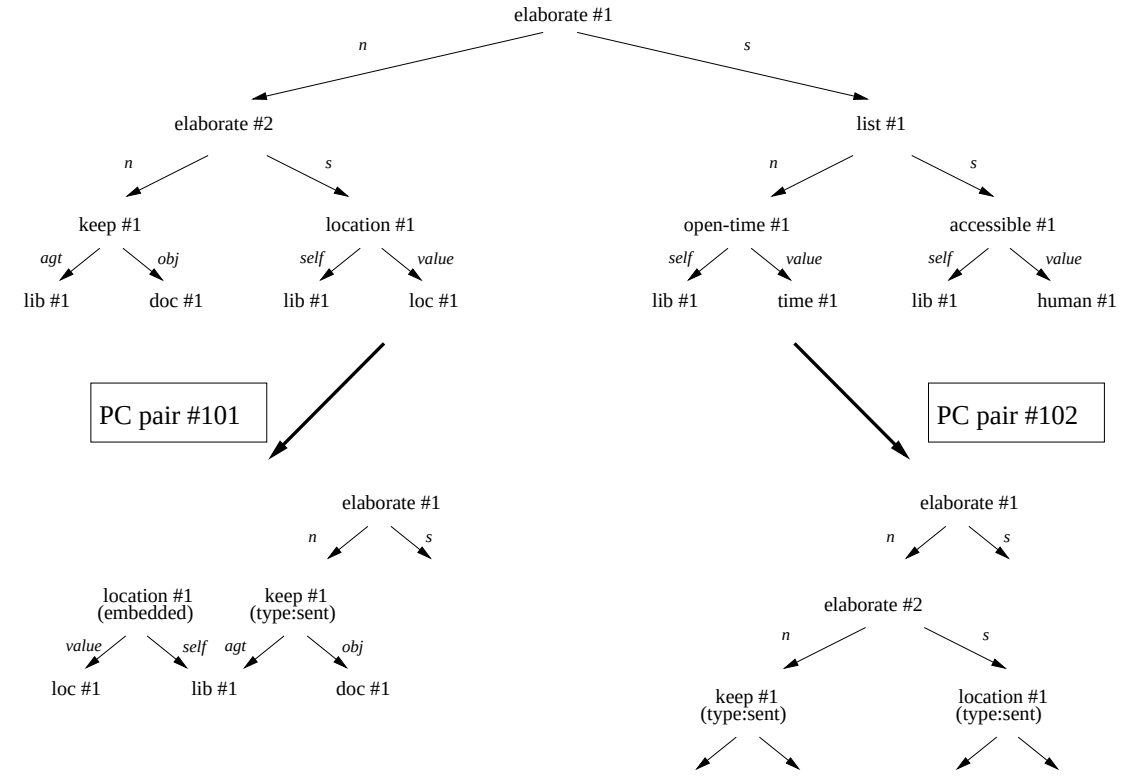


Figure 2.2: PC pair alternatives in WEIVER

A simplistic revision planner could, of course, become trapped in a cycle: for example, there may be some surface constraint which renders this second alternative unsatisfactory, and the revision planner then chooses PC pair #101 as an alternative, and so on, leading to an infinite loop. Inui *et al* deal with this by introducing a PC pair history, which records all of the choices made of PC pairs.

For small scale constraints, such as the sentence-level ones of WEIVER, this approach is not at all problematic. The revision planner just searches through the space of alternatives and picks any one which improves on the problem. However, in a larger, multi-paragraph text, this undirected traversal of the search space is very inefficient; to be at all workable, there needs to be a strategy for searching the space of all possible alternatives, and this is what the mathematical optimisation techniques described in Chapter 8 of this thesis achieve. These techniques also allow the choice of an optimal set of alternatives in one pass, once an initial text is produced (the starting point for RP, or the first draft for WEIVER), instead of constant revision and re-generation of text.

Also, the simple search strategy works well here as the constraints are all well correlated. That is, for the constraints described in the work (sentence length, depth of clause embedding, depth of modification relation in NPs, depth of centre embedding), when a revision solves one problem, it tends to take the text in the correct direction relative to the other constraints also. In the first alternative (2) above, moving the too-deeply embedded description out of the first sentence to produce (3) also reduced the sentence length, a generally desirable state of affairs.

In using global constraints—that is, constraints on the entire text, not just on individual sentences—however, it is typically the case that these constraints conflict. For example, moving a too-deeply embedded clause to a separate sentence, while decreasing average sentence length (a component of a readability index), generally increases *total* text length because of the need for an anaphor, and other lexical items, to make the resulting sentences grammatical. This compounds the inefficiency of a simple problem-by-problem search technique, as implemented in the WEIVER revision planner: if it dealt with inversely correlated constraints, it could keep choosing alternatives which solved one problem (that is, went towards satisfying one constraint), but which kept introducing other problems (that is, moved the text away from satisfying other constraints). Again, the purpose of the techniques of mathematical optimisation is to deal with situations where these sorts of conflicting constraints are the norm.

A difference in the fundamental frameworks of RP and WEIVER also means that this sort of revision isn't appropriate for text-to-text paraphrasing. In general, starting from a knowledge base, the generator can be considered to be omniscient. That is, it is known which items are significant, which ways of expressing the knowledge are equivalent. This means that all of the alternatives that the revision planner can choose from are known to be equivalent, so the text will be of the same quality from draft to draft:

> In our method, because the surface generator generates an alternative draft from the same rhetorical structure, it is ensured that the new draft keeps the intended meaning intact. Moreover, because the surface generator also considers the external dependencies, the new draft satisfies not only the grammatical and lexical constraints, but also the pragmatic and textual preferences. Thus, it is guaranteed that the surface change will not worsen the quality of the draft. [Inui *et al*, 1992: 227]

So, even if all of the drafts do not express quite the same information (although all of the examples given by Inui *et al* do), it is because the omitted pieces of information are known not to be significant, with this being expressed in the knowledge base.

However, when paraphrasing from text, it is not known with absolute certainty which pieces of information the author intended to be significant. Neither can it be asserted that a surface change will not worsen the quality of the draft: again, because it is not known with certainty what the author intended to be significant (and even a human editor may wonder, *Is this it-cleft which can be paraphrased to standard declarative form actually significant?*), but also because, for a large-scale text-to-text system based on a broad coverage parser, it is impractical to assume perfect textual and pragmatic knowledge. Hence, an estimate of the damage to the text is an appropriate way of helping to choose between alternative expressions, if the alternatives are not known to be (or not actually) equivalent. This again points to a mathematical optimisation model, where this damage measure acts as a guide through the search space of possible alternatives.

STREAK

The STREAK system of Robin (1994) is another that incorporates paraphrase as part of a revision-based generation architecture. The domain of generation is basketball summary reports: STREAK takes as input the box scores that are used for reporting basketball games, and produces as output a text—specifically, a summary sentence modelled on the lead sentence of newswire reports—which reports the results, including historical information that has opportunistically been added in. A typical output sentence is:

> Kenny Smith scored 28 points Sunday night to pace the Houston Rockets to a 99–94 victory over Orlando, giving the Magic their league-high 10th straight loss. [Robin, 1994: 19]

In this text, Kenny Smith's score of 28 points, the total game score of 99, and the team names are items of information taken from the box scores; the 10-long losing streak and the fact that this streak is the longest in the league are items of historical information, added in to give context.

The approach taken in STREAK is to report the significant new facts, and then progressively add in as much historical information as is reasonable. This reasonableness is determined by, as in WEIVER, surface form constraints, so historical information is added only until the sentence is about to become too long or too syntactically complex:

> The ability to convey historical information thus entails the ability *to perform final content selection under surface form constraints.* [Robin, 1994: 40; emphasis in original]

Robin uses this as the main argument for a revision-based architecture: that content selection—not just final surface form, as in WEIVER—can be decided only after evaluating a draft. Robin goes further towards actual paraphrasing of this draft than Inui *et al*, in that the proposed revisions are carried out on the draft, whereas Inui *et al* regenerate a new draft from the knowledge base. More precisely, a new fact is added at the Deep Semantic Specification (DSS) level, but the actual paraphrasing occurs at the Surface Semantic Specification (SSS) and Deep Grammatical Specification (DGS) levels:

> ... most revision operations are *non*-monotonic: in order to accommodate a new floating fact, they alter the linguistic realization of the draft content. Such an alteration is decomposed into several low-level actions cutting and pasting the linguistic contents of the draft. [Robin, 1994: 85]

An initial sentence is generated, containing just the significant new facts, and this draft sentence is paraphrased to add extra information, with this process being repeated until one of a set of constraints is met. This gives rise to a sequence of drafts, this example taken from Robin (1994: 93):

(4)
a. Dallas, TX – Charles Barkley registered 42 points Friday night as the Phoenix Suns routed the Dallas Mavericks 123–97.
b. Dallas, TX – Charles Barkley registered 42 points Friday night as the Phoenix Suns handed the Dallas Mavericks their 27th defeat in a row at home 123–97.
c. Dallas, TX – Charles Barkley registered 42 points Friday night as the Phoenix Suns handed the Dallas Mavericks their franchise worst 27th defeat in a row at home 123–97.
d. Dallas, TX – Charles Barkley registered 42 points and Danny Ainge added 21 Friday night as the Phoenix Suns handed the Dallas Mavericks their franchise worst 27th defeat in a row at home 123–97.
e. Dallas, TX – Charles Barkley registered 42 points and Danny Ainge came off the bench to add 21 Friday night as the Phoenix Suns handed the Dallas Mavericks their franchise worst 27th defeat in a row at home 123–97.
f. Dallas, TX – Charles Barkley matched his season record with 42 points and Danny Ainge came off the bench to add 21 Friday night as the Phoenix Suns handed the Dallas Mavericks their franchise worst 27th defeat in a row at home 123–97.

The paraphrases that led from one draft to another can be either monotonic, as the Adjoin-of-Classifier revision transforming the draft in (4b) to the draft in (4c) (by the addition of *franchise worst*), or non-monotonic, as the Adjunctization revision transforming the original draft in (4a) to the draft in (4b) (with the change of the full verb structure to the light verb structure).

In contrast with RP, STREAK, like WEIVER, looks only at sentence-level constraints: the length of the sentence, its maximum level of embeddedness. Where STREAK differs from WEIVER is in the generally conflicting nature of its constraints and goals: it aims to maximise informativeness (which is achieved, in practice, by the addition of as many floating facts as possible) while minimising space and maximising readability (measured by depth of syntactic embeddedness), these two 'goals' being, in practice, constraints,[1] maximum permitted values for sentence length and syntactic embeddedness. However,

[1] They are technically constraints, rather than goals, as there appears to be no attempt to choose the best revision, in terms of taking the smallest number of words and least embedding (given some trade-off factor):

> When [a specific debugging feature] is not present, the reviser uses the first rule whose `bls` [Base Layered Specification—the template to be matched] feature matches the current draft

the constraints themselves are generally aligned, in that it is possible to minimise space and maximise readability by restricting the information added to a minimum.

This is unlike in RP where, as mentioned in the discussion about WEIVER in this section, the constraints are generally conflicting, necessitating an architecture where the resolution of these conflicts is part of the model. The key difference between STREAK and RP is the way in which a paraphrase is chosen. STREAK chooses the first revision rule whose righthand side matches the current draft. Once the fact added by this revision rule has become part of the draft, it is permanent: there is no way of removing a fact, even though individual linguistic elements may be deleted by a further application of paraphrasing. Because this fact was chosen merely by being the first one encountered, there is no guarantee that this will in fact give the optimal result for the three 'goals': for example, adding one fact via an Adjoin operation may prevent another fact from being added because of the length constraint, even though this second fact could have been added had an Adjunctization operation been used instead. An concrete instance of this problem with such greedy algorithms is given in Section 8.1, which explains the motivation behind the RP model with a more mathematically detailed example. In contrast to what occurs under greedy algorithms, RP looks for the combination of paraphrase applications that is optimal in the mathematical sense: it searches the space of paraphrases to find the best (that is, optimal cost) set of paraphrases, rather than just choosing the first option encountered at each opportunity, so that the text is minimally changed.

In addition to this, the constraints chosen in previous systems have been fixed and not easily changed. This is not unreasonable, given that constraints such as sentence length and embeddedness are important; however, there is a very large number of possible constraints. For instance, as well as the four used in RP—length, readability, lexical density and sentence length variation—there are many others that might be added, such as a prescribed value for syntactic right-skewedness, a restriction on the syntactic similarity of sentences, and so on, some of which are suggested in Section 3.5. When the number of desired constraints becomes large, this also leads to the need for a model which is easily extensible in the number of constraints, and which embodies an intelligent search strategy for choosing paraphrases which best satisfy these constraints.

Robin stresses that his system, with its information-*adding* paraphrases, differs from the other systems that he discusses, such as WEIVER and YH, which, he says, have information-*preserving* paraphrases; and indeed, Inui *et al* specifically note that this is the case for WEIVER. However, this knowledge about information preservation is an artefact of generating from a knowledge base; the ability to *add* information even more so. Starting from text, with no other knowledge, it is not possible to add more information, or even be certain that information is being preserved; and, in general, to satisfy typical constraints (shorter total word length, improved readability index values) when starting from text, in RP it is necessary to use information-*lossy* paraphrases. With this information loss comes the need to use a completely different model, where information loss is measured and minimised.

---

and whose `adss` [Additional Deep Sematic Specification—description of information to be added] matches the first DSS in the ordered floating fact list. [Robin, 1994: 84]

Instead of finding the best revision, the revision process continues choosing the first relevant match until particular limiting (constraint) values are reached: 45 words for the sentence length constraint, and 10 levels in depth for syntactic embedding.

2.1.4 Summary

Overall, the main difference between RP and the NLG systems described here is that under RP, starting from text, it is not known, unlike the case when starting from some knowledge base, what linguistic realisations are equivalent, and what the effects of different ones are. Additionally, these NLG systems have not had a need to use a model which allows efficient searching of the space of potential solutions. Thirdly, also because the starting point is text, a formalism for describing text-to-text mappings is necessary, which is different from the knowledge-base-to-text functioning of the NLG systems.

2.2 Correcting Text

2.2.1 Style Checkers

Style checking

Both style checking and grammar checking can be seen as the imposition of constraints on a text. It may seem odd to compare the sort of surface constraints, like length, that are the focus of RP, to a correction such as *Fix subject-verb number disagreement*; however, they do exist on a continuum in which the boundaries are somewhat blurred. Style checkers operate in a way that is sometimes very close to surface constraint satisfaction—one of the principles they implement is that of conciseness, manifested through advice to remove 'useless' words—and at other times perform functions akin to grammar checking—for example, detecting a lack of syntactic parallelism, as in *You should drink tea rather than drinking coffee*, where it could be argued that it is a question of grammar as to whether the participial form of the verb should be allowed. It has been noted often that distinctions between grammar, style and usage are not precise; and in practical systems, the division between grammatical errors and stylistic infelicities can and does move (Richardson and Braden-Harder, 1988). The remainder of this section thus looks at style checking systems, and later at controlled language checking systems, as examples of systems with very precise constraints on text; it then compares them with the RP task, as examples of models which impose constraints on existing texts.

Style checking systems

Style checkers draw most of their rules from sources such as style guides, like those of Strunk and White (1979), Kane (1983), and Williams (1990). These generally contain rules about what and what not to do: prefer the active voice, don't split infinitives, and so on. The advice varies greatly: some rules are extremely specific, as the two previously mentioned; others are vague, such as *Remove deadwood* (from *The Oxford Guide to Good English* of Kane, 1983). Some rules, as noted above, are difficult to distinguish from issues of grammar, like the infinitive-splitting rule; others are quite strongly surface-oriented, such as Strunk and White's directive to *Be concise*.

This idea of style is quite different from the concept discussed earlier in Section 2.1.2. There, style was a descriptive interpretation of the form of the text, as characterised in Crystal and Davy (1969). In style guides, one particular style is assumed, a plain style which is aimed at facilitating clear communication—see MacArthur (1991) for an overview

of the evolution of the plain style into what is popularly seen as the default correct writing style.

Style checkers, generally speaking, tend to implement the more specific, grammar-based corrections of style guides in order to make a text conform to the plain style. An early style checker, cited in most of the literature, is the Unix Writer's Workbench (WWB) (Cherry, 1981). After identifying the parts of speech of the words in a text, the STYLE component of the WWB produces a summary of text characteristics: readability grades, sentence information (such as average sentence length), sentence types, word usage (including information on passive constructions and nominalisations), and sentence beginnings. The user then compares their text against the statistics for a prototypical document of the type that is being written. The DICTION component works in tandem by suggesting alternatives for phrases considered too verbose or too hackneyed. Although, because of the earliness of the system, there is no full parsing, and hence no identification of specific constructions deemed problematic, the WWB basically implements in an approximate way the sort of advice found in style guides: use the active voice, don't have sentences that are too complex, and so on.

A subsequent system, CRITIQUE (Richardson and Braden-Harder, 1988), based on IBM's EPISTLE system (Heidorn *et al*, 1982), was the first to use a parser, and so is fairly typical of current systems, but in other respects was similar to the WWB, in that it also gives advice on specific, grammar-based problems.[2] Systems related to this, such as that incorporated into Microsoft word-processing software, suggest corrections. These later versions produce parses of sentences, check through them, and suggest alternatives for those that match some specified pattern designated as incorrect; this approach is still based on style guides, which similarly specify incorrect or undesirable constructions, and give alternative phrasings.

Some other later systems do take slightly different approaches. The Computerised Comprehensibility System (CCS) of Kieras (1990) is based around psycholinguistic evidence about which constructions are difficult to understand (for example, evidence that suggests that nominalisations take longer to comprehend than equivalent clauses leads to nominalisations being a non-preferred form), rather than using the advice found in style guides. Payette and Hirst (1992), in their system STASEL, merge the approach of typical style checkers with Di Marco and Hirst's (1993) goal-directed approach, using only the goals of clarity and conciseness. In practice, STASEL appears to work similarly to conventional style checkers in a number of ways: for example, it flags anticipatory constructions, such as *it*-clefts, in much the same manner recommended by style guides.

Comparison with RP

There are three main characteristics that these systems share which differentiate the task they are attempting to achieve—conformance of a text to a plain style—from the RP task—conformance of a text to global surface constraints.

The first is the concept of correction itself. Style checkers (and the style guides on which they are based) have a model of what makes for good text, and good constructions within

[2] To be precise, CRITIQUE is a combination of style checker and grammar checker, so the larger part of its advice is on grammar; but even its stylistic advice is related to specific grammatical features, like the syntactic parallelism mentioned above.

that text. However, deciding what is 'good' is contentious, and opinions on it change. Many older sources of stylistic advice (Orwell, 1946; Strunk and White, 1979) recommend, unqualified, the use of the active voice as the way to write; more recent sources (Williams, 1990) acknowledge that the passive voice is preferred in some cases for thematic or fluency reasons. The US Document Design Centre for a time recommended 'whiz-deletion' (transforming a full relative clause into a reduced relative clause); this is now the subject of some argument (Huckin *et al*, 1986). This pattern occurs frequently whenever constructions are deemed good or bad. There has been some effort to determine, psycholinguistically, which constructions are good or bad based on how easy or difficult they are to comprehend (see, for example, Wright (1985)); as mentioned in Section 2.2.1, this has been introduced into at least one style checking system (Kieras, 1990). The area still causes problems, however. RP, by taking the original text as ideal, avoids these problems; paraphrases (the changing of one construction to another) are neither good nor bad in themselves, but instead are all damaging to the text, with the aim of RP being to minimise this damage.

The second respect in which RP is different is the idea of LOCAL checking. In style checkers, the aim is to fix all instances of a construction deemed incorrect or inadvisable. Thus, for example, all occurrences of the passive voice will be flagged for change in most systems, this decision being based only on the fact that the sentence is in the passive voice. However, in RP, not all occurrences of the same construction are treated in the same way: some instances of the passive voice may be changed, in order to fit the text to length restrictions, but others will not, if the text is already on target to meet the length constraint.

The third characteristic of style checkers is the difficulty of a comprehensive paraphrase representation. As noted above, the sorts of advice range from specific to quite vague. As a style checker aims to correct the undesirable features of text, it will, at some stage of development, need to fix (and hence represent) all of these. Representing paraphrases which vary in abstraction from *Change passive voice to active* to *Fix incoherent theme* requires a very expressive representation; developing an elegant one would be a major task. RP, however, does not need to have such a broad representation formalism. Its goal is not to fix a wide range of phenomena; instead, the paraphrases are just tools to alter the parameters of the text to conform to the constraints. It is unnecessary, for example, to have a representation covering paraphrases to alter the thematic structure of a text, if the text can be revised to fit the constraints just using simpler, more specific paraphrases.

These three differences mean that RP does not have a number of the difficulties—and contentious points—of style checkers. On the other hand, new issues do need to be considered—What is a damage function? How can damage be measured? and so on—and these are discussed in Chapters 4 and 8.

2.2.2 Controlled Languages

Definition of Controlled Languages

Controlled languages are a subset of the set of natural languages which have been created for specific purposes. Their creation is generally motivated by the complexity of natural languages and the difficulty this complexity causes in ensuring that a text is correct, or unambiguous, or possessed of some other desirable property. Controlled languages (CLs) are specified by restricting vocabulary, grammar and surface attributes of text; specific

instances of CLs are described in this section. CLs have in general been devised for large companies which want to use them in documentation: the reasoning is that documentation in a controlled language should lead to fewer errors in implementing instructions (hence the enthusiasm with which CLs have been taken up by manufacturers of large equipment such as tractors and aeroplanes), and also that translation of documentation should be more accurate, easier and cheaper.

Checking a document written in a controlled language is similar to checking unrestricted text, with two main differences. Firstly, an extra level of checking is carried out: as well as checking for non-words or incorrect syntax, the CL checker also looks for 'unapproved' (but not necessarily syntactically incorrect) use of language, which generally has some approved alternative. Secondly, CLs often have specific surface constraints which are not relevant to natural languages; this is the aspect that makes CL checking relevant to RP, and will be discussed later in this section.

Varieties of CLs

The most widely used controlled language is AECMA[3] Simplified English, devised and used by the aerospace industry; there is also a French equivalent, GIFAS[4] Rationalised French. Farrington (1996) gives an outline of AECMA Simplified English (SE); he notes that the aerospace industry is the sort of place to which CLs are particularly suited, giving as reasons the volume of documentation—his example is a not atypical set of manuals for one aircraft containing 66000 pages—and the requirements of the documentation—accuracy, completeness, relevance, conciseness, a convincing style, meaningfulness, and lack of ambiguity—which are more easily checked in a CL than in an unrestricted natural language.

AECMA SE attempts to achieve these aims principally by restricting vocabulary, grammar and surface attributes of English. There are three sources of words in AECMA SE: approved words, technical names and manufacturing processes. Approved words, of which there are about 950, have defined meanings, and may be used in only one sense and in only one part of speech. Technical names must also be defined, and can be used only as adjectives or nouns. Manufacturing processes are always used as verbs. The Guide specifying which words are approved also specifies a number of unapproved words and their approved alternatives; for example, the noun *airframe* should be replaced by the noun *structure* under Guide specifications.

The restrictions on syntax are less comprehensive; instead of there being an exhaustive enumeration of permissible items, as for vocabulary, there is just a set of rules (approximately 55 of them) which apply to some syntactic constructions. For example,

- Wherever possible, nouns must be preceded by: a definite article (*the*); an indefinite article (*a*, *an*); or a demonstrative adjective (*this*, *that*, *those*).
- Phrases containing the progressive (*-ing*) forms of a verb are not allowed.

The surface constraints are of the type applied to constituents no larger than a sentence. For example, sentences are restricted to 20 words in length for procedure definitions, and

[3] Association Européenne des Constructeurs de Matérial Aérospatial
[4] Groupement des Industries Françaises Aéronautiques et Spatiales

25 words for descriptive passages; noun clusters are restricted to four consecutive nouns, with any additional necessary information being expressed in relative clauses, etc.[5]

Non-AECMA varieties of controlled languages include Caterpillar's Technical English (Hayes, Maxwell and Schmandt, 1996), those built with the KANT framework at Carnegie-Mellon University (Nyberg and Mitamura, 1996), one using Perkins Approved Controlled English (Douglas and Hurst, 1996), and CLs for German and Swedish (Schachtl, 1996; Janssen, Mark and Dobbert, 1996; Almqvist and Sagvall Hein, 1996). Although each has special characteristics applicable to the environment in which it is used, all are broadly similar in the relationship they have to the relevant natural language; they are all, as AECMA SE, characterised by vocabulary, syntax and surface restrictions on natural language.

Controlled Language Checkers

The aim of CL checkers is to guide a technical writer to write documents which meet the restrictions imposed by the CL. Ideally this would be a two-stage process: the first stage is the identification of text which does not meet the restrictions, and the second stage is the automatic rewriting of this text so that it does conform. This second stage is a kind of paraphrase, and has been described as such:

> As we have defined it, a controlled language is a variant of an existing sublanguage, such that expressions in the sublanguage are linked, via a paraphrase relation, to expressions in the controlled language that satisfy specific additional constraints. [van der Eijk *et al*, 1996: 67]

In general, the second step has not been successfully implemented in systems, as it is quite complex. Even the simplest type of correction, replacing an unapproved word with an approved word, is more complex than it seems at first glance: it is not possible in many cases to just replace one word with the other, but instead the entire sentence needs to be restructured. For example, under the advice of the guide to replace the unapproved word *tend* with the approved word *cause*, the following sentence must be completely restructured:

(5) As the compressor loads up, lubricating oil and glycol tend to carry forward and deposit on the densitometers.

As another example, the guide specifies that *number* should not be used as a verb, and that hence (6a) should be replaced by (6b) (Douglas and Hurst, 1996).

(6) a. Number the components.
b. Give X numbers [to the components].

It is clear that in this example, extra information (the value of X) is required, and that more than just a lexical, or even syntactic, change is involved.

Thus Clémencin (1996) notes:

[5] There are a few other restrictions that do not fit into these categories. They are generally of an imprecise nature; for example, *Keep to one topic per sentence.* These are too abstract to be dealt with by current CL checkers.

> Our intuition is that the reformulation of sentences cannot currently be fully computed using the information produced by the sole linguistic analysis [produced by a CL parser]. [Clémencin, 1996: 40]

Currently, systems take some combination of two approaches with this difficulty with paraphrasing: one approach is to deal only with simple cases automatically; the other is not to attempt an automatic paraphrase at all, but just to provide the advice about the alternatives to a proscribed constituent as stated in the guide. Two systems which lie on different points of this scale are the Boeing SE Checker (Wojcik and Holmback, 1996), and the Cap Volmac CL checker (van der Eijk *et al*, 1996). The former attempts no automatic paraphrase:

> Boeing's SE checker does not attempt to propose revisions of sentences, nor are there any plans to make it do so. Most of our error reports simply convey the same advice that one finds in the SE Guide itself ... [Wojcik and Holmback, 1996: 25]

The latter has a paraphrase capacity which can transform (7a) into (7b).

(7) a. Check that leading edges conforms to values in teh [*sic*] table.

b. Make sure that the leading edges agree with the values in the table.

Comparison with RP

The basic similarity between this CL checking and RP is that both aim to rewrite text, by means of paraphrase relations (generally with only partial coverage in CL systems, as described above), so that it fits certain constraints. Indeed, some of the constraints—for example, the AECMA surface constraints limiting sentence length—are closely related to those found in RP. However, there are a couple of aspects which make the tasks different.

In the same way as in style checking, the rewriting in CL checking is intended to improve the text. A text written in a CL is supposed to be clearer and less ambiguous than an equivalent text in a natural language; application of paraphrases is thus supposed to result in text improvement. This is explicitly stated as a goal of CL checking:

> Documents paraphrased, or created from scratch, in the controlled language should be able to perform the communicative functions of the document at least as well as corresponding documents in the non-controlled language, throughout the various stages in the document lifecycle. [van der Eijk *et al*, 1996: 67]

Studies have been carried out which support the idea that CLs are indeed clearer. Holmback *et al* (1996) tested success in comprehension and translation between comparable SE and non-SE documents, and found in some cases that there were definite, statistically significant results favouring the SE documents. These SE documents were not, however, the result of an automated rewriting of the non-SE versions; rather, they were recreated by human experts. Expecting an automated application of paraphrase relations to be capable of this quality of result seems quite ambitious; as well as the intended, positive

effect of paraphrasing, there are negative, generally unintended effects (such as, say, accidental rearrangement of topic) that are beyond the 'understanding' of the paraphrase. RP does not do this; starting from the assumption that the original text is the ideal, and an application of a paraphrase makes the text worse, all of the system is designed around the minimisation of damage, so that more radical paraphrases are avoided where possible.

Also significantly different are the difficulties encountered: RP, given its framework, does not have the same problems as CL checking. Firstly, the size of problems in CL checking is often measured from user responses, and on this metric, incorrect vocabulary corrections rate most highly problematic (Barthe, 1996). Since RP does not aim to make lexical alterations[6] this significant problem does not exist. Secondly, CL checkers attempt more difficult sorts of paraphrases, as in example (6). This is because there is only a certain set of constructions which are allowed in a CL document, so a CL checker must deal with every construction outside this set. RP, however, does not have to do this: paraphrases are provided to rewrite—if it is deemed appropriate by the goal search mechanism—those constituents which can be rewritten. If there is a construction that is too complex for the paraphrase mechanism, it is not considered as a candidate for moving the text to constraint conformance. This is discussed elsewhere in this thesis, particularly Section 2.2.1: the difference between the local constraint satisfaction of checkers—where each construction of given classes (for CL checkers, this being all sentences) must be evaluated, and all members of those classes treated in the same way (for example, all uses of *number* as a verb transformed to nominal equivalents)—and the global constraint satisfaction of RP—where each construction of given classes may not be transformed in the same way, depending on the state of the text with respect to the global constraints.

2.3 Summary

So, as noted in this chapter, the RP task is different from those addressed in other systems. In RP:

- text is mapped to text, rather than from some knowledge representation to text where equivalence of outputs is known, in some 'paraphrase mapping';
- consequently, a notion of 'damage' to the text, caused by these paraphrase mappings, must be considered;
- changes are caused by surface constraints applied to the whole of a text, rather than by the need to correct individual constituents of the text;
- constraints can conflict with each other, leading to a large search space and the need to navigate it efficiently.

The last of these characteristics could, in fact, apply to NLG, but has not occurred in systems so far.

These are the four basic characteristics of the RP framework. The framework is not tied to any particular surface constraints or means of mapping text to text; however, to illustrate

[6] Except as a side-effect, for example, of the rewriting of a light verb construction as its full verb equivalent, where the noun in the former changes to a verb in the latter; this can, of course, be seen as a morphological process occurring on the same word (Quirk *et al*, 1985).

the framework it is of course necessary to use some particular constraints and paraphrases, and see what is involved in such use, thus the following two chapters gather together the components of the RP framework. Chapter 3 examines constraints that might be used under RP: those that are part of the systems described in this chapter are not necessarily appropriate, being either of an inappropriate type—prescriptive grammatical advice rather than surface requirements of a particular task—or positively correlated—a situation which will in general not occur in RP. Then, Chapter 4 examines paraphrases that might be used in RP: those of the present chapter are in general prescriptive, or focussed to a particular application, so Chapter 4 builds a general purpose, descriptive collection from a range of sources.

Chapter 3

Constraints

The basic elements of a model of Reluctant Paraphrase (RP), beyond a source text, are constraints which force the text to be modified, and paraphrases which are the tools for carrying out this modification. This chapter presents in a descriptive manner the four constraints used in this thesis. A more formal treatment is given in Chapter 8, where linguistic and mathematical formalisms are developed for precisely defining these constraints and their interactions with the model's paraphrases.

Often when writing, constraints are imposed on the source text, forcing it to be rewritten so that it is judged acceptable, for reasons of space, house style conformity, and so on. In constrast with the revision-based systems described in Chapter 2, these constraints are typically applied to the text as a whole. STREAK (Robin, 1994), for example, also has a length constraint, but this is a per-sentence limit of 46 words; the length constraint used in this thesis is the length of the whole text.

Four constraints on text—length, readability, lexical density and sentential variation—are presented in this chapter. This is not intended to be an exhaustive list of constraints that can be applied to text. Rather, it is a sample of some that are used in the production of written text, or are proposed for use by, for example, style guides and psycholinguists. The four given here are all surface constraints, applied to surface features of the text, and there is some debate about the appropriateness of applying surface constraints to text. The measure that is the focus of most of the debate is readability as a surface phenomenon; Section 3.2, discussing readability formulae, then looks at the validity of using such formulae, but the arguments can be more generally applied to the other surface constraints also.

In terms of developing the paraphrasing model of this thesis, the constraints described are, by virtue of their sometimes-conflicting nature, sufficient to show the need for a new model which, unlike existing models, caters for conflicting constraints that apply to the whole of a text, as discussed in Section 8.1. Additionally, the model proposed in this thesis is one where, should new extra constraints be desired, they can be easily fitted in by framing them as constraint equations (Section 8.3.1) and adding them to the mathematical model described in Chapter 8; examples of potential new constraints are given in Section 3.5.

3.1 Length

The simplest of the four constraints on text discussed in this thesis is length. In general, the constraint is a maximum length for the text (rather than a minimum, which would involve 'padding out' the text in order to meet the requirements[1]). Some of the time this is only a 'soft' constraint, in the form of a recommendation to be concise; such recommendations are typically found in style guides:

> Vigorous writing is concise. A sentence should contain no unnecessary words, a paragraph no unnecessary sentences, for the same reason that a drawing should have no unnecessary lines and a machine no unnecessary parts. This requires not that the writer make all his sentences short, or that he avoid all detail and treat his subjects only in outline, but that every word tell. [Strunk and White, 1979: 23].

> To write clearly, we have to know not only how to manage the flow of ideas but also how to express them clearly ... Usually, compress what you mean into the fewest words. Don't state what your readers can easily infer. [Williams, 1990: 115].

In particular, removal of unnecessary elements of the text is recommended to achieve a shorter text that is more effective.

> Literally deadwood means that a word fulfils no function—that is, it conveys no meaning and contributes in no significant way to the relationship between writer and reader or to the rhythm and movement of the sentence ... Deadwood violates the virtue of concision ... The problem is that deadwood interferes with communication. Dead words get in the way of vital words. [Kane, 1983: 372].

While the scientific basis for their prescriptions might at times be somewhat unclear, these style guides are used by large numbers of people throughout the world—as evidenced by the numerous reprintings of popular books such as *The Elements of Style* (Strunk and White, 1979)—and so their recommendations of shorter text are worth considering as a potential constraint for this reason alone. Other, less prescriptive, writers such as Crystal and Davy (1969) note that conciseness is a feature of different genres of text, and so, in order to conform to the style of that genre, a particular level of conciseness should be applied: for example, less formal, more conversation-like writing tends to be less concise, while scientific or technical writing is towards the more concise end of the spectrum. Results confirming this descriptive analysis of text have been found in a quantitive analysis of text genres by Biber (1988).

In other cases, there are specific, 'hard' constraints on length. One example is in conference papers: when writing a paper for a conference, it is often the case that the original draft is too long, and it is necessary to compress the original text to conform to the imposed

[1] This is not really feasible for a system which only uses syntactic paraphrases of existing text, as this sort of paraphrase cannot add any extra content; however, such padding is technically possible through the addition of function words.

word limit	page limit	other limit	no details	no limit
51	85	2	15	9

Figure 3.1: Distribution of length limits in a conference announcement corpus

limit. This length limit will generally take the form of either a maximum number of words, or a maximum number of pages. For this thesis a corpus of conference announcements was examined to verify the commonplace nature of these types of length constraints. The conference announcements in the corpus were all final calls for papers, as this is where submission format specifications are made, and were taken from the LINGUIST email list; the conferences were mostly in the field of computational linguistics, with others being in closely related fields such as theoretical linguistics, cognitive science and logic. The corpus consisted of announcements from 1 January 1996 until 11 April 1998, for a total of 149 announcements. Almost all of these had maximum length constraints of some sort, as demonstrated in Table 3.1.[2,3]

Another area where there are tight constraints on length is in the writing of news text in newspapers. After the original story is written, it generally needs to be cut down to fit the available space:

> The copy editor's first task is to reduce the volume of news to a level where it approaches what the outlet can use ... While the main means of reducing the flow is through acceptance and rejection of complete stories, abbreviating the stories which are accepted also serves to reduce. [Bell, 1991: 76]

It is in areas such as editing of news text that combinations of constraints occur: as well as length, news editors are also concerned about readability, the constraint discussed in the following section.

Note that there is an important distinction between application of this length constraint and the process of summarisation. Although summarisation can be viewed as applying a stringent length constraint (say, shortening a text to 10% of its original length) while keeping the most important elements, the length constraint in RP is one where the modified text is intended to be close to the original: the magnitude of length adjustment is just that that can be achieved by small-scale modification or editing, such as by the syntactic paraphrases described in this thesis in Section 4.2.

3.2 Readability Formulae

Producing good readable text is an important goal of writing. One approach to producing good text is to use explicit rule- or measurement-based approaches; that is, rules are given to guide the process of writing and rewriting, or a metric used to predict readability, or

[2]Note that the total of all entries in Table 3.1 is greater than 149 as some conferences specify limits in terms of both number of words and number of pages.

[3]The 'no details' column of Table 3.1 covers announcements where submission details are given in a place other than the email, for example, at a Web URL.

both. Rules are typically like those found in style guides, and they are generally fairly simplistic; measurement metrics come in for the same criticism.

The concept of a quantitative measure of readability is quite an old one—the earliest "readability formula" is that of Lorge, developed in 1939. Readability formulae were designed as a tool for standardising the evaluation of readability of texts with respect to a given group of readers, particularly in educational and military fields or for texts targeted to the general public. Using readability formulae to predict the readability of a piece of text became an alternative to measuring readability with test subjects, or to relying on instinct. However, because of their simplistic nature, there has been some debate over whether readability formulae should be used to guide writing. On the other hand, they are used in practice in a number of areas, notably in the writing of technical documentation, as detailed in this section; as such, they worth including as a constraint in a computational model of reluctant paraphrase. Moreover, this section argues that there are theoretially valid grounds for using readability formulae as a constraint guiding a computational paraphrase system.

Accurately judging readability is an important issue when writing for a specific target population, where there is a particular educational or communicative goal behind the writing; two specific examples of this are writing for children and writing technical documents. In doing the former, it is important to ensure that the text is constrained in such a way that the readability level is appropriate for a particular age group; with the latter, it is important to ensure that instructions can be easily comprehended, or that an argument is clear.

Evaluating the level of readability can be done in three basic ways—by intuition, by measurement, or by prediction. Intuition is the method often used by teachers and writers, and is developed through their experience in practically evaluating readability. It is generally the most qualitatively extensive measure of readability, as it takes into account factors that more formalised measures of readability overlook due to their essentially simplificatory nature; but it is difficult to use with consistency over large volumes of text. Klare (1974–75) notes that other researchers "have shown that human raters can be used in lieu of formulas even when judging as many as 80 books": presumably, it was not practical to use human raters to evaluate beyond 80 books.

The second alternative is measuring readability. This can be done in several ways: two common ones are multiple choice comprehension tests and cloze tests (tests where every nth—generally fifth—word is omitted, with the test subject asked to fill in each blank). This method has the advantage of consistency, but is expensive and time-consuming to implement.

The third alternative is using formulae to predict the readability of a piece of text. Formulae are also consistent across large volumes of text, are in addition cheap to implement, and fit the concept of an imposed constraint most easily; however, they only look at superficial indicators. These indicators are textual parameters like average length of sentence in words, average number of syllables per word, and number of 'difficult' words. Weightings are assigned to each parameter in the formulae, and often also checked by calculating a correlation with a popular testing standard, such as the McCall-Crabbs "Standard Test Lessons in Reading" (McCall and Crabbs, 1961). Most of the major formulae are decades old; some of the more widely used ones, as summarised in Klare (1974–75), are described below.

3.2.1 Well-known Formulae

The Lorge Formula Developed to evaluate children's reading material, this was first published in 1939. It uses one version of the Dale list of words, which listed common words that ordinary people should know. The revised and corrected formula given in Lorge (1948) is

$$\text{grade placement} = 0.06x_1 + 0.10x_2 + 0.10x_3 + 1.99$$

where

x_1 = average sentence length in words
x_2 = number of prepositional phrases per 100 words
x_3 = number of different 'hard' words not on the Dale list of 769 words

The Flesch Formulae In contrast to the Lorge formula, these were designed for general adult reading matter. Two of the most widely used (originally defined in Flesch, 1943) are

$$\text{grade placement} = 0.07x_m + 0.07x_s - 0.05x_h + 3.27$$

where

x_m = number of affixes
x_s = average sentence length in words
x_h = number of personal references

and

$$\text{reading ease} = 206.835 - 0.846w_l - 1.015s_l$$

where

w_l = number of syllables per 100 words
s_l = average number of words per sentence

Note that the grade placement formula was also recalculated in 1948, giving the equation above, after an error was found in Lorge's formula, on which it was partly based.

The Dale–Chall Formula This formula (Dale and Chall, 1948) and Flesch's Reading Ease score were the two most widely used formulae. Klare (1974–75) suggests that it was "the most accurate general purpose formula available up to 1960". The formula is

$$x_{c50} = 0.1579x_1 + 0.0496x_2 + 3.6365$$

where

x_{c50} = reading grade score of a pupil who could answer one-half of the test questions on a passage correctly
x_1 = Dale score, or percentage of words outside the Dale list of 3000
x_2 = average sentence length in words

The FOG Index Robert Gunning, the developer of the Fog Index, proposed counting 'hard words', which he took to be words of three or more syllables. The formula is

$$\text{reading grade level} = 0.4x_1 + 0.4x_2$$

where

x_1 = average sentence length
x_2 = percentage of words of three or more syllables

The resulting reading grade level approximates the number of years of education a reader would have to have to understand the text easily.

Other formulae There are many other formulae derived from these by, for example, recalculating the parameter weightings from a new corpus, or by using parameters that are slightly different and easier to calculate. Klare (1974–75), in his overview of readability formulae, describes these and more in some detail.

3.2.2 Use of Readability Formulae

There are, generally speaking, two uses of readability formulae: pure prediction, and production guidance. The former was the original intended use—evaluating a text to see whether it will be 'comprehensible', under some definition of the word, to a particular audience. It led fairly naturally to the latter use, which is as an aid to producing text, either in the original writing process or through revisions, so that it will be readable for a specific audience. This use then leads to an imposed constraint on the text—it must conform to specific readability formulae criteria in order to be judged acceptable. However, there is some argument about the validity of both uses, and about the idea of readability formulae in general.

In this thesis, I propose using readability formulae as one of the constraints under which text can be reluctantly paraphrased. The first argument for doing so is on the grounds of general usage: that is, many people use readability formulae for text production—and in some cases readability formula use is made mandatory by house style policies—and so it can be seen as a typical externally imposed constraint, which does not need to be reasonable in and of itself. A second argument is based on an evaluation of the literature on readablity formulae which concludes that, in the particular circumstances in which I propose to use them, they are reasonable. These arguments also extend, to varying extents, to the sorts of surface constraints on which RP is based.

Actual usage People do actually use readability formulae in a wide range of areas; this is noted almost universally by commentators on readability formulae, whether they are for or against the use of readability formulae. In education, they are used to decide "whether the texts are suitable for adoption in the classroom at different grade levels" (Frase, 1981); this is especially true for basal or remedial readers, and some States of the US require that basal series must pass a readability formula criterion if they are to be used (Bruce *et al*, 1981). Bruce *et al* also note that this is increasingly true of public documents, that "insurance policies, tax forms, contracts, and jury instructions must meet criteria stated in terms of

readability formulas" (*ibid*: 50). Duffy (1985) states that "most state legislatures have relied on a readability criterion for judging the plain English of consumer documents" (*ibid*: 114). Duffy's own work concentrates on the use by the military, which requires the use of readability formulae as a criterion of the contract covering production of their technical manuals. Drury (1985) comments that they are used by journalists—as the formulae were originally developed with popular magazines—and by textbook publishers. In all of these areas the readability formula is used as a legal or contractual criterion, whether required by publishers, governments, or some other authority; there is no disagreement that there are many situations where the use of readability formulae is mandatory.

The military has developed computational systems which guide writers using readability formulae. One large-scale system that was implemented and used was that of Kincaid, the psychologist in charge of the Readability Project for the US Navy Training Analysis and Evaluation Group; the system is described in Kincaid *et al* (1981). The system, called the Computer Readability Editing System (CRES), evaluated a text for readability using stored lists of basic words and readability formulae. It was a system similar to, albeit simpler than, other grammar checking / advice systems such as the Unix Writers Workbench (Cherry, 1981), except that it explicitly recommended use of a readability formula, the Flesch–Kincaid formula (a recalculated version of the Flesch Reading Ease formula), to improve editing:

> The readability grade level thus calculated by the editing system serves as a general guide to the writer concerning the appropriateness of the material for the intended readers. If the grade level is too high, the text should be simplified [Kincaid *et al*, 1981: 39].

The paper adds the proviso that "readability formulas provide only a general indication of the overall level of difficulty. The other components of CRES provide more specific feedback to writers about particular words and sentences" (Kincaid *et al*: 38). A more recent implementation of text revision software incorporating readability formulae, CCS, is also used by the US Navy, and is described in Kieras (1990). This is similar to earlier systems, but is more sophisticated in its use of text parsing and building of semantic structures.

Others recommend use of readability formulae in an advisory way, for instance as part of house style requirements. As an example, Xerox has a manual covering publishing standards for documentation (Xerox, 1988). In this manual, the use of readability formulae is advised, in particular, use of the Fry readability scale. The Fry readability scale (Fry, 1965) is effectively a graphical representation of the Flesch reading ease score defined earlier. The range of values is from 1 (elementary school level) to 17+ (college graduate) with the advice of the manual being to aim for an 8th grade level of readability. Similarly, Diane Ritchey, Head Technical Writer of On Command Video Corp, notes (personal communication) that her company has a Documentation Style Guide (DSG) which advises the use of the readability formulae found in grammar checkers. They use a standard word processor, Microsoft Word, which processor comes with a grammar checker which calculates, among other measures, Gunning's Fog Index and the Flesch Reading Ease score. As mentioned earlier, the Fog Index measures the approximate number of years of education a reader would need in order to understand a document easily. The DSG gives the following score–education correspondence for the Fog Index:

1–12	grade and high school level
13–16	college level
17–18	master's level college education
19–21	doctorate level

It recommends that a writer aim for a score of 10–11.

For the Flesch readability scale, there is a percentage score:

90–100	very easy
80–90	easy
70–80	fairly easy
60–70	standard
50–60	fairly difficult
30–50	difficult
0–30	very difficult

The DSG recommends a range of 60–70.

The procedure for writing a technical document involves writing a first draft, then reviewing it, then editing it to fit the guidelines, one of which is the readability formula target. As a guide to usage, the Xerox manual notes:

> Use readability formulas to evaluate a document *after* you've written it. Readability formulas should not dictate what and how you write. Counting words per sentence while you write will seriously impair your progress. If you find you've written your document at too high a grade level for your reader, you can revise it in several ways. For example, you can change long or abstract words, vary sentence length, or edit your writing for conversational style. You will have then used the formula as it is meant to be used—as a quantitative measure and as an indicator of possible modifications. [Xerox, 1988: 3–16]

So in summary, judging from current opinion, it seems that readability formulae are used fairly widely, which is sufficient justification for including a readability metric as a constraint in a computational system for reluctant paraphrasing.

Theoretical Reasons for Usage However, appealing to current opinion justifies inclusion only during such time as readability formulae are in favour; it is also important to look at the arguments and evidence about whether readability formulae should be used in editing, and decide on that basis. In this section, I will argue for use of readability formulae only in the specific context of their use as a constraint or goal in a computational system for Reluctant Paraphrasing.

In a more general context than that of RP, opinion on the use of readability formulae in text production and editing is sharply divided. Some writers assert that it is very important to use readability formulae in editing—Vervalin (1980) says that it is "the first step in defogging your communications"; others who also advocate its use see it in a more subsidiary role—Powell (1981) notes that readability formulae "have some weaknesses and should not be the final arbiter in how we write, but used with understanding, they can be a

useful tool in the writer's bag". On the other side of the debate, some say that readability formulae are useless—Redish (1981) quotes a Navy study which showed no link between use of readability formulae and improved comprehension; others contend that they are worse than useless, and actually harmful—Bruce *et al* (1981) discuss a study by Davison *et al* (1980) to demonstrate the point. Specific arguments against their use are given below, along with an evaluation of how they relate to the use of readability formulae as a constraint or goal in the computational paraphrasing model of this thesis.

Claim 1: Readability formulae don't specify the causes of reading difficulty.

Redish (1981) comments on this in her analysis of readability formulae, noting that having a high score (that is, an indication that the material is difficult to read) doesn't tell the writer what to change in order to make it easier to read; Frase (1981) tags the readability formulae as "analytically weak" because of this. However, readability formulae proponents acknowledge this: that using readability formulae as a guide in order to improve writing, for example by merely changing the textual parameters corresponding to a formula's parameters, "can provide no assurance of improving readability" (Klare, 1974–75: 98), as the parameters, typically word length and sentence length, are generally simple and superficial: "no argument that they [the parameters] cause ease or difficulty is intended; they are merely good indices of difficulty" (*ibid*: 97). The position of these advocates is that readability formulae are used to predict whether there will be reading difficulties, and if there are, 'clear writing techniques' are used to edit the text. Klare (1979: 82–83; quoted in Duffy, 1985) gives an algorithm for this kind of targeted editing:

1. Apply a formula to see if a piece of writing is likely to be readable to intended readers.

2. If the readability index suggests it is, and if other requirements for good writing have been met, stop there. Keep in mind, in other words, that while a poor index value predicts poor writing, a good index by itself need not mean good writing.

3. If the readability index suggests the piece of writing is not likely to be readable to intended readers, put the formula aside so as not to be tempted to "write to formula".

4. Rewrite the material, trying to discover and change those parts likely to cause trouble. Use the formula information only as a guide to where to begin.

5. Apply the formula again to see if the piece of writing is now more likely to be readable to intended readers.

6. If it is, and other requirements for good writing are met, stop there.

7. If it is not, repeat steps 3, 4 and 5 until an appropriate readability index is achieved.

Claim 2: Introducing clear writing guidelines doesn't help; they just create new problems.

Duffy (1985) asserts that a procedure such as the one outlined above will not work. Firstly, he questions whether it is possible for the writer to put the readability formula criterion

"to the side". He believes it is not, given that writers will be influenced, by factors such as budgetary constraints and production schedules, to "write to the formula" if this is what is required of them. Even though this is true, it is not a convincing argument to not use readability formulae when there is no alternative target metric for guiding editing. In either scenario—with or without readability formulae—there will be writers who are pressured, or are lazy, or for some reason produce sloppy work; if the guidelines are shown to work for the conscientious writer, they are still worth pursuing. In any case, a computational algorithm, the specific application of the readability formulae target discussed in this section, is not something that would be influenced by budgetary constraints or production schedules in making its revision choices.

Secondly, Duffy doubts that clear writing guidelines do produce more comprehensible text. Duffy and Kabance (1982) carried out a number of experiments in which aspects of vocabulary and sentence structure were altered in accordance with the guidelines, and the effects on comprehension evaluated. Duffy and Kabance found no significant effects on comprehension. There are other studies, however, which do show effects for clear writing guidelines, although they show that not all guidelines are uniformly useful. Coleman (1962) evaluated three simplification strategies, and found different effects for each. Breaking compound sentences at *and* was found to be ineffectual. Raising clause fragments (for example, participial phrases) resulted in some improvement in comprehension. The most reliable of the three strategies was the third, breaking sentences joined by coordinating conjunctions other than *and*.

Other studies suggest that the improvement from clear writing guidelines comes at least partly in the form of increased reading speed. Wright (1985) notes that this is true for several constructions she tested in psycholinguistic experiments: changing constructions containing nominalisations to have a full verb as head (for example, (1a) to (1b)) produced a very statistically significant effect, as did adverbial movement (for example, (2a) to (2b)). The other strategy evaluated was relative clause promotion (for example, (3a) to (3b)), where it was shown that reading speed did not necessarily increase with the promotion of relative clauses.

(1) a. The tenants wanted a reduction in the charge for electricity.
b. The tenants wanted the charge for electricity to be reduced.

(2) a. The cyclist pedaled along the footpath in absent-minded fashion.
b. In absent-minded fashion the cyclist pedaled along the footpath.

(3) a. The carpet was woven by craftsmen who had learned their trade in China.
b. The carpet was woven by craftsmen. They had learned their trade in China.

However, Wright tested promotion of relative clauses at only one level of embeddedness, and only gives as examples subject relative clauses; it seems reasonable to suppose that the rewriting of a sentence with multiples levels of embedding, particularly where linearity of narrative is not preserved, could result in the sentence being comprehended more quickly—for example, rewriting (4a) as (4b).

(4) a. Pity the rat (which) the cat (which) the dog saw ate.
b. The dog saw a cat; the cat ate a rat; pity the rat.

Similarly, 'garden path' sentences (for example, (5a)), which are noted in psycholinguistic literature as causing a slowing in reading speed disproportionate to the syntactic complexity of the sentence, could be rewritten to remove the cause of the garden-pathing. For example, (5a) could be rewritten as (5b).

(5) a. The horse raced past the barn fell.

b. A horse was raced past the barn. It fell.

A problem with this is that editors aren't always good judges of readability, and this leads to bad guidelines for clear writing. One conclusion that Wright (1985) draws from her experiments is that "editorial judgments about preferred forms of expression may not always correspond to the reader's relative ease of comprehension" (*ibid*: 90). For example, some clear writing guidelines recommend using 'whiz deletion' (i.e. removing the relative pronoun from a relative clause, as in *Joe wants the blazer which was designed by BMW* becoming *Joe wants the blazer designed by BMW*), which gives an improved result on most readability formulae. However, this often makes a text more difficult to understand (Huckin *et al*, 1986). Davison *et al* (1980), in a study on using readability formulae to revise text, also note that other common clear writing guidelines can actually increase the difficulty of the text. They examined some conversions of texts so that they would be suitable for children; an example from one such conversion is from (6a) to (6b).

(6) a. If given a chance before another fire comes, the tree will heal its own wounds by growing new bark over the burned part.

b. If given a chance before another fire comes, the tree will heal its own wounds. It will grow new bark over the burned part.

The sort of guideline behind this conversion would produce an improvement under most readability formulae, and so would be recommended under a revision system based on readability formulae; however, the new version requires the reader to make the inference that the growing of new bark is the way that the tree carries out the healing. Bruce *et al* (1981) use this as a justification for concluding that it is inappropriate to use readability formulae in conjunction with clear writing guidelines to edit text. However, Davison *et al* (1980) themselves say:

> Objective measurements do not tell a writer how to produce a text, as numerous writers on formulas have pointed out. Yet writers may try to tailor their sentences and use of words so that the formulas will yield the reading-level figure that is being aimed at. Of course adaptors do not follow readability formulas slavishly. In this study, we noted a good deal of conscientious and careful rewriting, but as the examples we will discuss presently show, it seems that vocabulary lists and restrictions on sentence length and passage length are often given primary importance at the expense of other factors which no one would deny are related to readability. These include the syntactic structure of sentences, the logical or discourse connections between sentences or clauses, coherence of topics, and similar categories. We will argue that more attention needs to be given to these factors for which there is to date no objective measurement. [*ibid*: 5]

So, a more appropriate conclusion is that the use of readability formulae is not wrong, it is merely not sufficient; and that clear writing guidelines should be evaluated before use in experiments such as those of Wright (1985), and only the effective ones used.

Claim 3: Readability formulae have strayed too far from their original usage and assumptions.

Redish (1981) notes that readability formulae were originally developed in the 1920s and 1930s to help textbook writers and publishers to target their work to children, and that in the 1940s Flesch produced formulae for readability of popular magazines; and that, since then, they have been applied to a much wider variety of fields—military, consumer, legal, and so on—without any real checking of whether assumptions made in the development of a formula hold in the new area of application. One example of such an assumption which is not checked relates to the level of comprehension. Since most readability formulae predict reading grades (or some related measure), the process of deriving a formula involves calibrating its parameters with respect to an independent measure of Reading Grade Levels (RGLs), so that the output from the formula is an appropriate reading grade. For example, Kincaid *et al* (1975) derived a readability formula for the military using cloze test results and performance in the Gates–MacGintie reading test (Gates and MacGintie, 1965). Comprehension, for an individual reader, was defined as scoring 35% or more on a cloze test for a specific text. Readers were grouped according to RGLs using the Gates–MacGintie test. Then, each group was tested for comprehension of a text; group comprehension was defined as 50% of the individual readers in the group comprehending the text, according to the definition of individual comprehension above. The passage was then assigned the lowest RGL of the groups which were deemed to comprehend the passage. However, notwithstanding the arbitrariness of the designated comprehension scores, this assumption of comprehension has little to do with many of the tasks for which readability formulae are used. The 35% cloze test result was assumed by the readability formula designers to be equivalent to a 70% score in a multiple choice test; but, according to reading teachers, a score of 70% indicates that the reader cannot adequately comprehend the passage without assistance (Entin and Klare, 1978). For many tasks, this is an inadequate definition of comprehension. Moreoever, this equating of the cloze and multiple choice results has been shown to be wrong: a score of 35% in the cloze test is closer to 50% on a multiple choice test, according to Duffy (1985).

However, the problems caused by the violation of assumptions like this affect only the exact predictive use of the readability formulae. Duffy (1985) defines two types of prediction: the sense of STRONG PREDICTION—that is, "not simply that text A is easier to comprehend than text B, but that the text will be comprehended by a reader at a particular reading level" (p117)—and the sense of WEAK PREDICTION, which is only a relative comparison of texts, ranking them in order of readability. Even Duffy, who concludes that there are no circumstances under which strong prediction is valid, concedes that his concerns are solely with the strong use of readability formulae (p118). This is because implicit changes in the assumptions are generally monotonic in nature. Continuing the example of the comprehension assumption above: if a new task has a required individual comprehension equivalent to a 50% cloze result instead of 35%, the strong prediction will no longer be valid; the RGL predicted by the readability formula will underestimate the actual RGL that would apply if the assumptions were correct. So, for instance, the readability formula

might predict an RGL of 10, but the actual RGL is 13. However, in the weak sense of prediction, this is not a problem. When comparing two texts, both will be affected by the error in the assumption in the same way, so the rank ordering will not change.

In the computational model proposed in this thesis, there is no prior specification of how the readability formula target should be defined. Using weak prediction is acceptable; there is no reason for the readability goal not to in principle be defined relative to another text—for example, "revise this text, A, so that it is approximately as readable as text B".

In any case, strong prediction is still potentially feasible—albeit probably impractical—if assumptions are identified and their sensitivity determined. So, for example, sensitivity to change in the comprehension assumptions could be determined by carrying out experiments, which might show that an increase of 15% in the cloze comprehension results will result in an increase of three RGLs.

Claim 4: Readability formulae have been shown to be inaccurate.

Kern (1979) analysed the predictions of military-based readability formulae on a set of military texts, and discovered that they were very inaccurate. The errors ranged as high as 9 RGLs; the statistical standard error of the formulae was only 1.6 RGL. However, Duffy (1985) notes that the errors occur because, in the process of calibrating the parameters of the formulae, too few sets of scores (only 20, in fact) were used. This finding does not invalidate the principle of readability formulae; rather, it indicates a need for a more rigorous statistical basis for the regression procedure involved in calculating the parameters.

Claim 5: There are many important factors that readability formulae don't measure.

Many writers have noted that readability formulae, in looking only at the word level and the sentence level, ignore important features of the text which affect readability. Bruce *et al* (1981) note that readability formulae don't take into account discourse cohesion, rhetorical structure, dialect, background knowledge required, and so on; Redish (1981) points out that typography, use of graphics and illustrations, and so on are important for readability, but are not included; Drury (1985) mentions visual presentation; Wright (1985) talks of theme and ambiguity in affecting readability difficulty.

These are all true; but readability formulae in themselves are not trying to find causes of reading difficulty, they are aiming to predict readability. It is still possible to predict a phenomenon without knowing what causes it. Long sentences and words are not necessarily the sole causes of a text being difficult to read; but there is generally a correlation between them and the causes of reading difficulty mentioned above, which make the readability formulae valid as a prediction tool. Other measures—such as the clear writing guidelines—are used to alter the causes of reading difficulty, with the readability formulae as a guide in the process. It is obvious that clear writing guidelines will not be sufficient to remove all causes of reading difficulty—the other causes above are also important, and in some cases more so—but they go part of the way; and, more importantly, they are the most computationally tractable of all those factors suggested.

Claim 6: Readability formulae also don't measure the reader's task or motivation.

As discussed above, readability formulae are not related to the reader's task—they give the same prediction regardless of whether the reader is reading for pleasure, or is studying for a test that is critical in getting a promotion, despite the fact that comprehension of the text will be different in different situations. Motivation is a factor here. As Bruce *et al* (1981) comment:

> A health information sheet describing the concept and treatment of hypertension, for example, may communicate quite effectively if a patient has enough time to read it and feels comfortable asking a physician for clarification. In a rushed, brusque encounter, however, the document would be much less comprehensible. [Bruce *et al*, 1981: 50]

However, this is again only an argument against strong prediction; that is, it is impossible to predict exactly how well the reader will comprehend the text, because of the varying situations under which the reading will take place. However, it does not discount readability formulae being useful as part of a process of paraphrasing a text to make it relatively more readable. Assuming clear writing guidelines do work (an assumption which should be tested psycholinguistically, as discussed earlier), using the readability formulae will produce texts that are more readable than previously. It seems reasonable to suppose that this would occur regardless of the motivation of the reader. This, admittedly, is another assumption that should be tested, that there are no situations in which using clear writing guidelines will decrease the readability of a text. For example, it should be tested whether using clear writing guidelines under conditions where there is no reading pressure increases reading difficulty: for instance, longer sentences slowing down a more expert reader in a specialist domain. However, this assumption seems intuitively reasonable.

Claim 7: There is a need for other measures besides readability formulae.

Readability formulae generally measure only sentence length and word length; it has been suggested that other measures may be at least as appropriate. Vervalin (1980) and Powell (1981) note that it is important to have variety in sentence length.

> When writing on a technical subject, focus on shortening sentences too. But have a balance, that is, not too many short sentences run together. Rather, orchestrate your writing. Two short sentences, a long one, a medium, a short, another medium, two shorts, etc. Modulating your writing in this way avoids the bumpy, choppy effect you get by having all your sentences short. [Vervalin, 1980: 88]

Redish (1981) notes the trade-off between sentence length and lexical density: it is possible to make a sentence shorter by, for example, compressing a phrase into a compound noun; however, this makes the text lexically denser—that is, it has more 'content words' per sentence—which makes reading more difficult by removing explicitly signalled relationships (see Section 3.3). Frase (1981) advocates multiple measures, although he does not specify exactly what sort:

> Use multiple measures. Written communication is complex, and single measures tell us little. Measure many variables to avoid losing sight of this complexity. [Frase, 1981: 49]

Plung (1981) recommends a measure of 'word precision'. Noting Flesch's comment that readability formulae were originally designed to identify conceptual abstraction as a measure of reading difficulty, Plung suggests that words should be categorised according to their degree of precision, as measured by the number of meanings. None of this, however, suggests that readability formulae are bad, merely not adequate.

These other proposed measures are also useful, and can in fact be combined with readability formulae. This is one of the motivations for using the constraint optimisation approach of the computational model of this thesis: that there can be measurable goals for a text—readability indices, target lexical density scores, lexical precision, and so on—which will at times compete with each other to be satisfied, and it is the goal of the model to optimise the revision to achieve the maximum overall satisfaction of these goals.

Claim 8: Readability formulae do not judge the inherent comprehensibility of the content, nor do they take account of its grammaticality.

Redish (1981) described a calculation of Dale–Chall and Fry readability formulae scores for an English translation of Plato's dialogue Parmenides; the scores indicated that it was appropriate reading material for fourth to eighth grade school children. A more general argument, found in Frase (1981), is that it is possible to have grammatically scrambled text which will score well on a readability index. For example, a text composed of sentences like

(7) Dog my car blue.

would score well on most indices, because of the short sentence length and short words. That is, the content is not comprehensible, but readability formulae indicate that it is.

However, in editing it is a reasonable assumption that the text will generally be comprehensible to some degree, since the author is intending to convey some information. It may be that it is first necessary to carry out some grammar correction, for example, but it is reasonable to take as a starting point that the text is adequately grammatical and the content is appropriate for the intended audience. This is particularly the case in Reluctant Paraphrasing, where an assumption is that the starting text is ideal, precisely conveying what the author wants, and any editing necesarily moves it away from this; the research thus only concentrates on the use of readability formulae from that point onwards.

Claim 9: Intuition is a better guide than rules are.

One school of editing believes that the rule-based approach produces bad text, and that intuition is the best guide, as it incorporates all of the factors involved in readability described above, obviating the need for readability formulae. This is an approach often used in the rewriting of texts for different reading ages, according to two editors at the publishing house Weldon's, Helen Bateman and Claire Craig (personal communication). Weldon's publishes children's books, with the writing targeted towards 8 to 13 year olds.

There is no specific process of simplification from adult texts; instead, there are a number of central visual images around which text is developed from scratch, and this means that there are no specific text simplification guidelines.

The texts are, however, all reviewed by the text editor to ensure appropriateness of readability. This editing stage is the one where text is altered so that it is not too complex for the reading audience. Craig describes the task as being an intuition-based one—she alters the text based on her knowledge and experience—and as one which is too complex for a simplistic quantitative approach. It is true that some components of the process are more complex than could be dealt with using only simple rules and targets—for example, ensuring logical topic ordering or a maximum of three topics; however, after some discussion, it did appear that syntax-based rules are used, but at a subconscious level—for example, the editor's intuition leads her to act as if governed by a rule stating:

> Phrases which post-modify nouns and which are headed by a participle are too complex, and should be split off to form a separate independent clause, taking into account changes in given/new status.

More importantly, a number of intuitions could be captured by formula-based targets. For example, Craig said that it was important to have sentences which are about the correct length, but which aren't monotonously regular, something which was noted by Vervalin (1980); see Section 3.4 for a concrete proposal of such a measure.

However, while this type of intuitive editing may be preferable for many human editors, it is not feasible for a computational system, which needs explicit rules.[4] Duffy (1985) concludes his review of readability formulae and clear writing guidelines by arguing that "guidelines and regulations in general will be ineffective when relied upon as the primary control strategy. Guidelines at best can provide the novice with general strategies for preparing a document". But, in this situation, the computer is just that: a novice. It does not have the text manipulation skills of a human, so using readability formulae as a constraint driving revision strategies seems like a reasonable approach.

3.2.3 Summary of Appropriateness of Readability Formulae

Readability formulae are fairly simplistic measures of ease of reading, generally based on average sentence and word lengths. This simplistic nature has led to disputes about the validity of using readability formulae in editing text. However, evaluating these arguments suggests that in the context of a computational paraphrasing system their use is reasonable:

- Although they don't specify causes of reading difficulty, they can, and should, be used in conjunction with guidelines which do address such causes.
- The criticisms of the strong predictive powers of readability formulae, particularly regarding demonstration of their inaccuracy and the violation of implicit assumptions, do not hold for using readability formulae for weak prediction, and can in some cases be corrected.

[4] This ignores approaches such as connectionist models of computation; however, at this stage, the task described is too complex for these sorts of models.

- The need for measures other than readability formulae is acknowledged, and is in fact an important part of the computational model described.
- The acknowledgement that a system based on the computational model is in effect a novice editor justifies using readability formulae, simplistic though they are, rather than more holistic editing techniques used by human editors.

Perhaps most importantly, readability formulae are used as editing targets by a wide range of writers and editors, in industry, government and the military; this alone is a major factor in choosing readability formulae to be one of the types of constraints incorporated into the computational paraphrasing model.

3.3 Lexical Density

Lexical density is a measure of how densely packed the text is. As an example taken from Halliday (1985b), (8a) and (8b) are paraphrases with different lexical density values: (8a) is denser than (8b), having the same information expressed with fewer words that carry no content.

(8) a. Sex determination varies in different organisms.

b. The way sex is determined varies in different organisms.

Lexical density is usually calculated as the proportion of content words to total words (where total words equals content words plus function words) in a text. Content words are generally taken to be members of grammatically open word classes: nouns, verbs, adjective, and adverbs. Function words are those from closed classes: articles, demonstratives, pronouns, prepositions, conjunctions and interjections (Quirk *et al*, 1985). However, there is a blurring between content and function words, and even between the open and closed classes on which their definition is based. Adverbs, for example, can be subdivided into two subclasses, the productive subclass (generally, an adjective plus *-ly* suffix) and the non-productive subclass (*now*, *there*, *forward*, *very*, and so on). The productive subclass counts as one of the open classes, there being a partial function from the class of adjectives to this subclass of adverb; the non-productive subclass fits in with the closed classes, having characteristics similar to other closed classes.[5] There are also subclasses of verbs that act like closed class words (or alternatively, act in a functional as opposed to contentful manner). The first, and more well established, of these are the auxiliary verbs: *do*, *be*, *have*. The other subclass is that of light verbs. In the phrase *make a distinction*, interchangeable (in some sense of the word, to be defined more precisely in Section 4.3) with the verb *distinguish*, *make* acts as a light verb: it is only necessary because English sentences require a verb in order to be grammatical, and from which to hang tense and aspect markers. *Make* itself, in this context, as with other light verbs, does not have any

[5]Quirk *et al* (1985) define three of these characteristics. The first is that the words in the classes are RECIPROCALLY EXCLUSIVE, with the decision to use one item in a given structure excluding the possibility of using any other; this is not so much the case for the non-productive adverbs as for other closed classes. The second is that they are RECIPROCALLY DEFINING, meaning that it is less easy to state the meaning of any individual item than to define it in relation to the rest of the system. The third is the ability to create new items: creating new nouns, for example, is acceptable, where creating new articles is generally not.

content[6]; each nominalisation (like *distinction*) which takes part in light verb constructions has its own corresponding light verb, so the class is not productive; and they can be distinguished from full verbs (and full usages of light verbs) in text, through both rule-based approaches (Wierzbicka, 1982) or statistical approaches (Grefenstette and Teufel, 1995; Dras and Johnson, 1996). Constructions like this occur frequently in sources describing paraphrasing, and so are discussed further in Section 4.2. However, in spite of this imprecision at the boundary between content words and function words, it is still a widely used dichotomy, and will be used in this thesis in the traditional manner, as outlined at the start of this section.

In terms of the usage of lexical density as a measure of text, Halliday (1985b) notes that the degree of lexical density is one aspect in which written language and spoken language differ: written text tends to be more compact, like (8a), with spoken text more like (8b). Often, particularly in scientific writing, there is a tendency to compress text very tightly, with a high proportion of compound nouns, complex noun phrases, and other constructions lacking function words (Sampson and Haigh, 1988). Often, also, this can make technical text difficult to read, so that it is desirable to paraphrase the text to become less dense (Jordan, 1994). Psycholinguistic studies have shown (for example, Perfetti, 1969) that there is a correlation between lexical density and sentence comprehension, with sentences having a high lexical density being more difficult to absorb, so controlling the lexical density of a text is one way of helping less able readers, particularly those who are reading a language that is not their native tongue (Bradac *et al*, 1977).

One of the problems with using readability formulae as a guide to paraphrasing text is that it can increase the lexical density of a text, actually making it more difficult to read (Redish, 1981). For example, paraphrasing the sentence (8b) to (8a) would improve the readability score, as there are fewer words over the same number of sentences. As such, it is useful to have lexical density as a constraint in order to act as a counterweight to the problematic side-effects of a readability formula constraint, in addition to being included on its own merits.

3.4 Sentence Length Variation

The three measures proposed so far as constraints, and described above, are all standard measures: formulae have been discussed in the literature, and their virtues debated. Sentence length variation has also been proposed as a way of characterising text, but the proposals have not been made in any concrete manner. Style guides (e.g. Williams, 1990) recommend alternating sentence length so that the text doesn't become monotonous. In particular, those who find readability formulae to be, by themselves, inadequate, often recommend a measure of sentence length variation in addition to the readability formula value (Vervalin, 1980; Powell, 1981); it is possible for readability formulae to lead to a text comprising only short, simple sentences, which, most writers on style agree, is undesirable. For example, consider the three short texts below, adapted from Jordan (1993).[7]

(9) a. The cars came cruising up Broadway. The cars are glittering. The paint is

[6]Other instance of *make*, as in *make a cake*, do of course have content, indicating an act of creation; this is the full sense of the verb.

[7]These examples were taken originally from Strong (1973).

harsh. The paint is metallic. The paint is highly waxed. There is a rumble of exhaust. Lights explode softly off the scene. The lights are for the street. The scene is primitive.

b. The glittering cars come cruising up Broadway. Their paint is harsh, metallic and highly waxed. There is a rumbling of exhaust. Street lights explode softly off the primitive scene.

c. The glittering cars, their paint harsh, metallic and highly waxed, come cruising up Broadway. The exhausts rumble. Lights on the street explode softly off the primitive scene.

The first two versions, while having different levels of syntactic complexity (which is also reflected in their differing average sentence lengths), both have low levels of variation in sentence length. This is particularly noticeable in (9a), and would become more evident in (9b) if more sentences of similar length and syntactic structure were added. By contrast, (9c) has a large degree of variation in sentence length.

This thesis proposes a concrete measure of variation of sentence length: the statistical variance of the length of the sentences in the text, as variance is the basic measure of deviation from a standard. This is defined as

$$\text{variance} = \frac{E(X - \bar{X})^2}{n - 1}$$

where

X = variable that the variance is being calculated for, here sentence length
$\bar{X}$ = average sentence length
n = sample size

Under this measure, the texts (9a), (9b) and (9c), with average sentence lengths of 5, 7.25 and 9 words, have variance scores of 1, 0.9167 and 31 respectively: corresponding to intuition, the third version is much more variable than the first two.

3.5 Other Possible Constraints

The four constraints defined above are just a sample set: this thesis does not argue that these will comprehensively cover every type of constraint that might be desired in an RP system, just that they are sufficient to demonstrate the paraphrasing model of the thesis. Indeed, there are many other possible constraints that could be added to this sort of system. Syntactic embeddedness is an obvious one, used in WEIVER (Inui *et al*, 1992) and STREAK (Robin, 1994). This measures how deeply embedded syntactic constituents are within the text. Two examples, taken from Perfetti (1969), are given below.

(10) a. The family has accepted an offer to purchase a book.

b. The use of credit by the consumer has obviously increased.

Roughly characterising the sentences, (10a) has a greater degree of embedding than (10b), because it has an embedded sentence, *to purchase a book*;[8] Perfetti has a $\bar{d}$ score of embeddedness which measures this.

It has been suggested that level of embeddedness is a factor in reading comprehension difficulty (by, for example, Davison *et al*, 1980), although the psycholinguistic evidence is not entirely clear (Yngve, 1960; Perfetti, 1969). Nonetheless, it seems intuitively plausible that more deeply embedded sentences will be more difficult to understand, which has led to embeddedness being used as a measure of text quality in existing systems; the relevant constraint in STREAK, for example, does not allow embedding any deeper than eight levels.

Another possible measure for evaluating text is its right-skewedness. Style guides such as that of Williams (1990) note that there is a 'weight' to words, and a principle of END WEIGHT which prefers heavier constituents to occur at the end of the sentence. Recently a performance theory of word order has been put forward (Hawkins, 1994) stating this in a more precise way, and using it to explain phenomena such as the preferences of alternative orderings of, for example, arguments of verbs. For example, Hawkins provides data to show that constructions like (11b) are preferred over (11a) because of the placement of heavier constituents at the end of the sentence (that is, because of the right-skewedness of the corresponding syntax trees), as seen in Figure 3.2 (corresponding to (11a)) and Figure 3.3 (corresponding to (11b)).

(11) a. Shannon put the large orange halo which had previously been owned by the Sultan of Brunei's nephew in the groundkeeper's cupboard.

b. Shannon put in the groundkeeper's cupboard the large orange halo which had previously been owned by the Sultan of Brunei's nephew.

This idea of right-skewed trees being more natural or easy to comprehend has been used in other areas, such as work on prepositional phrase attachment. Right Association was an early method for predicting the attachment of prepositional phrases that are ambiguous between noun phrase attachment and verb phrase attachment, based on the idea that right-skewed trees were easier to comprehend: choose the rightmost attachment point, that is, the one giving the most right-skewed tree (Kimball, 1973; Frazier and Fodor, 1979).

Modelling this right-skewedness as a constraint would be reasonably straightforward. In the same way that embeddedness can be measured by average number of left branches in a phrase structure tree per word or terminal symbol (Perfetti, 1969), a right-skewedness score could be calculated by averaging the number of right branches per word.

3.6 Summary

This chapter has presented one of the basic components of RP, namely the constraints on the text which force the rewriting. The four constraints which have been used—length, readability, lexical density and sentential variation—have been chosen because they illustrate a combination of constraints that are used in practice, that are embodiments of

[8] This will in general be represented in a generative grammar as a rewrite rule with an **S** as the lefthand nonterminal.

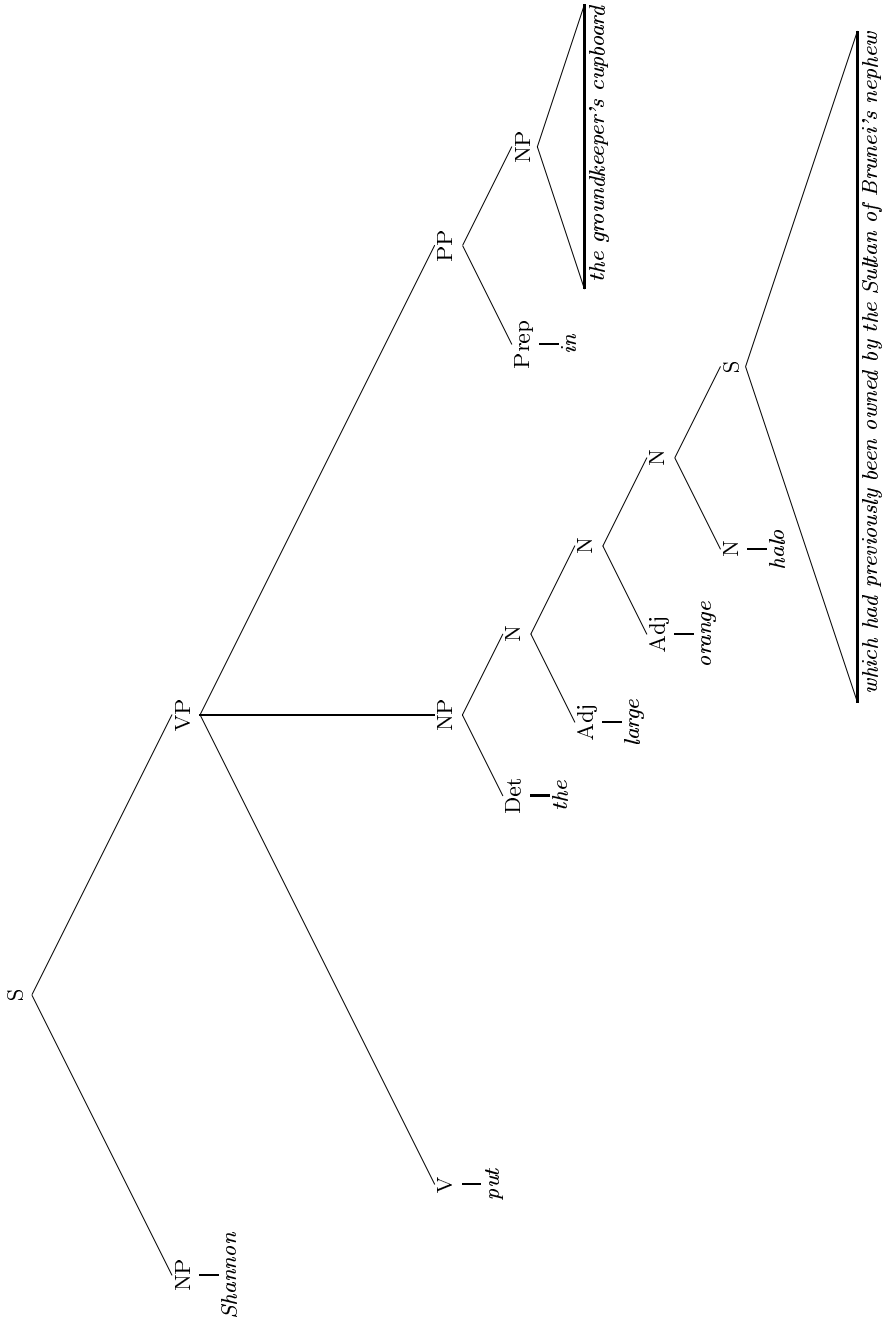


Figure 3.2: Non-right-skewed tree

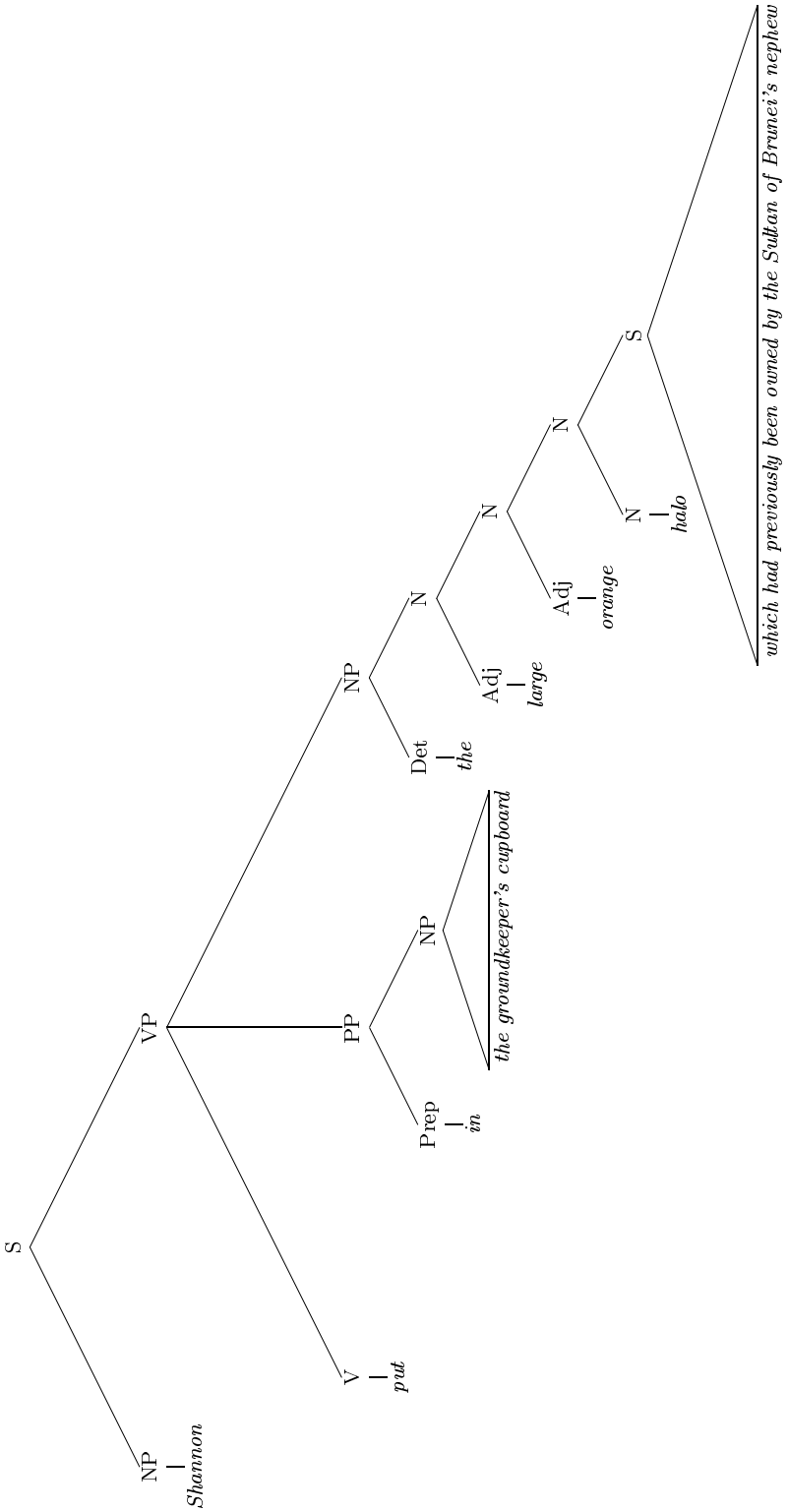


Figure 3.3: Right-skewed tree

desirable characteristics of whole texts, and that counteract or conflict with each other. The way in which these constraints interact, how they fit into the overall optimisation model of the thesis, and why, because of their characteristics, they require a different model of constraint handling than used in previous work, is covered in Chapter 8.

Chapter 4

Paraphrases

In addition to a set of constraints, such as those described in Chapter 3, the other basic element of a model of Reluctant Paraphrase (RP) is the set of the paraphrases which are the tools for carrying out the modifications forced by the constraints. As noted earlier, the concept of RP is not tied to any particular set of paraphrases, or even any particular type of paraphrase—semantic, syntactic, lexical, or other. But again, as with constraints, it is necessary to use some set of paraphrases; in order to illustrate RP in this thesis, syntactic paraphrases are the choice here, as a given syntactic paraphrase will be more widely applicable than, say a purely lexical one, and more closely related to surface phenomena (as are the constraints) than semantic paraphrases.

This chapter presents a general purpose collection of paraphrases which will be used in this thesis, along with a shallow taxonomisation and discussion of the effects of the paraphrases on a text. More importantly, it gives a progressively refined definition of what it means to be a paraphrase, beginning with the preliminary definition of Section 4.1, and culminating in the semantic model of Section 4.3. Later, a more formal treatment is given in Chapters 6 to 8, where linguistic and mathematical formalisms are developed for precisely defining these paraphrases and their interactions with each other and with the constraints of Chapter 3.

4.1 Fundamentals

The aim of this section is to make a first attempt at pinning down more precisely the notion of paraphrase; following are preliminary definitions which correspond to the usage of this thesis.

Definition 4.1.1 *A* PARAPHRASE PAIR *is a pair of units of text deemed to be interchangeable.*

Definition 4.1.2 *A* PARAPHRASE MAPPING *is the mapping between the two elements of a paraphrase pair.*

Definition 4.1.3 *A* PARAPHRASE *of a unit of text is the alternative to that unit of text given in some paraphrase pair.*

So, in terms of (1), a paraphrase pair is the pair (1a) and (1b); a paraphrase mapping is the relation between these two, with the aim of later chapters being to give a formal characterisation of this relation; and (1a) is referred to as a paraphrase of (1b), and vice versa.

(1) a. Steven made an attempt to stop playing Hearts.

b. Steven attempted to stop playing Hearts.

This idea of paraphrase differs from that of the systems described in Chapter 2, notably those revision-based generation systems described in Section 2.1.3, where the term paraphrase is used to describe generated units of text that are 'equivalent'. Here, paraphrases are directly from one unit of text to another, without any intermediate representation such as those used in generation. Inui *et al* (1992) in particular note that paraphrasing text by re-generating from an underlying representation preserves the meaning, that is, produces exactly equivalent texts (this exact equivalence being an artefact of the linguistic realisation process—only texts which express the meaning of the representation can be generated as alternatives). Robin (1994) introduced 'information-adding' paraphrases, where new information is added to a sentence, leading to a paraphrase process which incorporates the information into a sentence with a new syntactic structure, and often with different lexical elements. Again, because of re-generation from an underlying representation, equivalence of paraphrases has already been established. Under the definition of paraphrase used in this thesis, however, there is no underlying representation, hence no definition of inter-substitutability from that source; without this underlying representation, there can be no certainty of meaning equivalence. In fact, the paraphrases in this thesis do not produce equivalent texts, which has led to the looser notion of interchangeability; this is argued in Section 4.3.

Instead, paraphrase pairs have been taken from a variety of sources, which have deemed them to be interchangeable. These sources are:

- prescriptive style guides (Strunk and White, 1979; Kane, 1983; Williams, 1990), where the sources recommend that the paraphrases be applied to improve the writing;
- descriptive grammar texts (Crystal and Davy, 1969; Quirk *et al*, 1985), which merely note relations between text units deemed to be interchangeable;
- formal syntax texts (Lees, 1960; Delin, 1989; Vallduví, 1993), which give analyses to explain the relationship; and
- empirical derivation of paraphrases (Bell, 1991; Jordan, 1993; Jordan, 1994; Robin, 1994), where paraphrases are used in practical situations such as newspaper editing.

Like the constraints used in this thesis, the list of paraphrases in this chapter is not meant to be an absolute, definitive one; instead, it is a set used to demonstrate the ideas

behind the thesis. That is, the actual paraphrases (mappings between units of text) are tools rather than ends in themselves. This contrasts with the paraphrases as used in style checkers and controlled languages (Section 2.2), where paraphrases are corrective devices for fixing flaws in the original text, and hence are to some extent determined by the flaws being corrected. In terms of actual text mappings, there is some overlap, given the sources used in this thesis for obtaining paraphrases: for example, the passive–active paraphrase occurs both here and in style checkers. In style checkers, however, this particular paraphrase is expected to be applied at every possible opportunity, while under RP it is applied only when it is the best way of moving the text towards satisfying the global constraints.

To this end, the paraphrases of this thesis are restricted to being predominantly syntactic, with lexical changes applying only where made necessary by syntactic reorganisation. This characterisation of paraphrases as syntactic is not to attempt to explain paraphrase behaviour as syntactic, as the early work of Lees (1960) did, and ignore the movement to lexicalisation, where it is acknowledged that any mapping is lexically conditioned, possibly strongly so; the term syntactic is just a way of characterising those paraphrases which are predominantly about syntactic reorganisation. In fact, there is no reason not to use purely lexical paraphrases (for example, substituting *canine* for *dog*); it is just that syntactic paraphrases are more general—in that any given syntactic paraphrase will apply to a wider range of texts than a lexical one—which fits the concept of paraphrases as general-purpose tools. Again, this choice of syntactic paraphrase is not essential to RP, but does fit well because of the general applicability of syntactic paraphrases.

An additional characteristic of the paraphrases presented in this thesis is that they are, at least potentially, bidirectional. That is, applying them in either direction is in general equally valid. For example, there may be occasions when the active voice is changed to the passive; the direction of paraphrase application is determined by the constraints on the text, or more precisely, by the optimisation procedure fitting the text to the constraints as presented in Chapter 8.

The actual paraphrases are covered in Section 4.2, together with a shallow taxonomisation built bottom-up from the collected paraphrases; the effects of the paraphrases on the text, given the notion of 'interchangeability' under the definition of paraphrase in this thesis, are discussed in Section 4.3.

4.2 Types of Paraphrase

This section describes by example the paraphrases used in this thesis, with formal descriptions of the paraphrase types presented in Section 7.3, after the formal representation has been developed. The paraphrases fall fairly naturally into five classes, which is how they are listed in this section. The aim is not to produce a detailed taxonomy of paraphrases detailing which paraphrase types subsume which other, as, for example, Knott (1996) has done with his taxonomy of coherence relations, which was built up by examining the interchangeability of relevant cue phrases; rather, the aim is just to give a categorisation which makes it easy to refer to groups of paraphrases.

The five classes are Change of Perspective, Change of Emphasis, Change of Relation, Deletion, and Clause Movement. These are described and exemplified below: in each case, a general outline of the type is given, followed by specific subtypes and examples.

4.2.1 Type A: Change of Perspective

Type A (Change of Perspective) paraphrases are syntactic reworkings which involve a change in the part of speech of some key element of the text unit. They are so named here because they reflect a change in the way elements in the text unit are represented, for example, changing from an event perspective (generally represented using a verbal construction) to an object perspective (generally represented using a nominal construction). The change is generally caused by the insertion or deletion of one or more open-class elements that are effectively contentless. The most well understood of these contentless types of words are light verbs, with recognition of them dating back to at least Jespersen (1942), where examples such as *take* in *take a walk* are discussed. These verbs do not have any meaning of themselves; rather, they act as a prop for other non-verbal sentence elements, and are made necessary only by the requirement of English that sentences contain verbs, as a hook from which to hang tense and aspect markers. Often light verbs take nominalisations as arguments; sentences of this form, as in (2a) below, can then be rewritten so the nominalisation becomes a full verb, as in (2b). Other kinds of arguments for light verbs are possible, and are described by example in this Type A category. The class of light verbs is fairly small, although these verbs take a wide range of arguments, and occur in text very frequently (being among the most common verbs in English: *do*, *have*, *make*, *take*, and so on). There has been much discussion over whether the pairing of light verb and argument is idiomatic (Prince, 1974; Givón, 1979; Allerton, 1982) or subject to some, probably semantic, rules (Harris, 1957; Makkai, 1977; Wierzbicka, 1982), such as a rule stating that 'semi-voluntary actions which could cause one to feel better' will take *have*. Recent statistically-based work (Grefenstette and Teufel, 1995; Dras and Johnson, 1996) indicates that there probably is a pattern behind the pairings.

Other types of contentless open-class words (such as nouns like *fact* as head of an NP) are less well studied, although there is some discussion of them, as for example in Halliday (1985b). In any case, the method for paraphrasing them is essentially the same: the constituent that is an argument to the light element is moved to the matrix position. A generalised formalism for describing this kind of movement is Meteer's Text Structure (1991); see Section 2.1.3 for a description of this, and how it differs from paraphrases as used in this thesis.

The other main style of paraphrase in this category also involves changes in an element's part of speech, but a light element is not involved. For example, a nominalisation becomes a verb, as in (19), or an adjective becomes a noun, as in (20). This process of derivational morphology, a word changing from one part of speech to another, is discussed extensively in Quirk *et al* (1985); for example, they present a cline of nominalisations, various intermediate stages between pure verbs and pure nouns, including verbal nouns, participles and gerunds. Lees (1960) has also presented an extensive detailing of nominalisations, although within a Transformational Generative Grammar, rather than a descriptive, framework.

Following are the examples of Type A paraphrases gathered for this thesis. Each subtype is given a shorthand referent which describes very roughly the characteristic structures in the paraphrase; a more detailed representation of the change is not developed until Chapters 6 and 7. The shorthand here uses mostly standard linguistic abbreviations (for example, **NP** for noun phrase) along with a few others that are fairly self-explanatory (for example, **LV** for light verb) and, in some cases, the symbol $\rightleftharpoons$ indicating the paraphrase mapping. Thus this shorthand is somewhat akin to the Structural Analysis

of Transformational Generative Grammar, outlining relevant syntactic components of a paraphrase, rather than a Structural Change which describes how the change takes place.

LV + NP + inf-VP ⇌ V + inf-VP

(2) a. Steven **made an attempt** to stop playing Hearts.
b. Steven **attempted** to stop playing Hearts.

Here, there is a light verb with nominalisation complement and an infinitival VP following; in the paraphrase mapping the light verb disappears, the nominal complement becomes a full verb, and the infinitival VP remains.

LV + NP + PP ⇌ V + NP

(3) a. This **had a noticeable effect on** the research group's morale.
b. This **noticeably affected** the research group's morale.

(4) a. We noted that it was important for personal growth to **have a knowledge of** the game's rules.
b. We noted that it was important for personal growth to **know** the game's rules.

(5) a. Alpha's subsequent triumph **bore a strong resemblance to** gloating.
b. Alpha's subsequent triumph **strongly resembled** gloating.

(6) a. Unfortunately, playing the game **did grievous harm to** our reputations.
b. Unfortunately, playing the game **grievously harmed** our reputations.

Here, there is a light verb with nominal complement; in the paraphrase mapping the light verb disappears and the nominalisation complement becomes a full verb. Note that the PP modifying the nominal complement in (3a) becomes the NP complement of the verb in (3b). Also, adjectival modifiers of the nominal complement in (3a) become adverbs in (3b).

LV + AdjP + PP ⇌ V + NP

(7) a. Darrell's parents **were supportive of** his decision to become a drag queen and world-famous diva.
b. Darrell's parents **supported** his decision to become a drag queen and world-famous diva.

(8) a. Sister Mary Appendectomy **was helpful to** Darrell.
b. Sister Mary Appendectomy **helped** Darrell.

Similar to the above, but the complement of the light verb in (7a) is an adjective rather than a nominal group; this complement also becomes a verb, in (7b). The PP in (7a) becomes the nominal complement of the transitive verb in (7b).

LV + AdjP + PP ⇌ V + PP

(9) a. However, this **was different** from their original plans for him.

b. However, this **differed** from their original plans for him.

Similar to the above, but the underlying event (here, the event of 'differing') takes a prepositional complement, so in the paraphrase mapping the PP remains as a PP.

LV + PP + PP ⇌ V + PP

(10) a. Robin **was in collusion** with Tim when we last played 500.

b. Robin **colluded** with Tim when we last played 500.

(11) a. Even though we were **in competition** for the grand prize, it seemed unfair.

b. Even though we were **competing** for the grand prize, it seemed unfair.

Similar to the above, but the complement of the light verb is a PP. The underlying event ('colluding' in (10)) is intransitive, so the PP modifying it remains.

N + clause complement ⇌ NP

(12) a. We objected to **the way he moved** the chess pieces.

b. We objected to **his movement of** the chess pieces.

(13) a. I was amazed by **the fact that he rapidly drew** the picture.

b. I was amazed by **his rapid drawing of** the picture.

(14) a. **He is willing** to leave. **This** made Gillian upset.

b. **His willingness** to leave made Gillian upset.

(15) a. **He is free** to leave. **This** made Gillian happy.

b. **His freedom** to leave made Gillian happy.

(16) a. The pool **is deep. This** led to many deaths by drowning.

b. The pool**'s depth** led to many deaths by drowning.

Here there is an empty noun (such as *way* or *fact*), the purpose of which is to introduce a clause, as in (12a); in (12b) the clause complement becomes an NP, replacing the empty noun. The deleted constituent (an NP) has been replaced by another constituent with the same part of speech, an example of why this type of paraphrase is broader than just 'change of part of speech'.

N + AdjP ⇌ AdvP

(17) a. Marilyn carried on with her life **in a cheerful way**.
b. Marilyn carried on with her life **cheerfully**.

(18) a. She often remembered her choice **in a pensive manner**, however.
b. She often remembered her choice **pensively**, however.

Similar to the above subtype, but the modifier of the empty noun is an adjective rather than a clause, as in (17a); the adjective becomes an adverb, as in (17b).

Nominalisation with PP post-modfier ⇌ NP + inf-VP

(19) a. The tenants wanted **a reduction in** the charge for electricity.
b. The tenants wanted the charge for electricity **to be reduced**.

From (19a), a nominalisation with a PP postmodifier becomes an infinitival VP in (19b).

Nominalisation complement of subordinator ⇌ clausal complement

(20) a. Because **the soloist was ill**, they cancelled the concert.
b. Because **of the soloist's illness**, the cancelled the concert.

Nominalisation complement of Prep ⇌ reduced relative clause

(21) a. He was warned by **the repeated flashing of** a light.
b. He was warned by a light **flashing repeatedly**.

Noun compounding ⇌ PP

(22) a. The gamekeeper preferred to make **wildlife** television documentaries.
b. The gamekeeper preferred to make television documentaries **about wildlife**.

Noun compounding ⇌ relative clause

(23) a. **Spanish-speaking** people
b. people **who speak Spanish**

In this subtype, it is only the NP that is of interest. In situations where a paraphrase's units of text are taken to be sentences—as in Section 4.3 and the discussion of propositional content—assume sentences are formed using some kind of existential construction: for example, *There are people who speak Spanish.*

NP + attributive AdjP ⇌ predicative AdjP

(24) a. The examiner **who was kind** ...
b. The **kind** examiner ...

NP ⇌ qualifier

(25) a. Patrick Ewing scored 41 points. **It was a personal season high**.
b. Patrick Ewing scored **a personal season high of** 41 points.

PP ⇌ genitive

(26) a. The arrival **of the train**
b. The **train's** arrival

(27) a. The funnel **of the ship**
b. The **ship's** funnel

(28) a. The humming **of the machine** ...
b. The **machine's** humming ...

As earlier, when necessary the units of text in this paraphrase pair can be treated as sentences through the use of an existential construction, for example, *It is the funnel of the ship.* Note that not all PPs headed by *of* can take this paraphrase mapping. For example, *a man of courage* does not paraphrase to **courage's man.*

VP + NP ⇌ Agentive nominalisation + PP

(29) a. He's **selling** the car.
b. He's **the seller of** the car.

(30) a. He **studies** the book.
b. He**'s a studier of** the book.

Here, the verb corresponds to a nominalisation in an agentive form, and the sentence alternates between a subject–verb form and an equative form where the agent is equated with the agentive nominalisation.

4.2.2 Type B: Change of Emphasis

Type B (Change of Emphasis) paraphrases are those where syntactic restructuring alters the focal element the text unit for various communicative purposes. The term 'emphasis' is used here, rather than focus, as focus has a variety of technical meanings and does not coincide exactly with the paraphrases described here. This type of paraphrase is a more

general class of text mappings where the constituents are given varying prominence based on syntactic arrangement. This does, of course, have a lot in common with focus, so a brief discussion of focus is given here. In the information packaging framework of Vallduví and Engdahl (1996), the variations of focus are explained thus:

> Alternative sentential structures, differing in string order, in intonational structure, or in both, may be used to express the same propositional content. Nevertheless, these sentential structures are not interpretively equivalent in absolute terms, but rather add some extrapropositional contribution to meaning. [Vallduví and Engdahl, 1996: 459]

Vallduví and Engdahl examine various earlier proposals for explaining the choice between alternative sentential structures, and conclude that they can be grouped into two classes: those that divide the sentence into GROUND and FOCUS, and those that divide it into TOPIC and COMMENT. The ground–focus articulation splits the sentence into a known or expected part, the ground, and an informative or unexpected part, the focus. In a topic–comment articulation, the topic "performs the anchoring role to the previous discourse or the hearer's mental world", while the comment "makes some new contribution" (Vallduví and Engdahl, 1996: 465). Vallduví and Engdahl find that neither articulation by itself is sufficient for explaining choice of sentential structures, and so propose a ternary division of sentential material, with the major division of the sentence into focus and ground, and the ground further divided into link and tail; this is then used to explain syntactic emphasis and intonational prominence.

In keeping with the use of syntactic paraphrases in RP, only the syntactic aspect of information packaging is of interest here. Emphasis, the basis for this Type B category, is then used in a sense similar to that of the focus of Vallduví and Engdahl, as the part of the sentence given prominence because of its newness or interest; in the case of the paraphrases below, through devices such as clefting. Vallduví and Engdahl note, however, that it has been argued by Delin (1989), after extensive study of empirical data, that the basic role of clefts is not necessarily to realise ground–focus oppositions, but to indicate logical presupposition. *There*-sentences are often similar, in that they also give focus to particular sentential elements: "Clearly, the postverbal NP in *there*-sentences must be, in some sense, novel or hearer-new" (Vallduví, 1993: 31).

Other paraphrases in this Type B category also change the emphasis of the sentence, although emphasis here is used in a less precise sense than focus. Williams (1990) discusses reasons for choosing the passive voice over the active; by inverting the object and subject of the sentence, the passive can provide a better linkage of sentence topics. Also, this inversion can place 'heavier' constituents at the end of the sentence, in accordance with the principle of end-weight (Hawkins, 1994; see also Section 3.5); this is the natural placement of constituents that require more emphasis.

Overall, the paraphrases in this section shift the emphasis of a sentence, but notwithstanding this can be considered interchangeable. They are all well-known paraphrases—mapping between active and passive voice, clefting, pseudo-clefting, and so on—and so there is not as much explanation as in the previous section. The effects of the paraphrases in terms of shift in emphasis are discussed in Section 4.3.

Active $\rightleftharpoons$ passive

(31) a. The government forces soon **pursued** the retreating guerillas.

b. The retreating guerillas **were** soon **pursued by** the government forces.

(32) a. The butler **murdered** the detective.

b. The detective **was murdered by** the butler.

Cleft of subject

(33) a. John wore his best suit to the dance last night.

b. **It was John who** wore his best suit to the dance last night.

Cleft of direct object

(34) a. John wore his best suit to the dance last night.

b. **It was his best suit that** John wore to the dance last night.

Cleft of adverbial

(35) a. John wore his best suit to the dance last night.

b. **It was last night that** John wore his best suit to the dance.

Cleft of indirect object

(36) a. John wore his best suit to the dance last night.

b. **It was to the dance that** John wore his best suit last night.

Pseudo-cleft

(37) a. Andrew wants most to find a nice cover for his book.

b. **What** Andrew wants most **is** to find a nice cover for his book.

Reverse pseudo-cleft

(38) a. This annoys me most.

b. This **is what** annoys me most.

Existential sentences

(39) a. Something **must be** wrong.
b. **There must be** something wrong.

(40) a. Noone **was** waiting.
b. **There was** noone waiting.

(41) a. Plenty of people **are** getting promotion.
b. **There are** plenty of people getting promotion.

(42) a. Two bulldozers **have been** knocking the place flat.
b. **There have been** two bulldozers knocking the place flat.

Extraposition with inf-VP complement

(43) a. **It**'s a pity to make a fool of yourself.
b. To make a fool of yourself is a pity.

(44) a. **It** surprised me to hear him say that.
b. To hear him say that surprised me.

(45) a. **It** makes her happy to see others enjoying themselves.
b. To see others enjoying themselves makes her happy.

Extraposition with *that*-complement

(46) a. **It's on the cards** that income tax will be abolished.
b. That income tax will be abolished **is on the cards**.

Extraposition with *what*-complement

(47) a. **It doesn't matter** what you do.
b. What you do **doesn't matter**.

4.2.3 Type C: Change of Relation

Type C (Change of Relation) paraphrases involve altering a connective between clauses. Most of these paraphrase pairs have as their two components text units which are linked by different discourse relations (for example, sequence or causation). Typically, one paraphrase alternative is a complex sentence, where the constituents are linked with some coordinator, relative pronoun, or other similar connective; the other is a pair of sentences

linked just by sequence. In one direction, the paraphrases can be thought of as sentence splitting; in the other, as sentence combining.

The sentence splitting direction is the most common to apply. Most style guides recommend simpler (shorter) sentences, as more complex sentences are more difficult to comprehend, this view being backed up by psycholinguistic studies, such as that of Wright (1985). Jordan (1994) presents a collection of 'splitting' paraphrases derived from a corpus, with their aim being to make text easier to comprehend:

> An underlying motivation for this study derives from the growing interest in converting complex text into *plain language*—using forms more easily assimilable by readers who lack advanced linguistic comprehension skills. As complex noun phrases are a major cause of comprehension difficulty and can be a significant contributor to lower levels of *readability* as indicated by high index values, we need to understand how they can be written using simpler grammatical units. [Jordan, 1994: 77; emphasis in the original]

One subtype of this kind of paraphrase, going from alternative (a) to alternative (b), is exemplified in (48) to (56); another sort of example of this type is relative clause promotion, where a relative clause is promoted to a full sentence, as in (61) to (67). Davison *et al* (1980) have examined the production of children's texts from adult texts, where the aim is to simplify the adult text, through syntactic means and otherwise, so that it can be more easily comprehended; they found that the editors who adapted the texts did use paraphrases such as those of this category. They also noted, however, that the splitting process could actually make comprehension more difficult by removing explicit relations, which under the split version have to be inferred (as in, for example, (59), where the method relation indicator *by* is lost in the splitting); this effect is discussed further in Section 4.3.

The sentence combining direction is also used, however. Jordan (1993) notes that, "for prose intended for mature readers, writers must learn to use complex noun phrases in long sentences to emulate effective mature writing" (*ibid*: 39). This involves, typically, applying the paraphrases of this Type C category in the direction of alternative (b) to alternative (a).

Other paraphrases in this category are adapted from Robin's (1994) corpus-based derivation of the rewrites used in STREAK. Where Robin had information-adding paraphrases—taking a starting unit of text, and adding a constituent—the paraphrases in this Type C category present two alternatives with the same information: one where the information is added as in Robin's paraphrases, the other where it is added as a separate sentence.

The paraphrases in this category have different 'uses'—in the sense of helping the text to conform to its constraints—depending on the direction applied. When splitting sentences, readability values and lexical density values are generally improved; when combining sentences, length values are generally improved. This leads to the constraints conflicting with one another with respect to the paraphrases; this is discussed further in Section 8.1, as part of the motivation for the model put forth in this thesis.

NP splitting with PP post-modifier

(48) a. Similar interference with the gamma-ray burst detector on board the Japanese–Ginga satellite effectively blinded it during about a fifth of the same period.

b. Similar interference **occurred** with the gamma-ray burst detector on board the Japanese–Ginga satellite. **This** effectively blinded it during about a fifth of the same period.

(49) a. These cells would convert about 4 percent of the reactor generated heat into electricity. **The** considerable waste heat would be ejected through a set of radiator panels with a surface area of around 100 square meters.

b. These cells would convert about 4 percent of the reactor generated heat into electricity. Considerable waste heat **would result, and this** would be ejected through a set of radiator panels with a surface area of around 100 square meters.

(50) a. Women in long-term relationships with athletes may not be amused by Chamberlain's scorekeeping, nor are they apt to dismiss the Johnson episode as just another overhyped celebrity story—pathos aside—with hero worship, machismo and tabloid gossip.

b. **Some** women **are** in long-term relationships with athletes. They may not be amused by Chamberlain's scorekeeping, nor are they apt to dismiss the Johnson episode as just another overhyped celebrity story—pathos aside—with hero worship, machismo and tabloid gossip.

This paraphrase subtype is generally applicable where an NP has a long prepositional postmodifier. The anaphoric element introduced can be a specific one (for example, *he* or *she*), referring to the entity from which the PP is detached; or a general one (for example, *this*), which can refer to any such entity, as well as events. The NP with PP postmodifier becomes a sentence with a light verb connecting the NP and the PP.

NP splitting with PP post-modifier using existential *there*

(51) a. **The** recent unconfirmed accounts of former Soviet leader Leonid Brezhnev's penchant for astrology hardly surpass Nancy Reagan's famous astrological consultations.

b. **There have been** recent unconfirmed accounts of former Soviet leader Leonid Brezhnev's penchant for astrology. **They** hardly surpass Nancy Reagan's famous astrological consultations.

This is similar to the above, but the long NP becomes a sentence through the use of the existential *there*.

NP splitting with participial clause post-modifier

(52) a. A survey conducted by the Gallup Poll last summer indicated that one in four Americans takes cues from the stars or believes in ghosts.

b. An survey **was** conducted by the Gallup Poll last summer. **It** indicated that one in four Americans takes cues from the stars or believes in ghosts.

(53) a. **The** famous brown-haired, brown-eyed woman sitting in the early morning sunshine in the garden restaurant of Montreal's venerable Ritz-Carlton Hotel has undergone an astonishing metamorphosis.

b. **A** famous brown-haired, brown-eyed woman **is** sitting in the early morning sunshine in the garden restaurant of Montreal's venerable Ritz-Carlton Hotel. **She** has undergone an astonishing metamorphosis.

(54) a. The process would then wrap up with a summit conference in Ottawa from Feb 14 to 16 based on the general themes of Ottawa's constitutional package.

b. The process would then wrap up with a summit conference in Ottawa from Feb 14 to 16. **This would be** based on the general themes of Ottawa's constitutional package.

(55) a. Just five years ago scientists discovered a class of ceramic compounds **that** can conduct electricity without resistance at temperatures exceeding 100 kelvins.

b. Just five years ago scientists discovered a class of ceramic compounds. **These** can conduct electricity without resistance at temperatures exceeding 100 kelvins.

Here the NP is modified by a participial phrase; this can be turned into a separate sentence by the introduction of an auxiliary and an anaphor.

NP splitting with inf-VP post-modifier

(56) a. A proposal to amend the Professional Engineers Act to identify specifically sexual assault and sexual harassment as grounds for refusing to grant an engineering licence was introduced into the Ontario Legislature by MPP Norman Sterling, P. Eng., on December 6, 1990.

b. A proposal **has been made** to amend the Professional Engineers Act to identify specifically sexual assault and sexual harassment as grounds for refusing to grant an engineering licence. **It** was introduced into the Ontario Legislature by MPP Norman Sterling, P. Eng., on December 6, 1990.

Again, a light verb is introduced to form a sentence from an NP, this time one with an infinitival VP postmodifier.

NP + present participial clause post-modifier $\rightleftharpoons$ subordinate clause

(57) a. A satellite **passing** through the belt is subject to bursts of gamma rays as the positrons annihilate electrons in its outer skin.

b. **When** a satellite **passes** through the belt, **it** is subject to bursts of gamma rays as the positrons annihilate electrons in its outer skin.

An NP is modified by a phrase headed by a present participle, which can be turned into a tensed verb as part of a subordinate clause with subordinator *when*.

NP + PP post-modifier ⇌ subordinate clause

(58) a. A surgeon on his way to perform an operation was knocked off his bicycle and suffered a concussion.

b. A surgeon **was** on his way to perform an operation **when** he was knocked off his bicycle and suffered a concussion.

Split at *by*-clause

(59) a. The tree healed its wounds **by growing** new bark.

b. The tree healed its wounds. **It grew** new bark.

Here, *by* has as its complement a phrase headed by a present participle, as in (59a); this changes to a separate sentence—the discourse relation just turning into one of sequence—with the present participle becoming tensed, as in (59b).

Split at *through*-clause

(60) a. His solution is to equip Americans, through federally supported education and job-training programs, to "compete and win" in the global economy.

b. His solution is to equip Americans to "compete and win" in the global economy. **This is (to be) done** through federally supported education and job-training programs.

Subject relative clause promotion

(61) a. Robin, **who** put the sign in the window, was wearing lycra shorts.

b. Robin put the sign in the window. **He** was wearing lycra shorts.

(62) a. The pond **which** had frozen yesterday was melted by the bonfire.

b. The pond had frozen yesterday. **It** was melted by the bonfire.

(63) a. A skin test **which** tells in less than an hour whether or not a woman is going to become a mother has been announced by Dr Frederick H Falls, Dr V C Freda, and Dr H H Cohen, of the University of Illinois College of Medicine.

b. A skin test tells in less than an hour whether or not a woman is going to become a mother. **It** has been announced by Dr Frederick H Falls, Dr V C Freda, and Dr H H Cohen, of the University of Illinois College of Medicine.

(64) a. In response, the Tories had undertaken in last May's throne speech to introduce a law **that** would allow for "greater participation of Canadian men and women in constitutional change".

b. In response, the Tories had undertaken in last May's throne speech to introduce a new law. **This** would allow for "greater participation of Canadian men and women in constitutional change".

(65) a. ... to lead the New York Knicks to a 97 79 victory over the Charlotte Hornets **who** lost their sixth straight game.

b. ... to lead the New York Knicks to a 97 79 victory over the Charlotte Hornets. **The Hornets** lost their sixth straight game.

A relative clause, with the relative pronoun as subject of the relative clause, is promoted to being a separate sentence.

Object relative clause promotion

(66) a. The sign, **which** Robin put in the window, was in Hebrew.

b. Robin put the sign in the window. **It** was in Hebrew.

As above, except that the relative pronoun is the object of the relative clause.

PP-complement relative clause promotion

(67) a. The window, **which** Robin put the sign in, was made of bulletproof glass.

b. Robin put the sign in the window. **It** was made of bulletproof glass.

Split internal clause

(68) a. Indeed, United Nations guidelines prohibit the operation of reactors on board spacecraft **that** have not achieved either a stable Earth orbit or an interplanetary trajectory.

b. **Some** spacecraft have not achieved either a stable Earth orbit or an interplanetary trajectory. United Nations guidelines prohibit the operation of reactors on board **such spacecraft**.

(69) a. At the time, many of Schmidt's colleagues had good reason to question these results. Yet as modern astronomers review the evidence collected during the past 28 year, we find little room to doubt that Schmidt was right.

b. At the time, many of Schmidt's colleagues had good reason to question these results. Yet evidence **has been** collected during the past 28 years, **and** as modern astronomers review **this**, we find little room to doubt that Schmidt was right.

(70) a. Although **it** is unlikely at present, a collision between a nuclear reactor and one of the thousands of sizable objects traveling at a relative velocity of 10 kilometers per second could yield an abundance of radioactive fragments.

b. Thousands of sizable objects travel at a relative velocity of 10 kilometers per second. Although a collision between a nuclear reactor and one of **these objects** is unlikely at present, **it** could yield an abundance of radioactive fragments.

4.2.4 Type D: Deletion

Type D (Deletion) is the only category where the paraphrases are essentially unidirectional. Under this operation—which is used, for example, by editors of newspaper text, in order to shorten stories to fit space limitations—the constituents deleted are those which are in some sense peripheral. The least important constituents—in terms of candidates to delete—are those where no propositional content is lost: hedging verbs, empty nouns, relative pronouns, and so on, as in (71) to (78). Other constituents can also be deleted because of their peripheral nature, but in this case there is lost propositional content: adjectival participles, appositive NPs, parenthetical comments, and so on, as in (80) to (83). These deletion paraphrases will have an obvious effect on the text, unlike the paraphrases of other categories, where the effects are not so straightforward. This effect, and the consequent refinement of the notion of interchangeability in the definition of paraphrase, is discussed in Section 4.3.

V deletion with inf-VP complement

(71) a. Tripping over his own shoelaces **served to** start Mark contemplating his coordination skills.
b. Tripping over his own shoelaces started Mark contemplating his coordination skills.

(72) a. Nucleonics investigates the smaller particles that **go to** make up the nucleus of the atom.
b. Nucleonics investigates the smaller particles that make up the nucleus of the atom.

What is deleted here is an element sometimes referred to as a 'hedging verb', one with little semantic content.

N deletion with NP complement

(73) a. **The fact of** the war affected many people.
b. The war affected many people.

This is the noun parallel of the above.

Relative clause $\rightleftharpoons$ participial clause (whiz deletion)

(74) a. Joe wants the blazer **which was** designed by BMW.
b. Joe wants the blazer designed by BMW.

(75) a. The girl **who was** standing in the corner ...
b. The girl standing in the corner ...

(76) a. He was warned by a light **that flashed** repeatedly.
b. He was warned by a light **flashing** repeatedly.

A relative clause becomes a reduced relative clause.

Delete subject of non-finite clause

(77) a. The best thing would be **for you** to tell everybody.
b. The best thing would be to tell everybody.

Delete "to be" from appositives

(78) a. **Being** too nervous to reply, he stared at the floor.
b. Too nervous to reply, he stared at the floor.

(79) a. Seventy-three people have been drowned in the area, many of them **being** children.
b. Seventy-three people have been drowned in the area, many of them children.

Participial phrase $\rightleftharpoons$ PP

(80) a. The girl **standing** in the corner
b. The girl in the corner

In this subtype, a content word is deleted, turning a phrase headed by a participle into a PP.

Adverbial deletion

(81) a. The waterlogged conditions that ruled out play yesterday still prevailed **at Bourda** this morning.
b. The waterlogged conditions that ruled out play yesterday still prevailed this morning.

Appositive NP deletion

(82) a. Tempeste approached Blade, **a midnight dark and powerful figure**, and gave him a resounding slap.
b. Tempeste approached Blade and gave him a resounding slap.

Here, an entire appositive, being by its nature somewhat peripheral, is deleted.

Parenthetical deletion

(83) a. Some 17 million people entered the country, roughly half the total number of Europeans who migrated to the United States in the century after 1820 **(along with several hundred thousand Asians)**.
b. Some 17 million people entered the country, roughly half the total number of Europeans who migrated to the United States in the century after 1820.

Similar to the above, with deleted material, indicated by parentheses or dashes, also being to some extent peripheral.

4.2.5 Type E: Clause Movement

Type E (Clause Movement) paraphrases act less radically on the text than the other types; they merely move clauses around with respect to each other, without fundamentally altering them.

Move adverbial clauses

(84) a. The student copied the critical diagrams before returning the book.
b. Before returning the book the student copied the critical diagrams.

Move subordinate clauses

(85) a. She died, because she didn't know the rules.
b. Because she didn't know the rules, she died.

(86) a. She died, through not knowing the rules.
b. Through not knowing the rules, she died.

Reverse dependent and independent clauses

(87) a. She didn't know the rules; so she died.
b. She died; for she didn't know the rules.

4.3 Paraphrase Effects

This thesis argues that the paraphrases of Section 4.2 will cause some change to the text, and, under RP, any change effected by a paraphrase is taken to be a negative one. Developing an optimisation model which minimises this negative change thus requires a quantification of the effects that imposing a paraphrase on a text will have on that text. The rest of this section presents a framework of paraphrase change that allows changes to be evaluated and compared, and sketches methods for assigning a quantification to a paraphrase, leading to a function on the paraphrase representation, as presented in Section 7.3, and ultimately to the minimisation function for the model presented in Chapter 8.

A key assumption behind RP—that paraphrases to a text necessarily change it away from the author's original intent—is drawn from several schools of thought which hold that every choice made in constructing a text matters and has its own import, with the consequence that no unit of text can be considered to substitute for any other. Fairclough (1992) discusses the basic ideas of critical linguistics (Fowler *et al*, 1979), which has its origins in a combination of a linguistic text analysis with a social theory of the functioning of language in political and ideological processes. The text analysis component of this draws on Halliday's (1978) systemic functional linguistics, with its underlying assumption of the significance of choices; this assumption is made even stronger in critical linguistics, where these choices are linked to ideology.

> Two 'prevalent and related dualisms' in linguistic theory are rejected [by critical linguistics]: the treatment of language systems as autonomous and independent of the 'use' of language, and the separation of 'meaning' from 'style' or 'expression' (or 'content' from 'form'). Against the first dualism, critical linguistics asserts with Halliday that 'language is as it is because of its function in social structure' (Halliday 1973: 65), and argues that the language that people have access to depends upon their position in the social system. Against the second dualism, critical linguistics supports Halliday's view of the grammar of a language as systems of 'options' amongst which speakers make 'selections' according to social circumstances, assuming that formal options have contrasting meanings, and that choices of forms are always meaningful. [Fairclough, 1992: 26]

Fairclough also relates this view to other varieties of linguistic analysis with similar premises, including those developed by Foucault (1972) and Pêcheux (1982) and their related schools of deconstructionist thought, and to his own text-oriented discourse analysis, where texts are similarly treated as not being intersubstitutable.

In contrast, although retaining the idea that units of text are not perfectly intersubstitutable, in the sense of keeping the same meaning, effects, and so on, this thesis takes a less extreme position on paraphrasing and the substitutability of text. It uses the fact that people do, to some degree, treat text as being interchangeable, as evidenced by the sample from various sources used to build the set of paraphrases described in Section 4.2. However, the definition of paraphrase used in these sources and elsewhere, and the idea of when two alternative phrasings can be considered interchangeable, is usually fairly loose, and not satisfactory for the purposes of developing a model of RP. For example, Quirk *et al* (1985), in their extensive descriptive grammar of English, make use of paraphrase when referring to 'correspondences' between phrases such as in (88) and (89).

(88) He spoke these words $\simeq$ The speaker of these words

(89) A man who is timid $\simeq$ A timid man

Quirk *et al* do not define the scope of these correspondences, nor, of more relevance to this thesis, do they define the extent of interchangeability between such phrases; they appear to rely on an intuition by readers as to when two constituents mean, or refer to, the same thing, or at least an approximation of it.

The linguistic encyclopedia of Malnikjær (1991) gives two definitions of paraphrase. The first is in the context of Halliday's long-range cohesive devices, where the definition, taken from de Beaugrande and Dressler (1981), is "approximate conceptual equivalence among outwardly different material" (Malnikjær, 1991: 464). The second is drawn from Transformational Generative Grammar, where the definition of paraphrase is "that distinct sentences can, in particular respects, have identical interpretations" (*ibid*: 482–3). These are very broad definitions which do not circumscribe the notion of paraphrase to any great extent. 'Approximate conceptual equivalence' does not say anything more than the implicit reliance on intuition of Quirk *et al* (1985); nor does the term 'identical' in the interpretation of transformational generative grammarians.

An alternative definition within the transformational generative framework is that one constituent is a paraphrase of another if both are derived from the same Deep Structure—

for example, the active and passive forms of a sentence. The contentious nature of Deep Structure has been extensively discussed in linguistics literature (see, for an overview, Malnikjær, 1991: 482–497); reasons for choosing an alternative to Transformational Generative Grammar on formal grounds are discussed in Section 5.5.

The rest of this section proposes a definition of paraphrase interchangeability in the context of RP: this definition is based on the idea that interchangeability is possible, but is not necessarily only identity of meaning. The notion of paraphrase used here is thus less restrictive than the one found in critical linguistics and related areas, where no intersubstitution is possible, because no texts have the same meaning; but at the same time it is broader than that found in, for example, Transformational Generative Grammar, where intersubstitution is only possible in cases of identical meaning. The definition proposed covers all of those text mappings given as paraphrases in Section 4.2, and fulfills the requirement that the difference caused by application of paraphrases should be specified in enough detail that a quantification of the change is possible for the purpose of miniminising change between texts.

4.3.1 Meaning

In order to refer to and discuss 'meaning' and what it is, it is necessary to adopt a model of semantics. The basic model of semantics used in this thesis will be truth-conditional semantics, although information packaging concepts will also be used where appropriate.

The model of truth-conditional semantics provides only a fairly narrow of definition of meaning—within the systemic functional framework, by way of comparison, the kind of meaning truth-conditional semantics covers forms only one of the three strands of meaning, the ideational metafunction (Halliday, 1985a). Fundamental early work in truth-conditional meaning was carried out by Tarski (1935); under truth-conditional semantics, a text is viewed in terms of the propositions it contains. A useful definition of a PROPOSITION is that it is, roughly, "what a sentence on a certain occasion says about the world" (Allwood *et al*, 1977: 20). That is, the proposition expressed in the sentence *Snow is white* (an example from Tarski, 1935) is that snow is white; and this proposition has a truth value: it is either true or false. Similarly, the proposition expressed in *That man over there is my father*, given a particular referent for that man over there and a particular individual uttering the sentence, also has a truth-value.[1]

The truth of a proposition can be expressed in terms of POSSIBLE WORLDS, a concept put forward by Leibniz (1952). A possible world is the world that would exist if zero or more aspects of the actual world were different. The truth-set of a proposition is then the set of those worlds for which the proposition is true. Within this definition, there are three kinds of truth. A logically true proposition is one which, through the rules of inference in a system of logic such as propositional calculus, is true in all possible worlds. So, for example, the following (taken from Allwood *et al*, 1977: 23) is a logical truth:

(90) It's not the case that water both is and is not a chemical element.

An analytically true sentence is one that is also true in all possible worlds, but where the truth may depend on semantic relations, such as synonymy or hyponymy, between

[1] Note that this latter example raises distinctions of intensionality versus extensionality, or sense versus reference. For discussions of these issues, see, for example, Lyons (1977).

elements of the sentence expressing the proposition. For example, the following is an analytic truth:

(91) All spinsters are unmarried.

A synthetic truth is one whose truth or falsity depends on the state of the world—that is, it is true in some possible worlds and false in others. So, for example, the following sentence is a synthetic truth:

(92) Andrew Fisher was Australia's first Labor Prime Minister.

That is, there are some possible worlds (including the actual physical one) where the proposition is true, and other possible worlds where the course of history was different, where the proposition is not true. Most propositions expressed in natural language are of this synthetic type.

This framework then leads to a definition of paraphrase interchangeability for RP, as a refinement of Definition 4.1.1.

Definition 4.3.1 *Two units of text are interchangeable if, for the propositions A and B they embody, the truth-set of B is a (not necessarily proper) subset of the truth-set of A.*

This definition, using the term 'interchangeable', implies a degree of symmetry, even though subsethood is not a symmetric relation. Reasons for this will be discussed during the remainder of this section.

One reason for having this definition is that it allows for the intuition that paraphrases do not necessarily capture exactly the meaning of the original proposition. This covers cases like the correspondence of (89), described in Quirk *et al* (1985) and reproduced here.[2]

(93) A man who is timid $\simeq$ A timid man

The latter referring expression describes a man having the property of timidity at an unspecified time, while the former confines the referring expression to the present.

So, for example, in cases like the simple past, the paraphrases do not have exactly the same meaning: consider the sentences *A man who is timid knocked on the door* and *A timid man knocked on the door*. The truth-set of the former covers a subset of the possible worlds of the truth-set of the latter: the man in the latter may have been timid at the time of the knocking, but not at the time of the utterance, while this interpretation is not possible under the former; this can be verified by adjoining the clause *and, having done so, is now a brave one* to each formulation. Both, however, can have the interpretation that the man currently has the property of timidity. That is, the truth-set of the second proposition contains those possible worlds of men knocking at doors who either are or

[2]In cases where only an NP example is used, assume propositional content is discussed in relation to the corresponding sentence derived using an existential; for example, *There is a man who is timid.*

were timid, while the truth-set of the first proposition only contains those possible worlds with currently timid men.[3]

Quirk *et al* (1985) talk about a limited type of this phenomenon in terms of degrees of explicitness of postmodification, using the example of the noun phrase correspondence:

(94) the girl standing in the corner $\simeq$ the girl in the corner

In terms of possible world semantics, the truth-set of the second is a superset of that of the first: it covers cases where the girl is sitting, lying, or performing the tango, as well as standing. However, the concept of paraphrase defined in this thesis is broader than just the concept of explicitness of postmodification, covering also, for example, cases of differing lexical explicitness.[4] For example:

(95) a. The chihuahua at the counter demanded some Pal.
b. The dog at the counter demanded some dogfood.

The second formulation covers dogs other than chihuahuas and dogfood other than Pal. Similarly, the definition covers cases of differing explicitness of clausal connectives:

(96) a. The tree healed its wounds by growing new bark.
b. The tree healed its wounds. It grew new bark.

The second formulation covers other possible worlds where the relation is something other than causal: for example, *The tree healed its wounds after growing new bark*, *The tree healed its wounds despite growing new bark*, and so on.

While for the examples above Definition 4.3.1 seems to accord mostly with intuition, there are situations where it may be either more lax or more strict than expected. Such cases are discussed in Sections 4.3.2 and 4.3.3.

4.3.2 Loss of Meaning

Within Definition 4.3.1 proposed above there are varying degrees of meaning loss, where 'meaning' is here interpreted in terms of constraints narrowing the truth-set of a proposition. In some cases a default reading can be inferred; so, taking (89), the paraphrase *a timid man* does not represent a significant loss of meaning—the losses are only tense and aspect—as the paraphrase which has as its form a head noun plus postmodifying relative clause can be inferred, with the default tense being that of the clause in which it is embedded.[5] However, other paraphrases lead to a greater degree of meaning loss. For

[3] Note, however, that it is not strictly accurate that *A man who is timid* is confined to referring to the present—it is possible to make statements like *A man who is timid will not get far*, where the property of timidity can apply now or in the future, because English does not have a morphologically inflected future tense.

[4] Since the RP model of this thesis is only dealing with syntactic paraphrases, these lexical paraphrases will not feature to any great extent; the point is just that these lexical paraphrases are covered by the definition of paraphrase effect discussed here, and so that at some future stage the set of paraphrase tools could be extended to include lexical paraphrases.

[5] This gives the most likely reading when the sentence is taken without context. So, for example, (97a) would generally be paraphrased as (97b).

example, from a construction containing a head noun with postmodifying prepositional phrase, such as *the girl in the corner*, no verb can be inferred to build a construction containing a head noun with postmodifying participial clause, because verbs, being an open-class grammatical category, comprise an infinite set of possibilities.

It is possible for this paraphrase relation to be such that the subset relation required by the definition is not, in set theoretic terminology, a proper subset; but rather, the truth-sets of the two formulations are identical. This occurs, most often, with emphasis-shifting (Type B) paraphrases, as discussed below.

A common example is the unmarked/existential-*there* paraphrase. For example:

(98) a. Two panthers are striding down the street.

b. There are two panthers striding down the street.

In this case, the possible worlds that comprise the truth-sets of each formulation are the same; that is, they are, in terms of truth-conditional semantics, MEANING PRESERVING.

It should be noted, however, that not all paraphrases representing focus-shifts are meaning preserving, in the sense of maintaining truth-set equivalence. The active/passive paraphrase is one such; some instances of the paraphrases are meaning-equivalent, such as:

(99) a. The first mate snogged the ship's navigator.

b. The ship's navigator was snogged by the first mate.

Others, however, are not, such as this example from Chomsky (1957):

(100) a. Everyone in this room speaks two languages.

b. Two languages are spoken by everyone in this room.

These sentences can be read as ambiguous, but there is definitely a preferred reading for each sentence: the preferred reading of the first is that each of the people in the room speaks some subset (of size two) of the total set of the languages spoken in the room, while the preferred reading of the second is that everyone speaks the same two languages.

In addition, there are some pairs of sentences which intuition suggests could be paraphrases, but which are not covered under Definition 4.3.1. For example:

(101) a. The monkey opened the coconut by using a stick.

b. The monkey opened the coconut. She used a stick.

The truth sets of these overlap: in (101a) the sex of the monkey is not specified, and in (101b) the relation between clauses is not specified. Cases like these can still be covered under Definition 4.3.1, by seeing them as multiple paraphrases under the definition: in the case of (101), this is two paraphrases, where, for example, going from (101a) to (101b)

(97) a. A timid man knocked at the door.

b. A man who was timid knocked at the door.

However, a broader discourse context may render this default invalid.

involves dropping the relation (to become *The monkey opened the coconut. The monkey used a stick.*) and then expanding the anaphor. However, no such paraphrases are used in this thesis.

For the RP model of which these paraphrases are a part, one of the basic assumptions is that the text to be paraphrased reflects precisely the intentions of the writer, and so any paraphrase will be non-optimal. The paraphrasing is only carried out because external constraints, such as length limits or readability requirements, are imposed on the text; the paraphrasing is done reluctantly. The following subsections then discuss the effect on the text—necessarily negative, given these assumptions of RP—for the categories of paraphrases given in Section 4.2; the focus of the discussion of effects will be in terms of the truth-conditional framework described above, although other potential types of effect will also be mentioned.

Type A: Change of Perspective

These paraphrases alter the perspective of one part of the text being paraphrased, an alteration which typically involves a change in the part of speech of one or more elements of the text. Example (2) is reproduced here.

(102) a. Steven made an attempt to stop playing Hearts.
b. Steven attempted to stop playing Hearts.

This has two main effects on the text. Firstly, the paraphrase is generally not meaning preserving. In the example above, the second formulation could also paraphrase *Steven made two attempts to stop playing Hearts* or *Steven made many attempts to stop playing Hearts*. Admittedly, it is possible to qualify (102b) with the adverbial *once* in order to preserve the truth-conditional meaning; however, even if the ambiguity of this alternative is resolved—that is, ensuring by context or some other means that *Steven attempted once to stop playing Hearts* is not interpreted as being equivalent to *Steven attempted once upon a time to stop playing Hearts*—it sounds marked in comparison with (102b); and, given that (102a) is unmarked, the preferred paraphrase would be the similarly syntactically unmarked (102b).

Often, the loss is only of closed-class words, and as such is not especially significant. Sometimes, despite the fact that only words with little content are involved in the paraphrasing, a more significant alteration in truth-conditional meaning occurs. For example, (103a) can paraphrase either (103b) or (103c):

(103) a. His drawing of the picture surprised me.
b. The fact that he drew the picture surprised me.
c. The way that he drew the picture surprised me.

A second effect of the paraphrase is more an effect on the reader. Many researchers in psycholinguistics (e.g. Coleman, 1962; Duffy and Kabance, 1982; Wright, 1985) have investigated the effects of different paraphrase alternatives on readers' comprehension. Wright

notes that paraphrasing a sentence containing a nominalisation, so that the rewritten version uses the equivalent verb, improves comprehension time—that is, the meaning is more easily accessed. Effects of other changed perspectives, such as prepositional phrase to adverb, have not been similarly investigated.

Type B: Change of Emphasis

This covers such paraphrases as *it*-clefts, pseudo-clefts, existential-*there* transformations, and active/passive paraphrases; see (104) to (107), respectively.

(104) a. John wore his best suit to the dance last night.
b. It was John who wore his best suit to the dance last night.

(105) a. Andrew wants most to find a nice cover for his book.
b. What Andrew wants most is to find a nice cover for his book.

(106) a. Something must be wrong.
b. There must be something wrong.

(107) a. The government forces soon pursued the retreating guerillas.
b. The retreating guerillas were soon pursued by the government forces.

As noted earlier, the focus-shift paraphrases generally have identical truth-sets—that is, they preserve truth-conditional meaning—although, in a few instances, this is not the case. However, changing the emphasis of a text, using paraphrases based only on the syntax of individual sentences, affects the fluency of a text, sometimes to the point of unreadability; within the systemic functional framework, this actually constitutes a change in meaning, but within the textual metafunction rather than the ideational (Halliday, 1985a). This effect on fluency is particularly notable in dialogue-like situations; the paraphrase in (108) is acceptable, but the one in (109) is not.

(108) a. Who ate the water buffalo?
b. John ate the water buffalo.

(109) a. Who ate the water buffalo?
b. ?It was the water buffalo which John ate.

This is due to the way written English uses syntax to mark information that is new (or focus), as opposed to information that is a link to knowledge the hearer already has (or ground). In this example, the cleft complement (*which John ate*) marks the link, while the head of the initial NP construction (*the water buffalo*) is flagged as new.[6] Thus, introducing clefts can cause problems for a text in terms of non-propositional meaning.

Paraphrasing in the other direction is not as problematic. For example, in a written text either of two alternatives (110) or (111) is acceptable:

[6]For a more detailed overview of how the concepts of new and given relate to syntactic and intonational structures, as well as to other concepts such as topic and ground, see Vallduví (1993); for further discussion on the implications of clefting paraphrases, see Delin (1989).

(110) a. Who was at the door?

b. It was the plumber who was at the door.

(111) a. Who was at the door?

b. The plumber was at the door.

This is because it is possible to have an intonational structure—see Steedman (1991) or Vallduví (1993) for discussion of these structures—such that the plumber is indicated as being new, which will fit with the syntactically unmarked formulation.

However, Delin and Oberlander (1995) look at the effects of carrying out paraphrases that change the emphasis of a text by replacing a marked construction with an unmarked one, specifically in the case of *it*-clefts; they conclude that there are non-propositional effects here also. Removing cleft constructions sometimes causes loss of a contrastive reading (for a variety of *it*-cleft they term 'stressed-focus'), loss of the implication of shared knowledge or the 'known fact' effect (Prince, 1978), and loss of the idea that the presupposed information is intended as background in the discourse structure. They attribute these effects to the loss of the copula verb which disappears in the non-cleft version. This copula is a stative verb, which gives a 'factual', durative reading which doesn't appear to be a 'narrative step'. Therefore, the cleft actually seems to affect the aspectual reading of the information, changing it from an event to a state. All of these effects are, again, non-propositional.

Since this thesis only looks at syntactic paraphrases at the level of individual sentences or pairs of sentences, there is no way of evaluating such pragmatic factors. Consequently, the paraphrases in this category will be tagged as mildly undesirable—without knowing the discourse context there is some risk that the paraphrase will make the text disfluent. A context-based paraphrase, able to reflect more accurately and completely the effects on the text, is beyond the scope of this work.

Type C: Change of Relation

These paraphrases involve splitting a clause to produce two smaller clauses; or, in the other direction, combining these smaller clauses to produce a more complex one. Typically, splitting is done by breaking down a complex noun phrase, as these are fairly strongly correlated with sentence length (Sampson and Haigh, 1988), and hence offer the most scope for splitting. Example (48) is reproduced here.

(112) a. Similar interference with the gamma-ray burst detector on board the Japanese–Ginga satellite effectively blinded it during about a fifth of the same period.

b. Similar interference occurred with the gamma-ray burst detector on board the Japanese–Ginga satellite. This effectively blinded it during about a fifth of the same period.

In this type of example, the truth-conditional meaning is not substantially affected: even though the difference between the formulations involves an open-class word (the verb *occurs*), it is a word with little content, serving only as a hook for tense and aspect;

see the discussion of Type A paraphrases in Section 4.2 (also Halliday, 1985b; Dras and Johnson, 1996).

However, changing only the syntax leads to problems with the ideas of given and new (or ground and focus), as noted by Jordan (1994). An example he discusses is:

(113) a. [On October 10, Darryl Bean ... wrote the same letter to three women ...] The three women who received Bean's odious letter are determined to stand up for their rights and freedoms.

b. [On October 10, Darryl Bean ... wrote the same letter to three women ...] Three women received Bean's odious letter. They are determined to stand up for their rights and freedoms.

Here, he notes that using a syntactic sentence-splitting paraphrase leads to redundancy because of earlier information which is repeated in the post-modifier as a link. Similarly, he notes that some paraphrases produce statements which are true but obvious from the reader's knowledge of the world; for example:

(114) a. A survey conducted by the Gallup Poll last summer indicated that one in four Americans takes cues from the stars or believes in ghosts.

b. A survey was conducted by the Gallup Poll last summer. It indicated that one in four Americans takes cues from the stars or believes in ghosts.

Here, the first sentence of (114b) is redundant, as it is generally known that the primary purpose of the Gallup organisation is to conduct surveys, and that it does so during the summer. Others of this type are also not entirely straightforward, such as example (59), reproduced here:

(115) a. The tree healed its wounds by growing new bark.

b. The tree healed its wounds. It grew new bark.

Often the removed elements are inferable; however, the ability to carry out this process of inference is also dependent on knowledge of the world. This sort of process, like the one that changes a text from example (115a) to example (115b) above, is often used to simplify a text for children or other less mature readers because it produces a sentence with less complex syntax; however, as Davison *et al* (1980) note, these are often the readers who are less able to make the necessary inferences. So in (115b), where the relation is implicit, and given no other knowledge, the text could mean that the tree healed its wounds at the same time as growing new bark, or despite growing its bark, or many other alternatives. The fact that these inferences can only be made through knowledge of the world—rather than the simpler, syntax-based inferences required when paraphrasing from, say, an attributive to a predicative adjective—means that the loss of meaning for this type is more significant; however, it is difficult to estimate the extent of this effect.

Apart from the relatively minor complications of examples such as these, however, this type of paraphrase is substantially meaning preserving.

Type D: Deletion

These are paraphrases where constituents are deleted; there are various shades of effect caused by this type. Some appear to have no effect on the text at all, and either side of the paraphrase is inferable from the other; for example:

(116) a. Too nervous to reply, he stared at the floor.

b. Being too nervous to reply, he stared at the floor.

Some have no effect on the truth-conditional meaning of the proposition, but can make the text more or less difficult to read. Garden-path sentences are one instance: in a garden-path sentence, the initial reading, because of expectations held by the reader, leads to a false parse. They can occur because of, for example, *that*-deletion, where the relative pronoun *that* is deleted:

(117) a. The cotton clothing is made of grows in Mississippi.

b. The cotton that clothing is made of grows in Mississippi.

Garden path effects can also occur because of whiz-deletion, where the relative *wh*-pronoun and verbal auxiliary are deleted from a relative clause to give a reduced relative clause:

(118) a. The horse raced past the barn fell.

b. The horse which was raced past the barn fell.

The difficulty involved in processing these garden-path sentences has been investigated by Gibson (1995). In addition, for whiz-deletion, even in cases where a garden-path sentence is not produced, there is a contention that the resulting compressed form of the text is still more difficult to read (see Huckin *et al*, 1986, for a discussion of the issue). Despite the additional processing required, there has not been any change in propositional meaning: the deleted auxiliary, used only as a hook for tense and aspect, can be inferred from the remainder of the sentence.

Yet another kind of deletion involves a more significant change: this time the change is to truth-conditional meaning, and there is a loss of information conveyed by the paraphrase alternative. This occurs, for example, in the deletion of appositives:

(119) a. Richard, a cowardly king, fled from the scene of the poisoning.

b. Richard fled from the scene of the poisoning.

The loss here is more significant than the previous examples of deletion because it is not just that the information is harder to access, it is that it is no longer there.

These subtypes of the deletion paraphrases thus decrease in desirability, from the point of view of side-effects on the text, ranging from acceptable to potentially completely unacceptable.

Type E: Clause Movement

The fifth category covers those paraphrases where a mobile element, such as an adverbial, is moved around the sentence. For example:

(120) a. The student copied the critical diagrams before returning the book.
b. Before returning the book the student copied the critical diagrams.

This has no effect on the truth-conditional meaning; however, some psycholinguistic studies do note that the positioning of a mobile element is a factor in determining sentence comprehension times (Wright, 1985).

4.3.3 Categorisation of Paraphrase Effects

The aim of this section is to work towards a 'damage' function, a broad quantification of the effects paraphrases have on a text. The most important and quantifiable effects are those which affect the truth-conditional meaning of the text, so these will be the focus of the quantification. As described above, there are other effects on the text, but these are secondary to the truth-conditional effects; for example, effects on factors such as the speed of comprehension, how quickly the information content can be assimilated, are secondary to changes in the information content itself. Since the aim is only to have a rough quantification, the primary effects are sufficient as a start, although the model could be refined later.

The starting point for the desired quantification is a categorisation of the truth-conditional meaning effects, ranking them in order of importance. The definition of paraphrase interchangeability proposed in this thesis, Definition 4.3.1, depends on the notion of a subset relation between the paraphrase elements. This is only defined in terms of a Boolean function on the truth-sets of the propositions embodied by the paraphrase elements—whether one is a subset of the other (True) or not (False)—but it is possible to consider the size difference in the subset relation if this relation holds. There are two trivial cases: where the sets are equal, and where one set is the empty set. In the first of these cases, there is zero difference; in the second, the difference is maximal. However, it is possible to conceive of gradations on a scale ranging from zero to maximal, where the relative 'size difference' of the truth-sets ranges from zero to maximal. The validity of talking about size differences in truth-sets, where these are infinite, is discussed below; but first, some examples, already discussed earlier, are re-presented.

(121) a. Onlookers scrambled to avoid the car which was flashing its headlights.
b. Onlookers scrambled to avoid the car flashing its headlights.

(122) a. The salesman made an attempt to wear Steven down.
b. The salesman attempted to wear Steven down.

(123) a. There was a girl standing in the corner.
b. There was a girl in the corner.

(124) a. Tempeste approached Blade, a midnight dark and powerful figure, and gave him a resounding slap.

b. Tempeste approached Blade and gave him a resounding slap.

These examples give a range of different magnitudes in the relative size of the sets representing the possible worlds in which each of the paraphrase alternatives is true. Paraphrase pair (121) represents a fairly minimal difference: (121b) can be a paraphrase either of (121a) or of *Onlookers scrambled to avoid the car which is flashing its headlights.* The set of possible worlds in which (121b) is true is a proper superset of set of the possible worlds in which (121a) is true; but intuition suggests the sets are relatively close in size, (121b) only covering two different cases (present versus present and past tense) with respect to the altered constituents. Example (122) represents a slightly bigger paraphrase: (122b) can paraphrase statements asserting one attempt—equivalent to (122a)—two attempts, seven attempts, or many attempts. The size of the set difference here is consequently relatively larger than in (121). In (123), the difference is larger still, in that (123b) can describe situations where the girl is sitting, lying, dancing, and so on. The largest difference is in (124), where (124b) includes in its set of possible worlds, over and above the possible worlds in which (124a) is true, worlds in which Blade has other characteristics.

As stated, it may seem problematic to discuss the differences in size of sets which are infinite. It is more straightforward to discuss such issues in a mathematical context; so, consider the set of natural numbers $\mathbf{N}$, $\{0, 1, 2, 3, \dots\}$, which is of size $\aleph_0$, that is, countably infinite. Consider also the set $S_1 = \{-2, -1, 0, 1, 2, 3, \dots\}$. This is also of size $\aleph_0$, that is, also countably infinite, by virtue of a pairing of the elements of the sets starting from the first element in each. However, consider further the set $S_2 = \{0, 2, 4, 6, \dots\}$. This too is of size $\aleph_0$, but there is an intuition that the difference in size between $\mathbf{N}$ and S_2 is larger than the difference in size between $\mathbf{N}$ and S_1. What this intuition corresponds to is the set difference: $\mathbf{N} - S_1$ is the set $\{-2, -1\}$, of size 2, while $\mathbf{N} - S_2$ is the set of positive odd numbers $\{1, 3, 5, \dots\}$ of size $\aleph_0$. This has parallels to the language examples above. For example, paraphrase pair (121) has minimal differences: the changed relative pronoun has no effect on truth-conditional meaning, and the changed auxiliary verb *be* can only be one of a small and finite number of alternatives. In (122), the set is also finite, but larger, with more alternatives for the determiner. In comparison, the deletion of the open-class constituent in (123), the present participle *standing*, leads to a much greater set difference; and deleting multiple open-class words in (124) has a still larger effect. This line of reasoning suggests that a way of approximating the intuition about the difference in the relative sizes of possible world sets is by using parts of speech. An alteration in less significant parts of speech corresponds to a small size in set differences, and so on.

Note that Definition 4.3.1, being based on truth sets and subsethood, would allow paraphrase pairs with little or no meaning preservation. For example, the definition allows for a logically false statement to be a paraphrase for any other statement, as the truth-set for a logically false proposition—the empty set—is a subset of any other truth-set. For example, (125a) can be paraphrased by (125b).

(125) a. The snake moulted all over Kim's parquetry floor.

b. It both is and is not true that Gerry has surgically enhanced buttocks.

There are no paraphrases proposed in this thesis which allow this, but Definition 4.3.1 does not rule such paraphrases out, and so this would be a potentially valid paraphrase.

This may seem to make the definition too lax, but there are two main reasons for feeling that the definition is nonetheless appropriate. The first reason is that exactly where the boundary should be, between what counts as paraphrase and what does not, is blurry; analogous to this is perhaps the way that the distinction between grammatical and ungrammatical is not perfectly clear-cut. Definition 4.3.1 rather defines a continuum of diminishing acceptability, according to the size of the set difference. This leads to the second reason. A damage function weighting can be assigned within the RP model of this thesis such that this sort of paraphrase will never be chosen during the optimisation process of Chapter 8.[7]

A problem now arises related to the roughness of the analysis of effect: the problem is that in practice there will be errors in the paraphrase process, so the analysis of effect may not be accurate; that is, given a sentence, an error might be introduced in determining its paraphrase. This is a consequence of using only syntactic paraphrases, and only looking at sentences in pairs, and not at any context beyond that. One major area of potential error is anaphora. For example, when sentence splitting occurs, an incorrect pronoun could be generated if a simplistic algorithm were used (taking the last mentioned entity as the anaphor), as in paraphrasing from (126a) to (126b).

(126) a. Andrew, easily beating Meredith at squash, consequently took a long shower.

b. *Andrew easily beat Meredith at squash. She consequently took a long shower.

This is usually avoided by using the general pronoun *this* when possible, representing both entities and events, as in (48), reproduced here.

(127) a. Similar interference with the gamma-ray burst detector on board the Japanese–Ginga satellite effectively blinded it during about a fifth of the same period.

b. Similar interference occurred with the gamma-ray burst detector on board the Japanese–Ginga satellite. **This** effectively blinded it during about a fifth of the same period.

Similarly, in sentence combining, an antecedent could be misidentified or ambiguous (although, in general, it is assumed in this thesis that some oracular component provides this information). Another source of error is that, in looking only at prototypical examples of a particular syntactic paraphrase, the assessment of the paraphrase mapping's effect might not be valid for all paraphrases of that type. An example is the active–passive paraphrase discussed earlier; mostly, there will be no truth-conditional meaning effect, but in the Chomsky example of people in a room speaking various languages, there is.

However, these sources of error do not, in the end, pose too much of a problem. The analysis of effect is only used to give a damage function to rank paraphrases in order of desirability and thus to guide the optimisation process described in Chapter 8 to finding

[7] This is a standard technique in Integer Programming, used in during the optimisation process of Chapter 8. Variables that cannot be part of a solution are not disallowed, but are just assigned very large weights; artificial variables are such a case.

Category 1	This is the lowest category of effect; it is used for potential sources of error that in normal circumstances do not produce a change in truth-conditional meaning. Changes to anaphors (either adding or deleting) fall into this category.
Category 2	This is the second lowest category of effect. It covers the smallest changes of meaning: those that occur when words carrying tense or aspect are lost (or introduced, with possible error). Changes to information structure, with its source of error which may include propositional change, also fall into this category.
Category 3	This is the third lowest category of effect. It covers larger changes to truth-conditional meaning and relational structure. Changes to determiners fall into this category, as do changes to discourse relations.
Category 4	This is the most significant category of effect, and 'overshadows' other categories of effect so that they do not count. It covers the largest changes to truth-conditional meaning, when open-class words are altered, although note that this does not include content-free open-class words, such as light verbs; where relevant, the effects of these are covered by the other categories.

Figure 4.1: Categories of paraphrase effect

the minimally changed text. Variations in this damage function are not problematic unless they cause large variations in the resulting text; and Section 8.4 demonstrates that this is not the case for the mathematical model of RP as presented in this thesis. That is, even if there is error in the paraphrasing process, the model is stable enough that the errors do not have a significant effect for the situations investigated.

This does mean, though, that the potential sources of error should be included in the categorisation of effects; it allows an evaluation of the effect of more (or less) error than expected. This source of error, along with the ordering of truth-conditional effects above, gives a skeleton for a damage function, measuring the amount of change to the text. The categorisation of effects as the basis of such a damage function shown in Table 4.1 is proposed. Worth commenting on here is the notion of 'overshadowing' introduced in Category 4. This is analogous to an infinite set difference (changes in open class words) overshadowing a small and finite set difference (changes in determiners): combining the infinite set difference with the finite one gives an infinite set, so it is as if the finite set difference had not been there in the first place.

Given this categorisation, it is possible (and indeed likely) for a paraphrase to involve more than one category of effect. Take as an example (128).

(128) a. The last pre-war census, in 1910, counted about 13.5 million foreign-born people in the American population, in contrast to about 22.5 million in

1994.

b. The last pre-war census, in 1910, counted about 13.5 million foreign-born people in the American population. This was in contrast to about 22.5 million in 1994.

Here there is one Category 1 effect (introduction of anaphoric *this*), and one Category 2 effect (introduction of auxiliary tense—past—with verbal auxiliary *was*). Another example is (124), reproduced here.

(129) a. Tempeste approached Blade, a midnight dark and powerful figure, and gave him a resounding slap.

b. Tempeste approached Blade and gave him a resounding slap.

Here there are four Category 4 effects (deletion of four open-class words *midnight*, *dark*, *powerful*, *figure*), with the Category 3 effect (deletion of determiner *a*) not counting, as it is overshadowed by the Category 4 effects.

The effect of a paraphrase is thus the aggregate of all of its effects according to the categorisation of Table 4.1. Again, this analysis of total effect is only a rough one; the categorisation is further developed in Chapters 7 and 8, where it is integrated into the optimisation model of the thesis. There, guided by this categorisation of effect, the optimisation process chooses that combination of paraphrases that minimally changes the text.

4.4 Summary

In the literature, there is a variety of definitions for the notion of paraphrase; however, none of these are very precise, which is problematic when it comes to evaluating the effects of paraphrases on a text, necessary given that the aim under RP is to minimise this change. As noted earlier, the paraphrases in this thesis are only tools for making the original text conform to its constraints; they are not compulsory actions for correcting flaws in the text. Given this conception of paraphrase, a sample of such general-purpose paraphrase tools, taken from a variety of sources which deem the paraphrase elements to be interchangeable, has been described by example in Section 4.2, and divided for convenience into five types—Change of Perspective, Change of Emphasis, Change of Relation, Deletion, and Clause Movement. From this, a definition of paraphrase follows which is a consequence of the assumption of RP (held also by critical linguists, systemic functional linguists and others) that any paraphrase of the text is a change away from the original choices made by the text's author, supported by reference to specific examples. This definition provides a way of talking about the magnitude of a paraphrase's effect, and leads the way to a 'damage' function for RP.

4.5 A Sketch of RP So Far

At this stage, it is useful to take stock of the state of the modelling of the RP task. Section 2.3 described the major characteristics of RP:

- text is mapped to text, rather than from some knowledge representation to text where equivalence of outputs is known, in some 'paraphrase mapping';
- consequently, a notion of 'damage' to the text, caused by these paraphrase mappings, must be considered;
- changes are caused by surface constraints applied to the whole of a text, rather than by the need to correct individual constituents of the text;
- constraints can conflict with each other, leading to a large search space and the need to navigate it efficiently.

Four constraints have been set out in Chapter 3; a taxonomy of paraphrases, with a large number of examples, has been given in Chapter 4; and a damage function, following from the definition of paraphrase of Chapter 4, has been outlined in Section 4.3. The aim of the remainder of the thesis is to give precise characterisations of each of these. In terms of a precise formal definition for paraphrases, Chapter 5 gives background and definitional material on the grammar formalism used for doing so in this thesis, Synchronous Tree Adjoining Grammar (S-TAG); and Chapters 6 and 7 express paraphrase within this formalism. In the process of expressing paraphrase within S-TAG, it becomes clear that a redefinition of the formalism is necessary, and that this repairs aspects of the formalism that prove problematic for applications such as machine translation or the modelling of coordination.

Chapter 8 gathers together all of the earlier work, and shows how a mathematical model can be built to carry out the RP task, one which will take the components of RP along with a source text, and produce a resulting text that conforms to the imposed constraints.

Part II

The Grammar Formalism

Chapter 5

Tree Adjoining Grammars

This chapter gives an overview of Tree Adjoining Grammar (TAG), a mathematical formalism which we will use to describe in precise terms the paraphrases exemplified in Chapter 4. The chapter also argues that TAG has properties which make it a potentially suitable formalism for the description of paraphrases, albeit not the only one which would be appropriate for doing so.

Sometimes, it can be unclear as to why a precise formalism, and a mathematically-based one at that, is necessary at all. Systemic Functional Grammar (SFG), for example, avoids the sort of mathematical formalism found in generative grammar work, with some SFG proponents believing that such formalisms are too restrictive and too narrowly focussed on syntax. However, having a mathematical formalism can be helpful; Rambow (1994) gives an overview that is broadly in accord with the ideas in this thesis, and he notes three positive features of mathematical formalisms in their application to linguistics. Firstly, they allow representation in a precise way, and consequently allow conclusions to be made with more rigour. Secondly, they are useful as a means of expression, or a common framework for discussing linguistic issues. Thirdly, particular mathematical formalisms have different levels of expressive power and restrictiveness. One argument is that in describing language, it is desirable to have a formalism that is expressive enough to cover all features of natural languages, but which is not so expressive that it is possible to describe any permutation of words at all. The closer the formalism is to achieving the balance by virtue of the mathematics on which it is based, the less work necessary to further constrain it to accurately describe some feature of language. In general, it is also true that the more constrained a grammar formalism is, the more computationally tractable it is, in terms of issues such as parsability: witness the use of context free grammars (CFGs), rather than any more powerful formalisms, in the design and compilation of programming languages.

On the other hand, in less constrained grammar formalisms it may be easier to express particular structures, or to capture generalisations: for example, TAG tree families express a generalisation that, in less constrained formalisms, can be done more simply. However, for these less constrained formalisms there is a corresponding increase in the amount of effort necessary to disallow invalid descriptions. The choice in this case is thus to some extent a personal preference. This thesis, in using TAG, tends toward the more constrained end of the spectrum.

Rambow (1994) further discusses some of the mathematical formalisms used for linguistic

description. Three are unrestricted in formal power: first order predicate logic (FOPL), which has been used by Stabler (1992) as a representation of Chomsky's Government and Binding Theory (Chomsky, 1981); Head-driven Phrase Structure Grammar (HPSG), based on feature structures and unification (Pollard and Sag, 1987; Pollard and Sag, 1994); and Lexical Functional Grammar (LFG), combining phrase structure with functional characteristics of language (Kaplan and Bresnan, 1981).

Other theories use restricted formalisms. One is Generalized Phrase Structure Grammar (GPSG) (Gazdar *et al*, 1985), based on CFGs. However, CFGs are not expressive enough to describe natural languages[1], and cannot without modification neatly describe linguistic features such as subcategorisation and agreement. The belief that CFGs were inadequate in this way had earlier led to the development of Transformational Grammar, (usually) CFGs augmented by transformations; but the formalism then becomes unrestricted, as shown by Peters and Ritchie (1973), who demonstrate that transformational grammars generate all recursively enumerable sets, that is, they are equivalent to Chomsky's Type 0 grammars. Another alternative is the class of Mildly Context Sensitive Grammars (MCSGs) containing, among others, Tree Adjoining Grammar (TAG). These are more powerful than CFGs, in that as well as being able to generate all Context Free Languages (CFLs) they can generate some non-CFLs as well; but they are only slightly more powerful, in that the languages they generate have several properties such as semilinearity (Vijay-Shanker, 1987), which much more powerful formalisms, able to express a wider range of languages, do not. Both of these characteristics are applicable to the description of natural language, with MCSGs able to model aspects of natural language that were used to show the inadequacy of CFGs; and with semilinearity as a characteristic appropriate to natural language, as discussed in Joshi (1985) and Berwick and Weinberg (1984).

There are other MCSGs as well, such as Combinatory Categorial Grammar (CCG) (Steedman, 1985; Steedman, 1991). However, in terms of expressive power they are equivalent: it is shown in Vijay-Shanker and Weir (1994) that four MCSGs—TAGs, head grammars, linear indexed grammars and combinatory categorial grammars—are all equivalent. However, their different representation schemes mean that one type of grammar may be more appropriate than another for representing particular characteristics of language. TAGs in particular have other features, notably the fact that they are tree-based, that there is the extension to Synchronous Tree Adjoining Grammar (S-TAG), and that there is also the XTAG standard grammar of English, which make them a promising starting point for describing paraphrases. Additionally, there are results for TAG which guarantee polynomial parsability: there are a number of methods for parsing which are $\mathcal{O}(n^6)$ where n is the length of the input string (with original complexity result in Vijay-Shanker, 1987), as well as the possibility of asymptotically faster methods (Rajasekaran and Yooseph, 1995).[2]

The paraphrase representation of this thesis is based on Synchronous Tree Adjoining Grammars (Shieber and Schabes, 1990), noting that there is a parallel between applications of S-TAG as described in the literature—syntax–semantics mapping and machine translation, for example—and paraphrasing. The latter of the two, machine translation, is particularly applicable, as the way it has been used (e.g. Abeillé, 1990) is as a direct mapping from one syntactic structure to another, similar to the definition of paraphrase in this thesis. However, under the current definition of S-TAG it is not possible to charac-

[1] Shieber (1985); this is also discussed in Section 5.2.

[2] But see also Satta (1994), which argues that asymptotically faster methods will have large hidden constants.

terise every mapping necessary for a transduction between two sets of syntactic structures, as in machine translation, with this likely to be the case for paraphrase. The aim of this chapter is to set out these problems, and along the way to give the necessary background both for the discussion of the problems with S-TAG in Section 5.4 and for the paraphrase application and the solutions to the S-TAG problems in Chapters 6 and 7.

5.1 TAG Definitions

TAG is a tree rewriting system, in the way that Context Free Grammar (CFG) is a string rewriting system. There have been several incarnations of TAG, starting with the original in Joshi (1975). The style of the presentation here mirrors that of Vijay-Shanker and Weir (1994), with the aim being to give a formal and precise characterisation of the objects that will be discussed in the thesis.

First, we will give a fairly standard definition of a tree. Let $\mathbf{N}_+$ be the set of all positive integers, and $\mathbf{N}_+^*$ the free monoid on $\mathbf{N}_+$. The binary operation of concatenation is denoted by $\cdot$ and the identity by ϵ. For $p, q \in \mathbf{N}_+^*$, $p \leq q$ if and only if there exists an $r \in \mathbf{N}_+^*$ such that $q = p \cdot r$ (that is, p is a PREFIX of q); $p < q$ if and only if $p \leq q$ and $p \neq q$. D is a tree domain if it is a nonempty finite subset of $\mathbf{N}_+^*$ such that if $d \in D$ and $d = d_1 \cdot d_2$ then $d_1 \in D$; and if $d \cdot i \in D$ where $i \in \mathbf{N}_+$ then $d \cdot j \in D$ for all $1 \leq j \leq i$.

A TREE OVER Σ (or, when alphabets are unimportant, simply a tree) γ is denoted by a partial function $\gamma : \mathbf{N}_+^* \rightarrow \Sigma$ where $\mathrm{dom}(\gamma)$ is a tree domain and Σ is the set of node labels (and $\mathrm{dom}(\gamma)$ is the, usual, tree domain γ). We say that the elements of a tree domain are addresses of the nodes of the tree and $\gamma(d)$ is the label of the node with address d. For a node η with address $d = j_1 \cdot j_2 \cdot \ldots \cdot j_{k-1} \cdot j_k$, $\mathrm{parent}(\eta)$ is the node at address $j_1 \cdot j_2 \cdot \ldots \cdot j_{k-1}$. For any node η_i, if $\eta = \mathrm{parent}(\eta_i)$, the η_i is called a child of η. Note that ϵ is the address of the root node of the tree. As an example, the addresses of trees in Figure 5.1 are given in Table 5.2.

With this, we will give some definitions related to trees. If $d \in \mathrm{dom}(\gamma)$ for some tree γ then γ/d, the subtree rooted at d in γ, is defined such that for all $d' \in \mathbf{N}_+^*$, $\gamma/d(d') = \gamma(d \cdot d')$. The leaf nodes of a tree domain D are given by $\mathrm{leaf}(D) \subseteq D$ such that for all $d \in D$, $d \in \mathrm{leaf}(D)$ if and only if for all $d_1 \in D$, $d \not< d_1$. The leaf nodes are sometimes referred to as the frontier of the tree. The internal or interior nodes of the tree domain are given by $\mathrm{internal}(D) = D - \mathrm{leaf}(D)$.

Tree adjoining grammars manipulate trees whose nodes are labelled either by terminal symbols or by triples of the form $\langle A, \mathbf{sa}, \mathbf{oa} \rangle$ where A is a non-terminal symbol, **sa** is a set of tree labels (that determines which of the trees of the grammar can be adjoined at that node) and **oa** is either **true** (indicating that adjunction is obligatory) or **false** (indicating that adjunction is optional). We call **sa** and **oa** the adjunction constraints at that node. A node at which the value of **oa** is **true** is said to have an OA constraint. A node at which the value of $\mathbf{sa} = \emptyset$ is said to have an NA constraint. Substitutions can occur at some leaf nodes.

Let Σ be divided into two sets, V_N a set of non-terminal symbols, and V_T a set of terminal symbols, with $V_T \cap V_N = \emptyset$; V_L is a set of tree labels and $V_T^\epsilon = V_T \cup \{\epsilon\}$.

An INITIAL TREE is defined as follows. For each $A \in V_N$, $init(V_N, V_T, V_L, A)$ is the set of trees $\alpha : D_\alpha \rightarrow V_T^\epsilon \cup (V_N \times 2^{V_L} \times \{\mathbf{true}, \mathbf{false}\})$ where D_α is a tree domain and the

following hold.

- The root of α is labelled $\langle A, \mathbf{sa}, \mathbf{oa}\rangle$ for some $\mathbf{sa} \subseteq V_L$ and $\mathbf{oa} \in \{\mathbf{true}, \mathbf{false}\}$.
- All internal nodes of α are labelled $\langle B, \mathbf{sa}, \mathbf{oa}\rangle$ for some $B \in V_N$, $\mathbf{sa} \subseteq V_L$ and $\mathbf{oa} \in \{\mathbf{true}, \mathbf{false}\}$.
- All leaf nodes of α are labelled either $\langle B, \emptyset, \mathbf{false}\rangle$ for some $B \in V_N$, or by some $u \in V_T^\epsilon$.

Leaf nodes labelled with triples containing non-terminals are said to be 'marked for substitution': the operation of substitution, defined below, occurs at these nodes, while adjunctions occur at the internal nodes of a tree.

An AUXILIARY TREE is defined as follows. For each $A \in V_N$, $\mathrm{aux}(V_N, V_T, V_L, A)$ is the set of trees $\beta : D_\beta \to V_T^\epsilon \cup (V_N \times 2^{V_L} \times \{\mathbf{true}, \mathbf{false}\})$ where D_β is a tree domain and the following hold.

- The root of β is labelled $\langle A, \mathbf{sa}, \mathbf{oa}\rangle$ for some $\mathbf{sa} \subseteq V_L$ and $\mathbf{oa} \in \{\mathbf{true}, \mathbf{false}\}$.
- All internal nodes of β are labelled $\langle B, \mathbf{sa}, \mathbf{oa}\rangle$ for some $B \in V_N$, $\mathbf{sa} \subseteq V_L$ and $\mathbf{oa} \in \{\mathbf{true}, \mathbf{false}\}$.
- One leaf node of β is called the foot node and is labelled $\langle A, \mathbf{sa}, \mathbf{oa}\rangle$ for some $\mathbf{sa} \subseteq V_L$ and $\mathbf{oa} \in \{\mathbf{true}, \mathbf{false}\}$. The address of the foot node of β is denoted $ft(\beta)$.
- The remaining leaf nodes of β are labelled either $\langle B, \emptyset, \mathbf{false}\rangle$ for some $B \in V_N$, or by some $u \in V_T^\epsilon$.

Roughly speaking, auxiliary trees are those trees that can be adjoined into others.

Let

$$\mathrm{aux}(V_N, V_T, V_L) = \bigcup_{A \in V_N} \mathrm{aux}(V_N, V_T, V_L, A),$$

and let

$$\mathrm{init}(V_N, V_T, V_L) = \bigcup_{A \in V_N} \mathrm{init}(V_N, V_T, V_L, A).$$

We then define $\mathrm{elem}(V_N, V_T, V_L, A) = \mathrm{init}(V_N, V_T, V_L, A) \cup \mathrm{aux}(V_N, V_T, V_L)$.

Let $\mathrm{elem}(V_N, V_T, V_L) = \bigcup_{A \in V_N} \mathrm{elem}(V_N, V_T, V_L, A)$.

For each $\beta \in \mathrm{aux}(V_N, V_T, V_L)$ let $spine(\beta) = \{d \,|\, ft(\beta) = d \cdot d' \text{ for some } d' \in \mathbf{N}_+^*\}$, that is, $spine(\beta)$ are the addresses on the SPINE of β which is the path from the root to the foot node of β.

We now define the TREE COMPOSITION operation.

$$\nabla : \mathrm{elem}(V_N, V_T, V_L, A) \times \mathrm{elem}(V_N, V_T, V_L) \times \mathbf{N}_+^* \to \mathrm{elem}(V_N, V_T, V_L, A)$$

for every $A \in V_N$. For $\gamma \in \text{elem}(V_N, V_T, V_L, A), \gamma_1 \in \text{elem}(V_N, V_T, V_L), d \in \text{dom}(\gamma)$, we have $\gamma' = \nabla(\gamma, \gamma_1, d)$ where for all $d' \in \mathbf{N}_+^*$

$$\gamma'(d') = \begin{cases} \gamma(d') & \text{if } d \text{ is not a prefix of } d' \\ \gamma'(d'') & \text{if } d' = d \cdot d'' \text{ and } d'' \in \text{dom}(\gamma_1) \\ \gamma(d \cdot d'') & \text{if } \gamma_1 \in \text{aux}(V_N, V_T, V_L) \text{ and } d' = d \cdot d_f \cdot d'' \\ & \quad \text{such that } d'' \neq \epsilon \text{ and } d_f = ft(\gamma_1) \end{cases}$$

This composition operation describes taking a tree γ, 'splitting apart' some node in this tree, and inserting another tree. At this stage there are no restrictions on the trees or on the node, except that if a tree from $\text{init}(V_N, V_T, V_L)$ is used the remainder of γ below the split node is discarded; restrictions follow in the definition of the 'derives' relation below.

Definition 5.1.1 *A TAG is a seven tuple $\langle V_N, V_T, V_L, S, \mathcal{I}, \mathcal{A}, - \rangle$, where*

- *V_N is a finite set of non-terminal symbols;*
- *V_T is a finite set of terminal symbols, with $V_T \cap V_N = \emptyset$;*
- *V_L is a finite set of tree labels*
- *$S \in V_N$ is a distinguished non-terminal symbol;*
- *$\mathcal{I}$ is a finite subset of $\text{init}(V_N, V_T, V_L)$;*
- *$\mathcal{A}$ is a finite subset of $\text{aux}(V_N, V_T, V_L)$; and*
- *$- : \mathcal{A} \to V_L$ is a bijection, the auxiliary tree labelling function.*

In a TAG, the trees in $\mathcal{I} \cup \mathcal{A}$ are referred to as elementary trees.[3]
Given a TAG $G = \langle V_N, V_T, V_L, S, \mathcal{I}, \mathcal{A}, - \rangle$, the derives relation $\stackrel{*}{\Longrightarrow}$ is defined as follows.

- If $\gamma \in \text{elem}(V_N, V_T, V_L, A)$, $\gamma(d) = \langle B, \mathbf{sa}, \mathbf{oa} \rangle$ for some $d \in \text{internal}(\gamma)$, $\beta \in \mathcal{A} \cap \text{aux}(V_N, V_T, V_L, B)$, and $\bar{\beta} \in \mathbf{sa}$ then

$$\gamma \stackrel{*}{\Longrightarrow} \nabla(\gamma, \beta, d)$$

  Note that the same non-terminal, B, must label the composition (internal) node and the root and foot of the inserted tree.

  This type of composition is referred to as TREE ADJUNCTION.

[3]Note that this does not correspond to elements of $\text{elem}(V_N, V_T, V_L)$; 'elementary tree' is the conventional term for one of the minimal base trees from which others are constructed, whereas $\text{elem}(V_N, V_T, V_L)$ contains trees formed by composition also.

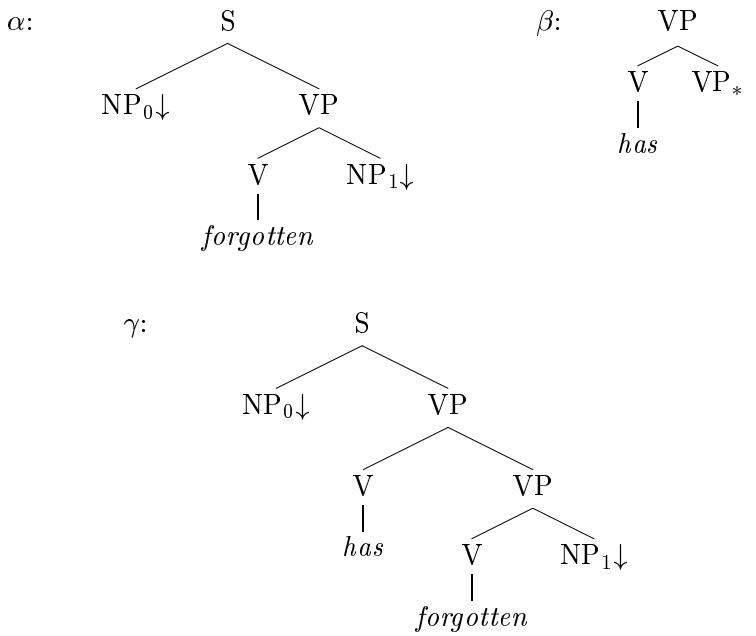


Figure 5.1: Example of adjoining

Tree	Node	Address
α	S	ϵ
	NP_0	1
	VP	2
	V	$2 \cdot 1$
	NP_1	$2 \cdot 2$
β	VP	ϵ
	V	1
	VP_*	2
γ	S	ϵ
	NP_0	1
	VP	2
	V	$2 \cdot 1$
	VP	$2 \cdot 2$
	V	$2 \cdot 2 \cdot 1$
	NP_1	$2 \cdot 2 \cdot 2$

Figure 5.2: Addresses for nodes in the trees of Figure 5.1

- If $\gamma \in \text{elem}(V_N, V_T, V_L, A)$, $\gamma(d) = \langle B, \emptyset, \textbf{false} \rangle$ for some $d \in \text{leaf}(\gamma)$, $\alpha \in \mathcal{I} \cap \text{init}(V_N, V_T, V_L, B)$ then

$$\gamma \stackrel{*}{\Longrightarrow} \nabla(\gamma, \alpha, d)$$

  Note that the same non-terminal, B, must label the composition (leaf) node and the root of the inserted tree.

  This type of composition is referred to as TREE SUBSTITUTION.

An example of tree adjunction is given in Figure 5.1, β adjoining into α: the VP node in α is 'pulled apart', β inserted, and the VP nodes merged to complete the adjunction, giving γ. An example of tree substitution is given in Figure 5.3, with α_2 substituted into α_1 at the NP node, giving γ.

In the examples given in this thesis, a TAG will generally only be given by its sets of initial trees (labelled α_i) and auxiliary trees (labelled β_i); by convention, the distinguished symbol will be S, the terminal alphabet the lower-case symbols, and the non-terminal alphabet the upper-case symbols. Also, when displaying a tree graphically we use several conventions. The non-terminal or terminal component of a node label is always shown. The adjunction constraints for a node labelled $\langle A, \textbf{sa}, \textbf{oa} \rangle$ are specified as follows. When **sa** is the empty set the node is annotated NA. The case in which **sa** includes the labels of all auxiliary trees in the grammar that are in $\text{aux}(V_N, V_T, V_L, A)$ (note that A is the same) is the default so no annotation is used. Otherwise, the set **sa** is given in full. The case in which **oa** is **false** is the default and no annotation is used. When **oa** is **true** the node is annotated OA. Nodes 'marked for substitution' (leaf nodes labelled with non-terminals) are indicated by a $\downarrow$; foot nodes of auxiliary trees are indicated by a $*$. In addition, in linguistic examples the trees follow XTAG notation (XTAG, 1995): the terminal alphabet, of natural language words, will be in italics, and the non-terminal alphabet, of abbreviations for category names, will be in non-italic font. Linguistic arguments in trees (such as the NPs in α) are subscripted according to argument order, as described in Appendix A.

Now, to define a sequence of derives relations, we note that for all $\gamma \in \text{elem}(V_N, V_T, V_L, A)$, $\gamma_1, \gamma_2 \in \text{elem}(V_N, V_T, V_L)$, $d_1 \in \text{dom}(\gamma)$ and $d_2 \in \text{dom}(\gamma_1)$,

$$\nabla(\nabla(\gamma, \gamma_1, d_1), \gamma_2, d_1 \cdot d_2) = \nabla(\gamma, \nabla(\gamma_1, \gamma_2, d_2), d_1) \tag{5.1}$$

Thus, for all $\gamma \in \text{elem}(V_N, V_T, V_L, A)$, $\gamma_1, \gamma_2 \in \text{elem}(V_N, V_T, V_L)$, $d \in \text{dom}(\gamma)$, if $\gamma \stackrel{*}{\Longrightarrow} \nabla(\gamma, \gamma_1, d)$ and $\gamma_1 \stackrel{*}{\underset{G}{\Longrightarrow}} \gamma_2$ then $\gamma \stackrel{*}{\underset{G}{\Longrightarrow}} \nabla(\gamma, \gamma_2, d)$ where $\stackrel{*}{\underset{G}{\Longrightarrow}}$ is the reflexive, transitive closure of $\stackrel{*}{\Longrightarrow}$. So under this identity it is equivalent to

1. take a tree formed by composition from several components $\gamma_1, \gamma_2, \ldots, \gamma_{n-1}$ and compose into it some tree γ_n, and
2. take a tree γ_1 and compose into it the tree formed by composition from $\gamma_2, \ldots, \gamma_n$.

This is used later in the definition of derivation trees; in inductively defining these we use the latter interpretation.

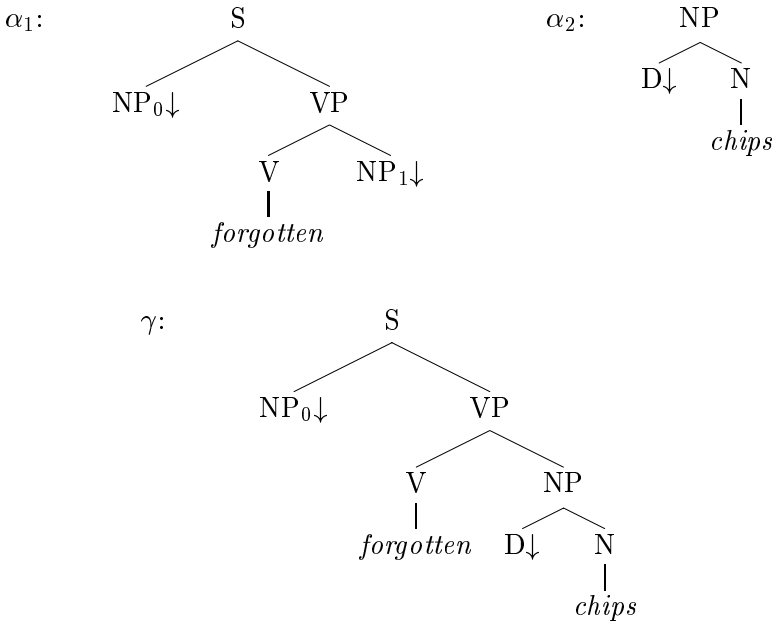


Figure 5.3: Example of substitution

Now we will define the objects of interest that a derivation produces. A tree γ has no OA nodes if for all $d \in \mathrm{dom}(\gamma)$ either $\gamma(d) = \langle A, \mathbf{sa}, \mathbf{false}\rangle$ for some A and **sa** or $\gamma(d) = u$ for some $u \in V_T^\epsilon$. The frontier of a tree γ has no nodes marked for substitution if for all $d \in \mathrm{leaf}(\gamma)$, $\gamma(d) = \langle A, \mathbf{sa}, \mathbf{oa}\rangle$ for some $A \in V_N$, **sa** and **oa**.

The TREE LANGUAGE (or tree set) $T(G)$ generated by $G = \langle V_N, V_T, V_L, S, \mathcal{I}, \mathcal{A}, -\rangle$ is defined as

$$\begin{aligned} T(G) \quad = \quad & \{\gamma \mid \alpha \underset{G}{\overset{*}{\Longrightarrow}} \gamma \text{ for some } \alpha \in \mathcal{I}, \\ & \text{and } \gamma \text{ has no OA nodes or frontier nodes marked for substitution}\} \end{aligned}$$

In defining yield(γ), the yield of a tree γ, we consider only symbols in V_T and V_N, that is, we do not give the SA or OA constraints of nodes involving non-terminals. For a tree γ

- if γ consists of a single node then if $\gamma(\epsilon) = u$ for some $u \in V_T^\epsilon$ then yield(γ) $= u$ and if $\gamma(\epsilon) = \langle A, \mathbf{sa}, \mathbf{oa}\rangle$ for some A, **sa** and **oa** then yield(γ) $= A$.
- otherwise if the root of γ has k children ($k > 0$) then

$$\mathrm{yield}(\gamma) = \mathrm{yield}(\gamma/1) \ldots \mathrm{yield}(\gamma/k)$$

The STRING LANGUAGE $L(G)$ generated by G is defined as

$$L(G) = \{\text{yield}(\gamma) | \gamma \in T(G)\}$$

Another structure, of more secondary interest and so not defined formally here, is the path set of a grammar G, which is the set of all paths from the root to the frontier for trees in $T(G)$.

For a given set of trees, different ordering of the composition operation can still produce the same derived tree. In earlier definitions of TAG (Vijay-Shanker, 1987), this was expressed by the idea of multiple simultaneous adjunction, where the ordering of the individual compositions was unimportant. Multiple simultaneous adjunction had the notation

$$\gamma[\gamma_1/d_1, \gamma_2/d_2, \ldots \gamma_k/d_k]$$

In the notation of this thesis, this covers many alternatives, for example,

$$\begin{gathered}\nabla(\nabla(\ldots \nabla(\gamma, \gamma_1, d_1), \gamma_2, d_{\gamma_2}), \ldots), \gamma_k, d_{\gamma_k}) \\ \nabla(\nabla(\ldots \nabla(\gamma, \gamma_2, d_2), \gamma_1, d_{\gamma_1}), \ldots), \gamma_k, d_{\gamma_k}) \\ \vdots\end{gathered}$$

We will therefore define a canonical ordering, using two identities, (5.1) above and (5.2) below.

For all $\gamma \in \text{elem}(V_N, V_T, V_L, A)$, $\gamma_1, \gamma_2 \in \text{elem}(V_N, V_T, V_L)$, $d_1 \in \text{dom}(\gamma)$ and $d_2 \in \text{dom}(\gamma)$,

$$\nabla(\nabla(\gamma, \gamma_1, d_1), \gamma_2, d_1 \cdot d_2) = \nabla(\nabla(\gamma, \gamma_2, d_2), \gamma_1, d_1) \tag{5.2}$$

Identity (5.2) reflects the reordering of trees composed into the same tree ('sibling ordering').

A BOTTOM-UP DERIVATION is one in which it is not possible to apply the lefthand side of identities (5.1) and (5.2).

These identities can be repeatedly applied to the expressions obtained through other derivations to obtain the one obtained through a bottom-up derivation. As an example, take the TAG trees of Figure 5.4, which produce the derived tree in Figure 5.5. α_2 is substituted into α_3 before this composite is substituted into α_1; α_4 and β_1 are also composed into α_1. The bottom-up derivation gives

$$\nabla(\nabla(\nabla(\alpha_1, \nabla(\alpha_3, \alpha_2, 1), 2 \cdot 2), \alpha_4, 1), \beta_1, 2)$$

Given $\gamma, \gamma_i \in \text{elem}(V_N, V_T, V_L, A)$, $d_i \in \text{dom}(\gamma)$, for $1 \leq i \leq k$, in such a bottom-up derivation we derive

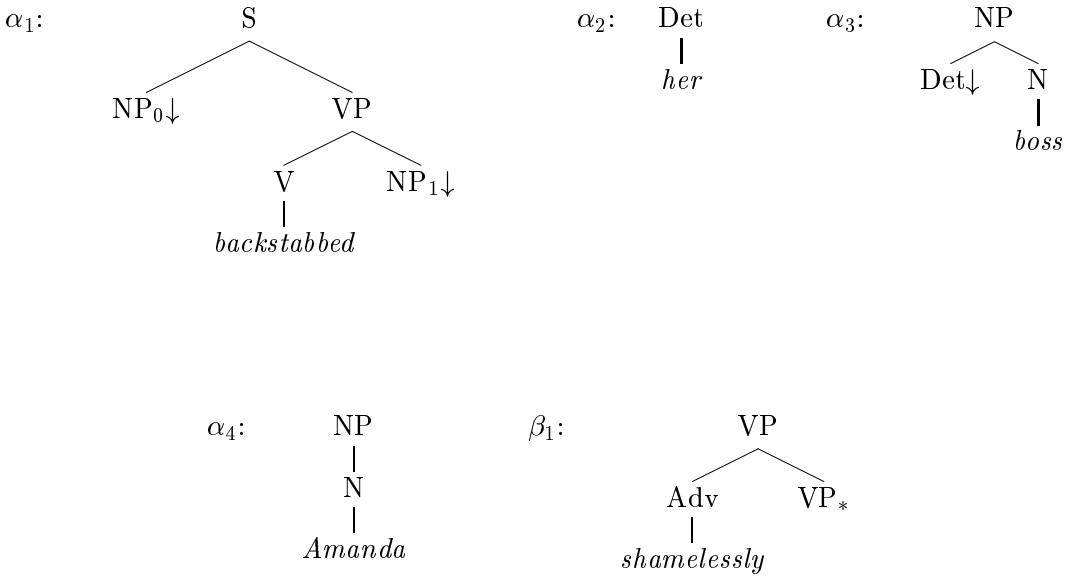


Figure 5.4: Elementary trees

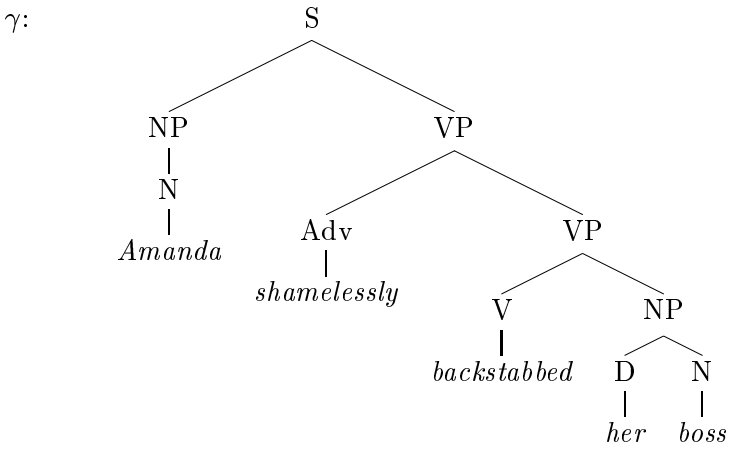


Figure 5.5: Derived tree

$$\nabla(\ldots\nabla(\nabla(\gamma,\gamma_1,d_1),\gamma_2,d_2),\ldots,\gamma_k,d_k)$$

This corresponds to composition into elementary trees only ($d_i \in \mathrm{dom}(\gamma)$); γ_i can be elementary trees or combined trees of the form $\nabla(\gamma_j,\gamma_k,d_j)$. We will thus use as a shorthand

$$\nabla_S(\gamma,\gamma_1,d_1,\gamma_2,d_2,\ldots,\gamma_k,d_k)$$

This then corresponds to the notation $\gamma[\gamma_1/d_1,\gamma_2/d_2,\ldots,\gamma_k/d_k]$ used in Vijay-Shanker (1987), with all d_i distinct.

A derivation tree is a structured representation of the derivation history. In TAG, the derived tree does not encode its derivation history (even ignoring ordering of composition); a given derived tree could have different composition modulo ordering (Weir, 1988: 22).

Vijay-Shanker (1987: 20) thus defines a derivation tree for TAG. The definition below is a modified form of this, to take into account the operation of substitution.

Nodes in the derivation tree correspond to trees in the TAG grammar. Therefore, let the set of elementary trees $\mathcal{I}\cup\mathcal{A}=\{\gamma_1,\gamma_2,\ldots,\gamma_m\}$ be given the names $\{\bar{\gamma}_1,\bar{\gamma}_2,\ldots,\bar{\gamma}_m\}$.

Definition 5.1.2 *A* TAG DERIVATION TREE *is defined as follows.*

- *If γ_i is an elementary tree with no OA constraints or frontier nodes marked for substitution, the derivation tree consists of a single node labelled $\bar{\gamma}_i$ where $1\leq i\leq m$.*
- *Consider a derivation in which γ' is produced as follows.*

  $$\gamma' = \nabla_S(\gamma_i,\gamma'_1,d_1,\gamma'_2,d_2,\ldots,\gamma'_k,d_k)$$

  where $1\leq i\leq m$ and $k\geq 1$, and all OA constraints and frontier nodes marked for substitution are included in $d_1,\ldots,d_k$.

  For each $1\leq j\leq k$, let γ'_j be derived from the elementary tree γ_{i_j}, where this derivation is represented by the derivation tree Υ_j. The root will be labelled by $\bar{\gamma}_{i_j}$.

  Let Υ'_j be the tree Υ_j except that its root is relabelled as $\langle\bar{\gamma}_{i_j},d_j\rangle$. The derivation tree for γ' will be rooted with a node labelled $\bar{\gamma}_i$ with k subtrees, Υ'_j, for $1\leq j\leq k$.

An example of a derivation tree, using the trees of Figure 5.4, is given in Figure 5.6, representing the derivation of the sentence *Amanda shamelessly backstabbed her boss.* When displaying a tree graphically, we will indicate substitution operations by dotted lines, and adjunction by solid lines. In addition, we will often write as the node labels only symbols such as γ (noting the bijection between names and trees), and addresses will only optionally be given, in parentheses after the label. In order to make recognition of the

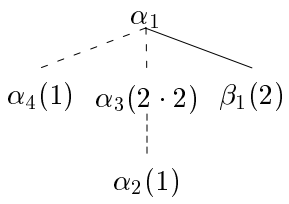


Figure 5.6: Derivation tree

corresponding trees easier, particularly in linguistic examples, sometimes the node label in the derivation tree is followed by its lexical anchor(s) in square brackets. For example, in Figure 5.6 the left child of the root might be labelled α_4[Amanda](1).

The DERIVATION TREE SET $T_D(G)$ for a TAG $G = (V_N, V_T, V_L, S, \mathcal{I}, \mathcal{A}, -)$ is defined as

$$T_D(G) = \{\Upsilon \mid \Upsilon \text{ is a derivation tree for some } \gamma \in T(G)\}$$

Given that each derivation tree specifies one derived tree, we can define a function from derivation trees to derived trees (this modified from Schabes and Shieber, 1994). Here and afterwards we will use the notation that for some pair x, x_L and x_R define the left and right projections of the pair respectively.

Definition 5.1.3 *A* TAG TREEYIELD FUNCTION $\mathcal{D} : T_D(G) \rightarrow T(G)$ *for some TAG* $G = (V_N, V_T, V_L, S, \mathcal{I}, \mathcal{A}, -)$ *and* $\Upsilon \in T_D(G)$ *is defined as follows.*

$$\mathcal{D}(\Upsilon) \quad = \quad \begin{cases} \gamma \quad \textit{if } \Upsilon \textit{ is a tree of one node and } \Upsilon(\epsilon) = \bar{\gamma} \\ \nabla_S(\gamma, \mathcal{D}(\Upsilon/1), \Upsilon_R(1), \ldots, \mathcal{D}(\Upsilon/k), \Upsilon_R(k)) \\ \qquad \textit{if } \Upsilon \textit{ has } k \textit{ children and } \Upsilon(\epsilon) = \bar{\gamma} \end{cases}$$

5.2 Some Basic TAG Properties

As mentioned in the introduction to this chapter, early work in mathematical linguistics used CFGs as the basis for modelling natural language. They have been shown not to be adequate for this; however, they are still useful as a baseline for comparison, as they have been extensively studied. This section will look at the basic properties of TAG, in part through comparison with the properties of CFG.

Generative Capacity

TAGs are between CFGs and CSGs (Context Sensitive Grammars) in power: that is, they can generate all languages generated by CFGs and more, but their own string languages are properly contained in the set of languages generated by CSGs.

In order to be more precise, the (standard) formal definitions of CFGs and CSGs are given below, along with that of another type of grammar, the regular grammar (RG).

Definition 5.2.1 *A phrase-structure grammar (PSG) is a quadruple* (V_T, V_N, S, P)*, where*

- V_T*,* V_N *and* S *are, as for TAGs, the finite set of terminal symbols, the finite set of non-terminal symbols, and the distinguished symbol* $S \in V_N$*, respectively; and*
- P *is a set of productions or rewrite rules with at least one non-terminal on the left hand side:* $P \subseteq \Sigma^* V_N \Sigma^* \times \Sigma^*$*, where* $\Sigma = V_T \cup V_N$*. A production* $(u, v) \in P$ *may be written* $u \rightarrow v$*.*

A CFG is a PSG where for each production $(u \rightarrow v) \in P$*,* $u \in V_N$*.*

A CSG is a PSG where for each production $(u \rightarrow v) \in P$*,* $u = u_1 A u_2$ *and* $v = u_1 x u_2$*, with* $u_1, u_2 \in \Sigma^*$*,* $A \in V_N$ *and* $x \in \Sigma^+$*.*

An RG is a PSG where for each production $(u \rightarrow v) \in P$*,* $u \in V_N$ *and* $v \in V_T \cup V_T V_N \cup \{\epsilon\}$*, where* ϵ *is the empty string.*

For a PSG G, the derives relation $\stackrel{*}{\Longrightarrow}$ is defined as follows. If $(u \rightarrow v) \in P$ and $\alpha, \gamma \in \Sigma^*$, then $\alpha u \gamma \stackrel{*}{\Longrightarrow} \alpha v \gamma$. $\stackrel{*}{\underset{G}{\Longrightarrow}}$ is the reflexive transitive closure of $\stackrel{*}{\Longrightarrow}$.

The string language generated by a grammar G, $L(G)$, is the set of all strings that a grammar can generate, that is, $L(G) = \{w \mid S \stackrel{*}{\underset{G}{\Longrightarrow}} w\}$. For the above three types of grammar, their associated languages are termed context free languages (CFLs), context sensitive languages (CSLs) and regular languages (RLs). When referring to a PSG, we will use the convention that terminals are given by lower-case symbols, non-terminals by upper-case symbols, and the distinguished symbol is the character S unless otherwise specified; thus only the set of productions will be given.

Two grammars G_1 and G_2 are WEAKLY EQUIVALENT if $L(G_1) = L(G_2)$. G_1 and G_2 are STRONGLY EQUIVALENT if they are weakly equivalent and for each $w \in L(G_1) = L(G_2)$, G_1 and G_2 assign the same structural description (tree) to w. Note that when comparing trees, we are only interested in symbols in V_T and V_N (so for TAG trees we ignore **sa** and **oa** constraints).

Joshi *et al* (1975) have shown that for every CFG G there is a TAG G' equivalent to G both weakly and strongly; thus TAGs are at least as expressive as CFGs.

As an example, take the following CFG G_1, from Joshi (1985).

$$\begin{aligned} S &\rightarrow aTb \\ T &\rightarrow Sa \\ T &\rightarrow TT \\ T &\rightarrow ab \end{aligned}$$

Take also the following TAG G_1' in Figure 5.7. G_1 is strongly equivalent to G_1'.

By contrast, take the CFG G_2.

$$\begin{aligned} S &\rightarrow aSb \\ S &\rightarrow \epsilon \end{aligned}$$

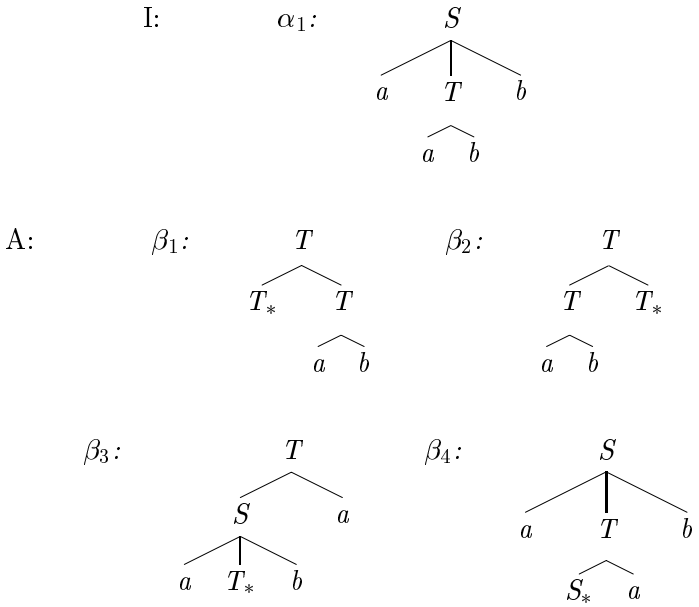


Figure 5.7: Grammar strongly equivalent to G_1

Take also the TAG G_2' in Figure 5.8. Both generate the context free language $L = \{a^n b^n \mid n \geq 0\}$. That is, G_2 and G_2' are weakly equivalent. However, they are not strongly equivalent, as G_2 produces centre-embedded trees, while G_2' produces right-skewed trees; and in fact there is no CFG strongly equivalent to G_2' (Joshi, 1985).

Some TAGs generate languages that CFGs cannot generate at all. Take the TAG G_3' in Figure 5.9. This generates the strictly context-sensitive language (CSL)

$$L(G_3') \quad = \quad \{a^n b^n c^n \mid n \geq 0\}$$

A language, such as $L(G_3')$, can be shown not to be context free by one of two types of

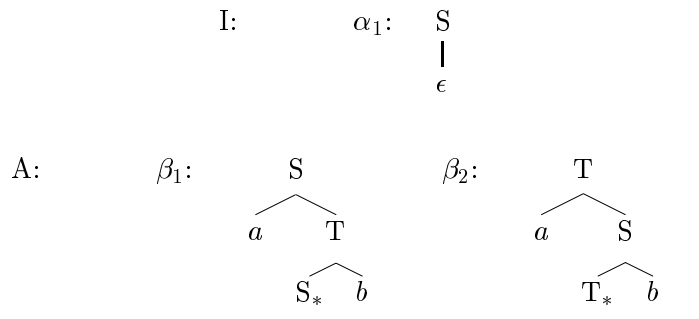


Figure 5.8: Grammar weakly equivalent to G_2

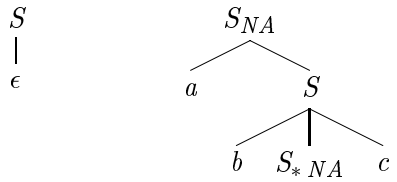


Figure 5.9: TAG grammar with constraints generating $a^n b^n c^n$

approaches. One is through various closure results: for example, as CFLs are closed under intersection with RLs, if $L(G_3')$ were intersected with an RL and the resulting language were not context free, then $L(G_3')$ could not be a CFL. More directly, a Pumping Lemma describes particular characteristics of strings in languages generated by some formalism, such that if a given language does not have these characteristics then it cannot be generated by that formalism. A Pumping Lemma for CFLs states that for some CFL L, a string $z \in L$, and for some sufficiently large constant n and $|z| \geq n$, we may write z as $uvwxy$ for $u, v, w, x, y \in V_T$ such that $|vx| \geq 1$, $|vwx| \leq n$, and for all $i \geq 0$, $uv^i wx^i y \in L$. If for some language L' it is not possible to write z as such a string $uvwxy$, then L' is not a CFL. The example here, $L(G_3')$, can hence be shown not to be a CFL (Hopcroft and Ullman, 1979: 128–129).

Derivations

In Section 5.1 we defined derivations, and derivation trees, for a TAG. Here we discuss some characteristics of the derivations that are useful later.

It can be seen from the form of the derivation rules used building the derivation tree under Definition 5.1.2—under which there is no 'memory' of operations beyond a single level in the tree—and has been shown by Weir (1988:22–23), that the derivation of a TAG is context free, with the following consequence.

Lemma 5.2.2 *(Weir, 1988) The set of derivation trees for some TAG G is a local set, that is, it can be encoded by a CFG; the path set of the derivation trees of G can be encoded by a regular grammar.*

In fact, Weir (1988: 45) characterises a TAG as a TAG yield function, such as the treeyield function $\mathcal{D}$ in Definition 5.1.3, applied to a context free grammar of derivation trees. TAG derived trees and TAG derivation trees are consequently referred to by Weir as OBJECT-LEVEL and META-LEVEL trees respectively.

This derivational aspect of TAGs can also be used to relate TAGs to dependency grammar (see, for example, Mel'čuk, 1988): derivation trees have strong correspondences to dependency trees, as they describe which trees are arguments or adjuncts of which others; the tree in Figure 5.6 gives dependency relationships, with subject and object nouns (α_3 and α_4 respectively) and the adverbial modifier (β_1) depending on the verb (α_1), and the determiner depending on its noun (α_2). For an exploration of this see, for example, Rambow and Joshi (1995) or Candito (1998).

Linguistic Characteristics

The definition of TAG, with its two operations of substitution and adjoining, leads to two properties which suit them particularly well to linguistic applications.

The first property is EXTENDED DOMAIN OF LOCALITY (EDL). The domain of locality is larger for TAGs than for CFGs. CFGs can effectively only have as their domain of locality a single level in a tree; TAGs, by definition, as many levels as desired. A CFG with the rules

$$S \rightarrow NP\ VP$$
$$VP \rightarrow V\ NP$$

specifies the relationship between a verb and its two arguments, subject and object, in two rules; there is no way of specifying the relationship in one rule while still retaining the VP node. Because TAG trees describe multiple levels of nodes, they allow localisation of more dependencies than CFGs, including long-distance dependencies such as filler-gap.

The second property is FACTORING RECURSION FROM THE DOMAIN OF DEPENDENCIES (FRD). When applied linguistically, there is an implicit requirement that elementary trees are 'minimal', codified under the Condition on Elementary Tree Minimality of Frank (1992: 54). That is, they describe a particular relationship, like the structure of a declarative sentence, in only a skeletal form, and specify the dependencies in that relationship, such as agreement, subcategorisation, and filler-gap. Any more complex forms—relative clauses between subject and verb, adverbial modifiers, and so on—which correspond to the dependencies being long-distance occur through the recursive operation of adjoining; dealing with long distance dependencies is thus factored out of the original specification of dependencies.

In linguistic applications of TAG, many words will select the same tree. In the standard XTAG grammar (XTAG, 1995), standard trees are named mnemonically: for example, tree α_1 in Figure 5.4 is an α**nx0Vnx1** tree, indicating that it is an initial tree (the α), that the lexical anchor is a verb (the capital **V**), and that it has two NP arguments (the **nx0** and **nx1**). A list of these is given in Appendix A. In addition, trees are often grouped together into TREE FAMILIES. Rather than having to specify individual trees for each word and all of its possible syntactic variations (such as extraposition, passive and so on), trees are grouped into families, and words assigned to the whole family of trees. These families are typically for verb trees, so when a verb selects a tree family it takes the associated trees as well. Tree families are similarly named mnemonically: the tree family for transitive verbs (containing, as well as the standard active voice declarative tree, the extraposed tree, the passive tree, and so on), for example, is denoted **Tnx0Vnx1**.

Dependencies

Another way in which TAGs are more powerful than CFGs, and more suitable for describing natural languages, is in their ability to describe specific kinds of dependencies, notably cross-serial dependencies (Joshi, 1985). In cross-serial dependencies, elements of the generated string are linked together in such a way that the links cross each other. These links are not an extra mechanism; they just indicate which terminals were produced during the same step of a derivation. See Rambow (1994: 37) for an exploration of this idea, and how it relates to strong generative capacity.

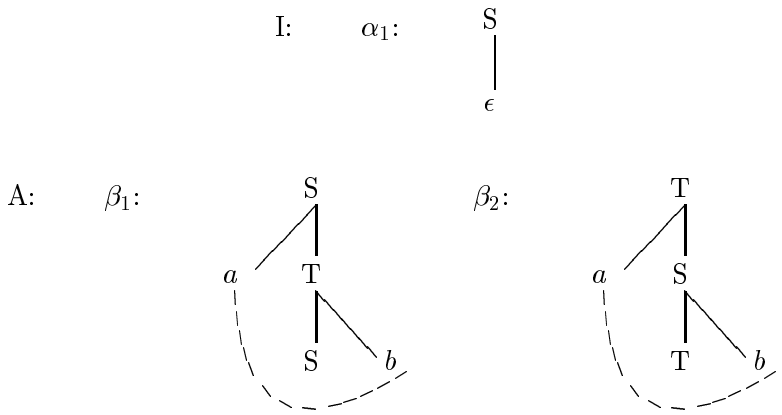
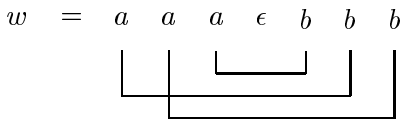


Figure 5.10: Trees with dependencies

Take the TAG G'_4 in Figure 5.10, where terminals in the resulting string are considered dependent if they came from the same application of an elementary tree during the derivation. Repeated applications of β_1 produce nested dependencies, and of β_2, cross-serial dependencies. Two adjunctions of β_1 and one of β_2 into α_1 and subsequent derived trees can produce the string



where the links between the elements of the string come from the links in the trees. CFGs produce only nested dependencies (as the innermost a and b in w above) or disjoint occurrences.

This is a feature of TAGs used to argue that TAGs are useful for describing natural languages, as cross-serial dependencies occur in some natural languages. Pullum and Gazdar (1982) reviewed attempts to show that natural languages are not context free, and found that at that time there were no convincing arguments that this was so: apparent problems with modelling such dependencies with CFGs could be explained by semantic or pragmatic considerations, or by the fact that previous attempts had considered only a subset of the language in showing non-context freeness, which does not of itself show that the whole language is not context free; an example from Partee *et al* (1993) is of the non-CFL $\{xx \mid x \in \{a,b\}^*\}$, which is a subset of the CFL $\{a,b\}^*$. Demonstrating non-context freeness via cross-serial dependencies in Dutch had these problems, as Dutch does not have case marking for constituents which are cross-serially linked, meaning that these could be explained by semantic or pragmatic factors, rather than by syntax. However, Shieber (1985) demonstrated the case using Swiss German, which has cross-serial dependencies similar to Dutch, but which indicates them via morphological case marking. He showed that intersecting Swiss German with a regular language produces a language of the form $\{wa^nb^mxc^nd^my \mid m,n \geq 0\}$, which can be shown by the Pumping Lemma for CFLs not to

be context free.

5.3 Extensions to TAG

There have been a number of extensions to the basic TAG formalisms, three of which are outlined here. The first two, the addition of feature structures and the requirement of lexicalisation, are not essential to this thesis, but, having become somewhat standard by virtue of the XTAG system, warrant inclusion and some discussion in later sections about how the ideas in the thesis relate to them. The third extension, however, the extension from trees as the elementary units to sequences of trees termed Multi-Component TAG (MCTAG), is used in later chapters in a number of ways.

5.3.1 Feature Structure-based TAG (F-TAG)

Elementary trees in a TAG, when used for linguistic modelling, have as one of their implicit characteristics that they capture dependencies, such as predicate-argument or filler-gap. Incorporating feature structures and unification into TAG (Vijay-Shanker, 1987) has the same goal of localising these dependencies; for example, agreement can be modelled locally. In F-TAG, feature structures are associated with nodes in elementary trees, and can describe two relationships: the relation of a node to its supertree, and the relation to its subtree. Thus for a node η there are two feature structures, t_η and b_η respectively. Having two feature structures fits with the operation of adjunction on trees: the node at which adjoining occurs is effectively split in two, and the auxiliary tree inserted, and the two feature structures can be separated when the node is split. Note that nodes marked $\downarrow$ for substitution have only one feature structure, since adjunction cannot take place at these nodes. If the auxiliary tree has root node ρ and foot node ϕ, then at the end of the derivation (if no other auxiliary trees are inserted at these nodes) t_η must unify with t_ρ, and b_η with b_ϕ.

The feature structure addition to TAG can also be used to represent constraints on nodes.[4] On a node η where adjunction is obligatory (an OA constraint), incompatible feature structures t_η and b_η force adjunction; otherwise, t_η and b_η should be compatible. Figures 5.11 through 5.14 contain examples with obligatory adjunction and unconstrained adjunction. In Figure 5.11, all top and bottom feature structures of tree α_1 can unify with no adjunctions necessary; but it is still possible for an adjunction to occur as normal, as for example by adjoining tree β_1 into α_1 in Figure 5.12, producing the tree of Figure 5.13. In tree α_2 of Figure 5.11, the feature structures on the tree ultimately will not be able to unify without some modification: from the VP node, the case value **none** will percolate up the tree, and will clash with the case of any tree substituted into node $\mathbf{NP}_0$ (for example, α_3). By adjoining the tree β_2 at this node, there will no longer be a case clash, as illustrated by the tree in Figure 5.14.

F-TAGs in this form are unconstrained, just as CFG-based unification formalisms are. However, Vijay-Shanker (1987) has shown that if the feature structures are bounded, the resulting formalism of Restricted F-TAG (RF-TAG) is equivalent to TAG. This restricted formalism is the one used in the XTAG implementation of TAGs.

[4] More precisely, feature structures can represent obligatory adjunction and selective adjunction, but null adjunction still needs to be explicitly marked on a node.

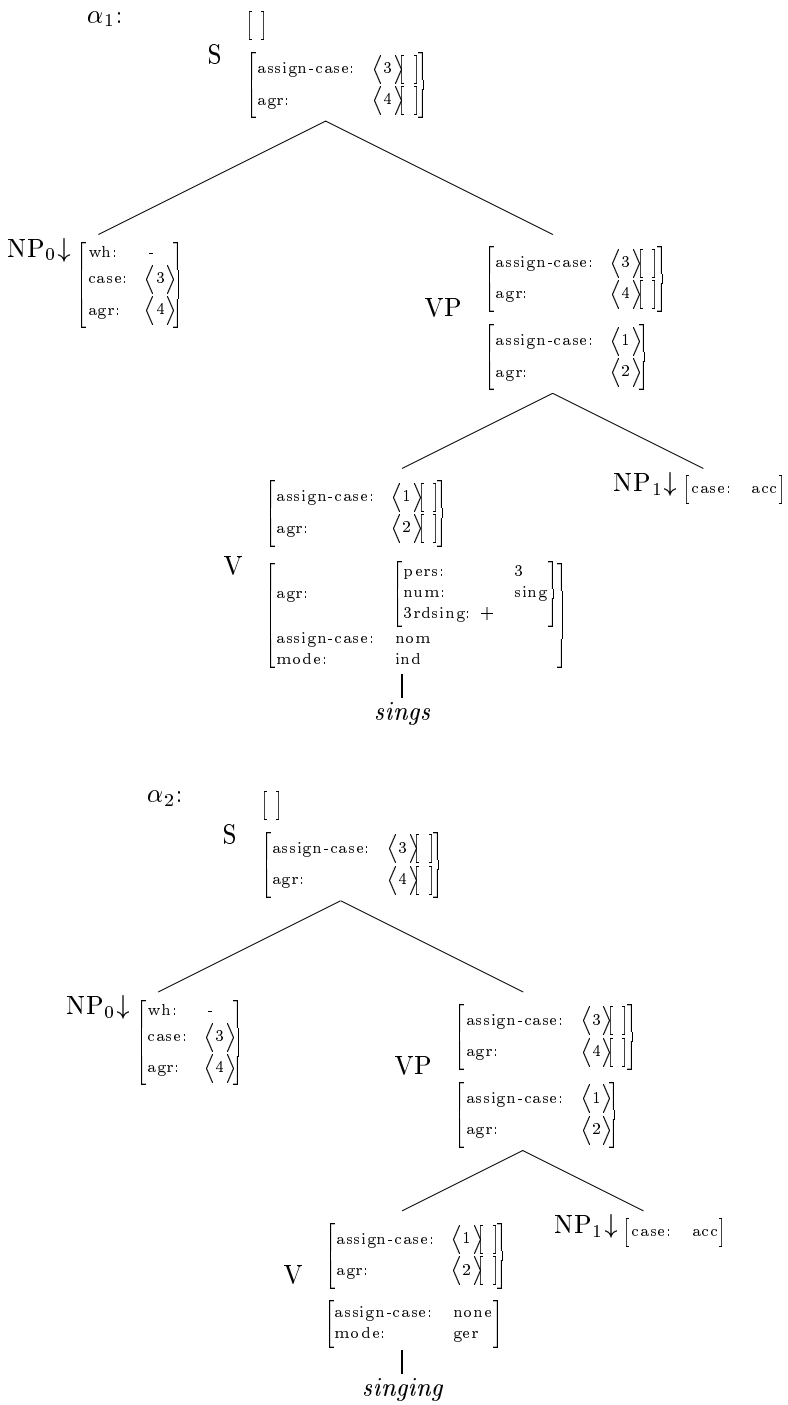


Figure 5.11: Example of composition with feature structures

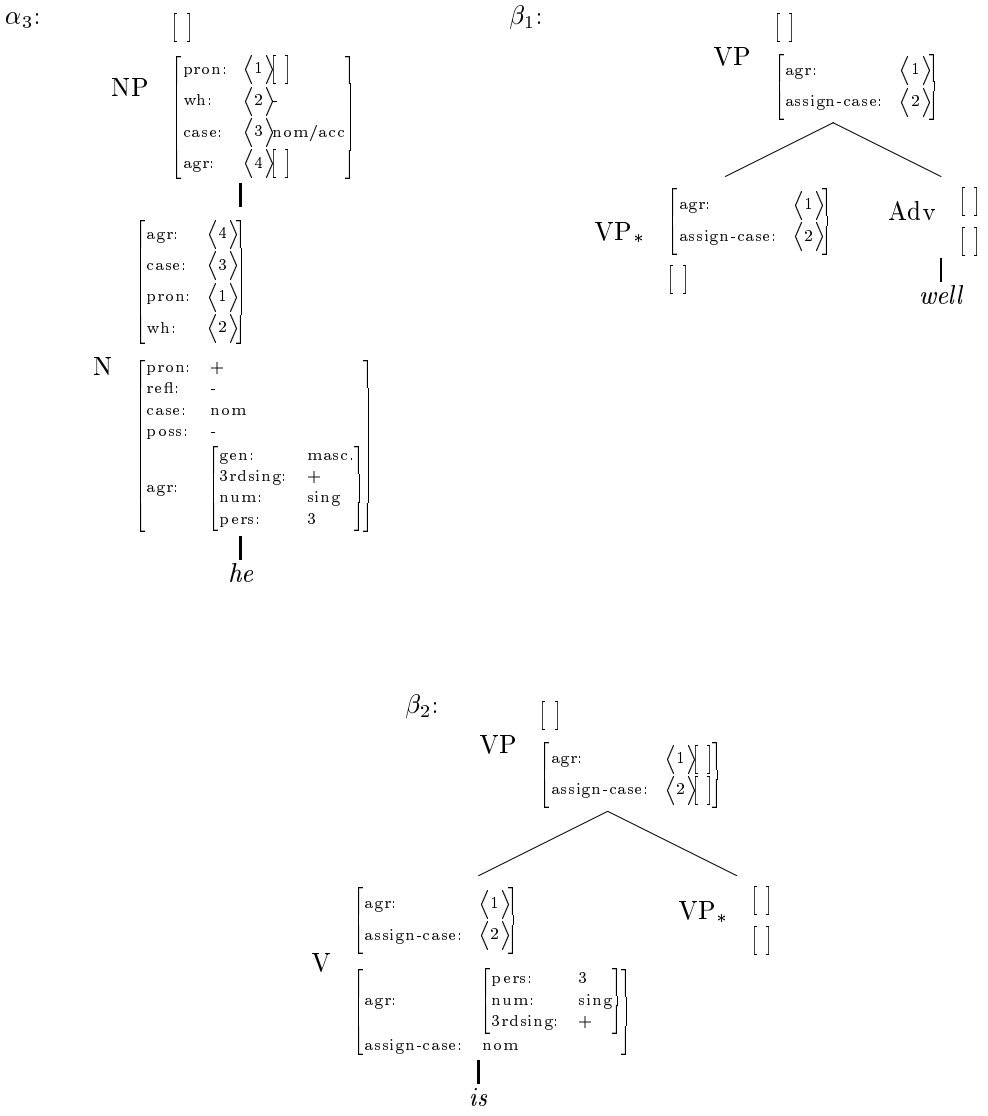


Figure 5.12: Example of composition with feature structures

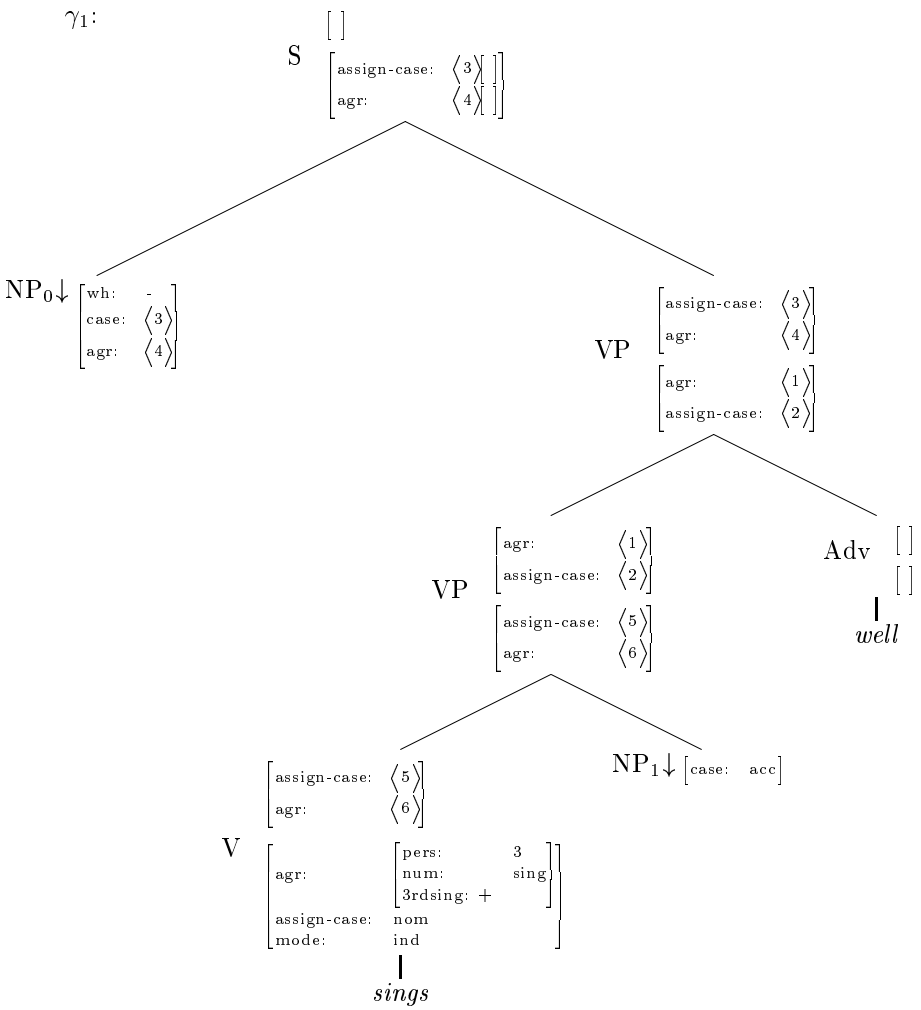


Figure 5.13: Example of composition with feature structures

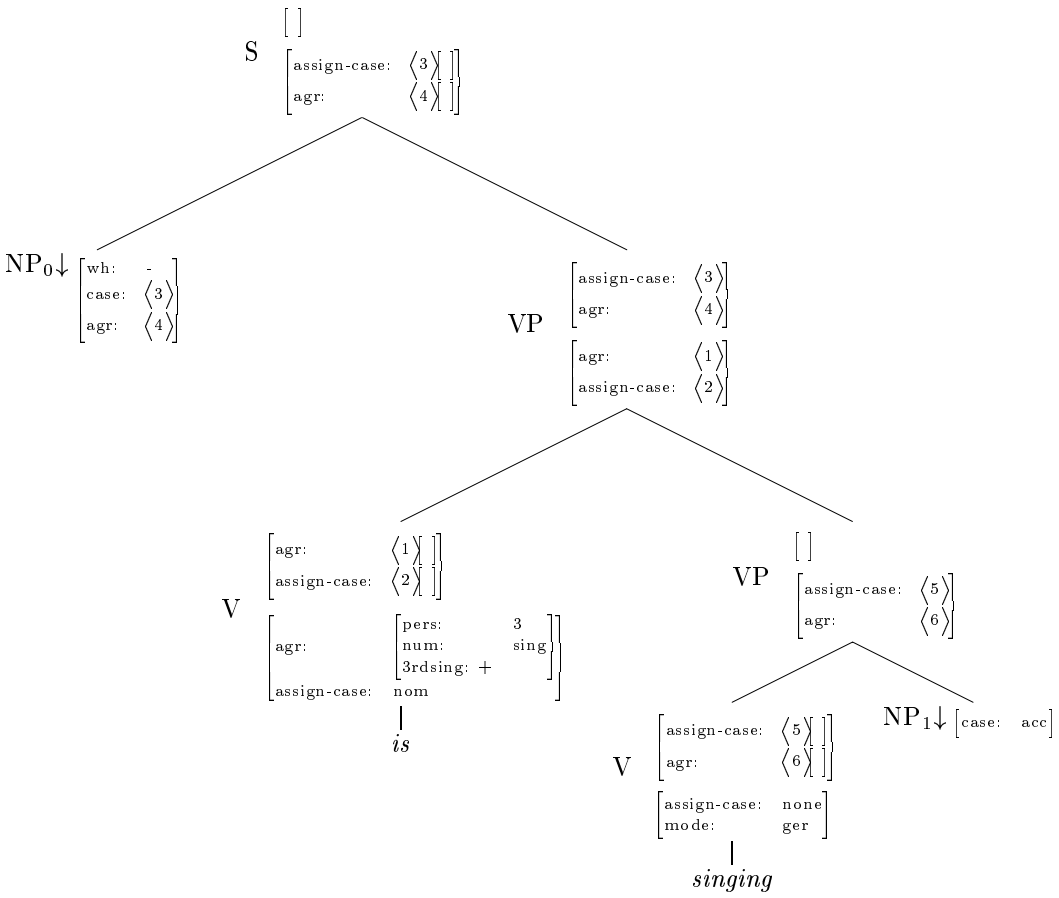


Figure 5.14: Example of composition with feature structures

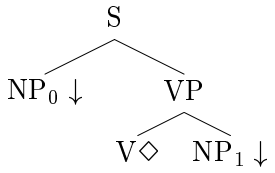


Figure 5.15: A lexicalised tree with anchor

5.3.2 Lexicalised TAG (LTAG)

The development of lexicalised TAGs came with the general movement in the linguistic application of grammars towards explaining phenomena lexically rather than syntactically. Schabes (1988) defines a lexicalised grammar as one

- with each lexical item (terminal symbol) being associated with a finite set of structures, and where the lexical item is in general the head of the structure; and
- with operations for composing these structures.

Having a lexicalised grammar has a number of advantages. In terms of formal properties, it can be shown (as summarised in Schabes and Joshi, 1996) that lexicalised grammars are finitely ambiguous, and that it is decidable whether a string is acceptable by a lexicalised grammar. In linguistic terms, the language-related properties of EDL and FRD are closely related to lexicalisation, and using a lexicalised grammar allows lexical idiosyncrasies to be more neatly explained. In addition, the use of a lexicalised TAG means that the syntactic paraphrases of this thesis will reflect the lexically-conditioned nature of many of the paraphrase mappings.

In practice, the same tree is associated with a number of lexical items: for example, most transitive verbs will have the associated tree with standard feature structures as in Figure 5.15. To indicate the non-terminal symbol which has a lexical item attached (termed the ANCHOR), the symbol $\diamond$ is written next to it.

Not all grammatical formalisms are naturally in a lexicalised form: CFGs, for example, are not, having rules of the form $S \rightarrow NP\ VP$. The process of lexicalising a grammar—which requires strong equivalence, that is, generating both the same language and the same tree sets—involves, for CFGs and similar formalisms, extending the domain of locality of the formalism.[5] Schabes (1988) has claimed that carrying out this process in a linguistically motivated way, so that lexical anchors heading each structure are appropriate, leads to TAGs. TAGs, because of their extended domain of locality, are naturally lexicalised.

5.3.3 Multi-Component TAG (MCTAG)

Another extension to standard TAG, first introduced in Joshi *et al* (1975) and investigated more fully in Weir (1988), is Multi-Component TAG (MCTAG). Here, the domain of locality is extended even further than for standard TAGs, by grouping trees together in a

[5] Even though, through transformation to Greibach normal form, each rule can be associated with a lexical item, this does not produce the same tree set, hence no strong equivalence.

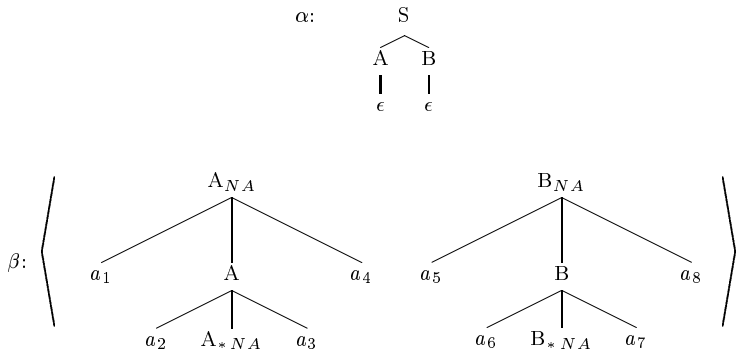


Figure 5.16: An MCTAG generating COUNT-8

SEQUENCE, with this sequence representing some relationship between trees which can be seen as a kind of locality.

Weir (1988: 32) notes that, given these sequences, there are a number of ways in which the sequences can be composed. These lead to the idea of different kinds of locality, tree-local and set-local (so termed in Frank, 1992). In tree-local MCTAG, all of the elements of a sequence are composed with a single tree. In set-local, all of the elements of a sequence are composed with the elements of another sequence. In both cases, there is a sequence with only one element, which the other must ultimately be composed into, so that the result is a single derived tree.

Tree-local MCTAG has the same strong generative capacity as standard TAG; however, there are linguistic reasons, related to the Condition on Elementary Tree Minimality (Frank, 1992), for why it might be preferred to use an MCTAG rather than a standard TAG.

Set-local MCTAG is more powerful than standard TAG. For example, the MCTAG in Figure 5.16 generates the language COUNT-8, $\{a_1^n a_2^n \ldots a_8^n \mid n \geq 0\}$, which, by the Pumping Lemma for TALs (Vijay-Shanker, 1987: 96), is beyond the generative capacity of standard TAGs. In addition, unlike for standard TAGs, the path sets of the derived trees may not be context free, and the paths are not independent.

In linguistic applications of MCTAG, the elements of the sequence are often related by LINKS (Frank, 1992). These links represent some kind of relationship between the elements such as (non-immediate) dominance or, more recently, c-command (Frank and Vijay-Shanker, 1998).

The derivation trees for MCTAGs are similar to those for TAGs: the only difference is that the derivation tree nodes are additionally annotated to identify the element of the parent sequence with which composition occurred.

Formal definitions follow, taken from Weir (1988: 33–39).[6] In this definition, an initial tree sequence is some $\alpha \in (\text{init}(V_N, V_T, V_L))^+$; an auxiliary tree sequence is some $\beta \in$

[6]Note that they are modified from the original to take account of the operation of substitution, which should allow the possibility of initial sequences of length greater than 1; also, the notation is slightly different so that it is consistent with that used for standard composition, presented in Definition 5.1.2, and later for Synchronous TAGs.

$(\text{aux}(V_N, V_T, V_L))^+$.

Definition 5.3.1 *An MCTAG is a seven tuple* $G = \langle V_N, V_T, V_L, S, \mathcal{I}, \mathcal{A}, - \rangle$ *where*

- V_N *is a finite set of non-terminal symbols;*
- V_T *is a finite set of terminal symbols, with* $V_T \cap V_N = \emptyset$*;*
- V_L *is a finite set of tree labels*
- $S \in V_N$ *is a distinguished non-terminal symbol;*
- $\mathcal{I}$ *is a finite subset of initial tree sequences;*
- $\mathcal{A}$ *is a finite subset of auxiliary tree sequences; and*
- $- : \mathcal{A} \rightarrow V_L$ *is a bijection, the auxiliary tree labelling function.*

Definition 5.3.2 *An MCTAG derived tree is well-formed if its corresponding derivation tree has as its root the label of an initial tree sequence* α *of length 1.*

The notation for composition is extended to denote multicomponent composition as follows:

$$\nabla_S(\langle \gamma_1, \ldots, \gamma_k \rangle, \quad \langle \gamma'_{1,1}, \ldots, \gamma'_{1,m_1} \rangle, \langle (i_{1,1}, a_{1,1}), \ldots, (i_{1,m_1}, a_{1,m_1}) \rangle, \ldots, \langle \gamma'_{n,1}, \ldots, \gamma'_{n,m_n} \rangle, \langle (i_{n,1}, a_{n,1}), \ldots, (i_{n,m_n}, a_{n,m_n}) \rangle)$$

That is, adjunction into $\langle \gamma_1, \ldots, \gamma_k \rangle$ occurs at the positions $(i_{1,1}, a_{1,1})$ through (i_{n,m_n}, a_{n,m_n}). Each position $(i_{p,q}, a_{p,q})$ is a pair that identifies a node with address $a_{p,q}$ in the $i_{p,q}$th tree in the sequence $\langle \gamma_1, \ldots, \gamma_k \rangle$. For each sequence $\langle \gamma'_{q,1}, \ldots, \gamma'_{q,m_q} \rangle$ being adjoined (with $1 \leq q \leq n$) there is a group of m_q positions preceding it, one for each tree in the sequence. The order of positions matches the order of trees in the sequence. Thus the pth tree in $\langle \gamma'_{q,1}, \ldots, \gamma'_{q,m_q} \rangle$ will be adjoined into $\gamma_{i_{q,p}}$ of $\langle \gamma_1, \ldots, \gamma_k \rangle$ at address $a_{q,p}$ for all $1 \leq q \leq n$ and $1 \leq p \leq m_q$.

The definition of derivation for MCTAG is then analogous to that of Definition 5.1.2, node labels becoming pairs of tree sequence names and tuples of address pairs, rather than of tree names and addresses; see Weir (1988: 34) for the original version of this definition.

5.4 Synchronous TAG (S-TAG)

Synchronous TAG (S-TAG) is, like the grammar formalisms described in Section 5.3, an extension to TAG; it is described here in more detail than the others, since one of the two foci of the thesis is to examine how to map between two paraphrase alternatives. This section looks at existing applications of S-TAG, and examines two major problems encountered.

Synchronous TAGs were introduced by Shieber and Schabes (1990) as a way of characterising correspondences between 'languages'; in the application of the 1990 paper the correspondences were between a natural language and a logical form representation. The source language and the logical form were defined by grammars in a TAG formalism, with the synchronous aspect being a link between structures of the two types such that adjunction and substitution are applied simultaneously to related nodes in pairs of trees.

Definition 5.4.1 *An S-TAG* $G = \{\langle L_i, R_i, \frown_i \rangle\}$ *is a set of triples such that*

- *there is a grammar of the left projection* $G_L = \langle V_N, V_T, V_L, S, \mathcal{I}, \mathcal{A}, - \rangle$ *such that* $L_i \in elem(V_N, V_T, V_L)$*;*
- *there is a corresponding grammar of the right projection* G_R*;*
- $\frown_i$ *is the linking relation between addresses in* L_i *and* R_i*.*

As a shorthand, when there is no ambiguity, we will generally write $G = \langle G_L, G_R, \frown \rangle$ to describe a set of triples which has G_L as the left projection and G_R as the right.

Such a grammar is intended to define a language of pairs $L(G) = \{\langle l_i, r_i \rangle\}$.

In the definition of the 1990 paper—referred to in Shieber (1994) as the 'rewriting' definition of derivation—the rewriting proceeds by choosing an initial tree pair with links $\langle I_L, I_R, \frown \rangle$ to be the current derived tree pair and repeatedly performing the following steps:

1. Choose a link $t_L \frown t_R$ between two nodes in the current derived tree pair.
2. Choose an auxiliary tree pair $\langle A_L, A_R, \frown' \rangle$ from the grammar such that A_L can be composed at t_L in I_L and A_R can be composed at t_R in I_R.
3. Modify the current derived tree pair by composing the chosen trees at the ends of the chosen link, yielding the modified derived tree pair. This becomes the new current derived tree pair.

The new link relation $\frown''$ includes all links from $\frown$ and $\frown'$, except for the link from $\frown$ chosen for the location of the composition.

The following example, with TAG trees given in Figure 5.17, is taken from Shieber (1994). By convention, we will draw S-TAGs as pairs of TAG trees, with the link relations indicated by diacritics of the form $\boxed{n}$. In α_2, the link $\boxed{3}$ connects the NP node in the left tree to the T node in the right tree;[7] both are marked for substitution here, which would leave links $\boxed{1}$ and $\boxed{2}$ in the derived tree. Tree β_1 can adjoin at $\boxed{1}$, but this raises the question of what to do with the link $\boxed{2}$. Shieber (1994) chooses to rewrite it at the root, rather than the foot, of the adjoining tree. Then the two orderings of adjoining give two different semantic representations (γ_2 and γ_3) for the same syntax tree (γ_1), as in Figure 5.18. These correspond to *int(twice(blink(john)))* and *twice(int(blink(john)))*, with differing scope for the intention. Thus S-TAGs under his definition allow multiple logical representations for a given element of the source language, a desirable feature in semantic modelling.

[7] The non-terminals in the logical form are F, R, T and Q, mnemonics for Formula, Relation symbol, Term and Quantifier.

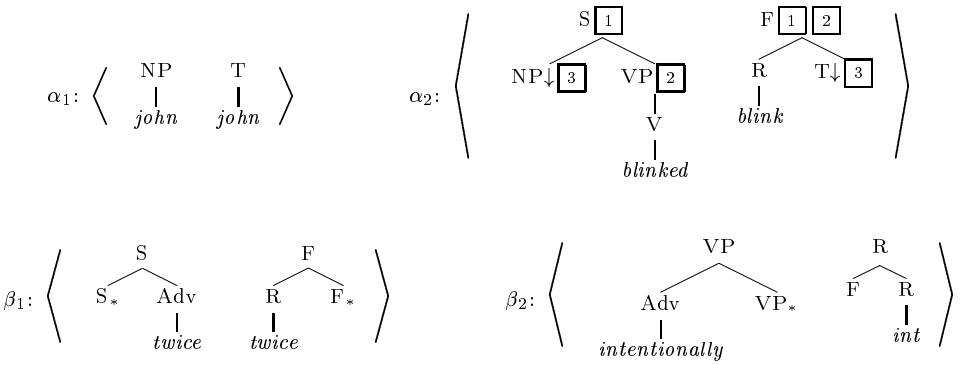


Figure 5.17: Syntax-semantics mapping: elementary tree pairs

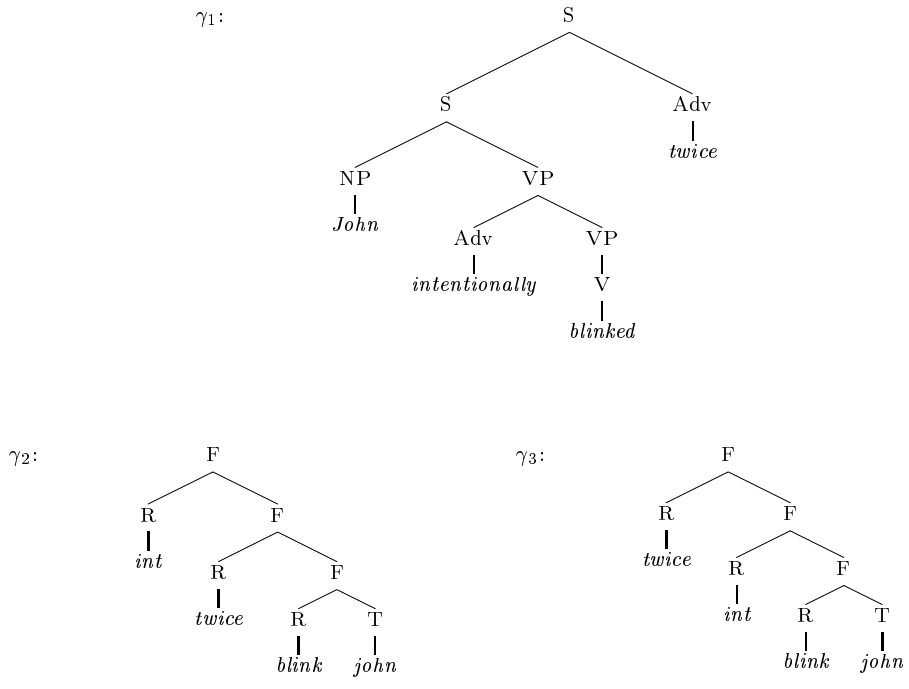


Figure 5.18: Syntax-semantics mapping: derived tree pairs

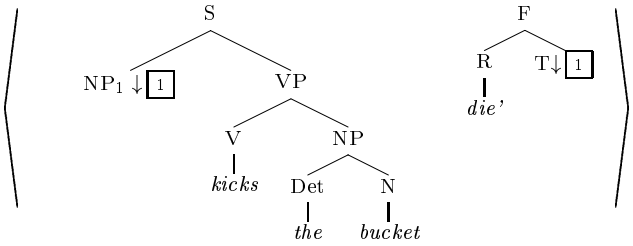


Figure 5.19: Syntax-semantics mapping: idioms

S-TAGs are also useful for semantic representations of idioms, Abeillé and Schabes (1989) noting that TAG's extended domain of locality is particularly suitable for capturing lexical aspects of idioms; for example, the idiom *kick the bucket*, used to mean *die*, as in Figure 5.19.

Abeillé *et al* (1990) discuss using S-TAGs for machine translation. In this application, there is a direct syntax-to-syntax mapping between languages, English and French. The avoidance of an explicit semantic representation is justified by the fact that the structures built by TAGs localise syntactic and semantic dependencies, and the work constitutes an investigation of the extent to which an explicit semantic level is necessary.

The tree pairs are then syntactic TAGs, again linked by a relation $\frown$. The following example is taken from Abeillé *et al* (1990). The tree pair γ in Figure 5.20 is constructed by taking pair α_1 and substituting α_2 at $\boxed{1}$ and α_3 at $\boxed{2}$, and adjoining β_1 at $\boxed{3}$. In α_1 the links allow for different argument positions in French and English: subject and object are reversed in the transfer. Also, the appropriate tree pair cannot be chosen until after the lexical item has been instantiated: subject and object are not always reversed, so a different transitive tree pair will be appropriate for other verbs.

Most often, in machine translation, the matching will be between pairs of elementary trees, with the link relations matching arguments of predicates—in Figure 5.20, these argument links are $\boxed{1}$ and $\boxed{2}$. The trees are instantiated to sets of morphological variants of a word, with the set for the lexeme *word* conventionally represented as $\backslash word \backslash$. Thus, the tree pair α_1 is represented as α_1' in Figure 5.21.

However, it is not always the case that the mappings will occur between elementary trees. Sometimes a single word will correspond to a phrase, or there will be some other discrepancy, leading to a tree pair containing an elementary tree and a derived tree. Abeillé *et al* (1990) mention two kinds, discrepancies in constituent structures and discrepancies in syntactic properties. As an example of the former they give (1) and (2).

(1) a. The baby just fell.
b. Le bébé vient de tomber.

(2) a. John gave a cough.
b. John toussa.

These are represented by the tree pairs in Figure 5.22. Note that $\boxed{2}$ in α links the noun *cough* with the verb *tousser*: this allows for modification of the act of coughing to be

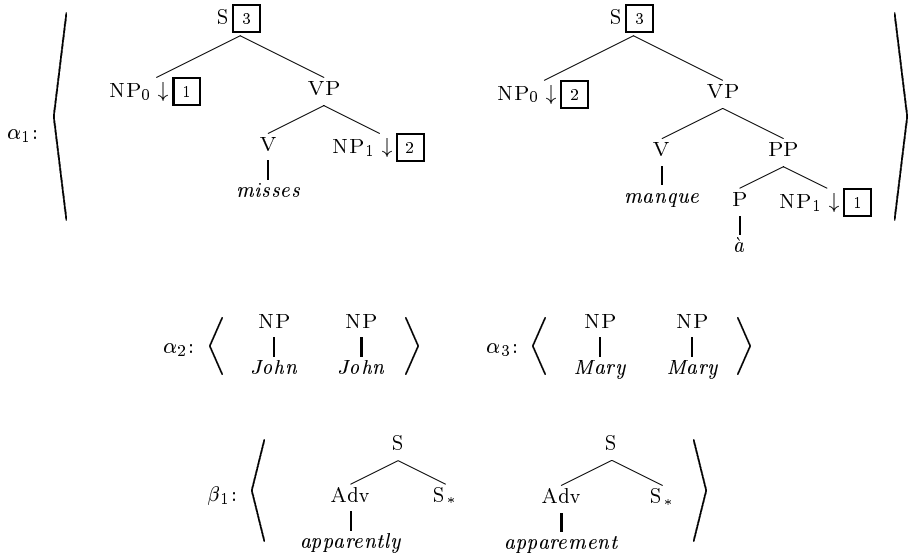


Figure 5.20: Machine translation using synchronous TAG

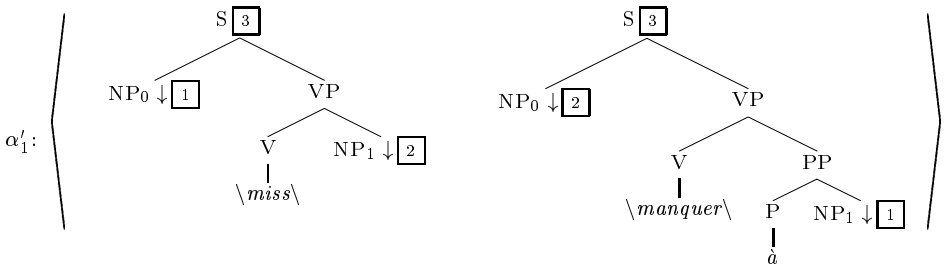


Figure 5.21: Machine translation using synchronous TAG: lemmatised forms

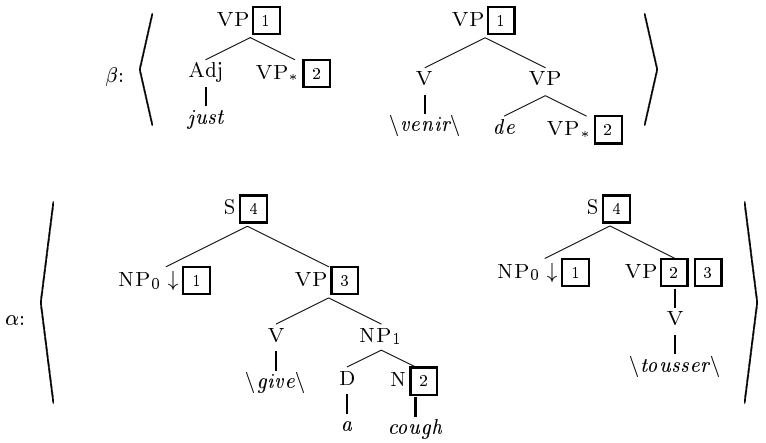


Figure 5.22: Machine translation involving different constituent structures

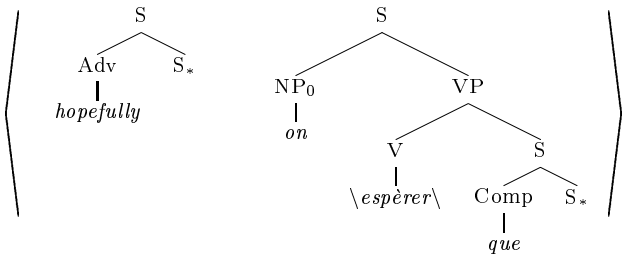


Figure 5.23: Machine translation involving different syntactic properties: translation pair

represented in ways syntactically appropriate to the language, through adjunction of an adjective to the English structure and an adverb to the French, giving sentences as in (3).

(3) a. John gave a weak cough.
b. John toussa faiblement.

As an example of discrepancies in syntactic properties, they give (4), with the qualifications of hopefulness given by the tree pair of Figure 5.23. The left tree, the English version, is represented by an elementary tree; the right tree, the French version, is a derived tree, consisting of at least separate trees for *on* and *espère que*: trees for the sentences in (4) are given in Figure 5.24, with trees β' and α'_3 comprising the derived French tree of Figure 5.23.

(4) a. Hopefully, John will work.
b. On espère que John travaillera.

In these pairs of sentences, the tense and other agreement features are shared. Abeillé *et al* do this in the transfer lexicon, through, for example, a transfer rule as below.

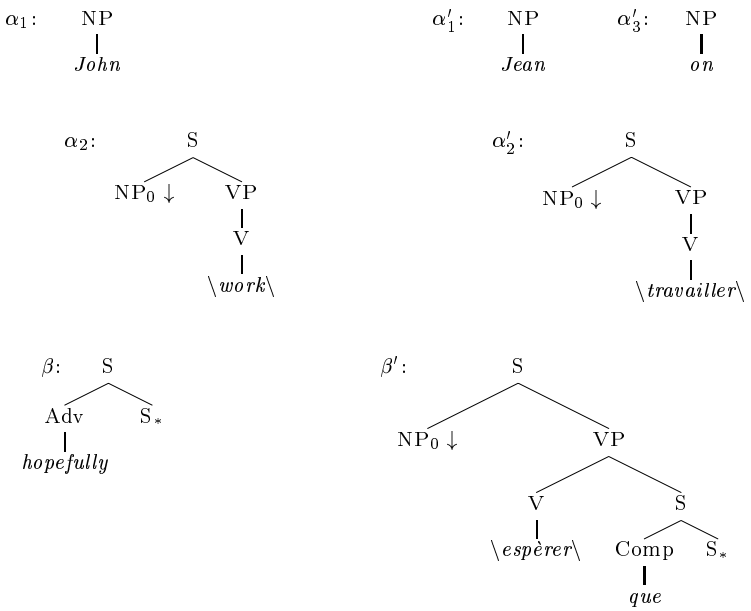


Figure 5.24: Machine translation involving different syntactic properties: elementary trees

\boy\, N [num=X] ↔ \garçon\, N [num=X]

In this rule, the agreement features are matched. In cases where the entries are not morphological sets, but are lexically fixed to account for idiosyncrasies, the rule is of the form

\luggage\, N [num=sing] ↔ \bagages\, N [num=pl]

Agreement feature structures of verbs are not matched, to allow for this type of idiosyncrasy; unification of feature strutures will ensure correct agreement.

The transfer of features attached to the sentential root nodes depends on how they are assigned in the target language. The tense feature is generally tranferred, but mode is not, as it depends on the verb of the matrix sentence if the sentence is embedded, as in (5).

(5) a. Jean wants Marie to leave.

b. Jean veut que Marie parte.

All of this work used the rewriting definition of S-TAGs—taking a derived tree and performing tree composition sequentially. However, Shieber (1994) showed that the expressive power of the formalism is then greater than for TAGs. The example Shieber used to demonstrate this uses the tree pairs α, β_1 and β_2 of Figure 5.25.

This generates the non-tree adjoining language $\{a^n b^n c^n d^n e^n f^n g^n h^n \mid n \geq 0\}$. Consequently Shieber proposed an alternative definiton for S-TAGs based on derivation trees

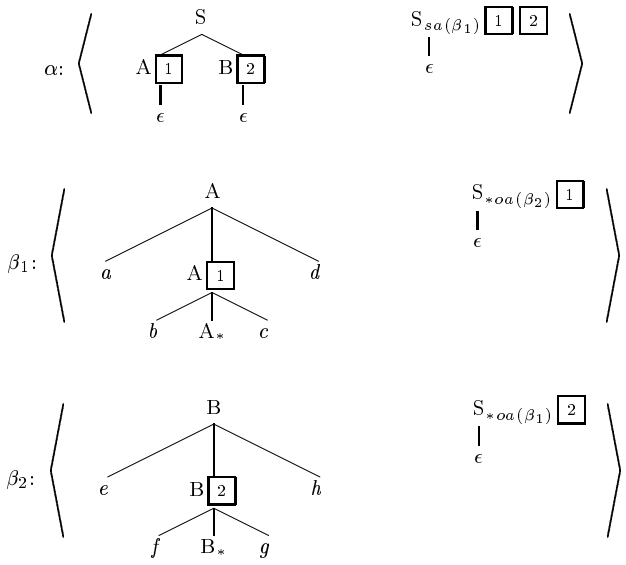


Figure 5.25: A grammar generating a non-TAL under the rewriting definition of S-TAG

such that the weak generative capacity of S-TAGs is only that of TAGs, noting that a derivation tree specifies a derived tree by virtue of the operations of adjunction and substitution, as in Definition 5.1.3. Shieber's 'natural definition of derivation' for an S-TAG then uses the following definition.

Definition 5.4.2 *A derivation for an S-TAG G is a pair $\langle D_L, D_R \rangle$ where*

1. *D_L is a well-formed derivation tree relative to G_L, that is, $D_L \in T_D(G_L)$.*
2. *D_R is a well-formed derivation tree relative to G_R, that is, $D_R \in T_D(G_R)$.*
3. *D_L and D_R are isomorphic. That is, there is a one-to-one onto mapping f from the nodes of D_L to the nodes of D_R that preserves dominance, i.e. if $f(\eta_l) = \eta_r$, then $f(\text{parent}(\eta_l)) = \text{parent}(\eta_r)$.*
4. *The isomorphic operations are sanctioned by links in the tree pairs. That is, if $f(\eta_l) = \eta_r$, then there is a tree pair $\langle (\eta_l)_L, (\eta_r)_L, \frown' \rangle$ in G. Furthermore, if $(\eta_l)_R \neq \epsilon$, then there is a tree pair $\langle \text{parent}(\eta_l)_L, \text{parent}(\eta_r)_L, \frown \rangle$ in G and $(\eta_l)_R \frown (\eta_r)_R$.*

The definitions of $T_D(G)$ and TAG treeyield functions will be extended to S-TAG derivation pairs as the set of derivation tree pairs, and treeyield functions applied to the pair elements, respectively.

S-TAG under this definition has the WEAK LANGUAGE PRESERVATION PROPERTY as defined in Rambow and Satta (1996), that is, the defined synchronisation mechanism does not alter the weak generative capacity of either of the formalisms being synchronised.

With the modified definition of derivation of Schabes and Shieber (1994), the treatment of semantic ambiguity through multiple adjunctions at a single node, as in Figure 5.17, is

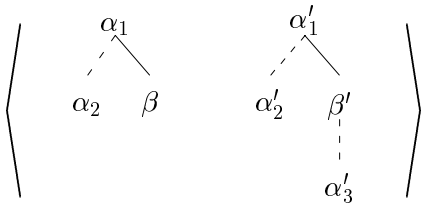


Figure 5.26: Non-isomorphic mapping

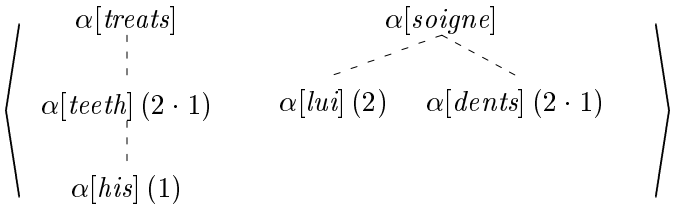


Figure 5.27: Non-isomorphic mapping

not problematic. However, problems are caused by the isomorphism requirement, which Shieber (1994) believes may be too strong. For (6), a modification of (4), the mapping between derivation trees node for node would not be isomorphic.

(6) a. Hopefully, John works.

b. On espère que John travaille.

TAG trees for the constituents of these sentences are given already in Figure 5.24. The derivation tree pair corresponding to this translation is given in Figure 5.26. Here the mapping from the right derivation tree to the left is not one-to-one, with both β' and α'_3 mapping onto β.

An example demonstrating problems with preserving dominance is given in (7).

(7) a. The doctor treats his teeth.

b. Le docteur lui soigne les dents.

This leads to a partial derivation tree pair, taken from Shieber (1994), of the form in Figure 5.27; here the actual structures of the trees are different, because the clitic *lui* is substituted directly into the verb tree, while the possessive pronoun *his* is substituted into *teeth*.

Another example, demonstrating inversion of dominance, is given in (8).

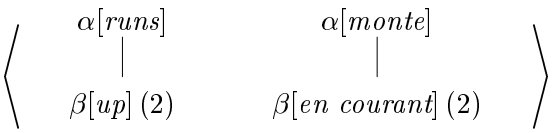


Figure 5.28: Non-isomorphic mapping

(8) a. Jean monte la rue en courant.

b. Jean runs up the street.

This gives the partial derivation tree pair in Figure 5.28; dominance is inverted, as $\alpha[monte]$ maps to $\alpha[up]$, and $\alpha[en\ courant]$ to $\alpha[runs]$.

A proposed solution is to allow 'bounded subderivation'—such as for *on espère que*—to be considered elementary for the purposes of the isomorphism requirement. This has the potential to solve the problem for the mappings in Figures 5.26 and 5.28: β' and α'_3 of Figure 5.27 can be treated together as a bounded subderivation, as can $\alpha[monte]$ and $\beta[en\ courant]$, or $\alpha[runs]$ and $\beta[up]$. However, Shieber (1994) notes that this is not the case in Figure 5.27: the structures are different, and because of the nature of clitics there can be arbitrarily many intervening nodes between $\alpha[treats]$ and $\alpha[his]$, making it impossible to specify a bijection preserving dominance for the grammar as a whole.[8] This means that, while neat, S-TAG is currently inadequate for describing mappings in machine translation: what constitutes a valid bounded subderivation is not precisely defined, and even if it were, there is still the problem with clitics.

The idea behind synchronous TAG, that of synchronised parallelism of grammars, has also been used in the modelling of coordination in TAG. Coordination poses difficulties for TAG, with a number of analyses proposed (Joshi, 1990; Joshi and Schabes, 1991; Sarkar and Joshi, 1996; Sarkar, 1997). The last of these (Sarkar, 1997) notes that previous attempts are somewhat inelegant, and do not follow the spirit of the formalism; sentences such as (9) and (10) either end up with unrooted trees or require structure merging on derived trees.

(9) Kiki frolics, sings and plays all day.

(10) Kiki likes and Bill thinks Janet likes soccer.

According to Sarkar, the difficulties occur because TAG conflates the ideas of constituency and dependency: the derivations do not, for coordination under these earlier approaches, reflect dependency of argument structure appropriately. Sarkar thus proposes splitting the representation of constituency and dependency, but keeping them linked via synchronisation. He defines a variant of S-TAG, Link Sharing TAG (LSTAG), which divides links into two disjoint sets Δ and Φ, where links from Φ are inherited, as under the rewriting

[8] Clitics are more extensively analysed from a TAG perspective, albeit one using the rewriting definition of S-TAG, in Abeillé (1992).

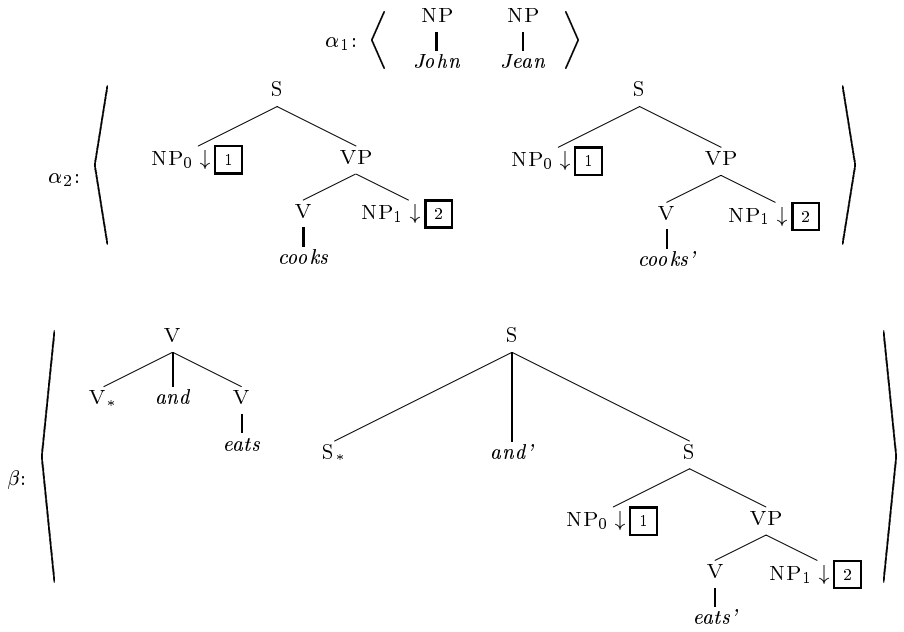


Figure 5.29: Coordination using LSTAG: elementary tree pairs

definition of S-TAG, but only for one composition, while those from Δ are not inherited. For example, in describing (11), the tree pair in Figure 5.29 could be used. In α, the links are both in Δ (that is, they are not inherited), while those in β are both in Φ (that is, they are inherited for the single adjunction of β into α).

(11) John cooks and eats beans.

Adjoining β into α gives derived and derivation trees as in Figures 5.30 and 5.31, which is not problematic. However, if an NP-rooted tree, such as α_1 from Figure 5.29, is substituted at the subject NP slot, the result is as in Figures 5.32 and 5.33. Sarkar refers to this as a 'tangled' tree, although it is in fact no longer a tree: the derivation 'tree' has become a directed acyclic graph, which poses particular difficulties given that the definition of S-TAG, Definition 5.4.2, depends on an isomorphism between well-formed derivation trees. It is also unclear what the formal properties of such a representation are.

Thus use of S-TAG is also problematic in coordination, with formal properties no longer clearly defined; this is also likely to be a problem in paraphrase, given the parallels between coordination and certain paraphrases, notably those involving multiple sentences (Katz and Fodor, 1963).

5.5 Suitability of TAG for Paraphrasing

There are, of course, other formalisms capable of describing paraphrases as well. This thesis does not propose an exhaustive analysis of the suitability of various formalisms for describing paraphrase; it is, in part, an exploration of how a specific family of formalisms,

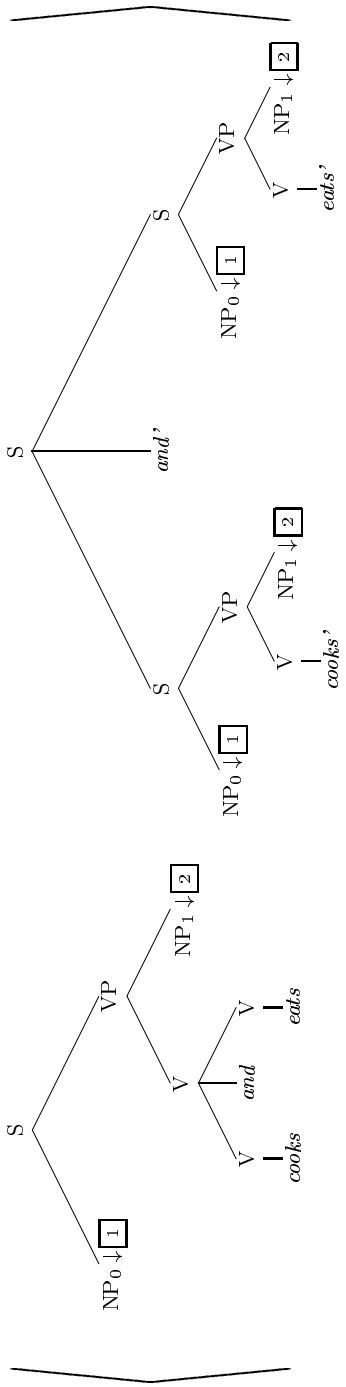


Figure 5.30: Coordination using LSTAG: derived tree pair

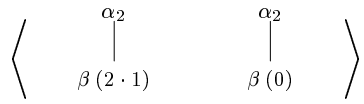


Figure 5.31: Coordination using LSTAG: derivation tree pair

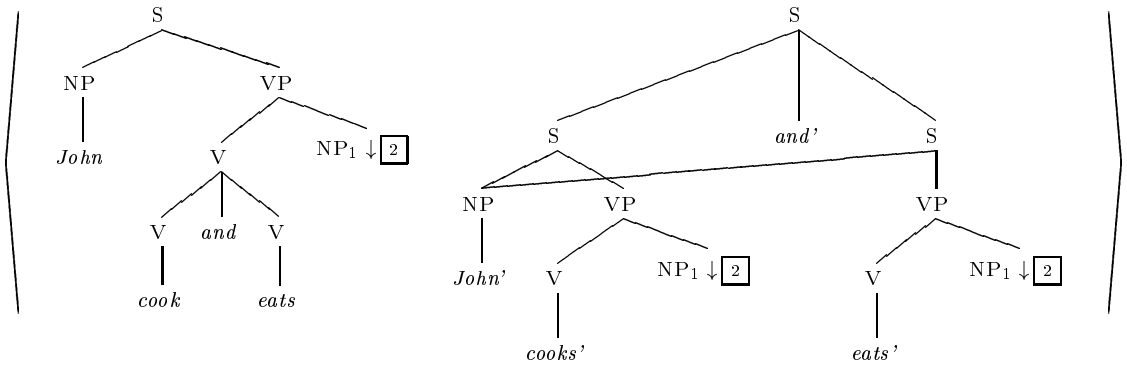


Figure 5.32: Coordination using LSTAG: derived tree pair

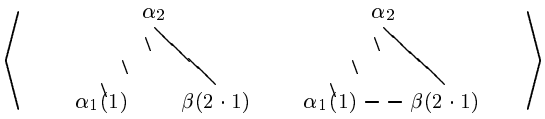


Figure 5.33: Coordination using LSTAG: derivation tree pair

TAG and its extensions, can be used to represent paraphrase, and what this exploration uncovers about the nature of TAG. However, an obvious alternative candidate for representing paraphrase, old though it is, is Transformational Generative Grammar—which unlike a number of other syntactic theories does have specific mechanisms for describing the relationships between interchangeable sentences, and which in fact holds this as a central aspect—which poses the question of why it should be of interest to look at another representation formalism. This section will briefly review problems with Transformational Generative Grammar, and, following from this, argue why TAG is potentially suitable for describing paraphrase.

Transformational Generative Grammar (TGG) in its original form was based on the work of Chomsky (see particularly Chomsky, 1957; Chomsky, 1965). The formalism was based on Phrase Structure Grammar, generally using CFGs[9] to give the underlying syntactic structure of a sentence, with a transformational component which was responsible for, in some sense, changing one structure into another. Three common functions have, at various stages, been performed by the transformational component: rules relating particular sentence types to each other, such as active and passive voice counterparts; rules that account for various morphological operations, such as number agreement between subject and verb; and rules for generating complex sentences. The first and third of these can be seen as covering essentially the ground necessary for the paraphrases described in Chapter 4 of the thesis: in other words, there is potentially a ready-made theory of paraphrase, with the relationship between paraphrase pairs described by transformations.

A typical transformation, producing the passive voice form of an active sentence, might look as below (Malnikjær, 1991: 485; modified from the original in Chomsky, 1957: 112).

SA: NP - Aux - V - NP

SC: X1 - X2 - X3 - X4 $\Rightarrow$ X4 - X2 + $_{\text{Pass}}$(BE + *en*) - X3 - ($_{\text{PP}}$by - X1)

The first part of the transformation, labelled **SA**, is the STRUCTURAL ANALYSIS, the form that the input sentence must match in order for the transformation to apply. The second part, labelled **SC**, is the STRUCTURAL CHANGE, specifying how the output structure is formed from the input structure. A given transformation could be either obligatory or optional: clearly, those related to paraphrase, as the passive transformation above, would be optional.

In terms of the suitability of TGG for describing paraphrase, there are three reasons for adopting an alternative approach in this thesis, related to constrainedness of the formalism, the types of structures used for mappings, and bidirectionality.

In terms of the first of these, an underlying problem with TGG is that it is unconstrained in formal power, as demonstrated by Peters and Ritchie (1973). As mentioned at the start of the chapter, whether a more or less constrained formalism is chosen is to some extent a matter of situation or of personal preference. It is the case that constrained formalisms may not express generalisations easily, and hence may require additional mechanisms to do this: an instance in TAG is that of tree families. Take as an example transitive verbs. A typical transitive verb will select the standard α**nx0Vnx1** tree, the passive α**nx1Vbynx0** tree, relative clause trees, and so on. There is no way in standard TAG of underspecifying a tree to capture the essential transitivity of a verb: the definition of the formalism, with initial

[9] Some variants used CSGs as the base formalism.

and auxiliary trees and their rules of composition, requires that there be individual trees fulfilling each function. Thus additional mechanisms are used to capture the generalisation, either tree families, or various approaches to constructing trees from parts—quasi trees (Vijay-Shanker and Schabes, 1992), DATR (Evans *et al*, 1995), hierarchical approaches (Candito, 1998), partial tree descriptions (Xia *et al*, 1998)—under which the common elements of a set of trees—say, the parts that express the essence of transitivity—form a base, and the individual trees are constructed from this base. However, unconstrained formalisms too require additional mechanisms; for them these mechanisms disallow invalid structures or analyses. Mechanisms such as Principles or Rules effectively constrain the formalism so that structures which do not occur in language are disallowed; take for example the Trace Principle of HPSG (Pollard and Sag, 1994: 172), which constrains the formalism through its requirement that every trace must be subcategorised by a substantive head, a condition which disallows certain structures for violating island constraints (*ibid*: 202). In this thesis, a constrained formalism is used, partly in the belief that this is more in keeping with the original spirit of generative linguistics: the original Chomsky hierarchy of grammar formalisms was, after all, developed to find a formalism that was expressive enough to describe natural language, but not overly expressive. Computationally, it is desirable to have a more constrained formalism so that complexity of associated algorithms is minimised, or at least has an asymptotic upper bound, a consideration which is the focus of much of the theoretically-oriented literature on parsing.

There have been attempts to constrain the power of transformations to this end. One such was the structure preserving constraint of Emonds (1976), which proposed that a transformation should be able neither to create nor destroy structure, but only to move lexical material around within already established structures. This ensured that the deep structure must have some lexicalised nodes, moving the formalism closer to the idea of lexicalised S-TAG, in much the same way that the lexicalisation of CFG, the base formalism, can be seen to produce TAG (Schabes, 1988).

A second reason for adopting an alternative approach is that it is clear that there is a discrepancy between the idea of transformational rules and the idea, encapsulated in the term 'structural change', of mapping between structures. The rules are flat and linear; however, their intent is to take entire substructures (say, the subtree correponding to the subject NP of a sentence) and place them in the resulting output structure. That this is the intent can be seen by the grouping of (PPby - X1) in the **SC** part, to form a PP substructure; but flat rules are not the natural environment for this. A much more natural way is to map between trees: that is, exactly what occurs in S-TAG. There are ways to extend TGG so that the mappings occur between trees, using a tree transformation language, and there is even standard software to do this, such as the XSLT tree transformation language for XML. However, given that S-TAG is already fundamentally based on trees, and given that its components and composition operations are already well-defined and amenable to formal analysis, this alternative seems at least as good on this criterion.

Thirdly, S-TAG has an inherent symmetry that TGG does not, so that the paraphrases are bidirectional. In a TGG transformation definition, a structure is constructed and operated on, and a paraphrase string produced. In S-TAG structures are built simultaneously, and there is no built-in direction.

Other formalism-related problems are not so relevant here. For example, in TGG difficulties with cycling in the ordering of constraints is not relevant here, as, because of the

Reluctant Paraphrase application, only one paraphrase is ever applied to a particular sentence; this also means that it is similarly unproblematic that the application of multiple paraphrases in S-TAG is currently undefined.

Note again that other formalisms also potentially have scope for representing paraphrases. For example, the HPSG formalism incorporates Lexical Rules (Pollard and Sag, 1994: 121), which are used to model active-passive paraphrases, among others. Using the active-passive case as an example, Pollard and Sag describe the mapping as a permutation of the subcategorisation lists of active transitive verb forms. For the verbs *consider* and *rate*, the permutation might look as follows:

(12) a. SUBCAT ⟨ NP, NP, XP[+PRD] ⟩ (consider, rate)

b. SUBCAT ⟨ NP, XP[+PRD], PP[*by*] ⟩ (considered, rated)

However, just as TGG is, this formalism is unconstrained, whereas TAG is constrained; and the rules are also flat, so, if for example the PP in (12b) is to be given any structure, it is done elsewhere in the appropriate feature structure. So the characteristics of TAG described above again lead us to consider that formalism: given the description in Sections 5.1 through 5.3, and the contrast with TGG, TAG appears to be a potentially useful formalism for describing paraphrases.

- TAG has the benefits of a precise mathematical formalism, and moreover one with constrained expressive power. Chapters 6 and 7 will investigate the level of expressive power required by different aspects of paraphrasing.
- There is an extension to TAGs, S-TAGs, which can be used to characterise mappings between TAG trees; and specifically syntax-to-syntax mapping as in the machine translation work of Abeillé *et al* (1990). This has parallels with the paraphrases discussed in Chapter 4—these are syntactically-based paraphrases, and so the machine translation machinery appears to be applicable to paraphrase, albeit possibly with modification.
- The extended domain of locality of TAGs, when used in TAG pairs, means that a structural description is assigned to both elements of the TAG pair. The resulting paraphrase will thus have an associated syntax tree, which is what TGG did only partially and in a way that did not really fit with the formalism; and which is moreover useful where the textual constraint may depend on this.
- There is a large-scale standard grammar and parser for English, the XTAG system (XTAG, 1995). This provides a standard library of trees to use to precisely define the paraphrases described in Chapter 4.

As stated at the start of this section, this thesis is, in part, an exploration of how paraphrase can be represented in TAG; it is also, conversely, an investigation of issues in TAG using paraphrases as a mechanism for this investigation. As noted, S-TAG has the potential to represent paraphrases, but there are aspects of the formalism which present difficulties for other applications, as discussed in Section 5.4, which are likely to be encountered in the representation of paraphrase. Chapter 6 investigates the representation of paraphrases in S-TAG, and Chapter 7 presents generalisations of the formalism which

provide a way of resolving the aforementioned difficulties evidenced both in paraphrase and in other applications, which arise from the fundamental nature of S-TAG as currently defined.

Chapter 6

Paraphrases and Synchronous TAG

The choice of Synchronous TAG (S-TAG) as a formalism for representing paraphrases was motivated by the recognition of the closeness of existing applications of the formalism, in particular of syntax-to-syntax machine translation, to the paraphrases discussed in Chapter 4. This chapter proposes a paraphrase representation within the S-TAG formalism. In this representation, referred to as S-TAG UNDER PARAPHRASE, the general idea is that each paraphrase pair is represented by S-TAG pairs including one predefined S-TAG pair specifying the structural rearrangement caused by the paraphrase; such predefined S-TAG pairs are called Structural Mapping Pairs (SMPs). For elements that are not part of the structural rearrangement—so they stay fundamentally the same—the S-TAG pairs can, on the other hand, just be generated from single trees; the functions that create a pair from a tree are called generating functions. An S-TAG grammar under paraphrase will thus consist of zero or more SMPs, and one or more generating functions (with this minimum generating function being the one that maps trees to pairs of identical trees).

This chapter in particular explores the differences from the use of S-TAGs in machine translation; these differences are predominantly due to the fact that tree pairs under paraphrase, by their nature, tend to have more complex derived structures. This more complex nature of paraphrase representation draws out the reasons behind the problematic aspects of S-TAG discussed in Sections 5.4 and 5.5, and leads to generalisations of S-TAG in Chapter 7 where these problems are resolved. In exploring these issues, this chapter also presents formal results showing the well-formedness of S-TAG under this paraphrase representation.

6.1 Fundamental Well-Formedness of S-TAG

While it is widely believed, given the argument in Shieber (1994), that the 'natural' definition of S-TAG (reproduced in Definition 5.4.2) has the weak language preservation property—that is, that the synchronisation does not change the properties of either component grammar in terms of the string languages generated—we note that it has not been formally proven that this is the case. In particular, it is not entirely clear exactly how Shieber's argument does demonstrate that the WLPP holds. The argument asserts that

it is possible to construct for any S-TAG a 'tree-set-local' MCTAG, where this appears to be a conflation of two version of MCTAG, tree-local and set-local, which have different formal properties. The argument then appeals to the fact that tree-local MCTAG is not greater in generative capacity than standard TAG; but the construction, in relabelling the nodes of the trees of one projection so that trees must compose into different elements of a sequence, appears to be set-local, which does have greater generative capacity than standard TAG.

Hence, before giving any formal results on paraphrase, we first prove that S-TAGs under Definition 5.4.2 have the weak language preservation property. In addition, the method of proof given here allows for generalisation of the idea of synchronisation of grammars, as discussed in Section 7.1.

Shieber's condition, and the key to the WLPP holding under the natural definition of S-TAG, is that the derivation trees are isomorphic, that is, that the tree structure stays the same, but the labels change. If it were the case that synchronisation paired all trees, demonstrating that the WLPP holds would be straightforward. However, this is not in general the case, as exemplified by Figure 6.1 with the S-TAG $G = \{\langle G_L, G_R, \frown\rangle\}$, where the grammars of the left and right projections are $G_L = \langle V_N, V_T, V_L, S, \mathcal{I}_L, \mathcal{A}_L, -\rangle$ and $G_R = \langle V_N, V_T, V_L, S, \mathcal{I}_R, \mathcal{A}_R, -\rangle$ respectively.

In this figure, $\mathcal{I}_L \cup \mathcal{A}_L = \{\alpha\text{S}, \alpha\text{N}, \alpha\text{D}, \beta\text{N}\}$, and
$\mathcal{I}_R \cup \mathcal{A}_R = \{\alpha\text{S}, \alpha\text{N}, \alpha\text{D}, \beta\text{N}, \alpha\text{NN}, \beta\text{NN}, \alpha\text{NNN}\}$.[1] The language of the left projection, $L(G_L)$, allows embedding of infinitely many NPs; the language of the right projection, $L(G_R)$, allows maximum embedding of two levels. Clearly, it is not possible for there to be a bijection between $T_D(G_L)$ and $T_D(G_R)$, as the former is infinite and the latter finite (of cardinality 9). In general, $(T_D(G))_L \subseteq T_D(G_L)$, and the same for the right projection; that is, the S-TAG pairing means that not all well-formed derivation trees in $T_D(G_L)$ are in the left projection of $T_D(G)$. Thus, we need to be able to discuss mappings between sets of trees, in much the same way as mappings between sets of strings, and show that if one set of trees has particular properties that lead to a given string language, then so does the subset that generates the paired language.

The key to S-TAG having the WLPP is that the derivations of TAGs are local sets, and we are mapping between them. Consequently, we introduce the notion of finite-state tree automata (Gécseg and Steinby, 1984), which can be used to prove results about tree languages in much the same way that ordinary finite automata can be used to prove results about string languages.

A tree over Σ can be defined as a Σ-TERM in the following way. For some tree $\gamma : \mathbf{N}_+^* \to \Sigma$

- if γ has domain $\{\epsilon\}$ (that is, is a node of one tree), then the corresponding Σ-term is $t = \gamma(\epsilon)$; otherwise
- if γ has domain $\bigcup_{i \leq m}\{i \cdot w \,|\, w \in \text{dom}(\gamma_i)\} \cup \{\epsilon\}$, $\gamma(\epsilon) = \sigma$, and t_i is the Σ-term for γ_i for $1 \leq i \leq m$, then the Σ-term for γ is $\sigma(t_1 \ldots t_m)$.

The set of Σ-terms for some Σ is T_Σ.

As an example, the Σ-term corresponding to $\alpha\mathbf{S}$, the left tree of α_1, in Figure 6.1 is

[1] The individual trees are from the standard TAG inventory, with some modifications (the +N and ++N categories) and simplified names.

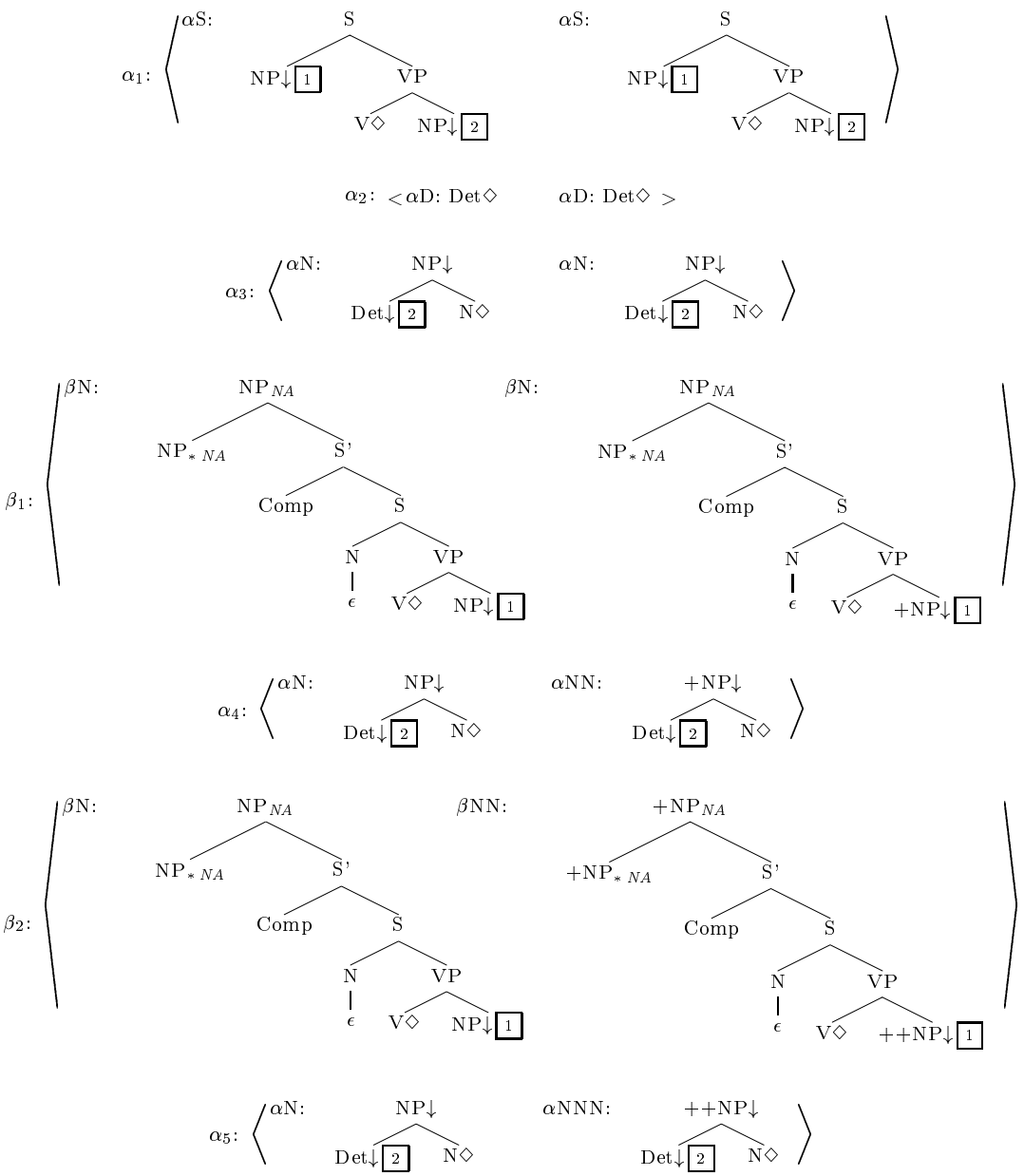


Figure 6.1: Pairing infinite and finite languages

S(NP VP(V NP)).[2] Henceforth this definition of a tree as a term will be used interchangeably with the definition of a tree from Chapter 5.

The following specific characterisation of tree automata is taken from Rogers (1994). A NON-DETERMINISTIC BOTTOM-UP FINITE STATE TREE AUTOMATON over Σ-terms is a tuple $\langle \Sigma, Q, M, F \rangle$ where

- Σ is a finite alphabet,
- Q is a finite set of STATES,
- F is a subset of Q, the set of FINAL STATES, and
- M is a partial function from $\Sigma \times Q^*$ to the powerset of Q with finite domain, the TRANSITION FUNCTION.

The transition function M associates sets of states with alphabet symbols. It induces a function that associates sets of states with trees, $\bar{M} : T_\Sigma \rightarrow 2^Q$, such that for some $t \in T_\Sigma$, $q \in \bar{M}(t)$ if and only if

- t is atomic with label σ and $q \in M(\sigma, \epsilon)$, or
- $t = \sigma(t_0 \ldots t_n)$ and there is a sequence of states $q_0, \ldots, q_n$ such that $q_i \in \bar{M}(t_i)$ for $0 \leq i \leq n$, and $q \in M(\sigma, q_0 \ldots q_n)$.

An automaton $\mathbf{A} = \langle \Sigma, Q, M, F \rangle$ accepts a tree $t \in T_\Sigma$ if and only if, by definition, $F \cap \bar{M}(t) \neq \emptyset$. The set of trees accepted by an automaton $\mathbf{A}$ is denoted by $T(\mathbf{A})$.

A set of trees is RECOGNISABLE if and only if, by definition, it is $T(\mathbf{A})$ for some automaton $\mathbf{A}$. This is the analogue of a regular set of strings, and has close connections with this (Gécseg and Steinby, 1996); the automaton is clearly similar to an ordinary finite automaton.

Now, the local sets can be related to the recognisable sets recognised by tree automata as follows.

Lemma 6.1.1 *(Thatcher, 1967) Every local set is recognisable. Every recognisable set is the projection of some local set.*

Rogers (1994) notes (expanding on Thatcher) that the projection is necessary because the automaton can distinguish between nodes labelled with the same symbol while the CFG (the derivations of which constitute the local set) cannot. The set of trees, with bounded branching, in which exactly one node is labelled A, for instance, is recognisable but not local. It is, however, the projection of a local set in which the labels of the nodes that dominate the node labelled A are distinguished from the labels of those that don't.

We now want to define the sort of mapping that selects those trees in one set that have the same structure (that is, domain) as at least one tree in another set. First we define an ALPHABETIC RELABELLING RELATION R as one where $R \subseteq \Sigma \times \Omega$, and extend this to trees expressed as terms as follows:

[2] Note that in this example, with TAG trees, we are ignoring **oa** and **sa** constraints, which by default for all of these nodes are **false** and $\mathcal{I} \cup \mathcal{A}$ respectively.

- $t_\Sigma R t_\Omega$ if $t_\Sigma \in T_\Sigma$ and $t_\Omega \in T_\Omega$ are atomic, and
- $\sigma(t_{\Sigma,1} \ldots t_{\Sigma,m}) R \omega(t_{\Omega,1} \ldots t_{\Omega,m})$ if $\sigma R \omega$ and $t_{\Sigma,i} R t_{\Omega,i}$ with $\sigma \in \Sigma$, $\omega \in \Omega$, $t_{\Sigma,i} \in T_\Sigma$, and $t_{\Omega,i} \in T_\Omega$, for $1 \leq i \leq m$.

Then we can define

$$\mathrm{RelImg}(R, T_\Sigma, T_\Omega) = \{t_\Omega \in T_\Omega \mid \text{if for some } t_\Sigma \in T_\Sigma, t_\Sigma R t_\Omega\}$$

This is the relational analogue of an inverse homomorphic image: all those trees in T_Ω which under R produce trees in T_Σ. We can now show the following result, along the lines of closure under inverse homomorphism.

Lemma 6.1.2 *Given an alphabetic relabelling relation R and two alphabets Σ and Ω, and given sets of terms over these alphabets T_Σ and T_Ω with T_Σ recognisable, then* $\mathrm{RelImg}(R, T_\Sigma, T_\Omega)$ *is recognisable.*

Proof: Let $\mathbf{A} = \langle \Sigma, Q, M, F \rangle$ be a tree automaton recognising T_Σ, and R an alphabetic relabelling relation. We construct a (non-deterministic) tree automaton $\mathbf{A}'$ that accepts $\mathrm{RelImg}(R, T_\Sigma, T_\Omega)$ by reading a symbol t_Ω in Ω and simulating $\mathbf{A}$ on t_Σ, where $t_\Sigma R t_\Omega$.

Formally, let $\mathbf{A}' = \langle \Omega, Q, M', F \rangle$ and, for $q \in Q$ and $\sigma \in \Sigma$, define M' so that the function it induces, $\bar{M}' : T_\Omega \to 2^Q$ is as follows: $q \in \bar{M}(t)$ if and only if

- t is atomic with label ω and $q \in M'(\omega, \epsilon)$, or
- $t = \omega(t_0 \ldots t_n)$ and there is a sequence of states $q_0, \ldots, q_n$ such that $q_i \in \bar{M}'(t_i)$ for $0 \leq i \leq n$, and $q \in M'(\omega, q_0 \ldots q_n)$.

where $\sigma R \omega$ for some $\omega \in \Omega$. By induction on the number of steps, $\bar{M}'(t_\Omega) = \bar{M}(t_\Sigma)$, where $t_\Sigma \in \Sigma$, $t_\Omega \in \Omega$, and $t_\Sigma R t_\Omega$; and hence $\bar{M}'(\omega, d) = \bar{M}(\sigma, d)$ for $d \in Q^*$. Therefore $\mathbf{A}'$ accepts t_Ω if and only if $\mathbf{A}$ accepts t_Σ. □

The automaton here is non-deterministic because R is a relation; this, rather than a deterministic automaton and a homomorphism, is necessary in showing the result in cases such as the mapping of the left projection of Figure 6.1 to the right projection, deriving $\mathrm{RelImg}(R, T_D(G_R), T_D(G_L))$.

Theorem 6.1.3 *S-TAGs, under Definition 5.4.2, have the weak language preservation property.*

Proof: Let $G = \langle G_L, G_R, \frown \rangle$ be an S-TAG for some TAGs G_L and G_R, let $\mathcal{D}$ be a TAG treeyield function such that $\mathcal{D}(T_D(G_L)) = T(G_L)$ and $\mathcal{D}(T_D(G_R)) = T(G_R)$, and let R be an alphabetic relabelling relation.

$T_D(G_L)$ is a local set, by Definition 5.4.2 and Lemma 5.2.2, and hence recognisable by Lemma 6.1.1.

$\mathrm{RelImg}(R, T_D(G_L), T_D(G_R))$ is recognisable by Lemma 6.1.2, and hence so is $\mathrm{RelImg}(R, \mathrm{RelImg}(T_D(G_L), T_D(G_R)), T_D(G_L))$, which is in fact $(T_D(G))_L$. Therefore $\mathcal{D}((T_D(G))_L)$ is a TAG, and $L(\mathcal{D}((T_D(G))_L))$ a TAL.

By the same reasoning, $L(\mathcal{D}((T_D(G))_R))$ is also a TAL. Hence S-TAGs, under Definition 5.4.2, have the WLPP. □

Note that this does not say anything about Shieber's concept of bounded subderivation, only about the cases when Definition 5.4.2 is strictly true and there is an isomorphism between derivation trees. The situation concerning bounded subderivation will be discussed in Section 6.2.2.

6.2 Representing Paraphrases

This section details a method for representing paraphrases, consisting of two components: STRUCTURAL MAPPING PAIRS (SMPs) and GENERATING FUNCTIONS. Structural mapping pairs are the S-TAG tree pairs that specify the syntactic rearrangement, while generating functions take a tree and produce a tree pair for those components not rearranged structurally, that is, those that are not altered by the paraphrase. The following two subsections give precise definitions for these generating functions and SMPs.

6.2.1 Generating Functions

In machine translation, it is obviously necessary to specify every tree pair, complete with links, as every word in the source language maps to a different word in the target language.[3] However, in paraphrase, most of the tree pairs will have the same left and right components. For example, take the set of tree pairs in Figure 6.2, representing the paraphrase relation between sentences with active voice and passive voice, for the pair of sentences in (1).

(1) a. The boy broke the frosted window.

b. The frosted window was broken by the boy.

Note that it is similar to the argument-swapping English-French pair of Figure 5.21.

Note further that α_2, α_3 and α_4 have identical left and right projections, including features and links, as these are the constituents that are not altered by paraphrase. Thus in a paraphrase representation only those pairs involved in specifying the alternation, structural mapping pairs, need be specified completely as tree pairs, like α_1 in Figure 6.2. These structural mapping pairs, and the conditions on them for well-formedness, will be defined more precisely later, in Section 6.2.2. Most tree pairs, however, will be generated by a single element.

[3] In practice, however, the pairs will probably be grouped in classes for mapping purposes, for example, Transitive-Verb-with-Argument-Ordering-Preserved, in a manner similar to tree families.

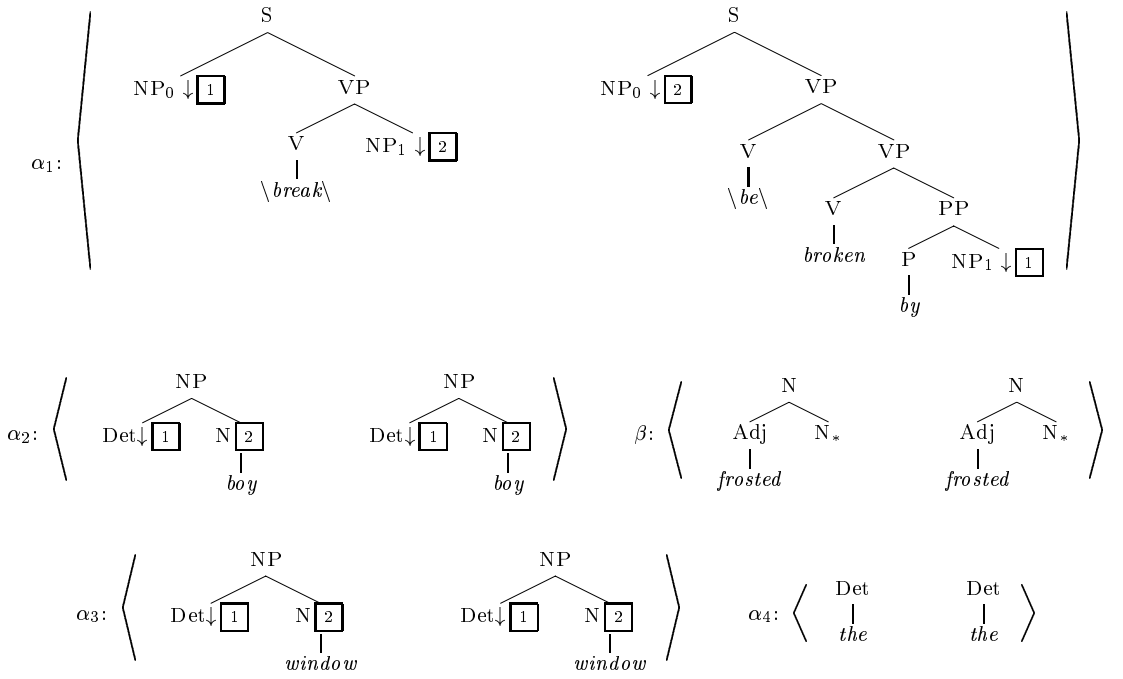


Figure 6.2: Active-passive mapping

Now we will want to define a function ψ, which will take a TAG tree and produce another tree, with which it can be paired. Such a function will obviously take a tree as one argument; it will also take a part of speech, so that ψ can 'transform' a tree of one part of speech type into a tree of another part of speech type. We will refer to an AUGMENTED SET OF PART OF SPEECH LABELS, designated $\mathcal{P}$. This contains the standard parts of speech, as are used in the rest of the thesis—**N**, **V**, **Adj**, and so on—plus 'null' parts of speech corresponding to each standard part of speech, ϵ_N, ϵ_V, ϵ_{Adj}, and so on. Let $\mathcal{P}_\epsilon \subset \mathcal{P}$ such that for all $p \in \mathcal{P}_\epsilon$, p is a null part of speech; $\mathcal{P}_p = \mathcal{P} - \mathcal{P}_\epsilon$. Note that $\mathcal{P}_p \subset V_N$ for some TAG G describing linguistic phenomena. The null parts of speech are explained and used later in this section.

The function ψ will thus be of the form

$$\psi : \mathcal{P} \times (\mathcal{I} \cup \mathcal{A}) \rightarrow (\mathcal{I} \cup \mathcal{A})$$

Before specifying the behaviour of ψ, we will investigate the properties we desire from such a function ψ by examining some tree pairs. Then, after defining ψ, we will define a function to generate S-TAG triples from a single tree.

In Figure 6.2, the function ψ is just the identity on the second argument: within each generated pair (that is, α_2, α_3, α_4), the elements are the same. The function here is

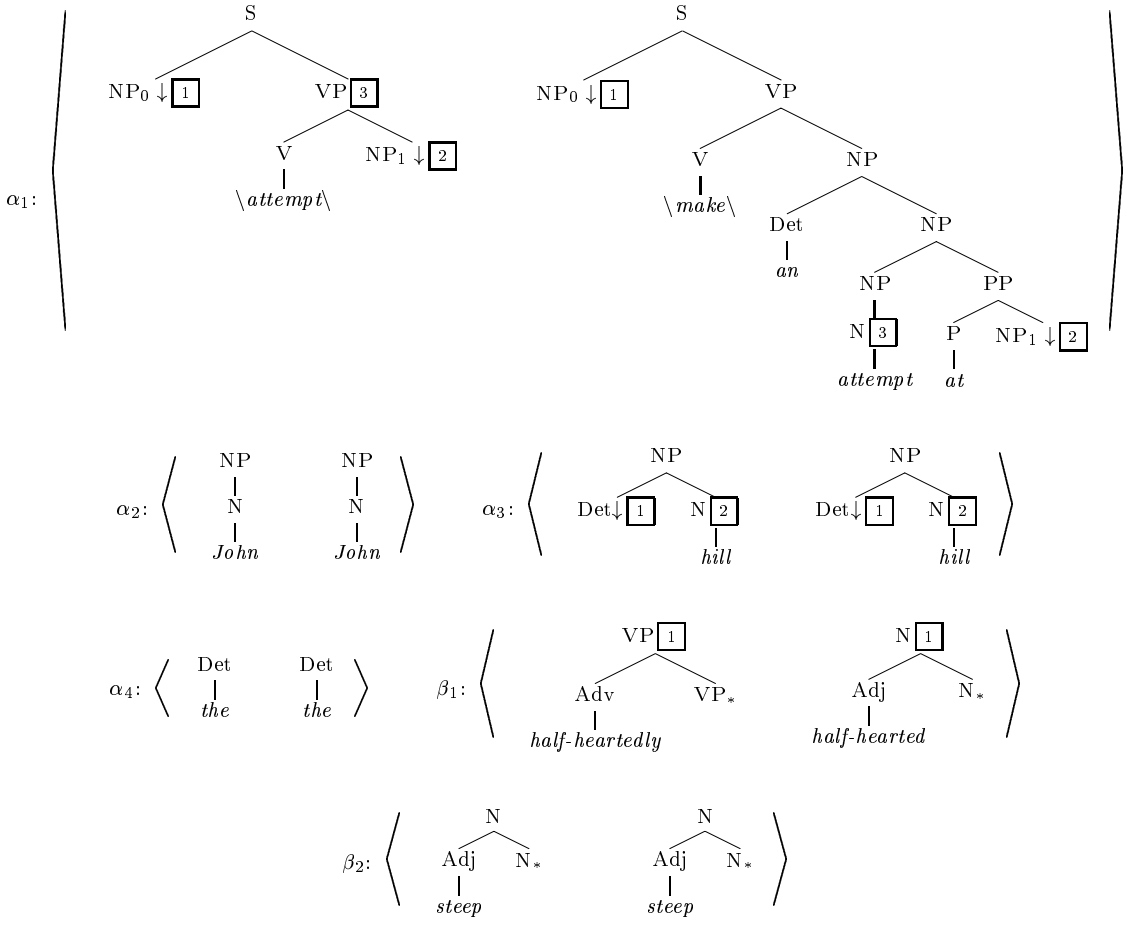


Figure 6.3: Tree pairs for example (2)

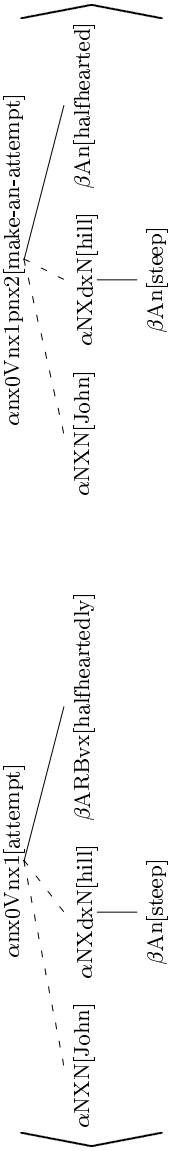


Figure 6.4: Derivation tree pair for Figure 6.3

defined to take as its part of speech the root of γ, to indicate that there is no transformation involved.

However, in some cases the function ψ is different. Consider the paraphrase in (2).

(2) a. John half-heartedly attempted the steep hill.
b. John made a half-hearted attempt at the steep hill.

In the corresponding trees of Figure 6.3, the structural mapping pair is α_1. In Figure 6.3, with tree pair β_1, both elements express the concept of half-heartedness, but one is realised syntactically as an adjective and the other as an adverb. Exactly which is the case depends on the structural mapping pair, which says, through link 3, that the left projection of α_1 takes an adverbial modifier and the right projection takes an adjectival modifier. The two elements of β_1 can still be generated by a pair $\langle \gamma, \psi \rangle$, where the function ψ is an appropriate morphological mapping.[4] In this case, it maps an adjective to its corresponding adverb if taking the left component as γ, and vice versa for the other direction.

The mapping is such that it is completely and uniquely determined by the label of the node into which it is adjoined or substituted: the Condition on Elementary Tree Minimality (CETM) (Frank, 1992) means that elementary trees should be minimal; thus there will not be two trees which, for example, represent simple adverbial modifiers corresponding to a simple adjectival modifiers. If the grammar was such that it did not obey the CETM, and there were two auxiliary trees representing simple adverbial modifiers, as say β and β' in Figure 6.5, both of which could correspond to simple adjectival modification, ψ could choose the simplest corresponding tree. In this thesis I will only include modifiers expressed by auxiliary trees here, such as adjective–adverb; noun–verb morphological processes, as in *destroy–destruction*, will be dealt with in SMPs, as these are 'core' mappings, and the auxiliary ones are determined by them. In circumstances where there is no appropriate morphological mapping, the part of speech $p \in \mathcal{P}$ gives an auxiliary tree of one node, the non-terminal labelling it being p. The reason for this will become clearer below in the discussion of null parts of speech ϵ_P.

We will call such a function that does this *morph*, and will define it as follows.

For some TAG G, morph : $\mathcal{P} \times (\mathcal{I} \cup \mathcal{A}) \to (\mathcal{I} \cup \mathcal{A})$, where for all $p \in \mathcal{P}$ and $\gamma \in (\mathcal{I} \cup \mathcal{A})$,

$$\text{morph}(p, \gamma) = \begin{cases} \gamma' & \text{if } \gamma'(\epsilon) = p \\ & \quad \text{and mapping from } \gamma \text{ to } \gamma' \text{ represents a 'simple morphological process'} \\ & \text{for some } \gamma' \in (\mathcal{I} \cup \mathcal{A}) \\ \gamma'' & \text{otherwise, where } \text{dom}(\gamma'') = \{\epsilon\} \text{ and } \gamma''(\epsilon) = \langle p, \mathcal{I} \cup \mathcal{A}, \mathbf{false} \rangle \end{cases}$$

At this point, the reason for the choice in Theorem 6.1.3 of a relation, rather than a homomorphism, becomes more clear. Consider the derivation tree pair for (2) in Figure 6.4, where projections of each pair of Figure 6.3 are labelled under the XTAG standard grammar notation (see Appendix A).[5] In one case, a β**An** from the right derivation tree is mapped to a β**An** label in the left, but in the other case to a β**ARBvx** label.

[4] Note that the pair here comprises a tree and a function; it differs from the pairs of S-TAG, which are pairs of trees.

[5] Except, to be precise, for the right projection of α_1: we will gloss over the structural complexity of structural mapping of structural mapping pairs until Section 6.2.2.

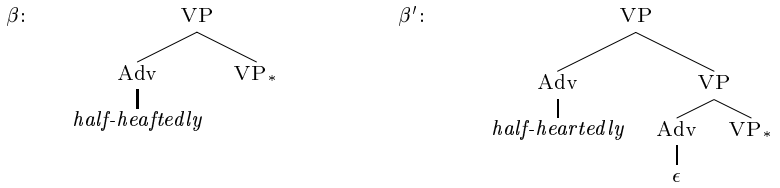


Figure 6.5: Alternative elementary trees for simple adverbial modification

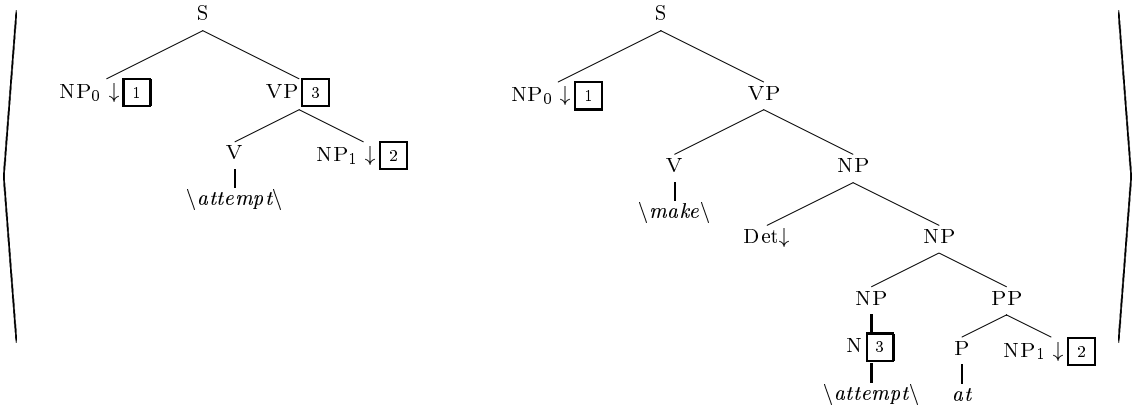


Figure 6.6: Generalisation of structural mapping pair α_1 from Figure 6.3

One problematic aspect occurs with the generalisation of α_1 from Figure 6.3. It is desirable to have the entire class of sentences of the form *John made an attempt at the hill*, *John made two attempts at the hill*, *John made many attempts at the hill* map to *John attempted the hill*. That is, the determiner in the structural mapping pair should be left as a substitution site, and the noun *attempt* as the lemmatised form, which on completion of the derivation will agree in number with the determiner via feature structure unification. Thus the more general structural mapping pair will be as in Figure 6.6.

Note that the determiner has no corresponding element on the left projection. This is an indirect consequence of our definition of paraphrase, Definition 4.3.1, where the propositional content of one element of the paraphrase pair is defined to be a subset of the propositional content of the other. That is, meaning can be lost during the paraphrase.

One approach is to let the substitution of a determiner tree occur by itself into the right tree of Figure 6.6. This is the easiest approach: if a node is not marked with a diacritic for linking, what occurs at that node is not replicated in the other tree.

However, this approach is not a desirable one. S-TAG is fundamentally a grammar of tree pairs, and the definition of S-TAGs depends on a bijection between paired derivation trees. If there is a composition operation performed on only one tree, this breaks down. Therefore, an alternative is to have a link between the **Det** node in the right tree and an arbitrary node in the left tree, as in Figure 6.7.

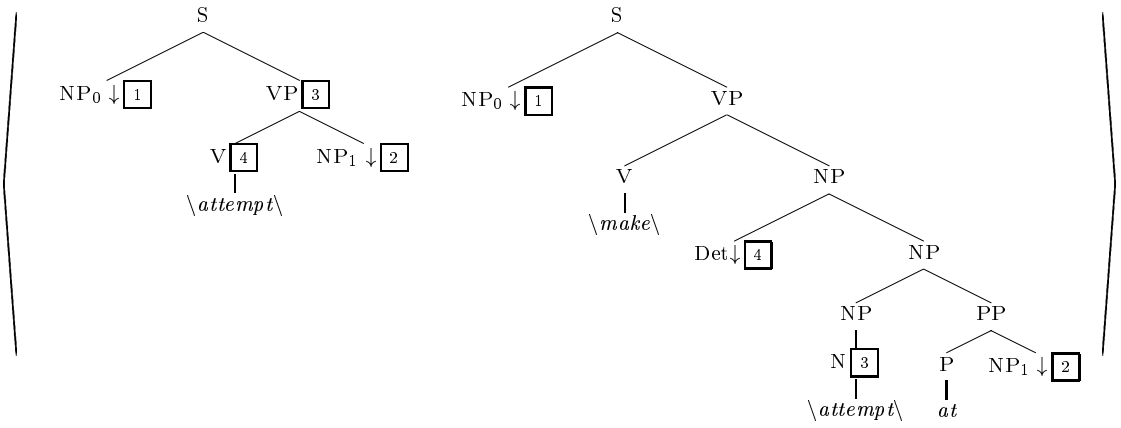


Figure 6.7: Synchronisation with null composition

The tree pair for composing with this will then be:

$$\beta\text{-}null\text{: } <\mathrm{V}_* \qquad \mathrm{Det}\diamond >$$

The left tree, an auxiliary tree consisting only of the node **V** (referred to as a 'null tree'), is just there to guarantee synchronous operation.

In this case, the definition of a function to generate *β-null* from a single tree is problematic. A function from the right tree to the left is OK: $\psi(\epsilon_V, \beta\text{-}null_R) = \beta\text{-}null_L$ (where $\beta\text{-}null_R$ and $\beta\text{-}null_L$ represent right and left trees respectively of *β-null*), for every lexical instantiation of **Det**. ϵ_V, the null part of speech for a verb, asks for the 'null operation', which is just the adjunction of an auxiliary tree consisting of a single node, the node label ultimately being determined by the tree into which it is adjoined. However, this function has no inverse. In order to define the function ψ from $\beta\text{-}null_L$ to $\beta\text{-}null_R$, one item is picked out to be the result of the function; for example, the determiner *an*. Then

$$\psi(\mathbf{Det},\ \beta\text{-}null_L) = \quad \begin{array}{c}\mathrm{Det}\\ |\\ an\end{array}$$

This corresponds to the idea of a default paraphrase where information is inserted, as discussed in Section 4.3.2 on loss of meaning in paraphrasing: if there is no context to suggest otherwise,[6] (3a) could be by default paraphrased to (3b).

(3) a. John attempted the hill.

b. John made an attempt at the hill.

[6] As noted elsewhere in this thesis, we are only looking at syntactic paraphrases without extra-sentential context. If such discourse context were used, the number of alternatives from which we select the default value—the inverse image of ψ—would be smaller.

Note finally that for all possible functions ψ, the feature structures notating the nodes in a tree γ are included as part of the tree, and so the resulting tree $\psi(p, \gamma)$ also has feature structures on its nodes.

Now, given these possible behaviours for ψ, we will define it as a function.

Definition 6.2.1 *A* GENERATING FUNCTION $\psi : \mathcal{P} \times (\mathcal{I} \cup \mathcal{A}) \rightarrow (\mathcal{I} \cup \mathcal{A})$ *for some TAG* $G = \langle V_N, V_T, V_L, S, \mathcal{I}, \mathcal{A}, - \rangle$ *is defined as follows.*
For all $p \in \mathcal{P}$ *and* $\gamma \in (\mathcal{I} \cup \mathcal{A})$,

$$\psi(p, \gamma) = \begin{cases} \gamma & \text{if } \gamma(\epsilon) = p \\ \text{morph}(p, \gamma) & \text{if } p \in \mathcal{P}_p \text{ and } \gamma(\epsilon) \neq p \\ \gamma'' & \text{if } p \in \mathcal{P}_\epsilon, \text{ where } \text{dom}(\gamma'') = \{\epsilon\}, \\ & \quad \gamma''(\epsilon) = \langle p', \mathcal{I} \cup \mathcal{A}, \textbf{false} \rangle \text{ and } p' \in \mathcal{P}_p \end{cases}$$

Now, given that we have a mechanism for generating one tree from another, we can generate an S-TAG triple.

Definition 6.2.2 *A* PAIRING FUNCTION $\Psi : \mathcal{P} \times (\mathcal{I} \cup \mathcal{A}) \rightarrow (\mathcal{I} \cup \mathcal{A}) \times (\mathcal{I} \cup \mathcal{A}) \times \frown$ *for some TAG* $G = \langle V_N, V_T, V_L, S, \mathcal{I}, \mathcal{A}, - \rangle$ *generates an S-TAG triple as follows.*
For all $p \in \mathcal{P}$ *and* $\gamma \in (\mathcal{I} \cup \mathcal{A})$

$$\Psi(p, \gamma) = \langle \gamma, \psi(p, \gamma), \frown \rangle$$

where for all $d \in \text{dom}(\gamma)$ *and* $d' \in \text{dom}(\psi(p, \gamma))$, $d \frown d'$ *where* $d = d'$.

That is, for generated S-TAG triples, all nodes are linked to their counterpart in the other tree.

6.2.2 Structural Mapping Pairs

Structural mapping pairs (SMPs) are the pairs that represent the syntactic rearrangement caused by a paraphrase, and must obviously be specified completely. This section explores the representation of SMPs in more depth.

Firstly, for the structural mapping pairs, it would be necessary, as in machine translation, to specify relationships between feature structures of the left and right trees as a pair—as, for example, the noun agreement features are mapped by Abeillé (1990)—since these don't come 'for free' as with a generating function ψ, which either just copies feature structures or, in the case of a morphological mapping, uses the appropriate basic features structures that are associated with the elementary tree. However, it is generally not necessary to specify feature structures for structural mapping pairs: also as in the machine translation

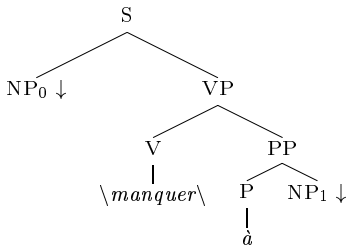


Figure 6.8: Elementary tree anchored by more than one lexical item

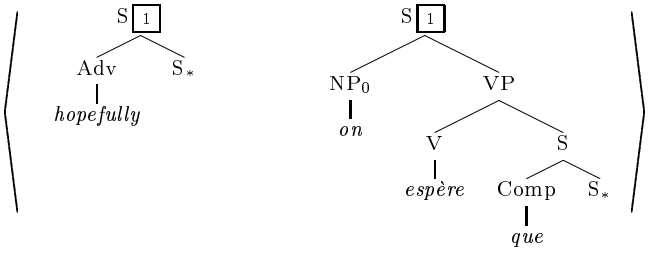


Figure 6.9: Mapping between semantically minimal pairs

approach of Abeillé (1990), verbs (and the structural mapping pairs which have them as anchors) do not need to have, for example, agreement features specifically mapped, as these are instantiated during unification with the agreement features of their arguments, which have their feature structures specified by ψ. For example, in α_1 of Figure 6.3, the agreement of the verbs $\backslash make\backslash$ and $\backslash attempt\backslash$ is determined by the subject NP, which is in both cases α_2, the tree pair for *John*, with the feature structure conformity guaranteed by the generation of pair α_2 from a single tree for *John* and the function ψ.

Now, it is useful to note that the sort of structures we deal with as elementary here—the trees of the structural mapping pairs—are different in character from the trees generally used in mappings, in that they are relatively complex derived trees. Frank's (1992) Condition of Elementary Tree Minimality (CETM) specifies that the basic trees that are the building blocks for all others should only have one lexical item associated with them as anchor. Abeillé (1990) uses a similar principle of semantic minimality: each basic tree embodies one semantic unit, thus allowing idioms, light verb constructions with their semantically empty verbs, and verbs with their obligatory function words (as *à* in Figure 6.8) to be included as elementary trees. Even some derived trees can be considered minimal in this sense: for example, the mapping between *hopefully* and *on espère que* in Figure 6.9.

Here, in the derived right tree, *on* can be considered to be semantically of little importance. Even though it is accorded its own elementary tree, it has much the same status as, say, the preposition *à*, which as a preposition has its own elementary tree, but which can still be included in the elementary tree of Figure 6.8 on the grounds of its functional role.[7]

[7] *On* is given its own tree because of the desire to minimise the number of trees through atomicity: it

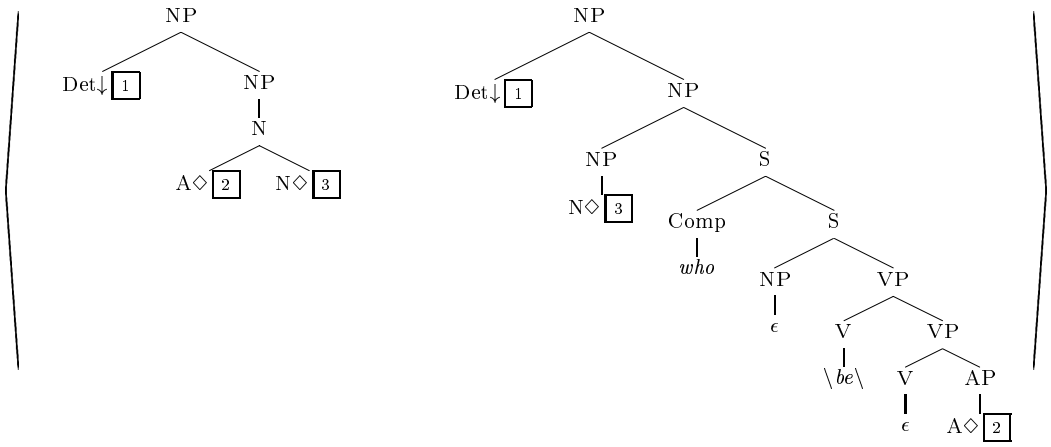


Figure 6.10: Semantically non-minimal pairs in paraphrase

By contrast, in paraphrase our basic trees are those of the structural mapping pairs, which do not necessarily represent one semantic unit. This is a consequence of the nature of syntactic paraphrase: when mapping between different syntactic structures, it is often the case that one or both are complex. Take, for example, the SMP in Figure 6.10, representing paraphrases of the form in (4).

(4) a. a timid man

b. a man who is timid

For (4), the actual trees used, taken from the standard XTAG grammar (XTAG, 1995), are as in Figure 6.11, with the S-TAG pairs apart from the SMP generated from these trees using the function ψ, with the appropriate part of speech determined by the label of the node in the SMP into which the tree is adjoined or substituted. The pair of derivation trees corresponding to the derived tree pair representing (4) is as in Figure 6.12. The regions of the derivation trees in Figure 6.12 formed by nodes in bold face correspond to the structural mapping pairs.

There are two possible approaches to establishing the well-formedness of S-TAG under paraphrase with respect to the SMPs, where this requires an isomorphism between the trees licensed by the S-TAG pairs, and in particular, the linking of the two regions corresponding to the SMPs in the derivation. The first is to treat each projection of an SMP as unitary, that is, as an elementary tree in its own right. This, however, is undesirable for several reasons: it does not capture the essence of the paraphrases, which is that they specify the mapping between arrangements of smaller units, the standard elementary trees; it goes against the spirit of TAG in creating new elementary trees with abandon, not respecting principles of minimality; and most importantly from a practical standpoint, it makes it difficult to establish isomorphism between pairs of derived trees. Therefore, we use a second approach, which is to specify SMPs in terms of derivation trees—that

is not desirable, for example, to have separate elementary (unitary) trees for *on espère que*, *il espère que*, *elle espère que*, and so on.

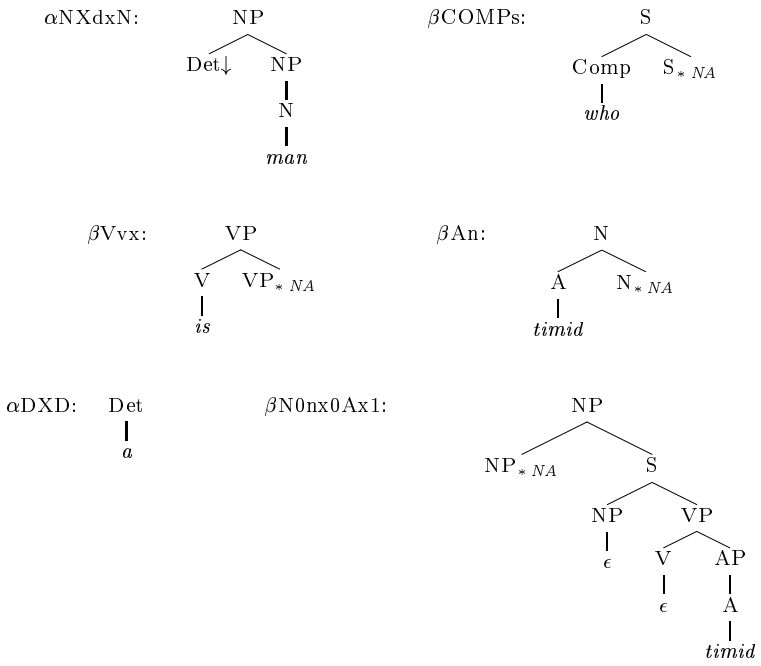


Figure 6.11: Component trees for semantically non-minimal SMP

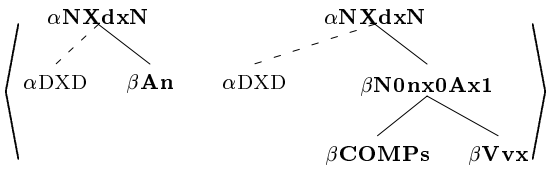


Figure 6.12: Derivation tree pair for semantically non-minimal SMP

is, detailing the structural arrangement of standard elementary trees obeying principles of minimality. Thus the derivation trees corresponding to the SMP in Figure 6.10 are precisely the regions in bold of the complete derivation trees in Figure 6.12.

Given this, it seems possible to specify paraphrases by means of valid derivation trees and their derived trees[8], with these valid derived trees corresponding to Shieber's idea of a bounded subderivation in a completed derivation tree.

However, take also, for another example, the tree pair of Figure 6.13. This tree pair represents a paraphrase involving promoting a subject relative clause to a full sentence, such as that of (5).

(5) a. The jacket which collected the dust was tweed.

b. The jacket was tweed. It collected the dust.

The mapping expressed by the trees in Figure 6.13 is the minimal essence of this paraphrase. From the standard XTAG grammar, the trees of Figure 6.14 could be used as the components of these minimal trees, and a derivation tree as in Figure 6.15 could be postulated to represent the left tree of Figure 6.13.

However, this proposed derivation tree of Figure 6.15 would only be valid if adjunction were allowed at substitution nodes. A conventional derivation can be seen by taking the derivation of the entire sentence, as in Figure 6.16: in this derivation tree, the relative clause and complementiser nodes depend from the NP which has been substituted into the matrix clause, rather than from the matrix (adjectival small clause) tree as in Figure 6.14.

For the paraphrase (5), this is mapped to the derivation tree in Figure 6.17. The parts corresponding to the structural mapping pair are indicated by nodes labelled in bold. Notice that the portion of the derivation tree in Figure 6.16 that corresponds to the structural mapping derived tree of Figure 6.15 is disconnected.

This poses a difficulty related to the clitic problem discussed by Shieber (1994): dominance relations are not preserved in the mapping. For the purposes of mapping between trees, the bold regions of Figure 6.16 cannot without restriction be treated as one constituent (which is Shieber's idea of treating multiple nodes as singular to give a bounded subderivation), as there can be arbitrarily many nodes between the two continuous regions.

For example, take the sentences in (6).[9]

(6) a. The jacket which collected the dust which covered the floor was tweed.

b. The jacket which collected the dust was tweed. It covered the floor.

These sentences have the derivation trees of Figures 6.18 and 6.19.

We can see that with unboundedly many relative clauses allowed, there will be unboundedly many nodes between the two halves of the disconnected region corresponding to the structural mapping tree for (6a).

[8] Both are used: the derivation tree to specify the components and their arrangement, the derived tree to indicate points of synchronicity.

[9] *It* in (6b) refers to the dust, which is not an obvious reading; clearly, having a more explicit referring expression is necessary. However, having a referring expression other than a fixed pronoun means that a link between multiple nodes on the same tree would be necessary in Figure 6.13, namely the two subject NP nodes in the right tree. This raises other issues, which are discussed in Chapter 7.

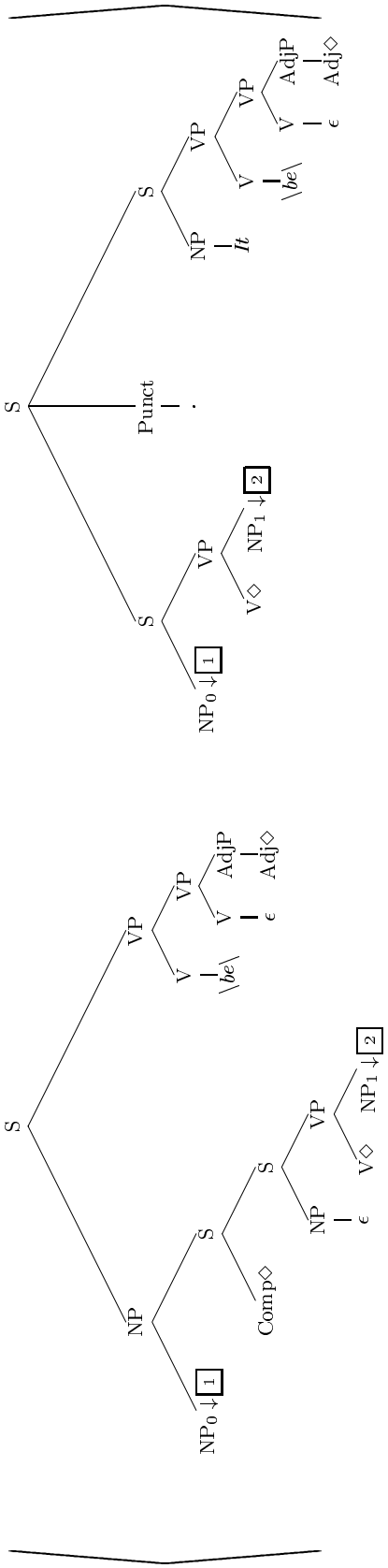


Figure 6.13: SMP for subject relative clause promotion

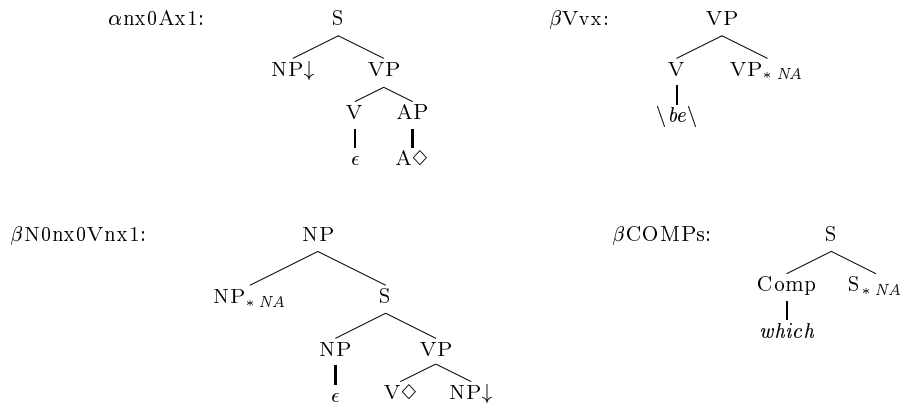


Figure 6.14: Component trees for Figure 6.13

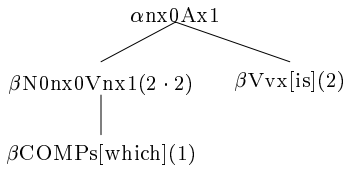


Figure 6.15: Postulated derivation tree for the left projection of Figure 6.13

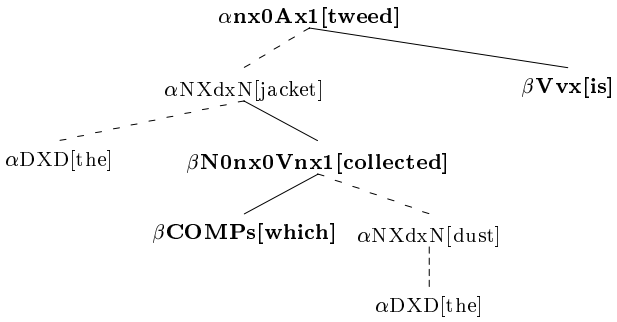


Figure 6.16: Derivation tree for (5a)

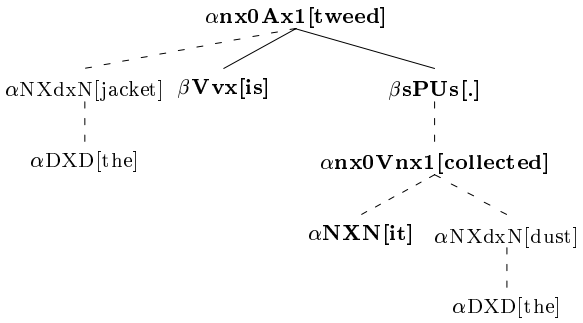


Figure 6.17: Derivation tree for (5b)

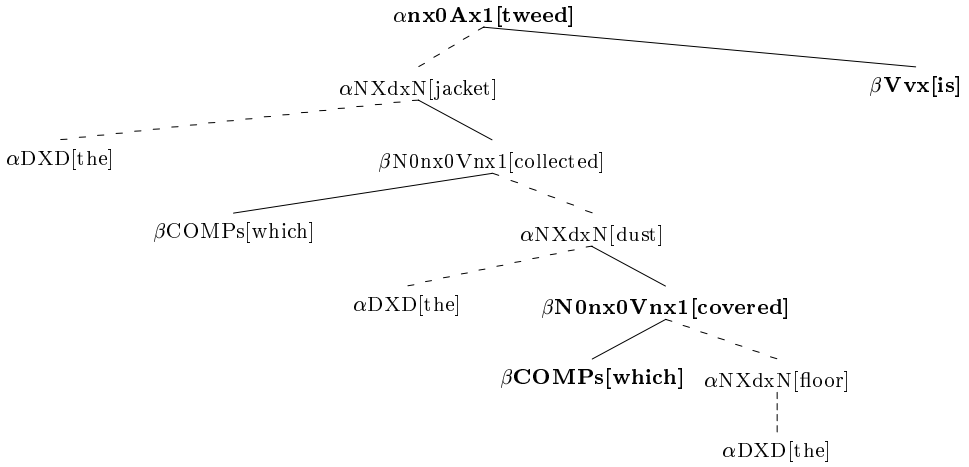


Figure 6.18: Derivation tree for (6a)

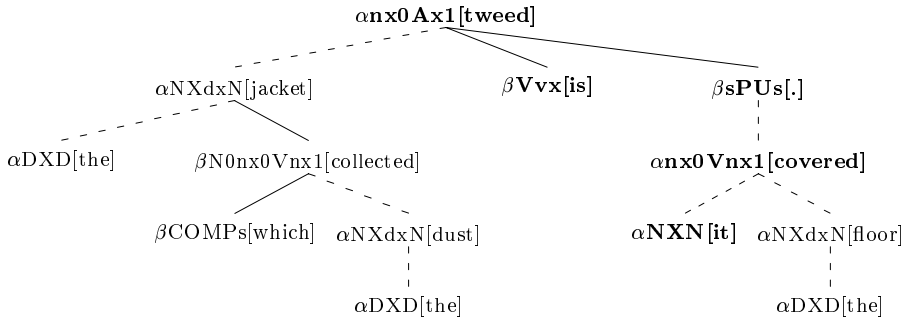


Figure 6.19: Derivation tree for (6b)

There are (at least) two solutions to this. The first one is the simplest, but neither elegant nor general. It involves treating the disconnected region as a single constituent by restricting the intervening region to being only a fixed number of nodes, for example, the single α**NXdxN** node representing *jacket* in Figure 6.16, along with any other dependents it might have that do not directly link the two regions (such as the α**DXD** node). If this is the case, then this α**NXdxN** node can be regarded as being dominated by this entire new constituent, and the mapping is OK. This corresponds to only allowing relative clause promotion to sentence at the top level of embedding; deeper embeddings cannot be promoted, as happens in the mapping of (6a) to (6b).[10]

A second approach is more general, recognising that the two disconnected halves of the region are actually related by some connection, no matter how much intervening material there is. This, of course, is suggestive of TAG's extended domain of locality. In fact, this appears to be a fairly natural way to treat the problem, defining an elementary tree which describes the disconnected constituents, and auxiliary trees to factor out the recursion.

Using this second option would solve the problem with unbounded relative clause promotion in paraphrase. However, this requires derivation trees with properties beyond the context-free, and S-TAGs are only defined for isomorphism between context-free derivation trees, so in this chapter we will use the first option for dealing with the discontinuities of relative clause promotion. The generalisation of Synchronous TAG that allows the second option is presented in Chapter 7.

Given this, we will define what makes a valid SMP, and take a modified version of Shieber's concept of a bounded subderivation (BSD), a bounded quasi-subderivation, that allows for these disconnected regions to be considered as a coherent whole.

First, however, it is necessary to define Shieber's concept of bounded subderivation in terms of what it means in a derivation tree.

Take some TAG G, some $\Upsilon \in T_D(G)$, and some $d \in \mathrm{dom}(\Upsilon)$. A PARTIAL SUBTREE of Υ at d, $\Upsilon/_p\, d$, is a tree such that it has a domain $D_p \subseteq \mathrm{dom}(\Upsilon)$ where for all $d_p \in D_p$, $d < d_p$.

A bounded subderivation is then a partial subtree in a derivation tree.

Shieber's proposal was that 'bounded subderivation would be considered elementary for the purpose of defining a relationship between the trees'. One way of viewing this is that, in effect, a surrogate derivation tree would be defined which, with the bounded subderivation becoming a single node, would be isomorphic to the corresponding derivation tree in the paired grammar. There are a number of ways (full) subtrees of the bounded subderivation could be treated; here, we will assume they are dominated by the entire bounded subderivation considered as a unit, and hence become subtrees of the new node in the surrogate. This might be represented formally as follows, where for a TAG G, derivation tree $\Upsilon \in T_D(G)$, and a bounded subderivation $\Upsilon/_p\, d_p$ with $d_p \in \mathrm{dom}(\Upsilon)$: for all $d' \in \mathbf{N}^*_+$,

[10] This differs from the clitic example in that under RP there is a choice about which paraphrases to model. Deciding to use only top level promotion is therefore not a problem. However, in modelling an existing language there is no such option: clitics cannot be made to limit their movement.

$$\dot{\Upsilon}(d') = \begin{cases} \Upsilon(d') & \text{if } d_p \not\leq d' \\ \eta & \text{if } d_p = d', \text{ where } \eta \text{ is some composite} \\ & \quad \text{node representing the bounded subderivation} \\ \Upsilon(d_p \cdot d'') & \text{where } d' = d_i \cdot d'', \\ & \quad d_i \in \mathrm{dom}(\Upsilon/_p\, d_p) \text{ and } d_i \cdot d'' \notin (\Upsilon/_p\, d_p) \end{cases}$$

A question here is, Does 'considering a BSD as elementary for the purposes of defining the relationship between the trees' still preserve the WLPP? It is possible, for any arbitrary collection of nodes, to imagine 'considering them elementary' for the purpose of isomorphism, which would really be establishing some general relation between the trees rather than an isomorphism.

Shieber's BSD, however, does satisfy the WLPP in a straightforward way. We can show that if the derivation trees of one projection are a recognisable set (which they are, being TAG derivations), then the set of surrogate derivation trees is also recognisable, if we define a relation between sets of trees as follows.

Given a TAG $G = \langle V_N, V_T, V_L, S, \mathcal{I}, \mathcal{A}, - \rangle$ and alphabets $\Sigma = V_N \cup V_T$ and $\Omega = \Sigma \cup \{\eta\} - \{\sigma_1, \sigma_2\}$ with $\sigma_1, \sigma_2 \in \Sigma$, a BSD RELATION R' is one where $R' \subseteq \Sigma \times \Omega$, extended to trees expressed as terms as follows:

- $t_\Sigma R' t_\Omega$ if $t_\Sigma \in T_\Sigma$ and $t_\Omega \in T_\Omega$ are atomic, and $t_\Sigma = t_\Omega$;
- $\sigma_1(t_{\Sigma,1} \ldots t_{\Sigma,i-1} \sigma_2(t_{\Sigma,i\cdot 1} \ldots t_{\Sigma,i\cdot n}) t_{\Sigma,i+1} \ldots t_{\Sigma,m}) R' \eta(t_{\Omega,1} \ldots t_{\Omega,i\cdot 1} \ldots t_{\Omega,i\cdot n} \ldots t_{\Omega,m})$ if $\sigma_1, \sigma_2 \in \Sigma$ form a BSD, where $\eta \in \Omega$ is some composite node indicating the BSD, and $t_{\Sigma,j} R t_{\Omega,j}$ with $t_{\Sigma,j} \in T_\Sigma$, and $t_{\Omega,j} \in T_\Omega$, for $1 \leq j \leq m$, and $j \neq i$, and for $i \cdot 1 \leq j \leq i \cdot n$;
- $\sigma(t_{\Sigma,1} \ldots t_{\Sigma,m}) R' \omega(t_{\Omega,1} \ldots t_{\Omega,m})$ otherwise, where $\sigma = \omega$ and $t_{\Sigma,i} R t_{\Omega,i}$ with $\sigma \in \Sigma$, $\omega \in \Omega$, $t_{\Sigma,i} \in T_\Sigma$, and $t_{\Omega,i} \in T_\Omega$, for $1 \leq i \leq m$.

By the same approach as Lemma 6.1.2, if T_Σ is recognisable then so is $\mathrm{RelImg}(R', T_\Sigma, T_\Omega)$, which is the set of surrogate derivations for $T_\Sigma = T_D(G)$. Thus since for any TAG G, $T_D(G)$ is recognisable, so is the set of surrogate derivations, and the WLPP holds. Conditions 3 and 4 of Definition 5.4.2 are then altered slightly, so that the isomoprhism is between these surrogate trees; we will refer to this as, for example, Condition 3 GIVEN BOUNDED SUBDERIVATION.

Note that such relations can be defined for any bounded region, but not for unbounded ones, which ties in with Shieber's suggestion.

Now we generalise the notion of a bounded subderivation to a bounded quasi-subderivation (BQSD).

Definition 6.2.3 *For some TAG G, and some $\Upsilon \in T_D(G)$, a* BOUNDED QUASI-SUBDERIVATION *is a set of partial subtrees $\Upsilon/_p\, d_1$, $\Upsilon/_p\, d_2$, ... , $\Upsilon/_p\, d_k$, where $d_1 < d_2 < \ldots < d_k$.*

That is, a BQSD is a set of partial subtrees where the root of the first dominates the root of the second, which dominates the root of the third, and so on. The portions of a

derivation tree between the roots of the partial subtrees the are not part of the partial subtrees will be referred to as SLOTS.

Now, valid derivation trees of one node are just those that correspond to projections of SMPs that are elementary trees, as the left tree of α_1 in Figure 6.3; those of more than one node correspond to the derived trees, as for the left projection of the SMP of Figure 6.10, with the bounded subderivation indicated by the region of bold labels in Figure 6.12. The other category is exemplified by the left projection of the SMP of Figure 6.13, and its bounded quasi-subderivation by the regions of bold labels in Figure 6.16 with intervening slot for node α**NXdxN**.

With a bounded quasi-subderivation, as with bounded subderivation, the components are treated as a single unit. This single unit dominates any node that is dominated by any component of the BQSD; a node filling a slot is similarly considered dominated by the entire BQSD. A relation R' can be defined to specify a mapping of these BQSDs to surrogates in the same manner as for BSDs.

Recall also that the SMPs need to be completely synchronised, so that it is not possible to compose with one projection and not the other; so all nodes in the trees of an SMP must be annotated with either diacritics (representing links) or NA constraints.

Definition 6.2.4 *For an S-TAG G, a well-formed* STRUCTURAL MAPPING PAIR *(SMP) is one whose two elements*

- *are connected trees corresponding to bounded quasi-subderivations in some derivation tree pair $\Upsilon \in T_D(G)$;*
- *are linked by a relation $\frown$, as a standard S-TAG pair.*

For this definition, exactly what can be part of an S-TAG triple is relaxed somewhat. Instead of just elements of $elem(V_N, V_T, V_L)$, the elements can be as in the first part of the definition above.

Related to the SMP definition, we define a Condition of Paraphrase Pair Minimality.

Definition 6.2.5 *Under the* CONDITION OF PARAPHRASE PAIR MINIMALITY *(CPPM), the trees in a structural mapping pair are such that they are the minimal expression of the paraphrase that they embody. That is, the derived trees in the SMP are made up of only the minimal number of elementary trees, each obeying the CETM, necessary for describing the syntactic rearrangement.*

For example, the left tree of the pair in Figure 6.13 satisfies this principle despite being semantically non-unitary and being derived from four elementary trees of Figure 6.15.

Before going on to demonstrate that, for the representation of paraphrase given here in Section 6.2, that the S-TAG representation is well-formed, and obeys the weak language preservation property, I will make a brief comment about underspecification of trees and notation.

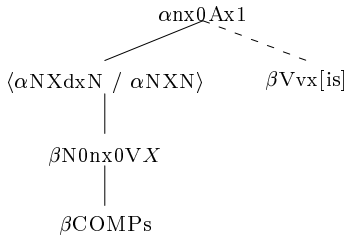


Figure 6.20: Derivation tree with slot and underspecified relative clause node

6.3 Underspecification and Notation

The structural mapping pairs presented so far are often more specific than is necessary for paraphrasing. For example, the left projection of the pair in Figure 6.13 only represents transitive relative clauses embedded into an adjectival small clause matrix. The same basic idea—that is, the same basic derivation structure—applies, for example, to relative clauses formed from ditransitive verbs, object-equi verbs, and so on. However, in TAG it is not possible to compose partial trees. In concept, though, any tree of the form[11] β**N0nx0V**X can validly be used, where X is a variable representing any valid string of arguments: **nx1**, **nx1nx2**, **nx1pnx2**, and so on. The derivation tree representing the underspecified left tree of the structural mapping is then as in Figure 6.20. The slot ⟨α**NXN / NXdxN**⟩ indicates the discontinuity in the derivation tree of this kind of mapping; it can be thought of as a 'place-holder' for a noun phrase node. Recall also that, under the restriction on disconnected regions adopted in this chapter, there can only be a bounded number of slots.

This has some connection with other ways of underspecifying trees: the sharing of structure in lexicalised trees (Vijay-Shanker and Schabes, 1992), DATR (Evans *et al*, 1995), the hierarchical approach of Candito (1996), the partial-tree descriptions of Xia *et al* (1998), and so on. Here, however, it is merely a shorthand for generalising across tree arguments. A more sophisticated representation could be employed to generalise across other aspects of the trees of the structural mapping, such as similarities between adjectival small clauses and other copular constructions; but that is not attempted here.

6.4 Well-Formedness of S-TAG under Paraphrase

Given the construction method for an S-TAG for paraphrase proposed in the preceding sections, the question is, Is S-TAG under paraphrase well-formed. This section shows that, given some TAG, the S-TAG that is constructed from it is well-formed.

In an S-TAG under paraphrase, as stated informally at the start of the chapter, we take some TAG $Gi = \langle V_N, V_T, V_L, S, \mathcal{I}, \mathcal{A}, - \rangle$ and some predefined SMPs and construct the S-TAG. This S-TAG is thus the mapping of the grammar onto itself,

$$G = \{\Psi(p, \gamma) \,|\, \gamma \in \mathcal{I} \cup \mathcal{A}, p \in \mathcal{P}\} \cup \{\langle L, R, \frown \rangle \,|\, \langle L, R, \frown \rangle \text{ is an SMP}\}$$

[11] Again, this uses the standard XTAG convention for naming trees, as described in Appendix A.

Note that both the BQSDs for the SMPs must come from trees in $T_D(Gi)$.

Now we demonstrate the well-formedness of this construction.

Lemma 6.4.1 *The derivation tree pairs of the S-TAG G under paraphrase are well-formed.*

Proof:

$\Upsilon \Rightarrow \langle \Upsilon_{\mathbf{L}}, \Upsilon_{\mathbf{R}} \rangle$

Take some derivation tree $\Upsilon \in T_D(Gi)$.

If Υ is a derivation tree of one node, then $\gamma = \mathcal{D}(\Upsilon)$ is an elementary tree, from Definition 5.1.2; and there exists a pair $\Psi(p, \gamma) = \langle \gamma, \psi(p, \gamma) \rangle$ for some part of speech $p \in \mathcal{P}$.

$\psi(p, \gamma)$ is a unique elementary tree under Definition 6.2.1. There is thus a set of pairs associated with each γ,

$$S = \{\langle \gamma, \psi(p, \gamma) \rangle \mid p \in \mathcal{P}\},$$

with cardinality $|\mathcal{P}|$. Note that S must contain at least one pair without a null tree, as the set of valid parts of speech must contain the part of speech of the root node of γ, that is, allowing the identity mapping on the second argument for the right projection.

As γ and $\psi(p, \gamma)$ are both elementary trees, they also have derivation trees of one node. Thus, there is a set of valid derivation pairs $\{\langle \Upsilon_L, \Upsilon_R \rangle\}$ of cardinality $|\mathcal{P}|$.

If Υ is a well-formed derivation tree of height > 1, then it can be defined recursively under Definition 5.1.2. The root node of Υ will correspond to an elementary tree γ. By the argument above, there exists a set S of pairs corresponding to γ. Given that there are immediate subtrees of Υ that are well-formed derivation trees, these can be attached to the node representing the left projection γ of the elements of S, by virtue of the same derivation rule used to create Υ,

$$\gamma' = \nabla_S(\gamma, \gamma_1, t_1, \ldots, \gamma_k, t_k)$$

The right subtree can similarly have subtrees attached using a derivation rule, with the appropriate parts of speech p_i determined by the address t'_i:

$$\gamma' = \nabla_S(\psi(p, \gamma), \psi(p_1, \gamma_1), t'_1, \ldots, \psi(p_k, \gamma_k), t'_k)$$

Thus, both left and right projections have valid derivation trees.

The third case is that Υ has a set of partial subtrees corresponding to a BQSD of the left projection of an SMP, $\Upsilon /_p d_i$, $d_i \in \mathrm{dom}(\Upsilon)$. If so, the argument is as for the recursive case above, except that $\Upsilon /_p d_i$ and their subtrees are used, rather than the root of Υ and its subtrees; the right projection is determined directly from the SMP, rather than by function ψ; and when a slot sequence is filled by a sequence of nodes, the lower partial subtrees in the BQSD are considered attached to the slot fillers.

Thus in all cases, a well-formed derivation tree Υ in the grammar implies well-formed derivation trees for an S-TAG pair.

$\langle \Upsilon_{\mathbf{L}}, \Upsilon_{\mathbf{R}} \rangle \Rightarrow \Upsilon$

The argument is similar to the case above, but simpler. The pairs $\langle \Upsilon_L, \Upsilon_R \rangle$ can be partitioned into equivalence classes by ignoring the part of speech $p \in \mathcal{P}$, and then taking the left projection to give Υ.

□

Note that as a consequence of this proof there will be at least one paraphrase for each well-formed derivation: the identity paraphrase, and any defined by SMPs on top of that.[12]

Now we define a mapping between the derivation trees of left and right projections of tree pairs, and show that it preserves structure.

For some $\Upsilon_L \in T_D(G_L)$ and $\Upsilon_R \in T_D(G_R)$, let $\delta : \Upsilon_L \rightarrow \Upsilon_R$ be a function that maps between derivation trees as licensed by the relation $\frown$ in the S-TAG G under paraphrase.

Lemma 6.4.2 *δ is a bijection given bounded subderivation.*

Proof:

Take a set of nodes $\{\eta_i\}$ in a derivation tree of the left projection, $\Upsilon \in (T_D(G))_L$.

If the $\{\eta_i\}$ correspond to a BQSD matching an SMP, then there is a corresponding BQSD in the right derivation tree by Definition 6.2.4.

If the $\{\eta_i\}$ do not correspond to a BQSD matching an SMP, then each η_i has a corresponding derived tree $\gamma_i = \mathcal{D}(\eta_i) \in (\mathcal{I} \cup \mathcal{A})$. The right projection is given by $\psi(p_i, \gamma_i)$, $p_i \in \mathcal{P}$. As in the proof of Lemma 6.4.1, p_i is fixed by the node with which the derivation tree for γ_i, the node η_i, has been composed; and by Definition 6.2.1 the right projection will be unique. As $\psi(p_i, \gamma_i)$ is an elementary tree or 'null' tree (by virtue of being part of a generated pair), it will correspond to a single node in the right derivation tree.

□

Lemma 6.4.3 *δ preserves dominance relations given bounded subderivation.*

Proof:

The argument is the same as in the proof of Lemma 6.4.1 for the recursive case, where the derivation rules used to construct well-formed derivation tree pairs ensure the dominance relations.

A special case occurs with SMPs whose BQSDs contain slots. Here, any nodes dominated by any part of a BQSD is considered dominated by the whole, by the unity of the BQSD. The tree corresponding to the slot filler α must be adjoined or substituted into the left

[12] Note that some possible paraphrases are mutually exclusive—for example, promoting relative clauses of differing depths, because they use related SMPs rooted in S—but others are not. Resolving this is beyond the scope of this thesis; and hence in Section 8.3.1 we will apply a condition in the optimisation component of the work to ensure that only a single paraphrase is used per sentence.

projection of the SMP, by the definition of its filling a slot; through the links corresponding to diacritics, $\psi(p, \alpha)$ will also be adjoined or substituted into the right projection. Note that no extra nodes can be inserted around the slot position once the BQSD has been established, by virtue of the context-free nature of TAG derivation trees (Lemma 5.2.2). Thus the node corresponding to the slot filler in the right derivation tree is similarly dominated by the BQSD on the right of the SMP.

□

We can now show that if a grammar produces well-formed derivation trees, then S-TAG under the paraphrase definition of this chapter is well-formed.

Theorem 6.4.4 *S-TAG under paraphrase is well-formed and has the WLPP.*

Proof: The S-TAG $G = \langle Gi, Gi, \frown \rangle$ under paraphrase is well-formed under Definition 5.4.2:

- Conditions 1 and 2 hold, from Lemma 6.4.1, by treating BQSDs as single nodes;
- Conditions 3 and 4 hold, from Lemmas 6.4.2 and 6.4.3.

□

6.5 Summary

This chapter has demonstrated that, given appropriate definitions for S-TAG pairs under paraphrase, the whole system is well-formed. The proofs are not an end in themselves, merely a way of guaranteeing that no unexpected mappings occur, and that the language that is generated by the mapping has the expected weak generative capacity. In this chapter the expected weak generative capacity of both projections is the same, and each will generate the same string language; in Section 7.1, this is extended to allow mappings between formalisms of differing generative capacity, for several motivating reasons.

The most important aspect of the chapter has been the constructive nature of the approach: the preparatory definitions show how to go about building paraphrases in a formal system, using structural mapping pairs (SMPs) and generating functions, with appropriate conditions. This foreshadows a solution to the problem of elimination of dominance as discussed in Shieber (1994), with another extension to S-TAG proposed, based on this idea; this is explored in Section 7.2.

To demonstrate the constructive nature of the approach, in Section 7.3 SMPs will be defined for a range of the paraphrases given informally in Section 4.2. Furthermore, a characterisation of the paraphrase effects of Section 4.3, as a function of S-TAG pairs, will be given; this function is a mapping from the formal model of this chapter to the paraphrases used in the optimisation models of Chapter 8.

Chapter 7

Generalising Synchronous TAGs

In the process of representing paraphrase in S-TAG in Chapter 6, a number of issues were raised. Firstly, the issue of referring expressions in multiple sentences: while using a simple pronoun means that standard S-TAG mapping methods can be used, modelling a more complex referring expression means that the linking relation implicitly links nodes in the same tree. Section 7.1 explores this and the implications it has for the S-TAG formalism.

Secondly, the idea of extended domain of locality for derivation trees appears to be a natural consequence of structural arrangements specified by derived trees. This suggests using a TAG grammar at the derivation (or meta-) level; this is investigated in Section 7.2.

Both of these ideas lead to generalisations of S-TAG that solve problems mentioned in Chapter 5 that would otherwise mean S-TAG was inadequate for modelling language. Then, with these generalisations, Section 7.3 outlines how the formalism is applied to examples of the paraphrases described in Chapter 4.

7.1 Extending S-TAG for Many-to-One Links

In Chapter 6 all of the links were such that they allowed standard TAG derivations to occur. However, in paraphrase it is sometimes necessary to have links that are many-to-one. This section looks at an example of this type, and concludes that an extension of S-TAG to allow mappings between broader classes of TAG-related grammars is a natural way to handle these. This extension also provides a natural way to handle coordination, a problem given various treatments under the TAG formalism by Joshi (1990), Joshi and Schabes (1991), Sarkar and Joshi (1996) and Sarkar (1997); the treatment here allows them to be closer in spirit to standard TAGs, in that it does not require any special extra mechanisms, such as derivation trees becoming directed acyclic graphs.

7.1.1 Many-to-One Links

The motivation for having many-to-one links is given predominantly by paraphrases of Type C (Change of Relation) in this thesis, those describing sentence splitting or combining, as well as by the phenomenon of coordination; these problems have as a potential solution the idea of many-to-one links, as independently proposed in Dras (1997a) and

Sarkar (1997) respectively. In these, the split sentences will both have the same subject.[1] Then we could have an SMP as in Figure 7.1, a variant of Figure 6.13, which could represent the paraphrase (1). Note the two $\boxed{1}$ links at $\mathbf{NP}_0$ and $\mathbf{NP}_1$ in the right tree: both have the same NP substituted here as in the left tree.

(1) a. The jacket which collected the dust was tweed.

b. The jacket collected the dust. The jacket was tweed.

Having these many-to-one links, however, changes the properties of the S-TAG formalism, so that the weak language preservation property no longer holds. This can be shown by the following example. Take the TAG tree pairs of Figure 7.2.

It will be clear that the right projection of this S-TAG generates the string language $\{a^n b^n c^n d^n a^n b^n c^n d^n \mid n \geq 0\}$, which is not a TAL; this can be confirmed by application of the Pumping Lemma for TALs (Vijay-Shanker, 1987: 96), which states that TALs must be of the form $u_1 v_1^i w_1 v_2^i u_2 v_3^i w_2 v_4^i u_3$, with $u_i, v_i, w_i \in \Sigma^*$, with Σ the alphabet of terminal and non-terminal symbols.

In fact, for some arbitrary value $k \in \mathbf{N}_+$, we can get the language $\{(a^n b^n c^n d^n)^k \mid n \geq 0\}$ by having k nodes in one tree linked together by the same diacritic. Similarly, we can obtain the k-copy language $\{w^k | w \in \{a, b\}^*\}$ for some integer $k > 1$ by the tree pairs of Figure 7.3.

Exactly what this does to the corresponding derivation trees is not clear. In Sarkar (1997) they become directed acyclic graphs; at any rate, they are not well-formed under the definition of derivations given in Chapter 5, Definition 5.1.2.

One way of adapting the S-TAG formalism to handle this phenomenon in a predictable way, retaining the well-formedness of the derivation trees, is to note that what is actually happening when these trees are being synchronously composed is that *multiple copies* of the trees are being composed into the parent. That is, there is a set of identical trees being composed ultimately into the single tree at the root of the derivation. A neat representation for this is thus a Multi-Component TAG (Section 5.3.3), with identical elements in the sequence; because the elements of a sequence are composed into elements of another sequence, the MCTAG is set-local. Figure 7.2 can thus be re-represented as in Figure 7.4.

As shown in Weir (1988), set-local MCTAG has a generative capacity beyond that of standard TAG. With a sequence of k elements for some integer $k > 1$, MCTAGs are able to generate the language COUNT-$4k$, that is $L_{c4k} = \{a_1^n a_2^n \ldots a_{4k}^n \mid n \geq 0\}$. If these elements are all the same, then the language generated is $L'_{c4k} = \{a_1^n a_2^n \ldots a_{4k}^n \mid n \geq 0;\ a_i = a_{i+4}, 1 \leq i \leq 4k-4\}$, that is, $L'_{c4k} = \{(a_1^n a_2^n a_3^n a_4^n)^k \mid n \geq 0\}$. Similarly, they can generate $\{w^k \mid w \in \{a, b\}^*\}$ for some $k \in \mathbf{N}_+$, as in Figure 7.3.

We do not need the full generative capacity of MCTAG; at this stage, only sequences with identical elements are used, although this will be extended in a limited way later. A characterisation of this will be presented further on; first, however, it is necessary to show that the mapping between standard TAG and MCTAG is a valid one.

[1] More precisely, they will both refer to the same entity, with possibly different referring expressions, but here we will start with the assumption that the lexical realisation of the entity is the same for each sentence.

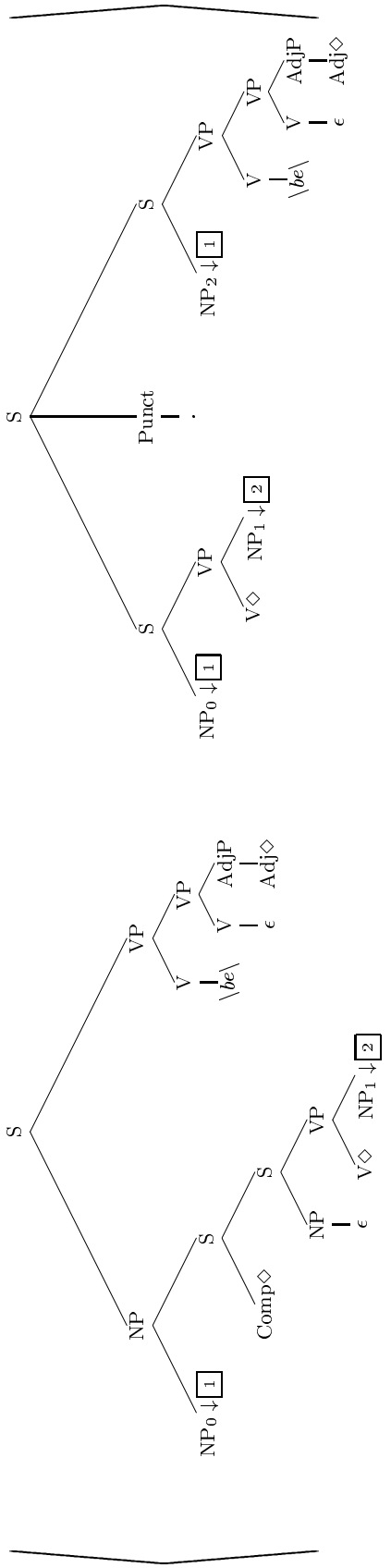


Figure 7.1: SMP with many-to-one links

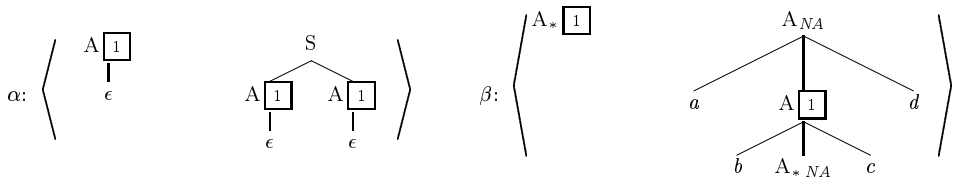


Figure 7.2: An S-TAG with many-to-one links generating $\{a^n b^n c^n d^n a^n b^n c^n d^n \mid n \geq 0\}$

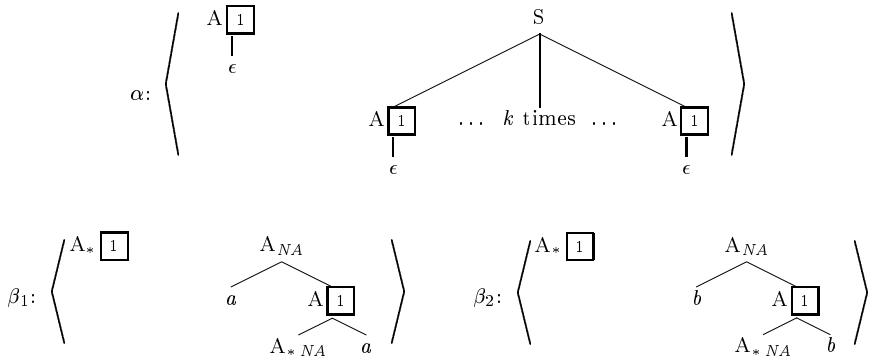


Figure 7.3: An S-TAG with many-to-one links generating $\{w^k \mid w \in \{a, b\}^*\}$ for some $k > 1$

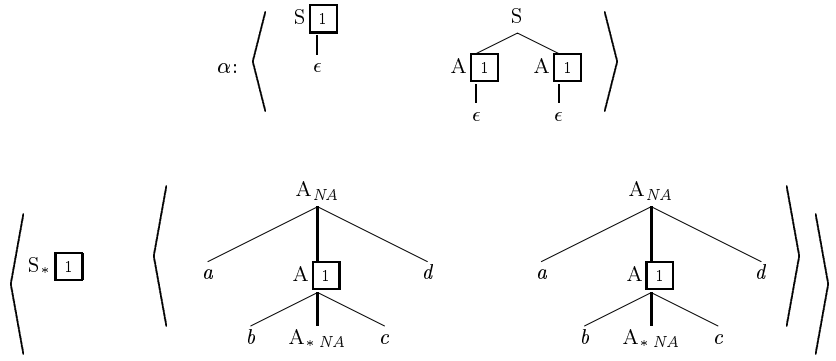


Figure 7.4: An S-TAG with many-to-one links represented via MCTAG

First, we give a more general result involving Linear Context Free Rewriting Systems (LCFRSs) which, although the proof is straightforward, is nonetheless important.

LCFRSs were defined in Weir (1988: 91–95) to be the class of grammar formalisms which have two particular properties: that the derivation trees form a local set; and that the composition operations be linear and non-erasing. CFGs, head grammars, TAGs and MC-TAGs all fall into this class of grammar formalisms. What makes each formalism different from the others in the class is the yield function, applied to the derivation structure to generate the object-level structure.

We will define a SYNCHRONOUS LCFRS to be one where there are two grammars G_1 and G_2 of possibly different types of LCFRS-class grammar formalisms, and elements of G_1 and G_2 are paired by a link relation $\frown$, such that Definition 5.4.2 holds. This link relation will not necessarily be the same as for S-TAG, where it is a relation between tree addresses, but could be defined to be a relation between term elements, for example.

Theorem 7.1.1 *A synchronous LCFRS (S-LCFRS) has the WLPP.*

Proof: By definition, the derivations of the projections of an S-LCFRS form local (and hence recognisable) sets. Given that there is an isomorphism between the derivation trees in a derivation pair, and Definition 5.4.2, the proof is the same as for Theorem 6.1.3, with the appropriate yield function to derive the object-level structure. □

The basic idea that this theorem embodies is that structures whose derivations are recognisable sets—or, in other words, which can be represented by context-free rules—can be mapped to each other. The weak language preservation property holds by virtue of the fact that the derivation structures are isomorphic; or, interpreted slightly differently, that both derived trees have the same derivation structure, with the labels read off the tree in a different way to produce different derived trees. This different yield function is what produces the potentially different generative capacities; the derivation tree is what the transduction takes place on, and its separation from the yield function is what allows the mappings between grammars with differing generative capacities (see Restriction 1 of the definition in Weir, 1988:94).

The application of this theorem that we will be using is only a very restricted one, mapping standard TAG to a restricted form of set-local MCTAG, as in the example discussed before the last theorem. This restriction is the one motivated by the example, such that sequences consist only of identical items.

This kind of sequence will thus be different in character from the sequences typically used in other applications of MCTAG. Most often there is some kind of ordering relation between elements, such as dominance or c-command. This allows, effectively, a wrapping of trees around some elementary tree; this can be seen as equivalent to an adjoining of this second elementary tree into a 'composition' of the trees in the sequence as ordered by the dominance or c-command orderings, but an adjoining which does not force root and foot nodes of the adjoined-in tree to be identical. This is used, for example, to model aspects of other languages such as clitics in Spanish (Bleam, 1994; Kulick, 1998) which pose problems for standard TAG analysis. In the sequence proposed under paraphrase, there is no ordering relation; rather, the elements in the sequence have some structural relation holding between them which, at this stage, will just be the identity.

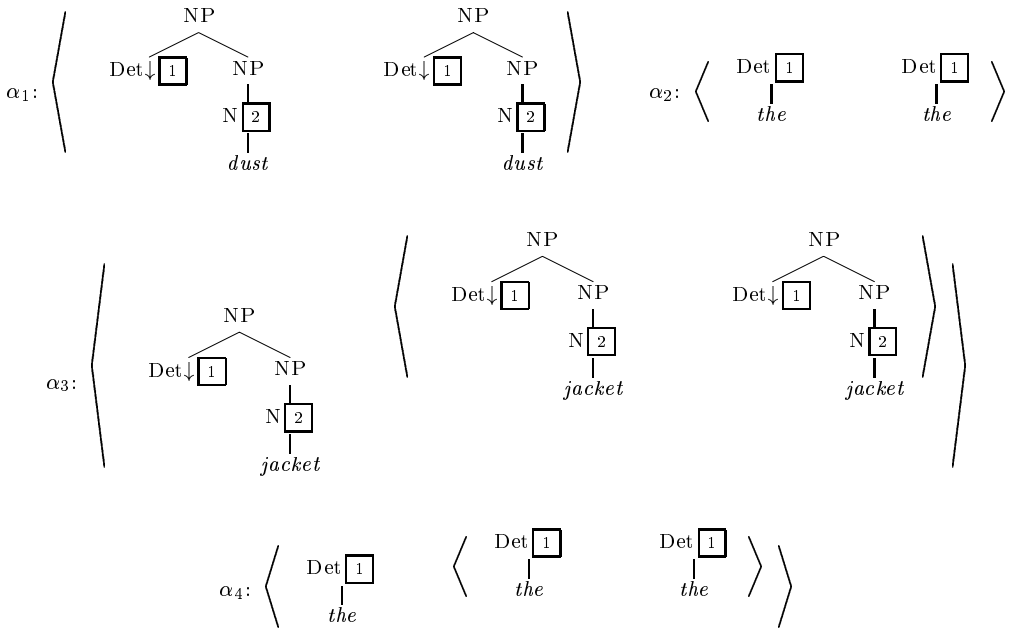


Figure 7.5: Tree pairs for (2)

Later in this section a constructive definition for this extension to paraphrase will be given, and the well-formedness of the synchronisation shown for this new definition also. First, an example will demonstrate the sort of derivation trees that an MCTAG for a paraphrase example generates, and how an isomorphism might be induced in a paraphrase context.

Take (1), reproduced below here as (2); an SMP as in Figure 7.1, anchored with the verb *collect* and the adjective *tweed*; and generated pairs as in Figure 7.5.

(2) a. The jacket which collected the dust was tweed.

b. The jacket collected the dust. The jacket was tweed.

In Figure 7.5, the pairs that correspond to a one-to-one link in Figure 7.1—α_1 and α_2—have simple TAG trees as the right projection, while those corresponding to a one-to-m link—α_3 and α_4—have a sequence of m (here 2) identical trees as the right projection. Note that the elements of the sequence in α_4 substitute into different elements of the sequence in α_3, making this a set-local use of MCTAG for the right projection.

A possible derivation tree pair is as in Figure 7.6. I am assuming here that the SMPs are constructed from a slightly different set of elementary trees from previously, with an initial tree (rather than an auxiliary tree) used for the S-rooted tree linking the two sentences.[2] As before, the parts corresponding to the SMP are in bold.

[2]Christie Doran (p.c.) has commented that while the choice of trees for lexical items has been strongly guided by linguistic considerations, choice of trees for punctuation is to some extent more arbitrary. The choice here between α**sPUs**[.] and β**sPUs**[.], the initial and auxiliary trees for sentence linking via a period, respectively, makes no difference linguistically, but the initial tree leads to a simpler exposition in the development of an isomorphism in this section than would occur by choosing the auxiliary tree.

For the right tree, the sequences in the MCTAG are given in Table 7.7.

In the derivation tree for the MCTAG in Figure 7.6, the addresses annotating each node are of the form $(t_1, a_1; t_2, a_2; \ldots ; t_k, a_k)$, where, for a sequence of k trees, t_i is the tree in the parent sequence into which tree i is composed, and a_i is the address at which this occurs. The 'copy' sequences S_2 and S_3 are substituted in parallel into the sequence S_1 representing the two component sentences, and this sequence in turn substituted into the discourse structure tree S_5, which combines the two sentences. An isomorphism that is natural in the paraphrase context is to treat the bolded regions within the derivation trees, which represent the SMP, as BQSDs that correspond to each other; and for sequences of (one of more) identical items to correspond to the single occurrence of that item in the left tree, as S_2 and S_3 to the singular α**NXdxN**[dust] and α**DXD**[the] (in the case with one item) in the MCTAG sequence.

Note that for this derivation tree pair, it is necessary to have a sequence like S_1, so that the copied items of S_2 can be composed into a sequence; if this were not the case, and the copied items were then necessarily copied into a derived tree, then the MCTAG would be non-local, which would be undesirable from the point of view of constraining generative capacity.

This rules out some structures allowed when mapping between standard TAGs. For example, if we wanted to use the β**sPUs**[.] tree, the second element of the S_1 sequence would need to be substituted into this, which would in turn need to be adjoined into the first element of the sequence S_1, and this is not possible. This would mean that the elements of S_1 could not be in a sequence; and in order for the MCTAG to avoid being non-local, the entire BQSD corresponding to the SMP would need to be treated as elementary. This would lead to problems with attachment—raising the question, Which part of this BQSD should the sequence S_2 attach to?—and with addressing: whatever the choice, the result would be a non-standard MCTAG derivation tree, and would necessitate the redefinition of derivation in MCTAG. This is obviously highly undesirable, so the alternative is to go with the restriction that is inherent in the first mapping described between derivation trees, that fewer possible paraphrases are expressible in MCTAG because of the requirement for certain trees to be in the same sequence, which means that they cannot be composed together.

Note that this is not a counterexample to Theorem 7.1.1. What that theorem states is that it is possible to synchronise two LCFRSs by virtue of an isomorphism between the derivation structures, and have the weak language preservation property hold. However, the derivation structures in MCTAG are more limited because of the need to put trees in sequence to satisfy any copying requirement, as well as the need to have mappings that are linguistically meaningful rather than just arbitrary correspondences of bounded subderivations. Thus an acceptable BQSD with many-to-one links using MCTAG has an additional implicit condition: that a derived tree in an SMP, where the grammar is an MCTAG and where there are many-to-one links, must have a corresponding BQSD which contains validly composed tree sequences, such that these tree sequences allow composition of sequences of copied items as licensed by the diacritics on the derived tree pair.

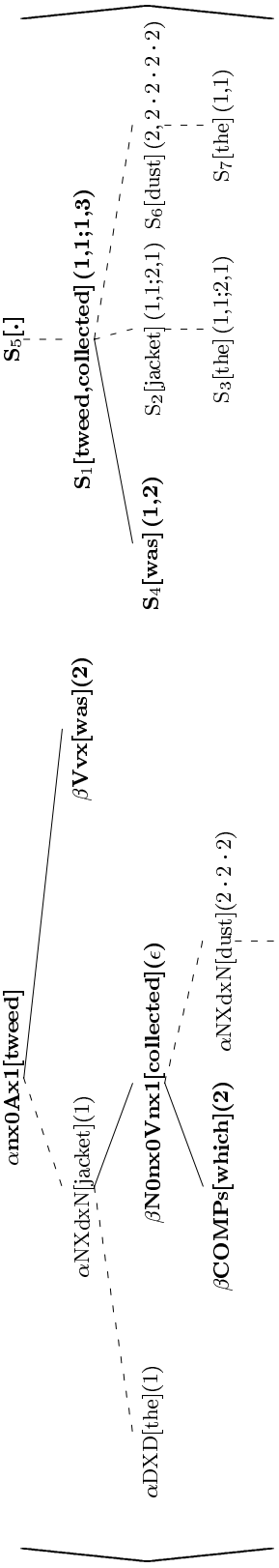


Figure 7.6: Derivation tree pair for (2)

$$
\begin{aligned}
S_1 &= \{\alpha\mathbf{nx0Ax1}[\text{tweed}],\ \alpha\mathbf{nx0Vnx1}[\text{collected}]\}\\
S_2 &= \{\alpha\mathbf{NXdxN}[\text{jacket}],\ \alpha\mathbf{NXdxN}[\text{jacket}]\}\\
S_3 &= \{\alpha\mathbf{DXD}[\text{the}],\ \alpha\mathbf{DXD}[\text{the}]\}\\
S_4 &= \{\beta\mathbf{Vvx}[\text{was}]\}\\
S_5 &= \{\alpha\mathbf{sPUs}[.]\}\\
S_6 &= \{\alpha\mathbf{NXdxN}[\text{dust}]\}\\
S_7 &= \{\alpha\mathbf{DXD}[\text{the}]\}
\end{aligned}
$$

Figure 7.7: MCTAG grammar sequences for Figure 7.6

7.1.2 Well-Formedness of S-TAG under Paraphrase with Copying

Expanding the term S-TAG to encompass mappings between different types of TAG, this section will demonstrate that S-TAG is well-formed when mapping between a TAG and an MCTAG, where sequences of identical items are used to satisfy many-to-one links. The proof will be fundamentally the same as in Theorem 6.4.4, showing that well-formed derivation trees are possible, and that there is an isomorphism between them licensed by the definition of paraphrase using SMPs and generated pairs. The difference is that generated pairs are now pairs of tree sequences, leading to a modified definitions for ψ and Ψ, the generating function and pairing function of Definitions 6.2.1 and 7.1.3. This section will thus first formally give these modified definitions, and show that the new ψ has the same properties as the original.

Definition 7.1.2 *Given some TAG* $G = \langle V_N, V_T, V_L, S, \mathcal{I}, \mathcal{A}, - \rangle$, *a* ONE-TO-$m$ GENERATING FUNCTION $\psi_m : \mathbf{N}_+ \times \mathcal{P} \times (\mathcal{I} \cup \mathcal{A}) \rightarrow (\mathcal{I} \cup \mathcal{A})^+$ *is defined as follows.*
For all $k \in \mathbf{N}_+$, $p \in \mathcal{P}$, $\gamma \in (\mathcal{I} \cup \mathcal{A})$, *and* $\gamma'_i = \gamma'$ *for* $1 \leq i \leq k$ *and any tree* γ',

$$
\psi_m(k, p, \gamma) = \begin{cases}
\langle \gamma_1, \ldots, \gamma_k \rangle & \text{if } \gamma(\epsilon) = p \\
\langle \text{morph}(p, \gamma_1), \ldots, \text{morph}(p, \gamma_k) \rangle & \text{if } p \in \mathcal{P}_p \text{ and } \gamma(\epsilon) \neq p \\
\langle \gamma''_1, \ldots, \gamma''_k \rangle & \text{if } p \in \mathcal{P}_\epsilon, \text{ where } \text{dom}(\gamma''_i) = \{\epsilon\}, \\
 & \quad \gamma''_i(\epsilon) = \langle p', \mathcal{I} \cup \mathcal{A}, \mathbf{false} \rangle \text{ and } p' \in \mathcal{P}_p \\
 & \quad \text{for } 1 \leq i \leq k
\end{cases}
$$

ψ_m works just as ψ, except that it takes an extra argument k, the length of the generated sequence. Each item in the sequence is identical. The value k is restricted such that it is equal to the largest value of m for the m-to-one links. The pairing function is similarly analogous.

Definition 7.1.3 *Given some TAG* $G = \langle V_N, V_T, V_L, S, \mathcal{I}, \mathcal{A}, - \rangle$, *a* ONE-TO-$m$ PAIRING FUNCTION $\Psi_m : \mathbf{N}_+ \times \mathcal{P} \times (\mathcal{I} \cup \mathcal{A}) \rightarrow (\mathcal{I} \cup \mathcal{A}) \times (\mathcal{I} \cup \mathcal{A})^+ \times \frown$ *generates an S-TAG triple as follows.*

For all $k \in \mathbf{N}_+$, $p \in \mathcal{P}$ *and* $\gamma \in (\mathcal{I} \cup \mathcal{A})$

$$\Psi_m(k, p, \gamma) = \langle \gamma, \psi_m(k, p, \gamma), \frown \rangle$$

where for all $d \in \mathrm{dom}(\gamma)$ *and* $d' \in \mathrm{dom}(\psi_m(k, p, \gamma))_i$ *for* $1 \leq i \leq k$, $d \frown d'$ *where* $d = d'$.

We now demonstrate the well-formedness of S-TAG in this context.

Lemma 7.1.4 *The derivation tree pairs of the S-TAG G under paraphrase with copying are well-formed.*

Proof: Take some TAG Gi from which the S-TAG G under paraphrase with copying is constructed. The proof is fundamentally the same as for Lemma 6.4.1. There are now sets of pairs of tree sequences associated with each $\gamma \in Gi$,

$$S = \{\langle \gamma, \psi_m(k, p, \gamma) \rangle \mid k \in \mathbf{N}_+, p \in \mathcal{P}\},$$

In recursively defining the derivation trees, the one for the left tree is the same as in Lemma 6.4.1; the one for the right tree has its part of speech p_i determined by the address t'_i, also as before, and the length of the tree sequence determined by the number of addresses t'_j with the same diacritic. The right tree node is thus uniquely determined by the tree corresponding to its parent in Υ, as in the proof of Lemma 6.4.1, and the rest follows. □

Theorem 7.1.5 *S-TAG under paraphrase with copying is well-formed and has the WLPP.*

Proof: This is fundamentally the same as for Theorem 6.4.4. The analogues of Lemmas 6.4.2 and 6.4.3 are straightforward, given that it is possible to have isomorphisms existing between grammars of differing generative capacities by Theorem 7.1.1, and using the same function δ as in Lemmas 6.4.2 and 6.4.3, arising from valid generated pairs and SMPs, for inducing the bijection.

Given the analogues of these lemmas, and Lemma 7.1.4, the proof of this theorem is also straightforward. □

A further extension that follows is to have Ψ_m generate not a sequence of identical items, but instead a predefined sequence. As an example, take (3), a modified form of (2), with the SMP of Figure 7.1.

(3) a. The green jacket which collected the dust was tweed.

b. The green jacket collected the dust. The jacket was tweed.

The S-TAG pairs corresponding to the subject NP are as α_3 and α_4 in Figure 7.5, with an extra tree pair for the adjectival modification, as in Figure 7.8.

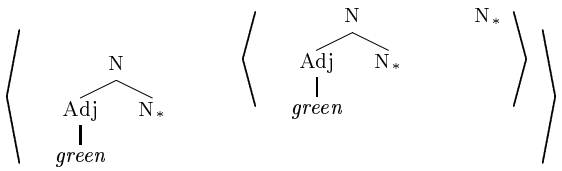


Figure 7.8: Adjectival modifier pair for many-to-one links

The function Ψ_m specifies which items are copied—as all items in the sequences in α_3 and α_4 of Figure 7.5, and the first item in the sequence of Figure 7.8—and which are replaced by 'null' trees which have no effect—as the second item of the sequence of Figure 7.8. Linguistically, this is determined by a ranking of which trees can in effect be 'deleted' first from a referring expression's derived tree.

7.1.3 Comments on Many-to-One Links

We noted earlier that an S-TAG projection that has an m-to-one link is able to generate, for some integer $k > 1$, the languages COUNT$-4k = \{a_1^n a_2^n \ldots a_{4k}^n \mid n \geq 0\}$ and k-COPY $= \{w^k \mid w \in \{a, b\}^*\}$; and as discussed under the paraphrase model presented in this section, this extension can be represented by MCTAG. This gives rise to a transduction between standard TAG and MCTAG, which could be viewed as undesirable: handling the m-to-one link phenomenon by increasing the generative capacity of the formalism. However, it is argued here that, given certain restrictions, this choice is one that is in fact appropriate and well-suited to paraphrasing as described.

These sorts of m-to-one links have been used in this section to model the coreference of two NPs, caused by splitting one sentence into two (or combining two into one). That is, the ability to perform a k-copy, or to count to $4k$, occurs when there are multiple sentences. In an informal way, it is possible to conceive of this 'extra power' as being 'distributed' over the multiple sentences so that each sentence has the 'power' of a standard TAG allocated to it. That is, if there are k sentences, and k-to-one links allowing languages like COUNT$-4k$ and k-COPY, this can be seen as it being possible to allocate the ability to count to four to each of the k sentences—the power of a standard TAG—or having one copy of a string $\{a, b\}^*$ per sentence.

7.1.4 Applications

The previous sections have shown that it is possible, and within the spirit of the formalism, to extend S-TAG to beyond mapping between just two TAG derived tees, including to cases of mappings between grammars with differing generative capacities. In particular, mappings between standard TAG and set-local MCTAG are shown to be possible, maintaining the weak language preservation property, and moreover are a natural way to represent the many-to-one links necessary for representing such phenomena as the coreference that occurs when paraphrase involves multiple sentences.

One application of this S-TAG definition using set-local MCTAG is in dealing with coor-

dination generally in TAG, not just in paraphrase. As mentioned in Section 5.4 and at the beginning of Section 7.1, there have been several treatments of coordination in the TAG literature, as it has been difficult to find a natural expression for coordination in the formalism. The most recent approach, in Sarkar (1997), uses a form of S-TAG, termed SLTAG, in a way similar to that proposed here for mapping a single sentence to two.

However, the formulation in that paper uses extra mechanisms to model coordination, such as declaring two different types of links which have different inheritance properties when trees are composed. Also, the derivation structures are directed acyclic graphs, which raises significant unresolved problems—in terms of language preservation when synchronising structures, and in terms of the formal properties of the structures derived from them—because they are unlike any structures previously used to represent derivations in TAG. A fundamental cause of the problems is the multiple occurrence of diacritics in a single tree, as seen in the diagrams in Sarkar (1997), repeated here as Figures 7.9 through 7.12: the 'tangles' are caused by drawing links from all identical diacritics to a single child tree.

The formulation here—mapping between a standard TAG and an MCTAG—could easily be recast to give an alternative conception of coordination, by merely replacing the punctuation trees conjoining two sentences with a conjunction tree. A standard derivation structure falls out naturally, as in Figure 7.6, and the weak and strong generative capacities are known and well-defined, and suitable for coordination in the same way as for multiple sentences. The derivation tree also captures information about coreferent expressions, a feature which is an attraction of the Sarkar proposal: instead of representing coreference by links to those nodes in the derivation structure which have as a child the subtree representing the entity, this information is represented in the addressing notation of the MCTAG derivation tree, with trees in the same sequence mapped by Ψ_m being coreferent. Obviously, however, more investigation is needed to determine the extent of the suitability of such a technique to model all aspects of coordination.

A second application of the results of this section is in machine translation. Existing machine translation work using S-TAG, such as that of Palmer (1998), has discussed mapping between standard TAG and tree-local MCTAG, even though this is not actually licensed by the definition of Shieber (1994); most likely, this is done on the basis that they are similar and related formalisms with the same weak generative capacity. The results of this section prove that these do form valid synchronised mapping pairs, on the basis of their derivation structures. Moreover, this can be extended to mappings between languages which are currently represented within formalisms with different generative capacities. For example, there are proposals to use set-local MCTAG to model clitic-climbing in Spanish (see Bleam, 1994; see also Kulick, 1998, for an alternative view). It may also be possible to extend the results to show that there can be mappings between more radically different formalisms: for example, between standard TAG and D-Tree Grammar (DTG) (Rambow *et al*, 1995), to allow a formally well-founded basis for translation between, say, English and German, or English and Korean, with both of the non-English languages having DTG-like models to handle phenomena such as scrambling.

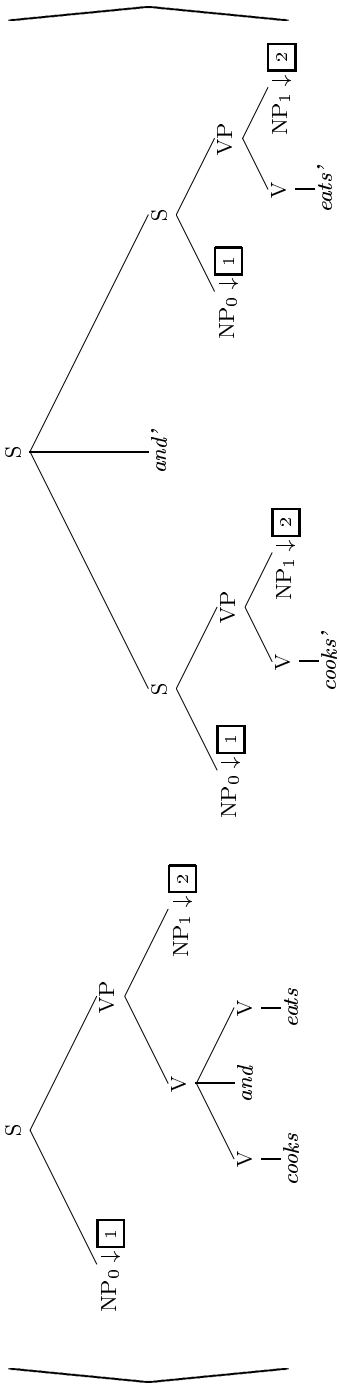


Figure 7.9: Coordination using LSTAG: derived tree pair

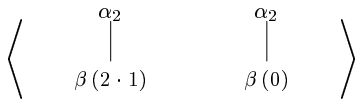


Figure 7.10: Coordination using LSTAG: derivation tree pair

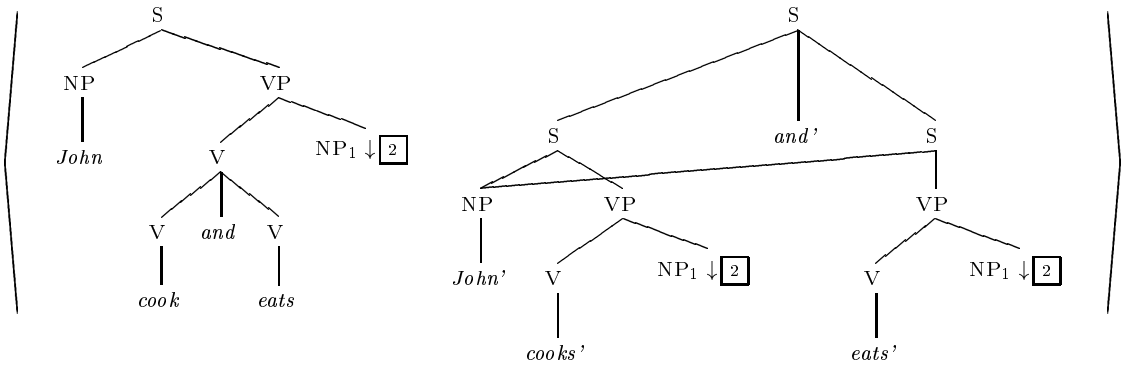


Figure 7.11: Coordination using LSTAG: derived tree pair

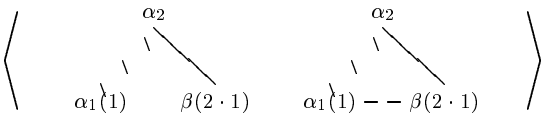


Figure 7.12: Coordination using LSTAG: derivation tree pair

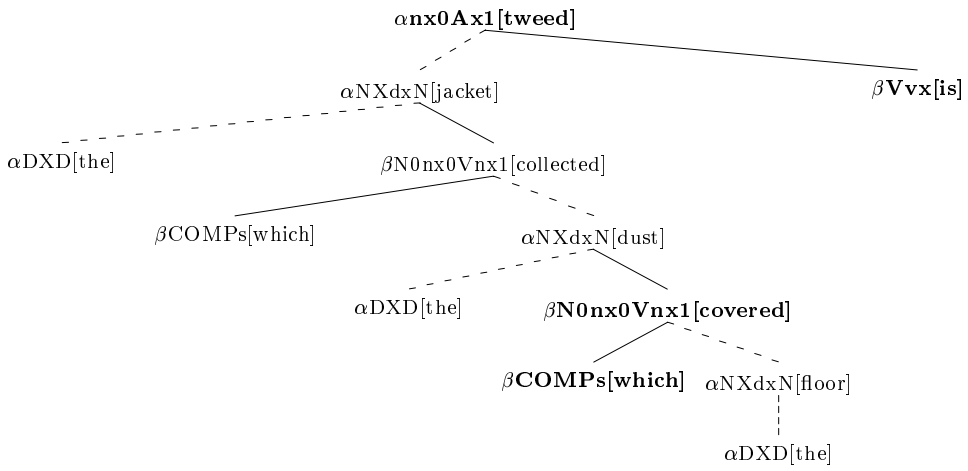


Figure 7.13: Derivation tree for (4a)

7.2 Extending S-TAG for Unbounded Separation

Section 6.2.2 discussed a problematic operation that occurs in paraphrase, that of promoting arbitrarily deeply embedded relative clauses to form separate sentences. This operation causes the same problems for the modified S-TAG definition of Shieber (1994) that the category of clitics does, in that an isomorphism between the derivation structures cannot be established without some restriction on the phenomenon. Section 6.2.2 mentioned as a possible solution, by recognising that the unbounded phenomena result in discontinuous regions that can nonetheless be considered as having the same domain of locality, that a TAG derivation structure may be more appropriate. The standard S-TAG definition, however, is based on context-free derivation trees; hence the inferior solution of Section 6.2.2 was to use these context-free derivation trees, with a restriction on the operation of relative clause promotion. This section will generalise S-TAG so that the derivation structures themselves can be TAGs, and will further show that this is possible without increasing the generative capacity of the resulting object-level structures.

7.2.1 TAG Meta-Derivation Structures

The example from Section 6.2.2 is reproduced here.

(4) a. The jacket which collected the dust which covered the floor was tweed.
b. The jacket which collected the dust was tweed. It covered the floor.[3]

These sentences have the derivation trees of Figures 7.13 and 7.14.

As noted before, with unboundedly many relative clauses allowed, there will be unboundedly many nodes—in this example, nodes of type α**NXdxN**, α**DXD**, β**N0nx0Vnx1** and

[3]We are again using the pronoun *it* rather than the less ambiguous referring expression*the dust*. Using *the dust* would involve many-to-one links, as discussed in Section 7.1, whereas the fixed pronoun *it* does not need these. So in the interests of a simpler exposition in this section, *it* will be used.

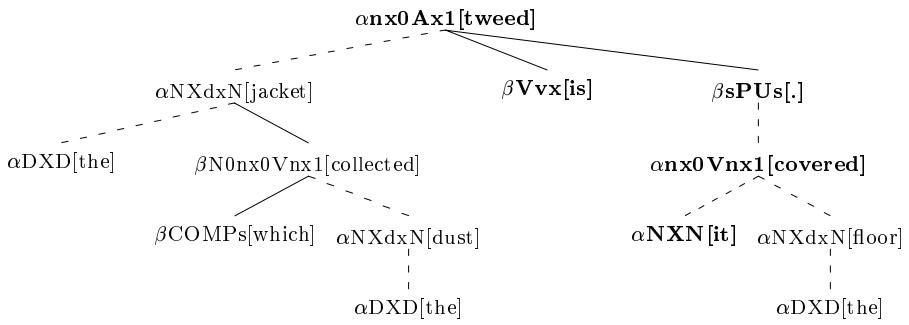


Figure 7.14: Derivation tree for (4b)

βCOMPs—between the two halves of the discontinuous region which correspond to the structural mapping tree.

The first solution, used in Section 6.2.2, was to treat the discontinuous region as a single constituent by restricting the intervening region to being only a fixed number of nodes. This corresponds to only allowing relative clause promotion to sentence up to a particular level of embedding; arbitrarily deep promotion cannot be carried out. This is obviously especially unsatisfactory in the parallel case of modelling clitics, which cannot be made to artificially limit their movement.

The second approach uses the fact that the two discontinuous halves of the region are actually related, notwithstanding intervening material: the two halves form a single 'substructure', and what the synchronisation of TAG grammars should do is establish an isomorphism between corresponding substructures in trees. Clearly, the degenerate case, where substructures are single nodes, is the basic S-TAG; and Shieber's idea of bounded subderivation is a limited attempt to capture substructure. In order to capture substructure fully, what is needed is a METAGRAMMAR—a grammar at the meta level—and the nature of the substructure here, discontinuous regions which are related regardless of intervening material, is suggestive of TAG's extended domain of locality. In fact, this appears to be a fairly natural way to treat the problem. This solution thus involves defining an elementary tree at the meta level which describes the disconnected constituents, and auxiliary trees to factor out the recursion.

For (4a), a TAG describing the derivation trees of Figure 7.13 might look something like the one presented in Figure 7.15. In this meta-grammar[4], the tree α_1 represents the grouping of disconnected constituents involved in the structural mapping of arbitrarily deeply embedded relative clauses. It contains nodes representing the discontinuous groups (the first group, starting from the root, of nodes **αnx0Ax1** and **βVvx**; the second group, representing the relative clause, of nodes **βN0nx0Vnx1** and **βCOMPs**), as well as the nodes filling the slots that connect them together (the upper **αNXdxN**, and **αDXD**) and nodes for connecting any other tree (the node **αNXdxN** marked for substitution). The tree representing the unbounded intervening material is given by tree β_1: this is an auxiliary tree, as this is the typical way of representing factored-out recursion in TAG.

[4]Note that, because these are meta-level trees, it is the nodes describing object-level trees, rather than the terminal symbols on the frontier of the tree, that are of interest.

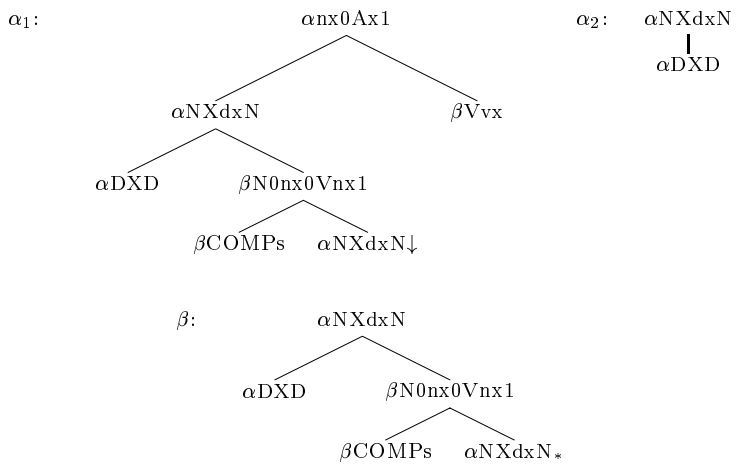


Figure 7.15: Meta-grammar for (6a)

The other tree here, α_2, is typical of any other tree in this meta-grammar that does not represent discontinuous groups or slots: it is an initial tree, of depth one, substituted into another tree; note that this is actually just the operation of concatenation that occurs in the construction of context-free derivation trees in standard TAG. A metagrammar for (4b) is more straightforward, with no trees with discontinuous regions; it is given in Figure 7.16. In defining an S-TAG metagrammar, then, the trees would be paired as described above, with diacritics indicating the points of simultaneous composition (for example, $\boxed{1}$ at α**NXdxN** and $\boxed{2}$ at α**NXdxN**$\downarrow$ in α_1 of Figure 7.15; similarly for Figure 7.16, with the $\boxed{1}$ on the α**NXdxN** that is the leftmost child of the root).

Note that for both projections, this is not the only metagrammar that could be defined.

For the purposes of Synchronous TAG, then, what we would want to do is synchronise the derivation structures of these meta-level grammars: as the meta-level is a pair of TAGs, the meta-meta-level will be a pair of context-free trees, and can be synchronised in the standard way. The meta-meta-level tree pair is as in Figure 7.17, which for this example uses the meta-level trees of Figures 7.15 and 7.16 for describing paraphrase pair (4). In a pair, the meta-level initial trees representing the SMP (as α_1) are mapped to each other; the single level trees that are concatenated (as α_2) are mapped to the corresponding single level trees that come from generating functions; and the multi-level auxiliary trees representing slot fillers (as β_1) are mapped to multi-level initial trees where each element of these meta-level trees corresponds by virtue of being related by the function Ψ. Note that an unbounded amount of intervening material (*The jacket which ... which ... which ... which ...*) will become just a chain of βs in both projections, which causes no difficulties in terms of a single node isomorphism.

A general method for constructing these TAG grammars for derivation trees—a TAG metagrammar—is then as follows.

Construction 7.2.1 *To build a TAG metagrammar:*

1. *An initial tree in the metagrammar is formed for each part of the derivation tree*

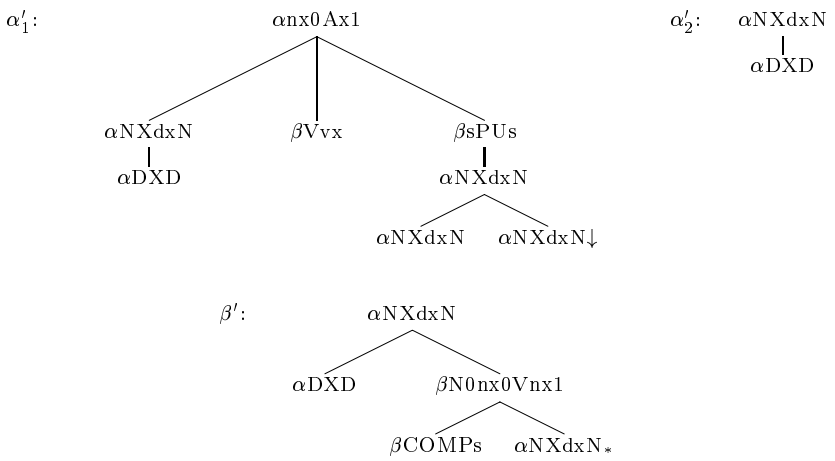


Figure 7.16: Meta-grammar for (6b)

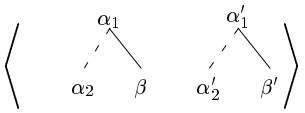


Figure 7.17: Meta-meta-level derivation tree pair using grammars of Figures 7.15 and 7.16

corresponding to the substructure (BQSD) of an SMP, including the slots so that a connected tree is formed. Any node that links these parts of the derivation tree to other subtrees in the derivation tree is also included, and becomes a substitution node in the metagrammar tree.

2. *Auxiliary trees are formed corresponding to the parts of the derivation trees that are slot fillers along with the nodes in the disconnected regions adjacent to the slots; one connected auxiliary tree is formed for each bounded sequence of slot fillers within each substructure (BQSD).*

3. *The remaining metagrammar trees then come from splitting the derivation tree into one-level trees, with the nodes on these one-level trees in the metagrammar marked for substitution if the corresponding nodes in the derivation tree have subtrees.*

Conceptually, this construction builds a metagrammar that is essentially context-free, like the standard derivation tree—the one-level trees in the metagrammar composed by substitution essentially mimic a context-free tree built by concatenation—except where TAG power is necessary for capturing unbounded dependencies.

7.2.2 An Infinite Progression of Synchronised TAGs

In fact, there is an infinite hierarchy of S-TAGs—not just the one extra level, the meta-meta-level used in the example in Section 7.2.1—and in this section we will establish this as a formal result. First, as a preliminary result, we will show that synchronisation can only occur between 'compatible' pairs of trees.

The ideas of this section draw on the infinite progression of grammars, between CFGs and indexed grammars (IGs) in generative capacity, that are presented in Weir (1988: 44–48). To develop this progression, Weir noted that for TAGs the derivation structure and the yield are separate.[5] He further noted that:

> Local sets have path sets that are regular languages, and string languages [the yield of the derivation tree] that are CFLs. TAG tree sets have path sets that are CFLs, and string languages that are TALs. We will continue this progression, characterising at the next stage tree sets with path sets that are TALs, and string languages that fall in some classes larger than TALs, say $\mathcal{L}_3$. At the ith stage, we have tree sets with path sets in the class $\mathcal{L}_{i-1}$, and string language in $\mathcal{L}_i$. The tree sets at every stage should have independent paths, and be labeled by members of a bounded sized alphabet. [Weir, 1988: 45]

The yield function for each level is given by composing the yield functions for previous levels: for example, a yield function whose output is TAG derivation trees would be of the appropriate type to be composed with the standard TAG yield function (which takes a tree, such as a TAG derivation tree, and reads off the nodes to produce a TAG tree), followed by the standard CFG yield function (which takes a tree and reads off the nodes

[5] This was a condition placed on LCFRSs in general, by Restriction 1 of the definition in Weir (1988: 94).

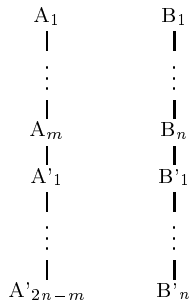


Figure 7.18: Trees with different path set languages

to produce a string language); composing these gives a complex yield function that can be applied to a local set to produce $\mathcal{L}_3$.

The properties of the languages produced by these higher level TAGs have not been investigated, although Rogers (1998) has given an algebraic characterisation of the progression.

Given this hierarchy, an obvious step is to investigate the synchronisation of elements of this progression. We will define a CHAIN to be an initial local set, and some number of tree sets recursively produced by applying a TAG treeyield function. Synchronisation will then be between two chains. In carrying out this synchronisation, it is not possible to arbitrarily choose one level from each chain and synchronise the chains at those levels. For example, assume we have two chains, one producing $\mathcal{L}_2$ as its string language (that is, a standard TAG), and the other $\mathcal{L}_3$. Then we take the tree sets that are two removed from production of the string language; these are TAG derivation trees (local sets) and TAG derived trees respectively. It is not possible in general to synchronise trees from these tree sets. For example, take the simple trees in Figure 7.18: even though the trees have the same branching structure, and the same number of nodes, it is not possible to induce an isomorphism between the two, as the mapping function cannot 'count', in order to equalise the number of As and A's in the left tree. In this example, although we can specify that all the trees in the right tree set are of that form (that is, with path sets of the form $A^n(A')^n$), as the path sets of TAG trees are context-free, we cannot do so with the left, as the path sets here are regular. A general proof is given below.

First, define the ORIGIN of a chain to be the local set that is the ultimate derivation tree. The DISTANCE of a tree set from the origin is the number of levels from the tree set to the origin, not including the tree set; the HEIGHT of the chain is the distance from the tree set producing the final string language $\mathcal{L}_i$ to the origin, that is, i.

Lemma 7.2.2 *There exists no isomorphism between tree sets T_1 and T_2 from chains P_1 and P_2, where T_1 and T_2 have different distances from the origin.*

Proof: The path sets of T_1 and T_2 are characterised by grammar formalisms of different generative capacities, under the definition of MLTAG from Weir (1988: 45). Assume, without loss of generality, that T_1 is characterised by a more tightly constraining grammar. A homomorphism from T_1 to T_2 would result in T_2 having the same structure, and being representable by the same grammar formalism, as T_1, since any grammar part

of an abstract family of languages is closed under homomorphism (Salomaa, 1973: 23, Theorem 3.5); and this is not the case. Therefore no isomorphism can exist. □

Here, we will restrict ourselves to looking at chains of the same length, and by convention will synchronise at the origin of each chain. Note that we could equally well synchronise tree sets at the same distance from the origin instead; however, given that results exist for substitutions and homomorphisms of local sets, the origin is an easy place to start.

Theorem 7.2.3 *There is a countably infinite number of synchronised chains of equal length.*

Proof: Each of the two chains in a synchronised pair is one of a countably infinite set by Weir (1988: 45). These chains are synchronised at their origins, as a consequence of Lemma 7.2.2. Thus there is an isomorphism between the set of individual chains, and the pairs of chains associated with it, and hence a countably infinite number of synchronised chains of equal length. □

Given that we are here synchronising in the same way as for the paraphrases under standard S-TAG as in Chapter 6—that is, that the synchronisation occurs between two local sets—it should be clear that the results there will carry over to this infinite chain also.

From the example given in Section 7.2.1, it should be apparent that we are only interested in going one level beyond standard TAG derivation trees; that is, to chains of height three. Apart from the fact that this is sufficient for solving problems of arbitrarily deep embedding, there is an intuition about why, for paraphrasing in general, this is so:

Even though mappings are explicitly specified at the derived tree (object) level, they are also to some extent specified at the derivation tree (meta) level. In the paraphrase model of this thesis, this is explicitly stated, in definitions of what constitutes an acceptable paraphrase in terms of combining elementary trees, with derivation trees being the embodiment of these substructures of acceptable combinations of elementary trees. In machine translation, this is more implicit, with previous work, such as that of Abeillé (1990), merely stating the derived trees in complex mappings in terms of their constituent trees. Given that the derivation (meta) level specifies the acceptable substructures, what we actually want to synchronise are the derivation trees of the derivation trees, that is, the origin of a chain of height three. This was not immediately apparent in previous work, as in applying S-TAG to machine translation (at least with closely related languages like English and French) many of the mappings are between elementary trees, which correspond to single nodes at the derivation level, which thus pose no problem in inducing an isomorphism licensed by S-TAG pairs. However, when the mappings correspond to more complex derivation tree structures, as is particularly the case for paraphrase—but also for more complicated linguistic phenomena like clitics—this generalisation emerges.

One consideration not yet addressed is the generative capacity of the resulting grammar if we have chains of height three, rather than the height two of standard TAG. Weir (1988) shows that $\mathcal{L}_i$ is a superset of $\mathcal{L}_{i-1}$; thus we will have languages with a greater generative capacity than standard TAGs generated by the S-TAG pairs. Even though some features of the language $\mathcal{L}_3$ are known—for example, that it has the property of semi-linearity (Weir, 1988: 96, Theorem 4.3.1), which is desirable for a grammar modelling natural language (Joshi, 1985)—the properties of this language have not in general been much investigated, nor its overall suitability for modelling natural language.

However, it is possible to remove this potential problem by observing that it is not necessary to use the full power of TAG at the derivation level. Construction 7.2.1 in fact preserves the context-free nature of standard TAG derivation trees, as shown below, in spite of being expressed in the TAG formalism.

Lemma 7.2.4 *The metagrammar given by Construction 7.2.1 is in the regular form of Rogers (1994).*

Proof: Any grammar which does not allow adjunction at any node along the spine of an auxiliary tree is in regular form (Rogers, 1994: Definition 1 and Lemma 4). By Definition 6.2.3, the bounded sequence of slots within a BQSD is headed by a node corresponding to an initial tree, thus the auxiliary trees of the metagrammar all have root and foot nodes corresponding to initial trees in the original object-level TAG grammar, by Construction 7.2.1. By the Condition on Paraphrase Pair Minimality (Condition 6.2.5) and Definition 6.2.3, the slots along a spine within a BQSD, apart from the topmost, will be a minimal number of nodes corresponding only to auxiliary trees from the original object-level TAG grammar: if any were initial trees, the corresponding derived tree in the paraphrase pair would not be connected. Thus no auxiliary tree in the metagrammar can adjoin into the spine of another, apart from at the root, and the metagrammar is consequently in regular form. □

Theorem 7.2.5 *The language generated by each projection of a paraphrase pair under S-TAG, with TAG derivation trees as under Construction 7.2.1, is a TAL.*

Proof: The metagrammar is in regular form, by Lemma 7.2.4. Thus the trees created by it are recognisable sets (Rogers, 1994: Proposition 1). These are the derivation trees for the projections of the paraphrase pairs, and since they fit the standard definition for TAG of being context-free, the resulting object-level trees are TAG trees, and their string languages are TALs. □

Thus no increase in generative capacity has been caused by this extension of S-TAG, which involves building a metagrammar to resolve problems with isomorphism.

7.2.3 Applications

This section has proved the existence of an infinite synchronised chain of TAG-related grammars. This extension provides a way to model unbounded phenomena that have previously caused problems for the S-TAG formalism, and moreover one that is natural: under the standard definition of S-TAG, the derivation level is used both for specifying arrangements of elementary trees for syntactic reorganisation, and also for synchronising the two grammars. By leaving the derivation level to specify elementary tree grouping, and moving to the meta-meta-level for synchronisation, a number of problems are solved. Within the paraphrase application, the promotion of arbitrarily deeply embedded relative clauses becomes possible; within machine translation, the unbounded derivation behaviour of clitics and consequent difficulties for the isomorphism requirement, as noted by Shieber (1994), are no longer problematic.

7.3 Representing Paraphrase Types

This chapter and the previous one have given formal results demonstrating the well-formedness of S-TAG under paraphrase, and generalisation of the S-TAG formalism to account for various phenomena that occur in paraphrase (as well as elsewhere). This section will use the definitions that have arisen as part of these formal results to show how the paraphrases presented in the taxonomy of Section 4.2 can be represented, and how their effects, as described in Section 4.3, can be defined.

This S-TAG-based represention of paraphrase in this thesis has some similarities to other representation schemes for individual paraphrase types. The S-TAG-based formalism is more general than these specific representation schemes, giving a uniform representation for all of the types of syntactic paraphrase discussed, and potentially for syntactic paraphrases beyond these; but some of the more TAG-related alternative representation schemes will be discussed in the relevant subsection below.

The following subsections present examples from the taxonomy of Section 4.2 represented in the generalised form of S-TAG, in order to give an idea of how each type is handled, along with some discussion of other potential representation schemes specific to that type; and then defines functions which map from the structures of the S-TAG representation to values which can be used in the optimisation model of Chapter 8.

7.3.1 Representation of the Paraphrase Taxonomy

Type A: Change of Perspective

The light verb to full verb paraphrases are fairly typical of this perspective-altering type. Take (5); the text in square brackets is not represented, although doing so would be straightforward.

(5) a. Steven made an unwilling concession to us [that we were right].
b. Steven unwillingly conceded to us [that we were right].

The trees giving the parses for the two sentences are as in Figure 7.19.

The SMP, and the corresponding BQSD pair, are in Figure 7.20. The generated pairs are straightforwardly derived from the trees in Figure 7.19, and are shown in Figure 7.21. The two α**NXN** trees generate tree pairs with ψ the identity on the secon argument. The β**An** and β**ARBvx** trees form a pair with one as the generator and ψ taking the label of the root of the other as the part of speech argument to generate the pair, with which one as generator and which one generated depending on the direction of paraphrase. The α**DXD** tree generates a pair by ψ with the null part of speech ϵ_V.

Type B: Change of Emphasis

The example paraphrase for this section will be the mapping between an unmarked and a clefted version of the same sentence, which is likewise fairly similar to others of the type. This type of paraphrase is the one that can easily be represented by the largest number of existing specific representation schemes, although most of the specific representation schemes are not actually used for this purpose.

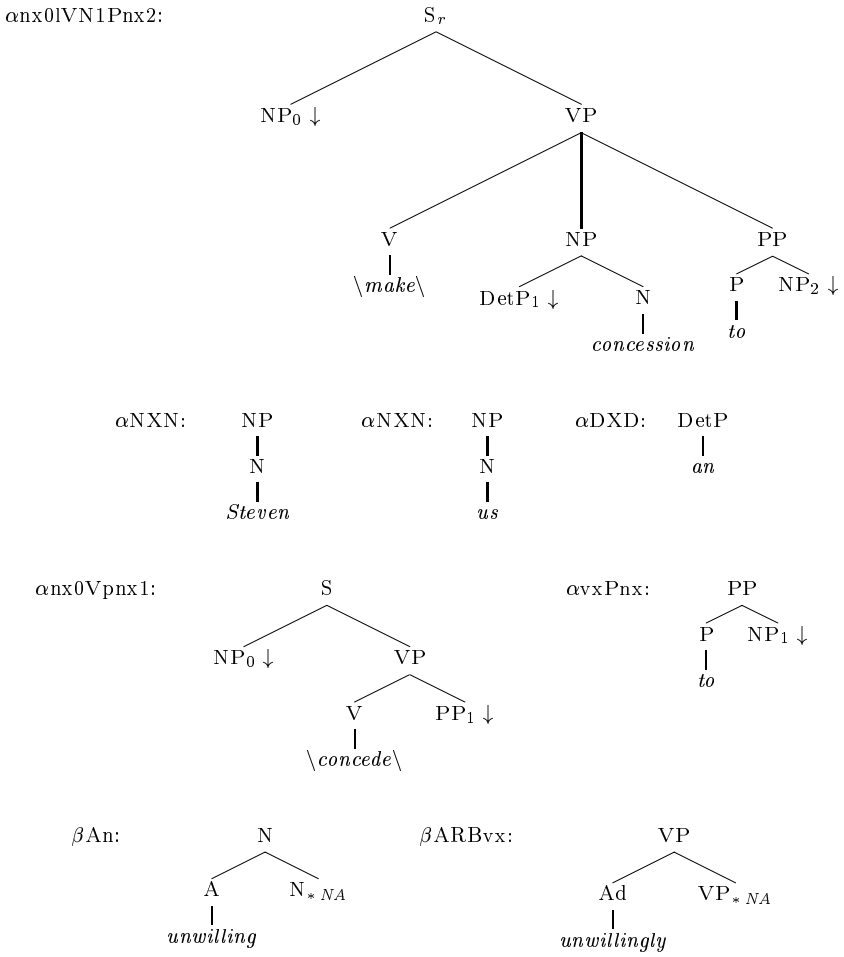


Figure 7.19: Elementary trees for (5)

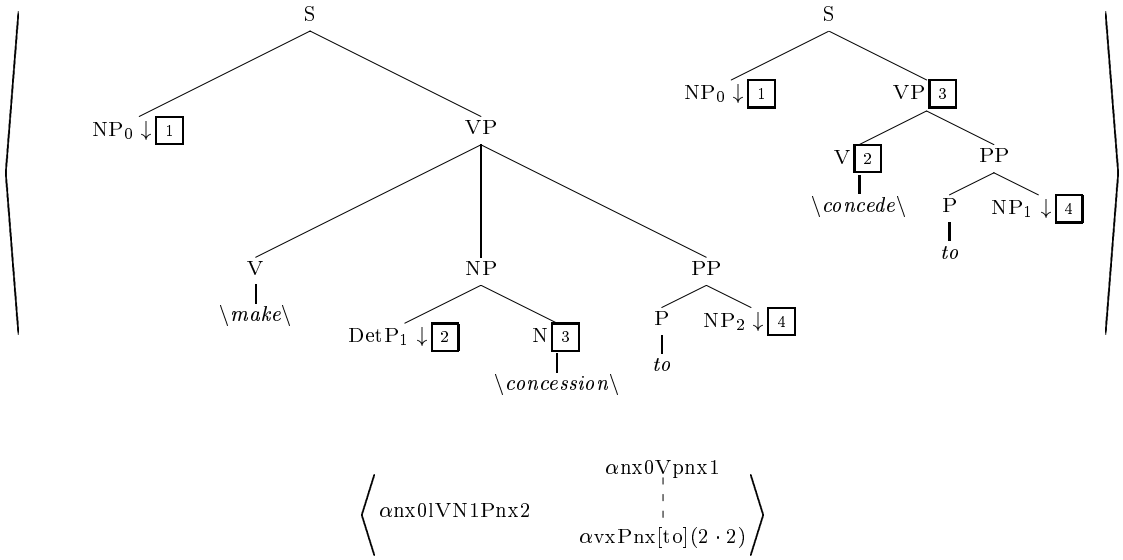


Figure 7.20: SMP and corresponding BQSD pair for (5)

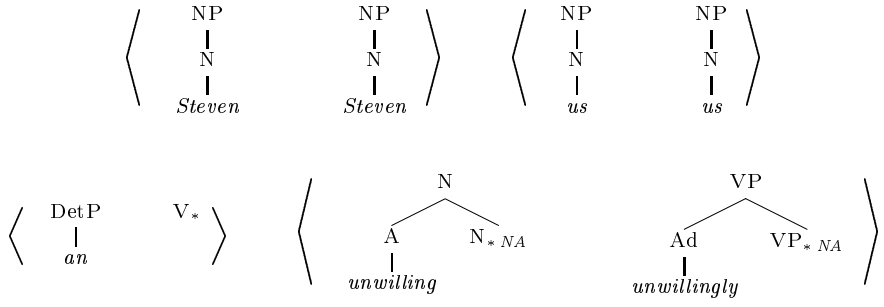


Figure 7.21: Generated tree pairs for (5)

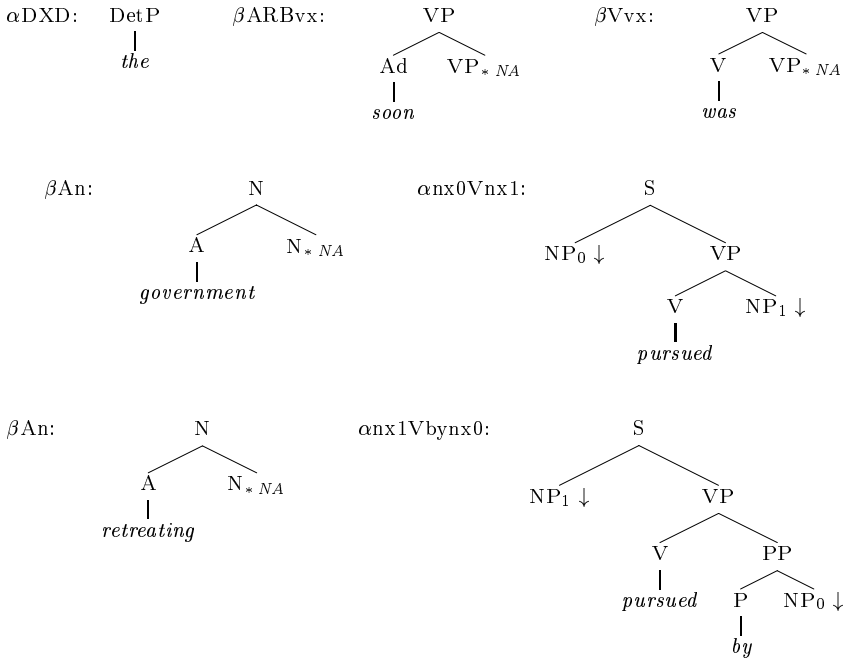


Figure 7.22: Elementary trees for (6)

As mentioned earlier, there are a number of schemes within the TAG framework which have been used to compactly encode TAG trees: quasi trees (Vijay-Shanker and Schabes, 1992), DATR (Evans *et al*, 1995), the hierarchical approach of Candito (1996), the partial-tree descriptions of Xia *et al* (1998), and so on. The motivation for these is to capture the dimensions of commonality among trees, so that when a change is made to a characteristic, it is propagated appropriately. For example, the extraposition characteristic, which is shared by many verb tree families, is separated out of individual trees, and recombined with base trees to form the original TAG tree.

Such representation schemes could easily be adapted for use here. However, they are both more general and more specific than the S-TAG formalism: more general in the sense that they by default give 'paraphrases' between texts which are not considered paraphrases here, such as between declarative and interrogative versions of a sentence; and less general, in the sense that they only deal with paraphrases of Type B.

The example paraphrase is given in (6); the trees in Figure 7.22; and the SMP and BQSD pair in Figure 7.23. All other tree pairs are generated by ψ the identity applied to the trees not included in the BQSD pair.

(6) a. The government forces soon pursued the retreating guerillas.

b. The retreating guerillas were soon pursued by the government forces.

An interesting characteristic of this type of paraphrase is that in XTAG, independently of the taxonomy of Section 4.2, the relations between the SMP mapping elements are almost always expressed within tree families: in the example in this subsection, the trees

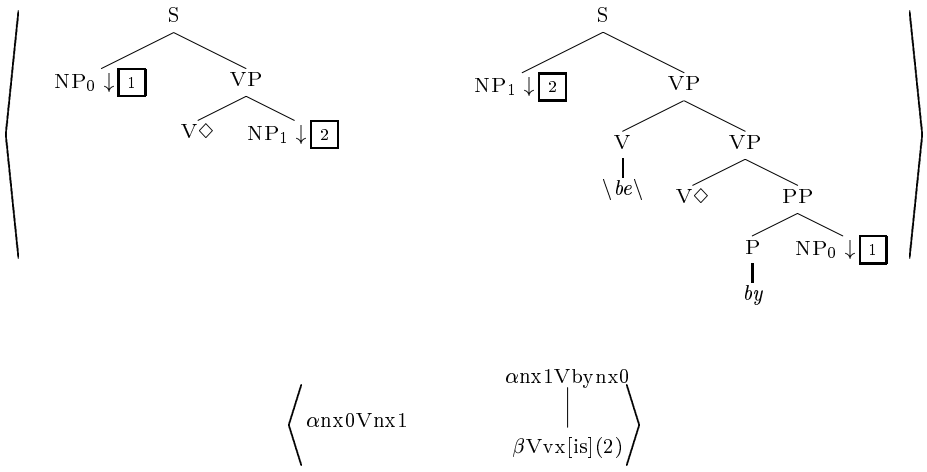


Figure 7.23: SMP and BQSDs for (6)

αnx0Vnx1 and **αnx1Vbynx0** are both from the same tree family, **Tnx0Vnx1**, that of transitive verbs.

Type C: Change of Relation

An example of this type of paraphrase, involving sentence splitting / combining, has been discussed extensively in Sections 6.2.2 and 7.2.1, and won't be repeated here.

A specific representation scheme for this type of paraphrase has been proposed in Chandrasekar (1994) and Chandrasekar *et al* (1996) using a finite state grammar approach. A later incarnation in Chandrasekar and Srinivas (1997), and Dras (1997a), independently propose TAG-based representations; the representation of this thesis is a development of that of Dras (1997a).

The main difference between the two approaches is that Chandrasekar is only interested in paraphrases of this type, and has the aim of simplifying text in a practical way, in order to make easier a variety of NLP applications such as machine translation. The representation scheme of this thesis is a more general one, covering a variety of different paraphrase types. This more general representation is necessary because of the RP framework, where it is not just text simplification that is the goal, but the modelling of conflicting constraints, of which simplification can be considered a subgoal.

More importantly, RP differs in that it concentrates on investigating the formal properties of the representation: showing that the weak language preservation property holds when paraphrasing, proposing a formally acceptable generalisation to allow subject copying, and so on. In Chandrasekar's representation, there is no constraint on transformational rules, and on what they do to the text, which is acceptable for a practically-oriented system; in this thesis, however, we are interested in developing a formalism that is in the constraining spirit of TAG, where predictions can be made about formal aspects of the operation of paraphrase.

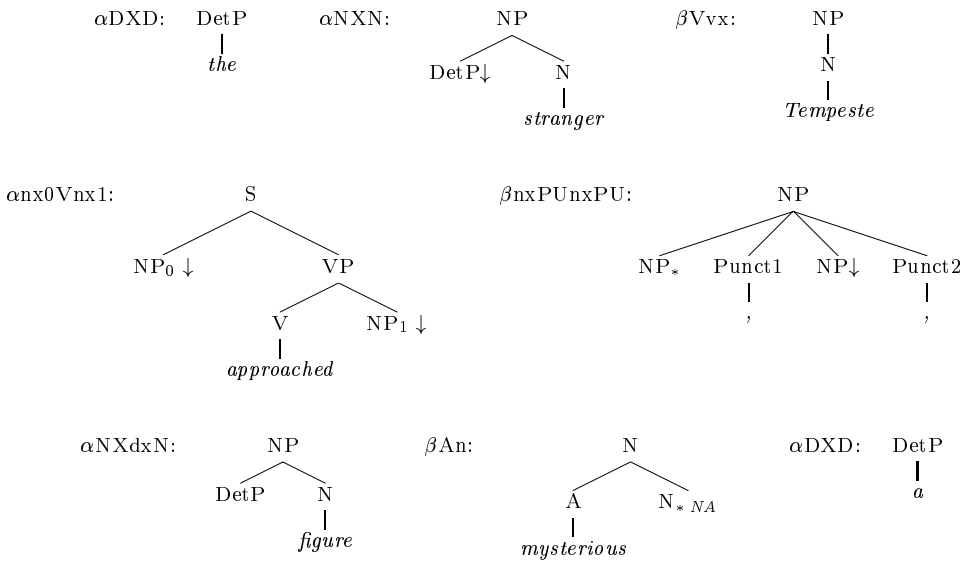


Figure 7.24: Elementary trees for (7)

Type D: Deletion

These are the deletion paraphrases; the example for this subsection is (7), the deletion of an NP appositive. TAGs have the capacity to describe constituents using the feature structures in order to identify functional roles of these constituents, and that could be used here—for example, there could be a functional feature with the value 'appositive' at the appropriate NP node—but we will only use purely syntactic information.

The trees are in Figure 7.24; the SMP and BQSDs are in Figure 7.25.

(7) a. The stranger, a mysterious figure, approached Tempeste.

b. The stranger approached Tempeste.

Again, the generated pairs are straightforwardly derived from the trees of Figure 7.24. Those that become subtrees dominated by the node $\mathbf{NP}_2$ in the right tree of the SMP are generated by ψ with a null part of speech. Note that the diacritic $\boxed{3}$ is placed on the **V** anchor node, so that ψ can only lead to a tree pair as in Figure 7.26.

Type E: Clause Movement

These paraphrases just involve movement of clauses. The example is (8), the corresponding trees in Figure 7.27, and the SMP and BQSDs in Figure 7.28.

(8) a. Keith smiled after thinking [for a moment].

b. After thinking [for a moment] Keith smiled.

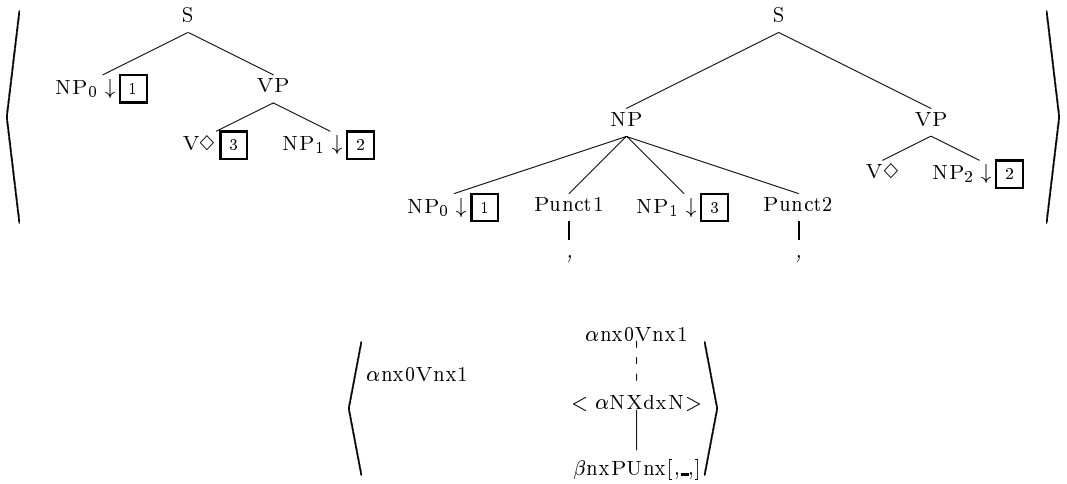


Figure 7.25: SMP and BQSDs for (7)

Figure 7.26: Tree pair derived from α**NXdxN** in Figure 7.24

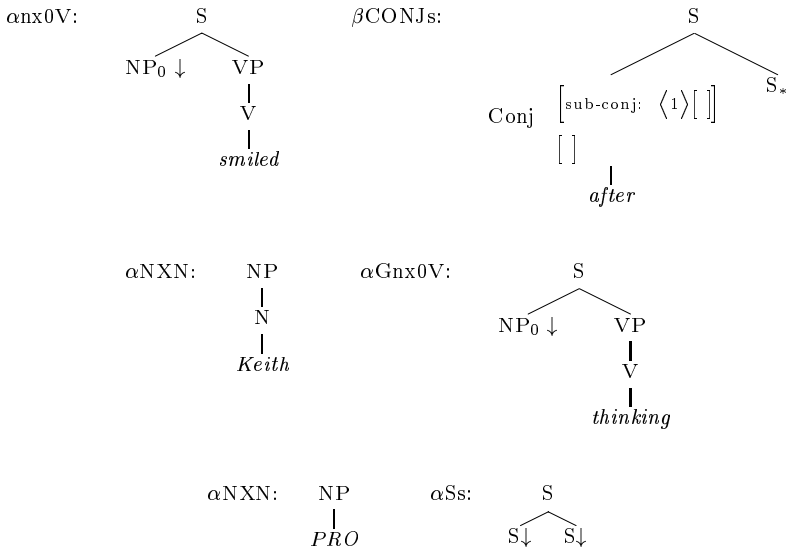


Figure 7.27: Elementary trees for (8)

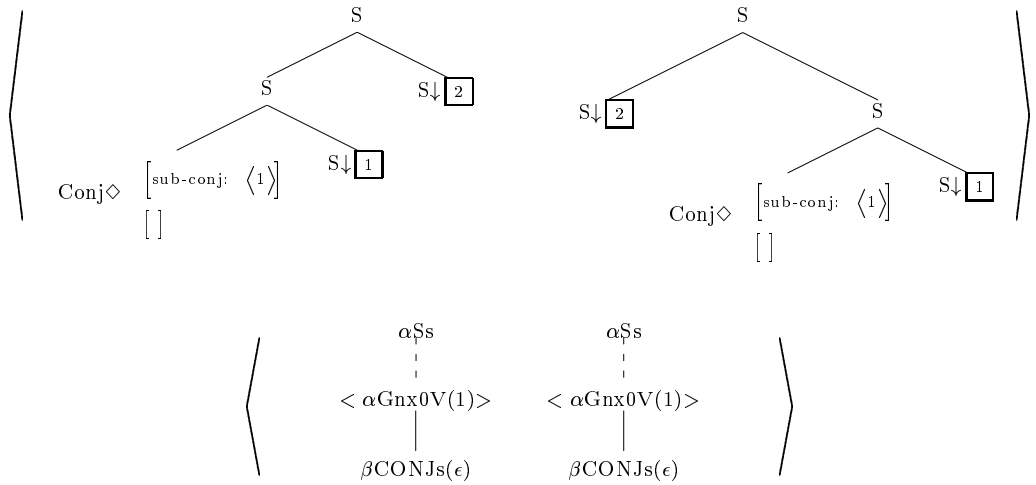


Figure 7.28: SMP and BQSDs for (8)

Figure 7.29: Tree pair with null tree

7.3.2 Paraphrase Effect Function

As noted earlier in earlier chapters, we want to have a measurable notion of paraphrase effect, in order to be able to choose a minimally-altering combination of paraphrases satisfying a set of global constraints. Section 4.3 proposed a semantic model which allows a rough quantification of paraphrase effect, based on the part of speech of altered constituents. Given the S-TAG representation of paraphrase, then we can define a function from tree pairs to a vector of values to provide this quantification.

In essence, what this function captures is those constituents that are eliminated (or introduced in a default way) as part of the paraphrase. In the S-TAG representation, this is represented by tree pairs generated by ψ with the null part of speech; for example, the tree pair in Figure 7.29.

Additionally, the use of particular trees or tree pairs that correspond to a specific paraphrase effect, as for the introduction of a default anaphoric element, or for pairs representing the mapping between basic declarative and clefted versions of a sentence, will also be captured by the function.

The formal definition of the function is then as follows.

Definition 7.3.1 *Given an S-TAG G, a function $f : T_D(G) \rightarrow \mathbf{R}^n$ maps from derivation tree pairs to a vector of reals of length n, where, for some tree pair $\Upsilon \in T_D(G)$,*

$$f(\Upsilon) = \begin{cases} e_i & \textit{if there exists a pair of BQSDs starting from the roots of } \Upsilon_L \textit{ and } \Upsilon_R \\ & \quad \textit{with a paraphrase effect of type } i \\ 0 & \textit{otherwise} \end{cases}$$

where e_i is the vector of length n with zeroes in all positions except for a 1 in position i.

The paraphrase effects can be, for example, the loss of a content word (represented by a pair including a single node indicating the null operation), or the effects of a specific mapping operation (represented by a SMP pair for, say, mapping between extracted and non-extracted forms of a sentence). The vector is just a set of pigeonholes for counting the effects of paraphrases, each paraphrase effect category corresponding to a single pigeonhole.

Definition 7.3.2 *Given an S-TAG G, a function $g : T_D(G) \rightarrow \mathbf{R}^n$ maps from derivation tree pairs to a vector of reals, where, for some tree $\Upsilon \in T_D(G)$,*

$$g(\Upsilon) = \begin{cases} f(\Upsilon) & \textit{if } \Upsilon_L \in \text{leaf}(T_D(G_L)), \Upsilon_R \in \text{leaf}(T_D(G_R)) \\ f(\Upsilon) + \sum f(\Upsilon_i) & \textit{otherwise} \end{cases}$$

where Υ_i are the subtrees depending from the pair of BQSDs at the root of Υ.

This function g of Definition 7.3.2 is just a recursive accumulation of the effects determined by the function f of Definition 7.3.1; that is, it is the sum of all paraphrase effect values represented as vectors by the function f. For the quantification proposed in Section 4.3.3, we are interested in g for the case where $n = 4$; that is, there is a vector of four reals, which is a count of the number of occurrences fitting each of the four categories of effect defined in Section 4.3.3

This function g is used later, in Chapter 8, for determining coefficients for both constraint equations and the objective function of the optimisation model.

7.4 Summary

This chapter has developed generalisations of S-TAG which solve problematic aspects of the formalism brought to light by various applications: in paraphrase, the promotion of arbitrarily deeply embedded relative clauses; in machine translation, the behaviour of clitics, as an example of unbounded phenomena there; in coordination, the separation of constituency and dependency leading to 'tangled' trees. The generalisations work with the same basic principles as S-TAG: there are two grammars operating synchronously in parallel, and the mapping is restricted, in order for the weak language preservation property to hold, by isomorphisms of meta-level derivation structures. The generalisations, however, allow grammars other than standard TAGs to be synchronised, introduce the notion of a metagrammar, and move upwards the level at which synchronisation occurs.

Then, using these, the chapter gives illustrations of the representation of paraphrases from Chapter 4; and further, the semantics of Section 4.3 are represented by a function on the S-TAG pairs defined in Chapters 6 and 7, which is itself defined in Section 7.3.2.

Chapter 8 returns to modelling the Reluctant Paraphrase (RP) task, putting it in an Integer Programming framework. It uses both the S-TAG representation to specify paraphrases, and the semantic function to develop a function for minimising change to the text.

Part III

The Integer Progamming Model

Chapter 8

An Optimisation Model

In this chapter we return to the modelling of the broad Reluctant Paraphrase (RP) task: given an original text, a set of syntactic paraphrases as proposed in Chapter 4—which can be specified precisely in the S-TAG formalism as described in Chapters 5 to 7—and a set of constraints as proposed in Chapter 3, such a model needs to find a text which fits the constraints while minimising the change to the source text.

This chapter loks at a particular mathematical framework, Integer Programming (IP), and how the modelling of RP can be done in a natural way within this framework. Further, given that there is scope for a range of different IP models, the chapter explores a number of such models and makes conclusions about which is the best of those presented.

8.1 Motivation for a New Model

Existing systems in NLG that carry out paraphrasing (see Section 2.1.3 for an outline of these) do so at the level of the clause or sentence. Under RP, paraphrases also work on single sentences or on pairs of sentences. However, whereas these existing systems specify constraints at this same level, the sort of constraints that are allowable in RP include whole-of-text constraints: overall length of text, overall readability of text, and so on. Another key difference is that the constraints in existing systems generally do not conflict, whereas in RP it is frequently the case that they do: the readability score for a text can be improved by splitting sentences, requiring the addition of referring expressions and verbal auxiliaries to the text, which generally moves the text in the wrong direction with respect to the length constraint. These differences mean that qualitatively different models are necessary for describing RP: when working with sentence-level text, the methods used by existing systems are adequate, as discussed below; but when scaling up to take the text as a whole, with the consequent large increase in paraphrase combinations as potential solutions, and with conflicting constraints, navigating the search space becomes more problematic, and a new model is needed to enable searches of the space to be applied in a principled and efficient way.

Current systems use two basic approaches. Both are simple, fundamentally greedy algorithms: the first type makes changes to a text which does not violate constraints, and keeps making them until a constraint boundary is encountered, and then stops (termed here UP-TO CONSTRAINT SATISFACTION); the second type produces a text and then restructures it

if a constraint is violated in that draft (REDO-UNTIL CONSTRAINT SATISFACTION). The following two subsections describe each of these types, and explain why the search techniques employed are not satisfactory for larger scale, RP-style, models.

'Up-to' constraint satisfaction

STREAK (Robin, 1994) and YH (Gabriel, 1988) are two revision-based systems which fall into this class, although the earlier NLG systems described in Section 2.1.1 could also be included here. All of these systems take the first choice of paraphrase (or in the case of earlier systems, the first choice of realisation of knowledge base entry) that does not lead to constraint violation and apply that. In YH the first choice is a locally optimal one: Gabriel (1988) notes that the program "generates text from left-to-right, making locally good decisions at each step". It is possible in YH to evaluate individual choices for local optimality as each choice has an associated 'measure of importance'. STREAK doesn't have such a way of ranking its choices, so all candidate facts are treated equally. The one actually chosen is just the first one encountered during the search, as is its linguistic realisation (see Section 2.1.3). Once this choice is made, it is permanent, even though individual surface elements may change through later application of what are termed in STREAK 'non-monotonic paraphrases'. Making choices using this simple greedy heuristic means that better alternative texts are missed, texts which would be considered if more than just the local effect of a paraphrase were considered. For STREAK, the goal is to maximise informativeness (in terms of facts included in the text) while satisfying length and embeddedness constraints. By choosing facts and their expressions in the way it does, STREAK misses out on finding the true, global optimum level of informativeness; better solutions can be found almost by inspection. An example of a solution from STREAK is re-presented in Figure 8.1, taken from Robin (1994: 93).

At, say, Draft 3, a different choice could have been made which would have had the effect of making the final text more informative. At that point, a decision was made to add in the floating fact relating to Danny Ainge coming off the bench. If the linguistic realisation of this had been as an Adjoin-of-Coevent-Clause-to-Clause revision (described in Appendix A of Robin, 1994), an extra word would have been saved in Draft 3, as in Figure 8.2. This would have percolated through the drafts and allowed the extra fact which terminated the revision sequence.[1] This is not too problematic for STREAK: despite Robin's description of the maximisation of informativeness as a constraint, it is actually only a goal, with the adding in of facts being optional, and with no minimum limits on the number of facts added in or suchlike.

However, if the algorithm were applied to a model with conflicting *true* constraints (that is, constraints having hard minimum or maximum limits), its choice of globally non-optimal facts and realisation would be a problem. If, as well as its hard constraint of the 45 word maximum, STREAK had another true constraint specifying a minimum of six floating facts, STREAK's algorithm would have failed, while an algorithm that either planned ahead, or that looked at combinations of paraphrases in parallel, would have satisfied the constraints. Moreover, even if it were not a hard constraint, the goal of maximising informativeness

[1] The STREAK output given in Robin (1994) does not show what this extra fact and its realisation were, but does show that it added three words and no extra levels of embedding. This means that, if the alternative paraphrase were chosen at Draft 3, the total would have been 45 words and eight levels of embedding, which does not violate the constraints.

```
Draft 0:
Dallas, TX -- Charles Barkley registered 42 points Friday night
as the Phoenix Suns routed the Dallas Mavericks 123-97.

Draft 1 (lex-num = 27 depth = 7):
Dallas, TX -- Charles Barkley registered 42 points Friday night
as the Phoenix Suns handed the Dallas Mavericks their 27th
defeat in a row at home 123-97.

Draft 2 (lex-num = 29 depth = 8):
Dallas, TX -- Charles Barkley registered 42 points Friday night
as the Phoenix Suns handed the Dallas Mavericks their franchise
worst 27th defeat in a row at home 123-97.

Draft 3 (lex-num = 34 depth = 8):
Dallas, TX -- Charles Barkley registered 42 points and Danny
Ainge added 21 Friday night as the Phoenix Suns handed the
Dallas Mavericks their franchise worst 27th defeat in a row at
home 123-97.

Draft 4 (lex-num = 39 depth = 8):
Dallas, TX -- Charles Barkley registered 42 points and Danny
Ainge came off the bench to add 21 Friday night as the Phoenix
Suns handed the Dallas Mavericks their franchise worst 27th
defeat in a row at home 123-97.

Draft 5 (lex-num = 43 depth = 8):
Dallas, TX -- Charles Barkley matched his season record with 42
points and Danny Ainge came off the bench to add 21 Friday night
as the Phoenix Suns handed the Dallas Mavericks their franchise
worst 27th defeat in a row at home 123-97.

Draft 6 (lex-num = 46 depth = 8):
```

Figure 8.1: STREAK draft output

```
Draft 0:
Dallas, TX -- Charles Barkley registered 42 points Friday night
as the Phoenix Suns routed the Dallas Mavericks 123-97.
        ⋮

Draft 3 (lex-num = 34 depth = 8):
Dallas, TX -- Charles Barkley registered 42 points and Danny
Ainge added 21 Friday night as the Phoenix Suns handed the
Dallas Mavericks their franchise worst 27th defeat in a row at
home 123-97.
Draft 4 (lex-num = 38 depth = 8):
Dallas, TX -- Charles Barkley registered 42 points and Danny
Ainge came off the bench, adding 21 Friday night as the Phoenix
Suns handed the Dallas Mavericks their franchise worst 27th
defeat in a row at home 123-97.
Draft 5 (lex-num = 42 depth = 8):
Dallas, TX -- Charles Barkley matched his season record with 42
points and Danny Ainge came off the bench, adding 21 Friday
night as the Phoenix Suns handed the Dallas Mavericks their
franchise worst 27th defeat in a row at home 123-97.
Draft 6 (lex-num = 45 depth = 8):
Dallas, TX -- ??
```

Figure 8.2: Modified STREAK output

would be better satisfied by one of these algorithms.

Additionally, it is not generally the case that applying revisions in STREAK moves the text in different directions with respect to the specified constraints. So, for example, STREAK revisions will not in general reduce the embeddedness of the text (moving the text away from violating the embeddedness constraint) while increasing the length of the text (moving the text towards violating the length constraint); movement is generally in the direction of violation for one or both constraints, for all paraphrases. Simple greedy algorithms are adequate for these positively correlated constraints, even in spite of missing possible optimal solutions, as the text is always moved towards constraint violation. In cases where the starting text already violates the constraints, and the paraphrases attempt to move the text towards constraint satisfaction—as for RP—the situation would be similar.

However, under RP, where the original text violates constraints, the constraints do conflict with respect to a subset of the paraphrases, and the greedy algorithms of STREAK and so on are not appropriate. For example, all of the paraphrases in Type C (see Section 4.2) move a text in different directions with respect to length and readability constraints: when the paraphrases are used in the 'splitting' direction, the text increases in overall length (through the addition of referring expressions and auxiliary verbs) while decreasing in average sentence length (the content of a single sentence being split across two), which is a major component of most readability formulae. The simple greedy algorithm of STREAK or YH would be inappropriate here: it could conceivably work by choosing one constraint to satisfy, and then another, alternating through the set. However, it is clearly possible (and easy) to end up with a cycle of paraphrases: for example, a paraphrase could be used in one direction to improve the length, but has the effect of worsening the readability formula value; and then its inverse is applied to improve the readability, but this worsens the length. With no other changes to the text, or other method for preventing cycles, this greedy algorithm would clearly continue picking this pair of paraphrases.

'Redo-until' constraint satisfaction

In STREAK it is not possible to fall into an endless cycle of paraphrasing: each paraphrase takes place when a new, non-empty floating fact is proposed for addition to the sentence, so the paraphrases must start from a different point each time. In WEIVER, paraphrases are applied to a sentence where none of the content changes between paraphrase applications. This means that it is possible to apply a paraphrase, and, if this fails, apply its converse, and then the original paraphrase again, and so on, endlessly. Inui *et al* (1992) consequently keep a history of paraphrases used so this will not occur.

However, there is still a problem when working at a multi-sentence scale. The Inui *et al* paper does not make clear whether the set of paraphrases is closed under composition, in the sense of containing all possible composites of elementary paraphrases. For example, going from (1a) to (1b) might use the elementary paraphrase which splits prepositional phrases from head nouns; going from (1a) to (1c) might use the elementary paraphrase which coordinates sentences; and going from (1a) to (1d) could be achieved by either the application of the previous two paraphrases successively (or simultaneously) or by the application of a single paraphrase which is the composite of the two, if such exists.

(1) a. The document is kept in the library in the most inner part of the next

building on the 4th floor. The library is open from 9 to 5 and is open to the public.

b. The document is kept in the library. The library is in the most inner part of the next building on the 4th floor. It is open from 9 to 5 and is open to the public.

c. The document is kept in the library in the most inner part of the next building on the 4th floor, is open from 9 to 5 and is open to the public.

d. The document is kept in the library. The library is in the most inner part of the next building on the 4th floor, is open from 9 to 5 and is open to the public.

If these composites exist, then the search only needs to find one paraphrase: this is the first paraphrase that will turn the source text, in whatever state it begins, into one which fits the constraints (and it must exist, if the constraints are satisfiable, because of the closure under composition assumption). For a multi-sentence text, this isn't feasible: it is equivalent to enumerating every single paraphrase combination (and isomorphic to producing every possible variant of the original text), and then performing a linear search through these combinations until a satisfactory one is found. With n_i possible paraphrases per sentence, this gives the exponentially increasing $\prod_i n_i$ for the number of combinations of basic paraphrases from which the composites are derived. Similarly, if the composites do not exist, WEIVER's search algorithm traverses a tree with $\prod_i n_i$ leaves; if there is no heuristic guidance for the tree traversal or method for pruning branches, the search is likely to be very inefficient.

This need for a method to guide traversal of a search space, particularly when this search space is complicated by conflicting constraints which invalidate simple heuristics like greedy search, is, in fact, the *raison d'ètre* of the field of mathematical programming, used as the basis for the rest of the model presented in this thesis.

As can be seen from this section, the strategies used by existing systems will not scale up well when there are multiple conflicting constraints. This is especially true when these constraints span an entire text: there are many more possible combinations to be considered, and the space of potential solutions to be searched is much larger.

The next section will describe a class of models, Integer Programming (IP) models, which are applicable to problems having this structure. Integer programming models have a number of features which make them suitable for the reluctant style of paraphrase described in this thesis.

- The sort of surface constraints used in RP and described in Chapter 3 are numerical, predominantly linear, and often conflicting. IPs are designed to model these.
- There are a large number of possible interactions of paraphrases, producing a large search space. IP solution methods, described in the following section, allow a globally optimal solution to be found, but do so in an efficient way, through pruning of the search space.

- Given that the original starting text is assumed ideal—the assumption is that the author knew what he or she was doing better than a computational editing system—the aim is to minimise change to the text while still satisfying constraints. IP problems can model this through an OBJECTIVE FUNCTION which is to be either minimised or maximised; the goal is to find the solution with the optimal (minimum or maximum) value. A solution which merely satisfies constraints is not necessarily satifactory from the point of view of this fundamental assumption of RP, as it may solve the constraint problem by radically altering the text, which is undesirable; this corresponds to the notion of a damage function proposed as one of the characteristics of Section 2.3.

- Modelling RP in this way—defining variables, an objective function, and so on—also allows efficient approximation methods (such as genetic programming, simulated annealing or tabu search) to be used to find a near-optimal solution. These are not in general described in detail in this thesis, with the focus being on the modelling of the problem and finding an exact solution. However, Section 8.5.2 presents a technique which uses features of language and the concept of symmetry-breaking to reduce the size of the model and hence the size of the search space; and Section 8.5.6 discusses how more general approximation methods might be applied to the problem.

8.2 Mathematical Programming Models

The field of mathematical programming covers a number of different frameworks, including Linear Programming (LP) and Integer Programming (IP). IP is a more restricted, less general variant of LP; this section first describes the common features of both by describing LP, and then describes the extra features of IP. For an overview of the terminology and techniques, see, for example, a text such as Ignizio and Cavalier (1994).

LP models have a number of components:

- DECISION VARIABLES. These are the variables whose value is being optimised.

- CONSTRAINTS. These are inequalities whose lefthand sides are linear combinations of decision variables, and whose righthand sides are constant values.

- an OBJECTIVE FUNCTION. This is also a linear combination of decision variables; the aim is to find the minimum or maximum value of this function such that the constraints are satisfied.

This gives a model of the form

$$
\begin{array}{lrcl}
\text{minimise / maximise} & z = & \sum c_i x_i & \\
\text{subject to} & \sum a_{1i} x_i & \leq & b_1 \\
& \sum a_{2i} x_i & \leq & b_2 \\
& \vdots & & \\
& \sum a_{mi} x_i & \leq & b_m \\
& x_i & \geq & 0, \quad i = 1 \ldots n
\end{array}
$$

An optimal solution is one where all of the constraints are satisfied and, given this, that the value of the objective function z is a minimum or maximum. These constraints determine a POLYHEDRAL SET: this is a set formed by the intersection of a finite number of halfspaces, each halfspace created by a constraint dividing the total space in half. In the formulation above, there are m halfspaces defined by the constraints $\sum_j a_{ij}x_j \leq b_i$, and n halfspaces defined by the constraints $x_i \geq 0$. The polyhedral set is then the space of all possible solutions. The objective function can be thought of as a moving hyperplane, whose movement is controlled by some algorithm in such a way that it will ultimately find the optimal solution.

The most common technique for finding an optimal solution from the polyhedral set for an LP problem is the simplex algorithm. This is an algorithm which, given the polyhedral set defined by the constraints of the problem, moves from one EXTREME POINT (a point defined by the intersection of the hyperplanes that form the boundaries of the polyhedral set) to an ADJACENT EXTREME POINT (a distinct extreme point which, joined by a line segment to its neighbour, would form an edge of the polyhedral set). This continues until an optimal extreme point, one for which the objective function value cannot be improved, is found.

The simplex algorithm is not of polynomial time complexity; Klee and Minty (1972) have constructed pathological examples demonstrating that it is, in the worst case, exponential. Instead, there are polynomial algorithms which are alternatives to the simplex, such as Khachian's ellipsoid algorithm (Khachian, 1979) or Karmarkar's projective algorithm (Karmarkar, 1984). However, it is well known that, in practice, the simplex algorithm does perform well (Ignizio and Cavalier, 1994), and in many cases better than the polynomial alternative.

In general, extreme points on a polyhedral set, including the optimum, will be vectors of real numbers, the vectors consisting of the optimal solutions for the decision variables. However, many variables are naturally integer-valued; these lead to Integer Programming (IP) problems. For example, it is not possible to apply half a paraphrase to a sentence: paraphrases are only represented naturally by integer variables. If it is the case that the optimal solution produced by the simplex is a vector of non-integer values, it is not possible to merely use the simplex algorithm to find a real-valued solution to the set of constraints and then, say, to round each element of the vector to the nearest integer: this will often produce non-optimal or even infeasible vectors. A number of techniques are used to find the optimal integer value enclosed by a polyhedral set, such as cutting planes and branch-and-bound; both of these are guaranteed to find the global optimal solution. The method of cutting planes works by adding extra constraints which successively slice parts of the CONVEX HULL (the boundary of the polyhedral set) until the convex hull contains only extreme points that are integer vectors. However, the process of reducing the convex hull so that it bounds only integers is often difficult or impossible. Branch-and-bound is an intuitive approach which enumerates solutions but prunes those sections of the space of solutions which are known to be infeasible or sub-optimal; this is the method of solving IP problems used in the rest of the thesis. An explanation of the technique, with examples, follows.

Branch-and-bound involves the calculation of the solutions to the LP RELAXATION of the problem (that is, the LP problem that corresponds to the IP problem, without the additional constraints that the variables have integer values). Then, for a decision variable

x_i which has a non-integer solution α_i, two alternative constraints are added which rule out that portion of the space defined as the maximal space containing this non-integer value but no integer values. The starting IP can be viewed as forming the root node of a tree containing the problem and a solution. The two halfspaces created by ruling out a portion of the total space embodied in this node can then be considered two mutually exclusive child nodes of this parent. Both child nodes contain the IP of the parent with one extra constraint: for the left child $x_i \leq \lfloor \alpha_i \rfloor$, and for the right child $x_i \geq \lceil \alpha_i \rceil$. Solutions are calculated for these, and the process continues recursively. The terminating cases are

- when a vector containing all integers is found;
- when a node is infeasible (that is, there are no possible solutions given the constraints); or
- when the objective function value for a node is less than the current optimality candidate.

Note that it is not actually necessary to carry out the full simplex procedure at each node; it is possible to use the 'dual' of the LP at the parent node, add the extra constraint, and solve in a small number of extra steps, often one step (Ignizio and Cavalier, 1994).

A short example follows, taken from Ignizio and Cavalier (1994: 424–429). The base integer program, P_0, is given below in (8.1).

$$\begin{array}{llrcrcrcl} \text{max.} & z & = & -x_1 & + & 4x_2 & & & \\ \text{s.t.} & & & -10x_1 & + & 20x_2 & \leq & 22 & \\ & & & 5x_1 & + & 10x_2 & \leq & 49 & \\ & & & 8x_1 & - & x_2 & \leq & 36 & \\ & & & & & x_1, x_2 & \geq & 0, & x_1, x_2 \in \mathbf{N} \end{array} \tag{8.1}$$

The solution to the LP relaxation is given graphically in Figure 8.3: the shaded region is the area bounded by the polyhedral set, the dotted line represents the objective function, the asterisks are the integral points within the shaded region, and the circle is the optimal solution to the relaxation. At this circled optimal point, $x_1 = 3.8$, $x_2 = 3$, and $z = 8.2$. This integer program P_0 and its solution form the root node of the branch-and-bound search tree. To exclude the non-integer solution but no all-integer solutions, two subproblems are created, considering the regions where $x_1 \leq 3$ and $x_1 \geq 4$. The left child, for example, is then the integer program P_1, below in (8.2).

$$\begin{array}{llrcrcrcl} \text{max.} & z & = & -x_1 & + & 4x_2 & & & \\ \text{s.t.} & & & -10x_1 & + & 20x_2 & \leq & 22 & \\ & & & 5x_1 & + & 10x_2 & \leq & 49 & \\ & & & 8x_1 & - & x_2 & \leq & 36 & \\ & & & x_1 & & & \leq & 3 & \\ & & & & & x_1, x_2 & \geq & 0, & x_1, x_2 \in \mathbf{N} \end{array} \tag{8.2}$$

Given that there is a corresponding right child P_2, where $x_1 \geq 4$, this gives the search tree below.

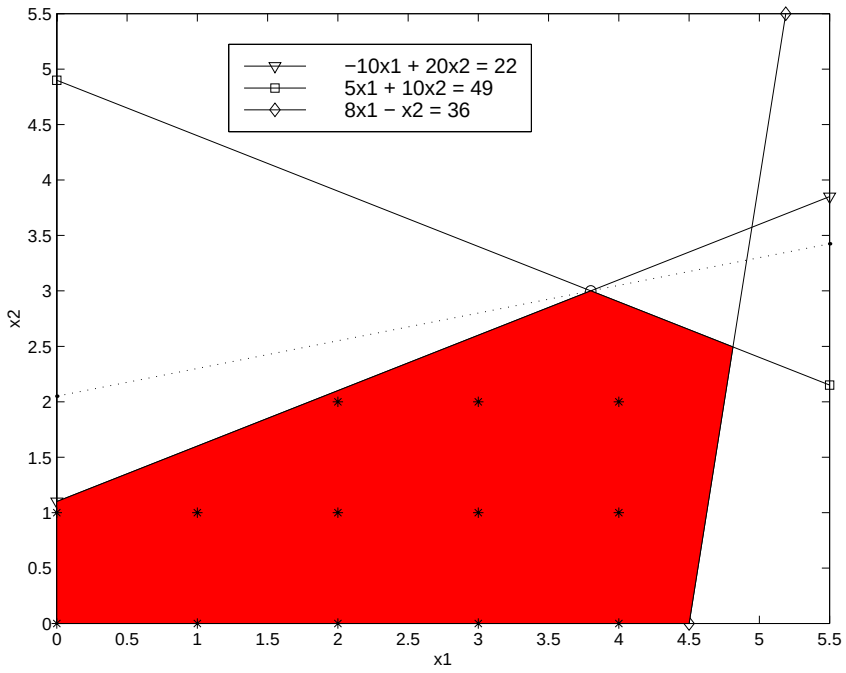
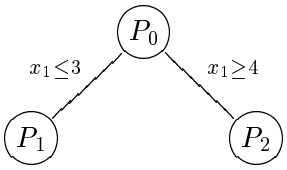


Figure 8.3: Solution to LP relaxation of P_0 in (8.1)



If subproblems are solved in a depth-first, right-to-left manner, the final search tree is as in Figure 8.4.

8.3 A Basic Model of Paraphrasing

Applying the idea of integer programming to paraphrasing, the most straightforward model is to have a decision variable for each paraphrase, representing whether or not the paraphrase is applied. In this situation, the choice, and hence each variable, is binary. That is,

p_{ij} = 0/1 variable representing the jth potential paraphrase for sentence i

In terms of the S-TAG formalism as described in Chapters 6 and 7, a p_{ij} will correspond to the specification of a paraphrase by a derivation tree pair Υ_{ij} in the S-TAG formalism, which gives the component trees and their arrangement.

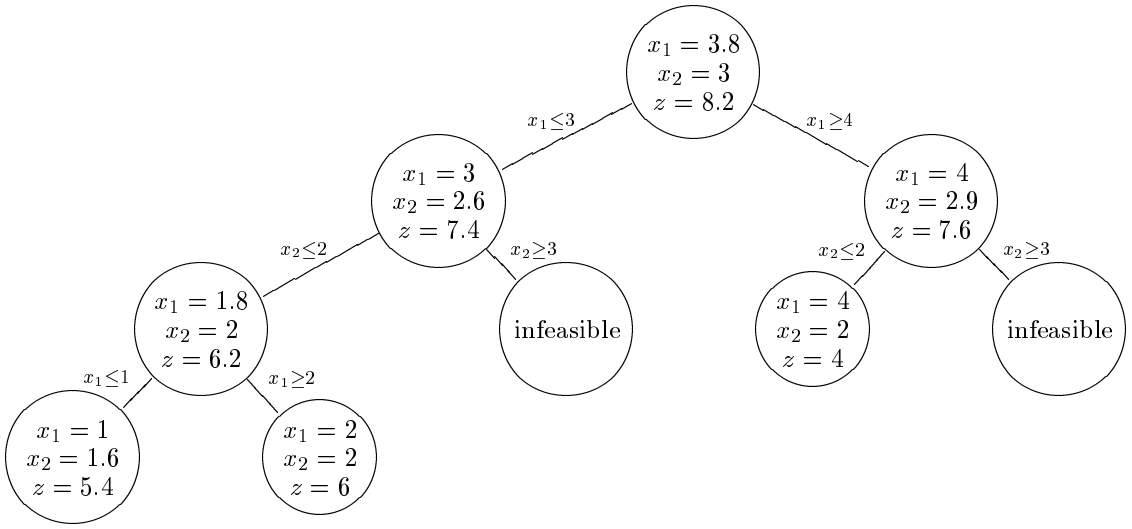


Figure 8.4: Pruned search tree

The objective function, the function to be optimised, is, for RP, a measure of the change to the text. With c_{ij} being the effect (or cost) of each paraphrase, if applied, this function has the form

$$z = \sum c_{ij}.p_{ij} \tag{8.3}$$

Relating this to the semantic function definition in Section 7.3.2, the cost c_{ij} is the sum, weighted by importance, of all effects for a particular paraphrase, with the vector representing the effects being $g(\Upsilon_{ij})$ as defined in Section 7.3.2; what this means will be explained in more detail in Section 8.4. Each element of the vector of effects will have its own weight, thus there is a vector of weights $\bar{v}$, equal in dimension to $g(\Upsilon_{ij})$. The total effect of a paraphrase is thus the product $c_{ij} = \bar{v}.g(\Upsilon_{ij})^T$ (that is, a scalar quantity derived from multiplying the vector $\bar{v}$ by the transpose of the vector resulting from the function g).

8.3.1 Constraint Modelling

To model paraphrasing with IP, the constraints need to be able to be represented as inequalities whose lefthand sides are polynomial (generally linear) functions of the decision variables. This section (see also Dras, 1998) will demonstrate how such constraint equations arise from the constraints described in Chapter 3 of this thesis.

Length Constraint

The constraints take the form "total length must be decreased by at least some constant value", or "readability formula value must be no greater than some constant value". The length constraint is simplest; expressed mathematically, it is

$$\sum w_{ij}.p_{ij} \quad \leq \quad k_1 \tag{8.4}$$

where

w_{ij} = change to length of sentence i caused by paraphrase ij
k_1 = required change to the length of the text in words

Readability constraint

For a readability constraint, the one component that almost all readability formulae have in common is average sentence length, with the aim of readability formulae being to prevent it from being too great. That is,

$$\frac{W + \sum w_{ij}.p_{ij}}{S + \sum s_{ij}.p_{ij}} \quad \leq \quad k_2$$

where

s_{ij} = change to number of sentences in the text by paraphrase ij
W = total words in original text
S = total sentences in original text
k_2 = required average sentence length; $k_2 \geq 0$

This can be expressed as a linear combination of variables as follows.

$$\sum (w_{ij} - k_2.s_{ij})p_{ij} \quad \leq \quad k_2 S - W \tag{8.5}$$

A constraint that more closely reflects specific readability formula requirements includes a second term modelling word difficulty in addition to the average sentence length component. The Lorge formula described in Section 3.2, for example, uses proportion of 'hard' words to total words as a measure of word difficulty, where 'hard' words are defined to be those not included in a specific list. This can be modelled by the constraint below.

$$a\left(\frac{W + \sum w_{ij}.p_{ij}}{S + \sum s_{ij}.p_{ij}}\right) + b\left(\frac{H + \sum h_{ij}.p_{ij}}{W + \sum w_{ij}.p_{ij}}\right) \quad \leq \quad k_2' \tag{8.6}$$

where

a = weighting for average sentence length component
b = weighting for word difficulty component
h_{ij} = change to number of hard words
H = total number of hard words in original text
k_2' = required readability score

Rearranging, this gives

$$a(W + \sum w_{ij}.p_{ij})^2 + b(H + \sum h_{ij}p_{ij})(S + \sum s_{ij}p_{ij}) \leq k_2'(S + \sum s_{ij}p_{ij})(W + \sum w_{ij}p_{ij})$$

Now,

$$\begin{aligned} LHS &= a(W^2 + 2W\sum w_{ij}p_{ij} + (\sum w_{ij}p_{ij})^2) \\ &\quad +b(HS + S\sum h_{ij}p_{ij} + H\sum s_{ij}p_{ij} + \sum s_{ij}p_{ij}\sum h_{ij}p_{ij}) \\ &= aW^2 + 2aW\sum w_{ij}p_{ij} + a\sum_{ij}\sum_{kl} w_{ij}w_{kl}p_{ij}p_{kl} \\ &\quad +bHS + bS\sum h_{ij}p_{ij} + bH\sum s_{ij}p_{ij} + b\sum_{ij}\sum_{kl} s_{ij}h_{kl}p_{ij}p_{kl} \end{aligned}$$

and

$$RHS = k_2'(SW + S\sum w_{ij}p_{ij} + W\sum s_{ij}p_{ij} + \sum s_{ij}p_{ij}\sum w_{ij}p_{ij})$$

Grouping like terms,

$$\begin{aligned} &\sum_{ij}\sum_{kl}(aw_{ij}w_{kl} + bh_{ij}s_{kl} - k_2's_{ij}w_{kl})p_{ij}p_{kl} + \\ &\sum_{ij}(2aWw_{ij} + bHs_{ij} + bSh_{ij} - k_2'Sw_{ij} - k_2'Ws_{ij})p_{ij} \leq \\ &\qquad\qquad k_2'SW - aW^2 - bHS \end{aligned} \tag{8.7}$$

Modelling the type of readability formula which uses average length of words in syllables as a measure of word difficulty is very similar. The new term describing word difficulty (that is, replacing the second term in equation (8.6)) has the same form,

$$b\left(\frac{T + \sum t_{ij}.p_{ij}}{W + \sum w_{ij}.p_{ij}}\right)$$

where

T = the total number of syllables in the text

t_{ij} = the change in number of syllables caused by paraphrase ij

The algebra to produce a polynomial inequality is identical to that leading to equation (8.8).

Note that although polynomial, this constraint is not linear in p_{ij}, the decision variables. Solving this kind of problem requires some modification to the solving of the LP relaxation. There are a number of methods for solving constraint problems that contain polynomial functions of the decision variables (see, for example, Bazaraa, Sherali and Shetty, 1993); where the degree of the polynomial is two, these are known as Quadratic Programming Problems (QPPs). One such approach is the method of complementary pivoting, which involves interchanging the non-linear constraint and the objective function (Shapiro, 1979). The objective function then becomes a constraint requiring a limit; this might be a value

chosen so that, say, there is no more than 5% difference from the score for the original text. The new non-linear objective function is then minimised or maximised. The modified problem is thus similar to a standard IP: the polyhedral set is still constructed from halfspaces defined by linear constraints, and only the moving objective function has changed shape.

Lexical density constraint

As noted in Section 3.3, there are a number of different ways of defining lexical density. Following Halliday (1985), lexical density is defined as the proportion of function (or non-content) words to total number of words; the lexical density constraint then requires the proportion of function words (taken here to be all closed class words) to total words to be greater than some constant value.[2] It has the form

$$\frac{F + \sum f_{ij}.p_{ij}}{W + \sum w_{ij}.p_{ij}} \quad \geq \quad k_3$$

where

f_{ij} = change to number of function words caused by paraphrase ij
F = total number of function words in original text
k_3 = required proportion of function words to total words; $0 \leq k_3 \leq 1$

This can be expressed as a linear combination of variables as follows.

$$\sum (f_{ij} - k_3.w_{ij})p_{ij} \quad \geq \quad k_3 W - F \tag{8.8}$$

Sentential variability constraint

The sentential variance constraint aims to make sure that there is a minimum level of variety in sentence structure (as reflected in sentence length). A variance is the expected sum of the squares of the relevant variables, normalised with respect to the mean; that is,

$$\begin{aligned} \text{variance} &= \frac{E(X - \bar{X})^2}{n-1} \\ &= \frac{E(X^2 - 2X\bar{X} + (\bar{X})^2)}{n-1} \\ &= \frac{EX^2 - \bar{X}^2}{n-1} \end{aligned}$$

where

X = variable that the variance is being calculated for, here sentence length
$\bar{X}$ = expected sentence length
n = sample size

[2] Note that, given this definition for lexical density, it is generally desirable to have a higher value—that is, a less dense text—hence the $\geq$ inequality in the constraint.

In our case, then, we will take X to be the length of sentence i, $\frac{\sum_i (W_i + \sum_j w_{ij}p_{ij})^2}{S + \sum_{ij} s_{ij}p_{ij}}$; $\bar{X}$, the average length of sentences, to be approximated by $\frac{W}{S}$; and n, the number of sentences, to be $(S + \sum_{ij} s_{ij}p_{ij})$. Thus

$$\frac{\frac{\sum_i (W_i + \sum_j w_{ij}p_{ij})^2}{S + \sum_{ij} s_{ij}p_{ij}} - \left(\frac{W}{S}\right)^2}{(S + \sum_{ij} s_{ij}p_{ij}) - 1} \quad \geq \quad k_4$$

where

W_i = length of sentence i
k_4 = minimum variance of sentence length

Multiplying both sides by $((S-1) + \sum_{ij} s_{ij}p_{ij})(S + \sum_{ij} s_{ij}p_{ij})S^2$,

$$\begin{aligned}
\text{LHS} &= S^2 \sum_i (W_i + \sum_j w_{ij}p_{ij})^2 - W^2(S + \sum_{ij} s_{ij}p_{ij}) \\
&= S^2 \sum_i W_i^2 + 2S^2 \sum_i W_i \sum_j w_{ij}p_{ij} + S^2 \sum_i (\sum_j w_{ij}p_{ij})^2 \\
&\quad -W^2 S - W^2 \sum_{ij} s_{ij}p_{ij} \\
\text{RHS} &= k_4 S^2((S-1) + \sum_{ij} s_{ij}p_{ij})(S + \sum_{ij} s_{ij}p_{ij}) \\
&= k_4 S^3(S-1) + k_4 S^2(2S-1) \sum_{ij} s_{ij}p_{ij} + k_4 S^2 (\sum_{ij} s_{ij}p_{ij})^2
\end{aligned}$$

Grouping like terms, we have

$$\begin{aligned}
&A_1 \sum_i (\sum_j w_{ij}p_{ij})^2 + A_2 (\sum_{ij} s_{ij}p_{ij})^2 \\
&\quad + B_1 \sum_{ij} W_i w_{ij}p_{ij} + B_2 \sum_{ij} s_{ij}p_{ij} \quad \geq \quad C
\end{aligned} \tag{8.9}$$

where

$$\begin{aligned}
A_1 &= S^2 \\
A_2 &= -k_4 S^2 \\
B_1 &= 2S^2 \\
B_2 &= -W^2 - k_4 S^2(2S-1) \\
C &= k_4 S^3(S-1) - S^2 \sum_i W_i^2 + SW^2
\end{aligned}$$

This is a polynomial of degree two—the terms with coefficients A_1 and A_2 are each polynomials of degree two—and thus is similar to the QPP readability constraint described earlier.

Other constraints

Given that there are n_i paraphrases for each sentence (with n_i varying for each sentence), the paraphrases will potentially conflict with each other. To simplify the application of the paraphrases, an extra constraint is added, stating that there cannot be more than one paraphrase applied per sentence:[3]

$$\sum_j p_{ij} \leq 1$$

These are a type of mutual exclusion condition (MEC), which prevent more than one paraphrase being chosen for each sentence.

To illustrate the model, an example is presented in the next section. The small size of this example does not allow a real demonstration of the usefulness of the approach, since the problem can be solved almost by inspection, but it does illustrate how the task is modelled by IP constructs. In larger problems this method of modelling allows the use of techniques such as branch-and-bound which make the solution of the problem feasible, where the solution would otherwise be impractical because of the problem's exponential complexity; such a larger problem is discussed in Section 8.4.

8.3.2 An Example

As an example, take the short text:

> The cat sat on the mat which was by the door. It ate the cream ladled out by its owner. The owner, an eminent engineer, had a convertible used in a bank robbery.

The arbitrarily chosen requirements are that there is at worst no compression of text length ($k_1 = 0$), that average sentence length is no greater than 10 ($k_2 = 10$), and that function words comprise no less than 52.5% of the text ($k_3 = 0.525$).

The values of F, W and S (from (8.4), (8.5) and (8.8)) are 17, 33 and 3 respectively.

Possible paraphrases of individual sentences, using just relative pronoun deletion, post-modifier split, and parenthetical deletion,[4] are:

(2) p_{11}. The cat sat on the mat by the door.

p_{21}. It ate the cream. It had been ladled out by its owner.

[3] Although it is possible in particular cases for paraphrases to overlap and produce satisfactory text, there is no easy way in advance to decide this; so for a model of an automated system the above constraint is necessary, at least until a much more detailed analysis of paraphrase interaction has been carried out.

[4] Here, we will only refer to paraphrases informally, rather than using the notation of the S-TAG formalism of Chapters 6 and 7. However, it should be understood that these precise definitions underlie the informal ones used here.

paraphrase ij	f_{ij}	w_{ij}	s_{ij}
11	-2	-2	0
21	+3	+3	+1
31	+3	+3	+1
32	-1	-3	0

Figure 8.5: Variable coefficients

$$
\begin{array}{rrrrrrrrrrrr}
\text{min.} & z & = & & c_{11}p_{11} & + & c_{21}p_{21} & + & c_{31}p_{31} & + & c_{32}p_{32} & \\
\text{s.t.} & & & - & 2p_{11} & + & 3p_{21} & + & 3p_{31} & - & 3p_{32} & \leq & 0 \\
& & & - & 2p_{11} & - & 7p_{21} & - & 7p_{31} & - & 3p_{32} & \leq & \text{-3} \\
& & & - & 0.95p_{11} & + & 1.425p_{21} & + & 1.425p_{31} & + & 0.575p_{32} & \geq & 0.325 \\
& & & & & & & & p_{31} & + & p_{32} & \leq & 1
\end{array}
$$

Figure 8.6: Basic IP formulation

p_{31}. The owner, an eminent engineer, had a convertible. It had been used in a bank robbery.

p_{32}. The owner had a convertible used in a bank robbery.

This gives decision variables p_{11}, p_{21}, p_{31}, and p_{32}, with associated coefficients shown in Table 8.5. The s_{ij} column in this table is the most straightforward: p_{21} and p_{31} increase the number of sentences in the text by one each, while p_{11} and p_{32} do not change it. In the w_{ij} column, p_{21} and p_{31} increase the number of words in the text by three each, while p_{11} decreases it by two, and p_{32} by three. The f_{ij} column is similar to w_{ij}, since most of the words added or deleted are function words, except in the case of p_{32}, where *eminent* and *engineer* are content words, and only *an* is a function word. The coefficients in Table 8.5 then give the basic IP formulation, as in Figure 8.6.

This IP problem has two alternatives which are feasible solutions: $p_{11} = p_{21} = p_{31} = 0, p_{32} = 1$ or $p_{31} = 0, p_{11} = p_{21} = p_{32} = 1$. There are two corresponding values for the objective function, $z = c_{32}$ and $z = c_{11} + c_{21} + c_{32}$. Since $c_{ij} > 0$ for all i, j—under the RP assumption all changes involve a positive cost—the best alternative is the first, with only the second paraphrase for the third sentence being applied. The resulting text is then as the original with *an eminent engineer* deleted:

> The cat sat on the mat which was by the door. It ate the cream ladled out by its owner. The owner had a convertible used in a bank robbery.

8.4 Application to Actual Text

8.4.1 A Larger Example

In the previous section, with the small example discussed, there were only two possible solutions; and, for those solutions, the algebraic expression of the objective function for

one was a subexpression of the algebraic expression of the objective function for the other, allowing comparison. However, this will not in general be the case, so a numeric objective function must be defined, that is, values given for c_{ij}.

Section 4.3 presented a way of roughly quantifying the effects of paraphrases on a text, which involves assigning components of the effect to categories. This was defined in terms of a function on S-TAG pairs in Section 7.3.2. Informally, to get an overall effect, the values 1 to 4 are assigned to the four categories—represented as a single vector of length four in the function definition of Section 7.3.2, $\bar{v} = (1, 2, 3, 4)$—and the total calculated for all the effects of a particular paraphrase. So, for example, take the paraphrase:

(3) a. The last pre-war census, in 1910, counted about 13.5 million foreign-born people in the American population, in contrast to about 22.5 million in 1994.

b. The last pre-war census, in 1910, counted about 13.5 million foreign-born people in the American population. **This was** in contrast to about 22.5 million in 1994.

In this paraphrase, there is one anaphoric element introduced (Category 1) and one auxiliary verb carrying tense (Category 2), represented in vector notation as $(1, 1, 0, 0)$; this gives a total cost for the paraphrase of $(1, 2, 3, 4).(1, 1, 0, 0)^T = 3$. As another example, take the paraphrase

(4) a. Historians know a great deal about those earlier immigrants—why they came, how they ended up, what their impact was on the America of their day.

b. Historians know a great deal about those earlier immigrants.

Here there have been five content (open-class) words deleted, a Category 4 effect: *came, ended, impact, America, day.* This gives a cost of $(1, 2, 3, 4).(0, 0, 0, 5)^T = 20$ for this paraphrase.

The text used in this section was taken from the *Atlantic Monthly.* This is a periodical which contains opinion pieces about topical issues; the text tends to be more complex than newspaper stories, and the articles significantly longer, but the text is not as syntactically complex as scientific writing tends to be. As such, there is a lot of scope for applying paraphrases in both directions: many paraphrases can make sentences more complex when applied in one direction, and less complex when applied in the other, so the *Atlantic Monthly* was chosen as a text which allows scope for both.

The text used in this section is the first two paragraphs of the article *Can We Still Afford to Be a Nation of Immigrants?*[5], a combined analysis of historical opinions of levels of immigration to the US and argument to continue such levels of immigration; the two paragraphs are given in Figure 8.7. The selection of paraphrases potentially applied to the text is given in Figure 8.8.

The original text has 346 words over 12 sentences. Assume a hypothetical requirement for the text that it become no longer ($k_1 = 0$), but that the average sentence length, as under a required simplified readability formula, be no more than 18 words ($k_2 = 18$). Table 8.9 then contains the relevant coefficients.

[5] *Atlantic Monthly,* November 1996.

1. The question in my title implies a premise: that historically the United States has well afforded to be a nation of immigrants—indeed, has benefited handsomely from its good fortune as an immigrant destination.
2. That proposition was once so deeply embedded in our national mythology as to be axiomatic.
3. More than a century ago, for example, in the proclamation that made Thanksgiving Day a national holiday, Abraham Lincoln gave thanks to God for having "largely augmented our free population by emancipation and by immigration".
4. Lincoln spoke these words when there were but 34 million Americans and half a continent remained to be settled.
5. Today, however, the United States is a nation of some 264 million souls on a continent developed beyond Lincoln's imagination.
6. It is also a nation experiencing immigration on a scale never before seen.
7. In the past three decades, since the passage of the Immigration and Nationality Act of 1965, the first major revision in American immigration statutes since the historic closure of immigration in the 1920s, some 20 million immigrants have entered the United States.
8. To put those numbers in perspective: prior to 1965 the period of heaviest immigration to the United States was the quarter century preceding the First World War, when some 17 million people entered the country—roughly half the total number of Europeans who migrated to the United States in the century after 1820 (along with several hundred thousand Asians).
9. The last pre-war census, in 1910, counted about 13.5 million foreign-born people in the American population, in contrast to about 22.5 million in 1994.
10. Historians know a great deal about those earlier immigrants—why they came, how they ended up, what their impact was on the America of their day.
11. Whether America's historical experience with immigration provides a useful guide to thinking about the present case is the principal question I want to address.
12. I want not only to explore the substantive issue of immigration but also to test the proposition that the discipline of history has some value as a way of knowing and thinking about the world.

Figure 8.7: Text from *Can We Still Afford to Be a Nation of Immigrants?*, divided into sentences

$p_{1,1}$ The question in my title implies a premise: that historically the United States has well afforded to be a nation of immigrants.

$p_{1,2}$ The question in my title implies a premise: that historically the United States has well afforded to be a nation of immigrants. Indeed, it has benefited handsomely from its good fortune as an immigrant destination.

$p_{3,1}$ For example, in the proclamation that made Thanksgiving Day a national holiday, Abraham Lincoln gave thanks to God for having "largely augmented our free population by emancipation and by immigration".

$p_{3,2}$ More than a century ago, for example, Abraham Lincoln gave thanks to God for having "largely augmented our free population by emancipation and by immigration". This was in the proclamation that made Thanksgiving Day a national holiday.

$p_{3,3}$ More than a century ago, for example, in the proclamation that made Thanksgiving Day a national holiday, Abraham Lincoln gave thanks to God for having "largely augmented our free population by emancipation and by immigration", speaking these words when there were but 34 million Americans and half a continent remained to be settled.

$p_{5,1}$ Today, however, the United States is a nation of some 264 million souls. They are on a continent developed beyond Lincoln's imagination.

$p_{5,2}$ Today, however, the United States is a nation of some 264 million souls on a continent developed beyond Lincoln's imagination, and is also a nation experiencing immigration on a scale never before seen.

$p_{5,3}$ Today, however, the United States is a nation of some 264 million souls on a continent developed beyond Lincoln's imagination, experiencing immigration on a scale never before seen.

$p_{7,1}$ Since the passage of the Immigration and Nationality Act of 1965, the first major revision in American immigration statutes since the historic closure of immigration in the 1920s, some 20 million immigrants have entered the United States.

$p_{7,2}$ In the past three decades, since the passage of the Immigration and Nationality Act of 1965, some 20 million immigrants have entered the United States.

$p_{8,1}$ To put those numbers in perspective: prior to 1965 the period of heaviest immigration to the United States was the quarter century preceding the First World War, when some 17 million people entered the country.

$p_{8,2}$ To put those numbers in perspective: prior to 1965 the period of heaviest immigration to the United States was the quarter century preceding the First World War, when some 17 million people entered the country. This was roughly half the total number of Europeans who migrated to the United States in the century after 1820 (along with several hundred thousand Asians).

$p_{8,3}$ To put those numbers in perspective: prior to 1965 the period of heaviest immigration to the United States was the quarter century preceding the First World War, when some 17 million people entered the country—roughly half the total number of Europeans who migrated to the United States in the century after 1820.

$p_{9,1}$ The last pre-war census counted about 13.5 million foreign-born people in the American population, in contrast to about 22.5 million in 1994.

$p_{9,2}$ The last pre-war census, in 1910, counted about 13.5 million foreign-born people in the American population.

$p_{9,3}$ The last pre-war census, in 1910, counted about 13.5 million foreign-born people in the American population. This was in contrast to about 22.5 million in 1994.

$p_{10,1}$ Historians know a great deal about those earlier immigrants.

$p_{11,1}$ There is a principal question I want to address. That is whether America's historical experience with immigration provides a useful guide to thinking about the present case.

$p_{12,1}$ I want not only to explore the substantive issue of immigration. Rather, I want also to test the proposition that the discipline of history has some value as a way of knowing and thinking about the world.

Figure 8.8: Possible paraphrases for the text of Figure 8.7

paraphrase	cost to text	c_{ij}	w_{ij}	s_{ij}	optimal value
$p_{1,1}$	-6 content words	24	-12	0	0
$p_{1,2}$	+ new anaphor	1	1	1	1
$p_{3,1}$	-1 content word	4	-5	0	0
$p_{3,2}$	+ new anaphor, + new tense	3	2	1	1
$p_{3,3}$	- tense	2	-1	-1	0
$p_{5,1}$	+ new anaphor, + new tense	3	2	1	1
$p_{5,2}$	- anaphor	1	0	-1	0
$p_{5,3}$	- anaphor, - tense	3	-5	-1	0
$p_{7,1}$	-3 content words	12	-5	0	1
$p_{7,2}$	-10 content words	40	-17	0	0
$p_{8,1}$	-14 content words	56	-24	0	0
$p_{8,2}$	+ anaphor, + tense	3	2	1	1
$p_{8,3}$	-4 content words	16	-6	0	0
$p_{9,1}$	-1 content word	4	-2	0	0
$p_{9,2}$	-4 content words	16	-8	0	0
$p_{9,3}$	+ anaphor, + tense	3	2	1	1
$p_{10,1}$	-5 content words	20	-17	0	1
$p_{11,1}$	+ anaphor, + tense, + there, Δ article	9	4	1	1
$p_{12,1}$	+1 content word	4	2	1	1

Figure 8.9: Coefficients for text of Figure 8.7

The optimal values for the variables, after solving the IP using branch-and-bound, are also given in Table 8.9; the optimal solution vector means that paraphrases $p_{1,2}$, $p_{3,2}$, $p_{5,1}$, $p_{7,1}$, $p_{8,2}$, $p_{9,3}$, $p_{10,1}$, $p_{11,1}$ and $p_{12,1}$ are chosen to be applied. The corresponding text given the application of these paraphrases is in Figure 8.10, with changes in bold. As a comparison of search space sizes, modelling the problem as an IP and solving using branch-and-bound required a search space of 193 nodes, while explicitly enumerating every solution to find the best one would require 2^{19} nodes.

8.4.2 Sensitivity Analysis

The only requirement of an assignment of values to categories, for the purpose of obtaining a numeric objective function, is to ensure that the relative rankings of categories is preserved; beyond that, the assignment of the vector of values 1 to 4 used in the previous section is arbitrary. Choosing different cost weightings has the potential to produce different optimal solutions. In mathematical programming, there are techniques of sensitivity analysis for evaluating the stability of a model (that is, the extent to which the optimal solution changes given a change in a model coefficient); however, these techniques can only be meaningfully applied to LPs, where a change in a coefficient will produce a continuous change in the objective function. IPs cannot use such analytical techniques that are independent of the specific mathematical programming problem to predict the exact behaviour of the IP when a coefficient is changed (Williams, 1995), but it is still possible to evaluate the stability of the model numerically, by comparing the optimal solutions

The question in my title implies a premise: that historically the United States has well afforded to be a nation of immigrants. **Indeed, it** has benefited handsomely from its good fortune as an immigrant destination. That proposition was once so deeply embedded in our national mythology as to be axiomatic. More than a century ago, for example, Abraham Lincoln gave thanks to God for having "largely augmented our free population by emancipation and immigration". **This was in the proclamation that made Thanksgiving Day a national holiday**. Lincoln spoke these words when there were but 34 million Americans and half a continent remained to be settled. Today, however, the United States is a nation of some 264 million souls. **They are** on a continent developed beyond Lincoln's imagination. It is also a nation experiencing immigration on a scale never before seen. In the first three decades, since the passage of the Immigration and Nationality Act of 1965, [...] some 20 million immigrants have entered the United States. To put these numbers in perspective: prior to 1965 the period of heaviest immigration to the United States was the quarter century preceding the First World War, when some 17 million people entered the country [...]. The last pre-war census, in 1910, counted about 13.5 million foreign-born people in the American population. **This was** in contrast to about 22.5 million in 1994. Historians know a great deal about those earlier immigrants [...]. **There is a principal question I was to address. That is whether America's historical experience with immigration provides a useful guide to thinking about the present case**. I want not only to explore the substantive issue of immigration. **Rather, I want** also to test the proposition that the discipline of history has some value as a way of knowing and thinking about the world.

Figure 8.10: Optimally paraphrased text

obtained for different coefficients.

For the type of coefficient change here—investigating the effect of different category–value assignments—it is possible to make some general statements about the effects. Assume we start with original costs c_{ij}, new costs by c'_{ij}, and the objective function

$$\text{minimise} \quad z = c_1x_1 + c_2x_2 + \ldots + c_nx_n$$

The new objective function is then

$$\text{minimise} \quad z' = c'_1x_1 + c'_2x_2 + \ldots + c'_nx_n$$

Now, let $c'_i = \kappa c_i + \delta_i$. Then the objective function becomes

$$\text{minimise} \quad z' = (\kappa c_1 + \delta_1)x_1 + (\kappa c_2 + \delta_2)x_2 + \ldots + (\kappa c_n + \delta_n)x_n$$

In the case where $(\forall i)\delta_i = 0$,

$$\text{minimise} \quad z = \kappa c_1x_1 + \kappa c_2x_2 + \ldots + \kappa c_nx_n$$

If $z'' = \kappa z'$, the IP becomes

$$\text{minimise} \quad z'' = c_1x_1 + c_2x_2 + \ldots + c_nx_n$$

That is, the objective function can be viewed as a moving hyperplane: to find the optimal point of the IP, this hyperplane is moved outwards towards the convex hull. Thus a constant factor κ applied to all costs has no effect on the optimal solution, as it is the same hyperplane with a different rate of movement; the only difference is the actual value of the objective function.

In cases where each δ_i term is small relative to κc_i, this is approximately the case also. That is, optimal solutions will be the same in almost all cases where two IPs have the same constraints but one has the original costs and the other has the modified costs. As δ_i increases relative to κc_i, this becomes less the case.

As an experiment, various different values were assigned to the four categories. These were:

weight vector $\bar{v}$	category 1	category 2	category 3	category 4
$\bar{A}$	1	2	3	4
$\bar{B}$	11	12	13	14
$\bar{C}$	1	11	21	31
$\bar{D}$	1	4	9	16

(k_1,k_2)	weight vector $\bar{v}$	paraphrases applied
(0,18)	$\bar{A}$	$p_{1,2}, p_{3,2}, p_{5,1}, p_{7,1}, p_{8,2}, p_{9,3}, p_{10,1}, p_{11,1}, p_{12,1}$
	B	$p_{1,2}, p_{3,2}, p_{5,1}, p_{7,1}, p_{8,2}, p_{9,3}, p_{10,1}, p_{11,1}, p_{12,1}$
	C	$p_{1,2}, p_{3,2}, p_{5,1}, p_{7,1}, p_{8,2}, p_{9,3}, p_{10,1}, p_{11,1}, p_{12,1}$
	D	$p_{1,2}, p_{3,2}, p_{5,1}, p_{7,1}, p_{8,2}, p_{9,3}, p_{10,1}, p_{11,1}, p_{12,1}$
(0,16)	$\bar{A}$	infeasible
	$\bar{B}$	infeasible
	$\bar{C}$	infeasible
	D	infeasible
(0,20)	$\bar{A}$	$p_{1,2}, p_{3,2}, p_{5,1}, p_{8,2}, p_{9,3}, p_{10,1}, p_{12,1}$
	B	$p_{1,2}, p_{3,2}, p_{8,2}, p_{9,3}, p_{10,1}, p_{11,1}, p_{12,1}$
	C	$p_{1,2}, p_{3,2}, p_{5,1}, p_{8,2}, p_{9,3}, p_{10,1}, p_{12,1}$
	D	$p_{1,2}, p_{3,2}, p_{5,1}, p_{8,2}, p_{9,3}, p_{10,1}, p_{12,1}$
(-20,18)	$\bar{A}$	$p_{1,2}, p_{3,2}, p_{5,1}, p_{7,2}, p_{8,2}, p_{9,3}, p_{10,1}, p_{12,1}$
	B	$p_{1,2}, p_{3,2}, p_{5,1}, p_{7,2}, p_{8,2}, p_{9,3}, p_{10,1}, p_{12,1}$
	C	$p_{1,2}, p_{3,2}, p_{5,1}, p_{7,2}, p_{8,2}, p_{9,3}, p_{10,1}, p_{12,1}$
	D	$p_{1,2}, p_{3,2}, p_{5,1}, p_{7,2}, p_{8,2}, p_{9,3}, p_{10,1}, p_{12,1}$
(20,18)	$\bar{A}$	$p_{1,2}, p_{3,2}, p_{5,1}, p_{7,1}, p_{8,2}, p_{9,3}, p_{10,1}, p_{11,1}, p_{12,1}$
	$\bar{B}$	$p_{1,2}, p_{3,2}, p_{5,1}, p_{7,1}, p_{8,2}, p_{9,3}, p_{10,1}, p_{11,1}, p_{12,1}$
	$\bar{C}$	$p_{1,2}, p_{3,2}, p_{5,1}, p_{7,1}, p_{8,2}, p_{9,3}, p_{10,1}, p_{11,1}, p_{12,1}$
	$\bar{D}$	$p_{1,2}, p_{3,2}, p_{5,1}, p_{7,1}, p_{8,2}, p_{9,3}, p_{10,1}, p_{11,1}, p_{12,1}$

Figure 8.11: Sensitivity of changes

$\bar{A}$ is the set of default values used in the previous section; $\bar{B}$ is a set of values with large δ_i relative to κc_i ($\kappa = 1, \delta_i = 10$); $\bar{C}$ is a set of values with small δ_i relative to κc_i ($\kappa = 10, \delta_i = -9$); and $\bar{D}$ is a set of values with quadratic δ_is. The results of the objective function tested at various points around the solution space are as in Table 8.11. The five points given in Table 8.11 have as their centre the choice of $k_1 = 0$ (change to length) and $k_2 = 18$ (readability) used in the previous section, with the other four points symmetrically around it. These four points are some distance away: the k_2 value is varied by approximately 14%, and k_1 value varied by around 12%. The solution vector for each of these alternative points represents the paraphrases that are chosen.

As can be seen from Table 8.11, in all cases but one the solution vectors are the same no matter which category-value assignment system is used. As expected, it is $\bar{B}$ which has a different solution vector, as this was the category-value set with the high $\frac{\delta_i}{\kappa c_i}$ ratio. Even this different solution is fairly close to the others, however, with only one paraphrase being swapped in to the solution. To check whether different category-value assignments would have different results on other texts, it would be necessary to map out the solution space of those texts; however, this is not an unrepresentative text in any particular way, so it would be expected that for other texts similar results should hold.

8.5 A General IP Model

This section discusses the idea of symmetry in IP models, a feature which leads to unnecessarily large search spaces. It shows how the basic model presented in Section 8.3 can be reformulated so that the search space is reduced, and presents some experimental work to show that this is actually the case.

8.5.1 The Problem of Symmetry

An undesirable feature of the basic model of Section 8.3 is potential SYMMETRY of solutions, which occurs when two solutions are interchangeable. For example, in the IP of Figure 8.6, $p_{2,1}$ and $p_{3,1}$ are, ignoring the mutual exclusion constraint on $p_{3,1}$, equivalent. That is, at the level of abstraction used for the IP modelling of the paraphrases, they will have the same effects on the text; so if there is an optimal solution where only one is required—say, solution vector $\bar{p} = (0, 1, 0, 0)$, indicating only $p_{2,1}$ is chosen—this will be equivalent to using the other paraphrase instead—$\bar{p} = (0, 0, 1, 0)$, that is, only $p_{3,1}$ chosen. Having this redundant symmetry increases the size of the search space; much of the search is effectively devoted to finding the ordering of these equivalent solutions.

However, this concept of symmetry, and ways to reduce it, has been explored in the artificial intelligence subfield of solving constraint satisfaction problems (CSPs), an area closely related to mathematical programming. Work has been done recently combining techniques of constraint satisfaction with branch-and-bound, and applied to the area of natural language processing, specifically finding semantic representations of text (Beale, 1997); the aim is to achieve a smaller search space than constraint satisfaction techniques or branch-and-bound would alone. Another useful idea from the field of constraint satisfaction programming is the idea of SYMMETRY-BREAKING, removing symmetry from problems to reduce the size of the search space. Other work in this area has been in finding and exploiting symmetry in propositional encodings of CSPs; for example, in scheduling problems representing the delivery of packages between cities via aeroplanes, where the deliveries are represented propositionally, there is no point in a search which differentiates between identical aeroplanes encoded in different propositions. Initial work looked at symmetries in the propositional encodings themselves (Benhamou and Sais, 1992; Benhamou, 1994; Crawford *et al*, 1996). Later work (Joslin and Roy, 1997) looks at 'lifted' descriptions of CSPs, minimal descriptions of the domain: for example, for the map colouring problem, these are the identities of the countries, the available colours, the adjacencies, and the axioms specifying the problem (that is, that adjacencies are symmetric, and that adjacent countries cannot share colours). These lifted descriptions can be expanded out to full propositional encodings, but Joslin and Roy demonstrate that it is better to carry out symmetry-breaking at this lifted level, in terms of the times taken to show whether a problem is unsatisfiable.

The idea of symmetry in IPs is noted in Williams (1995), in particular the fact that symmetry leads to pointlessly larger search spaces. In order to see this, a comparison can be carried out between two versions of a problems, one with symmetry, and one without. In IP, this can be done by defining a GENERAL IP (that is, one with variables not restricted to binary values), and reformulating it as a 0–1 IP, which will have redundant symmetry. As a concrete example, take the general IP in Figure 8.12. This general IP has the solution has the solution $y = 1$; using the branch-and-bound technique for solving IP problems, this

$$\begin{array}{rrcll} \max. & z & = & 5y & \\ \text{s.t.} & & & 4y & \leq \quad 6 \\ & & & 6y & \leq \quad 12 \end{array}$$

$$0 \leq y \leq n, \quad y \in \mathbf{N}$$

Figure 8.12: Sample one-variable general IP

$$\begin{array}{rrcll} \max. & z' & = & 5x_1 + 5x_2 + \ldots + 5x_n & \\ \text{s.t.} & & & 4x_1 + 4x_2 + \ldots + 4x_n & \leq \quad 6 \\ & & & 6x_1 + 6x_2 + \ldots + 6x_n & \leq \quad 12 \end{array}$$

$$0 \leq x_i \leq 1, \quad x_i \in \mathbf{N}$$

Figure 8.13: Equivalent binary reformulation of IP in Figure 8.12

involves calculating one linear programming (LP) solution, and one subsequent branching and visiting of two nodes.

In transforming this general IP to a 0–1 IP problem, y can be rewritten as the sum of n binary variables, $x_1 \ldots x_n$; define this as the LINEAR REFORMULATION of the general IP. The binary IP problem is then as in Figure 8.13.

Explicitly enumerating the combinations of n variables gives a search tree of $2^{n+1} - 1$ nodes, where the nodes represent the successive fixing of values for the binary variables. So, for three variables, the tree looks like Figure 8.14; and node 6, for example, represents the point in the traversal of the tree where the values of variables x_1 and x_2 are fixed to the values 1 and 0 respectively. The size of the search tree can be cut down using IMPLICIT ENUMERATION, a version of branch-and-bound tailored to binary variable problems (see, for example, Winston, 1991).

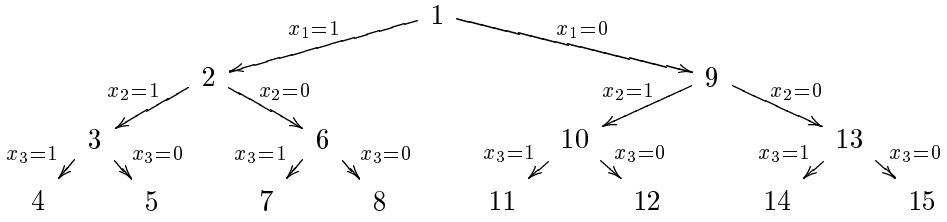


Figure 8.14: Search tree for explicit enumeration over 3 variables

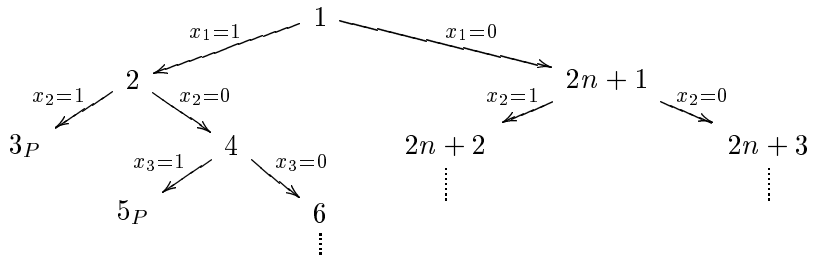


Figure 8.15: Search tree for implicit enumeration (linear reformulation)

For the original problem given above, the solution of $y = 1$ means that only one binary variable in the reformulation has a value of one; the others are zero. With n 0-1 variables, there are n equivalent optimal solutions that are equivalent to $y = 1$—$(1, 0, 0, \ldots, 0)$, $(0, 1, 0, \ldots, 0)$, and so on—and finding these gives the search tree in Figure 8.15.[6]

In this search tree, to find the ith solution, for an n variable problem with one variable equal to one, requires searching an additional $2(n - i + 1)$ nodes. There is also the single, non-optimal, all-zero solution at the last node searched.[7]

The total number of nodes searched, for an n variable problem of this form, is then

$$\begin{aligned} T &= 1 + \sum_{i=1}^{n} 2(n - i + 1) \\ &= n^2 + n + 1 \end{aligned}$$

This is quadratic in n (the number of variables), obviously far larger than the general IP search tree (three nodes) in terms of number of nodes searched. Part of this increase in search comes from the undesirable symmetry of the binary variables, with the search space covering the n interchangeable solutions, each with one $x_i = 1$ and all other $x_i = 0$. Particularly in this case, when the binary variables are the result of an artificial division of the original IP, it is unimportant to know the ordering of these n binary variables (that is, which particular variables are equal to one and which to zero). Instead, it is only necessary to know that one $x_i = 1$; but much of the search is effectively devoted to resolving possible orderings.

Consequently, if it were possible to move in the other direction, from 0-1 IPs to general IPs, a kind of symmetry-breaking could be achieved, and the search space of the problem reduced.

Note that another common, and more sophisticated, reformulation involves rewriting an IP variable whose values range from $0 \ldots n$ as a sum of $\lfloor \log_2 n + 1 \rfloor$ binary variables, where

[6] The subscript P on nodes 3 and 5 indicates that the tree has been pruned at this point by the process of implicit enumeration; nodes 6, $2n + 2$ and $2n + 3$ all have subtrees which are not fully drawn in this diagram.

[7] This is because of the arbitrary choice of left-branching on $x_i = 1$; it would be the first solution found when left-branching on $x_i = 0$, but otherwise everything is the same.

the ith binary variable has coefficient 2^i; define this as the EXPONENTIAL REFORMULATION of the general IP. That is,

$$y = 2^m x_1 + 2^{m-1} x_2 + \ldots + 2^2 x_{m-1} + 2x_m + x_{m+1}$$

where

$$0 \leq y \leq n, \quad y \in \mathbf{N}$$
$$m = \lfloor \log_2 n + 1 \rfloor$$
$$0 \leq x_i \leq 1, \quad x_i \in \mathbf{N}$$

Under this reformulation, the variable y given above can be re-expressed as:

$$y = 4x_1 + 2x_2 + x_3$$

This gives the following IP:

$$\begin{array}{lrcrcrcrl} \text{max.} & z'' & = & 20x_1 & + & 10x_2 & + & 5x_3 & \\ \text{s.t.} & & & 16x_1 & + & 8x_2 & + & 4x_3 & \leq \quad 6 \\ & & & 24x_1 & + & 12x_2 & + & 6x_3 & \leq \quad 12 \end{array}$$

$$0 \leq x_i \leq 1, x_i \in \mathbf{N}$$

In general, for a solution where the original variable y has an optimal solution of 1, and y has the range $0 \ldots n$, then the number of nodes searched is $2m+1$, where $m = \lfloor \log_2 n \rfloor$. It should be noted that, as it stands, the reformulation allows values for the original variable that are invalid, in the range $[n+1 \ldots 2^k - 1]$, so it is necessary to add to the IP an extra constraint of the form

$$2^m x_1 + 2^{m-1} x_2 + \ldots + 2^2 x_{m-1} + 2x_m + x_{m+1} \leq n$$

However, this exponential reformulation is of less interest here than the linear one, for reasons given in the next section.

8.5.2 Variable Aggregation

From the 0-1 IPs of the last section, it is apparent that all of the binary variables corresponding to a single general IP variable have a particular pattern across all the constraints and the objective function.

Definition 8.5.1 *Let a variable's* C-SET *be defined as the set of coefficients of a variable across a subset of the constraints and the objective function. A* TOTAL C-SET *is the c-set across all constraints and the objective function. A* PARTIAL C-SET *is the c-set across the objective function and all constraints except those that represent mutual exclusion conditions (MECs). A* CONSTRAINT C-SET *is the c-set across all constraints excluding those representing MECs.*

Then, for the first type of reformulation described, the linear reformulation, all the binary variables that can be aggregated into a single integer variable will have the same total c-sets (which is the same as the total c-set of the original general IP variable); for the second, exponential reformulation, the elements of each binary total c-set are related by factors of 2 to the elements of each other binary total c-set of variables that can be aggregated into a single integer variable.

Reformulating 0-1 IPs as general IPs thus requires forming equivalence classes of binary variables whose total c-sets share in the appropriate relation, and forming a single integer variable from those that do. In general, it is not likely to be the case that variables will have coefficients that fit this pattern across all constraints; IPs modelling physical phenomena can have large and widely varying coefficients (a survey of models of problems in areas such as factory planning, refinery optimisation, agricultural pricing, and so on can be found in Williams, 1995). This is especially the case for fitting the exponential pattern: while coefficients in actual problems are widely varying, within a given constraint they are generally of the same order of magnitude; but the exponential reformulation, after the first few variables, involves coefficients of very disparate magnitudes; investigating linear reformulations thus offer more opportunity for aggregating variables.

The sort of language problem treated in this thesis is amenable to a reformulation from 0-1 to general IP, reducing symmetry and decreasing the search space: the binary variables generally have small, and quite similar, coefficients, nothwithstanding the fact that they represent different phenomena. This is because they represent quantities like change in number of words caused by a particular paraphrase, and these paraphrases only have a limited scope for changing the text, restricted as they are by syntactic and other considerations.

Moreover, it is possible to rephrase the problem by 'tweaking' the coefficients, with the aim being to achieve variables with identical c-sets. In applying paraphrases, there is some flexibility regarding the resulting text, and consequently regarding the coefficients representing the effects of the paraphrase. Most notably, there is flexibility in choosing how an entity is referred to, as in, for example, when one sentence is split into two. For example, take the following paraphrase mapping:

(5) a. He desperately wanted the black leather jacket worn by the model.

b. He desperately wanted the black leather jacket. X was worn by the model.

In this paraphrase (5), for example, there are a number of alternatives for X: a pronoun (*it*), or some other more specific referent (*the jacket*, *the black jacket*, *the leather jacket*, *the black leather jacket*)[8]. This leads to, in this case, four variants of the post-modifying paraphrase, all with fairly similar properties in terms of constraint equations and objective function. This can be viewed as a single variable with coefficients that can also be varied to a limited extent.

Definition 8.5.2 *Let a variable's* k-FLEX *be defined as a set whose elements are the integers corresponding to the amount of variation to the text caused by the paraphrase variants, with respect to constraint* k.

[8]There are, of course, many other possible referring expressions if lexical changes are allowed—for example, *the piece of clothing*—but in this thesis, only syntactic paraphrases are considered.

The constraint c-set for the paraphrase of (5), using the coefficients of p_{ij} from equations (8.4), (8.5) and (8.8) of Section 8.3, is then $\{2+Y, 2+Y-k_2, 1-(2+Y)k_3\}$, with k-flex Y taking the values $\{0,1,2,3\}$ for each k; that is, we have four sets of coefficients (the constraint c-set) corresponding to the four values of the variant referring expressions. From the discussion in the previous section, it would seem desirable to use the variants which allow aggregation of variables, that is, to choose the values of the c-set variables so that the c-sets of different decision variables match; this should produce a smaller total search space. If the general IP has m variables with ranges $n_1, n_2, \ldots, n_m$, and there is a linear 0-1 IP reformulation, then the total number of solutions for the general IP is $\prod_i n_i$, while the total number of solutions for the binary IP is $2^{\sum_i n_i}$: for the IP of Figure 8.6, there are 12 possible solutions for the general IP as against 16 for the binary formulation. However, even though it is clear that the total space of solutions will be smaller when the aggregable variant is chosen, it may not be the case that the space is smaller when search pruning techniques, such as branch-and-bound, are used. For the single integer variable case, as in Figure 8.12, it is possible to demonstrate analytically that the actual search space with branch-and-bound is smaller. However, it cannot in general be shown analytically that the actual search space with branch-and-bound is smaller where the number of variables is greater than one: if it were possible, for example, to state at which node the ith solution would be found, and hence calculate the size of the actual search space, the enumeration-based methods for IP problem solving would be unnecessary.

Therefore, the following sections investigate the size of the search space of general IP versus binary IP problems where a pruning technique such as branch-and-bound is used. The two questions to be the focus are:

- For a multiple variable case, is it in fact better to have the general IP formulation? On average, how many nodes are searched for each variant formulation?
- If it is the case that the general IP formulation is better, what should be the mapping between c-sets of binary variables and integer variables? Given that there is a range within which the binary variable c-sets can move, it is possible (and in fact likely) that an individual binary variable could match with more than one other. If so, should it aggregate with the largest set of binary variables with common c-sets? Or should the sets of variables to be aggregated together be balanced as far as possible?

8.5.3 Comparing Models

The aim of the experiment in this section is to test whether there is a significant difference in the size of the search space between a formulation where there is no aggregation of variables, and two alternative formulations where paraphrase variants are used such that aggregation of variables can be carried out.

The two algorithms for aggregating variables are POLE and BALANCE, in Figures 8.16 and 8.17. The algorithms vary the coefficients where possible; choose a paraphrase variant whose c-set, for a particular referring expression, matches the c-set of at least one fixed-coefficient 'target' variable; and aggregate those two variables together. The difference between the algorithms is that, when faced with a choice between target variables, POLE chooses the potential aggregate variable with the largest range (that is, the one that has had the largest number of variables aggregated into it) while BALANCE chooses the one with

```
PROCEDURE POLE_ALGORITHM
   foreach flexible coefficient
     generate list of potential target variables
     aggregate to most often chosen variable
     mark coefficient as inflexible
   extract subproblem
   return subproblem
```

Figure 8.16: POLE aggregation algorithm

```
PROCEDURE BALANCE_ALGORITHM
   foreach flexible coefficient
     generate list of potential target variables
     aggregate to least often chosen variable
     mark coefficient as inflexible
   extract subproblem
   return subproblem
```

Figure 8.17: BALANCE aggregation algorithm

	binary IP	general IP
$A = 0, B = 0$	19	19
$A = 3, B = 2$	19	7
$A = 3, B = 1$	19	9
$A = 2, B = 1$	19	7

Figure 8.18: Search spaces for IP with varying coefficients

the smallest range. Thus BALANCE produces a set of aggregated variables with smaller ranges than does POLE.

For example, take the following IP with varying coefficients.

$$\begin{array}{rrcl} \text{max.} & z & = & 7x_1 + (4+A)x_2 + (5+B)x_3 + 6x_4 \\ \text{s.t.} & & & 10x_1 + (4+2A)x_2 + (5+2B)x_3 + 8x_4 \leq 13 \end{array}$$

$$0 \leq x_i \leq 1, \quad x_i \in \mathbf{N}$$

Let A and B be the 1-flexes (that is, the variant values for constraint 1) for x_2 and x_3 respectively; and assume A and B can take integer values in the range [0, 3]. It is then possible to pick values of A and B so that the coefficients of x_2 and x_3 match those of x_1 and x_4 in both the objective function and the constraint, the fixed coefficient variables. If A and B do take these values, binary variables can be merged to form a new variable whose range is equal to the number of variables used in its construction. Table 8.18 shows the results, for various values of A and B, of comparing the binary IP that comes from no aggregation of variables, against aggregating the variables wherever possible.

The case where $A = 0$ and $B = 0$, in row 1 of the table, has no possibility of aggregation, so the general IP model is the same as the binary one. In the cases in rows 2 and 4 of the table, one variable becomes the POLE (x_1 and x_4 respectively), and the other binary variables are aggregated to that one to form a new aggregate integer variable with range $[0, 3]$. In the case in row 3 of the table, the two variables with varying coefficients are spread evenly over the other two, as under the BALANCE algorithm.

In the example above, POLE produces a better search space than BALANCE, and both are better than the default of not varying coefficients. To test on a large set of data the null hypothesis that all three methods produce the same size search spaces, binary variable IP problems were randomly generated in the following manner. An initial problem was generated with an objective function and one constraint, of size 100 variables. Each variable had coefficients randomly generated from a uniform distribution with the range $[-10, 10]$; the constraint had equal chance of being $\leq$ or $\geq$, and the constraint constant was generated from a uniform distribution with lower limit zero and upper limit equal to the sum of the variable coefficients. Each coefficient also had associated with it a flexibility factor representing the extent to which it could vary; this maximum flexibility was generated from a uniform distribution with the range $[0, 3]$. This was then repeated with 2 and 3 constraints, starting with initial number of variables 400 and 2000 respectively. The variables were aggregated together using POLE and BALANCE; any problems generated where the two algorithms gave the same aggregations were discarded for the purposes of

differentiating between the two.

In order to construct problems that could be solved within a reasonable time, subproblems were extracted from these randomly generated problems: only those variables which took part in the aggregation, under either algorithm, were included in the subproblem; singleton variables, those not involved in the aggregation, were left out. These subproblems then had an average of just over 21 variables.

The subproblems, once calculated, were then formulated in the three different ways described above—the default no-aggregation, aggregation using the POLE algorithm, and aggregation using the BALANCE algorithm—and solved using a branch-and-bound technique.

Results

The mean search space sizes (in number of nodes) and their standard deviations are shown in Table 8.22. Sample sizes are 1493, 421 and 44.[9] It is clear from the table that hypothesis tests assuming the normal distribution are not likely to be appropriate for testing difference between means: the distributions are very skewed, with standard deviations much larger than the mean, and all values positive. This can be seen from Figures 8.19 to 8.21, which, for one constraint, plot histograms of the search space sizes; the distributions are clearly not Gaussian. Consequently, the non-parametric Wilcoxon paired sign-rank test was used for comparison. For one constraint, the algorithms were pairwise compared: taking as the null hypothesis that the means were equal, the probabilities that this was the case were 1.18×10^{-5} (default vs. POLE), 1.07×10^{-4} (default vs. BALANCE), and 1.26×10^{-4} (POLE vs. BALANCE). For two and three constraints, only POLE (the better of the two algorithms) and the default were compared, giving probabilities of 1.74×10^{-21} and 7.63×10^{-5} respectively. Hence the aggregation algorithms are clearly better than the no aggregation alternative for one constraint; and, choosing POLE as the representative aggregation algorithm for two and three constraints, this is significantly better than the no aggregation alternative here also.

8.5.4 Applying Aggregation to Actual Text

To apply this aggregation technique to actual text, the first step is to calculate the flexes of the paraphrases. The only paraphrases with variant referring expressions are $p_{1,2}$, $p_{5,1}$, $p_{8,2}$ and $p_{9,3}$. These variants are as in Figure 8.23.

This gives the values in Table 8.24. The values a_{hij} are the coefficients for constraint h, equal to w_{ij} and $(w_{ij} - k_2 s_{ij})$ for constraints 1 and 2 respectively. The flex values come from the paraphrase variants given in Figure 8.23 (with variant referring expressions in bold); note that the k-flex sets are the same for $k = 1$ and 2, since the paraphrase variants move a_{1ij} and a_{2ij} in the same direction and with the same magnitude (that is, a_{1ij} and a_{2ij} have the same derivative with respect to w_{ij}, the only component of a_{1ij} and a_{2ij} altered by the variants that is relevant to the paraphrasing). Variables $p_{1,2}$, $p_{3,2}$, $p_{5,1}$, $p_{8,2}$, $p_{9,3}$ and $p_{12,1}$ then aggregate to form variable q_1, which can take integer values in the range $[0, 6]$; variables $p_{3,1}$ and $p_{7,1}$ aggregate to q_2 with range $[0, 2]$; and variables $p_{7,2}$ and

[9] The sample sizes are unusual because of the discarding of the IPs where the POLE and BALANCE formulations were the same.

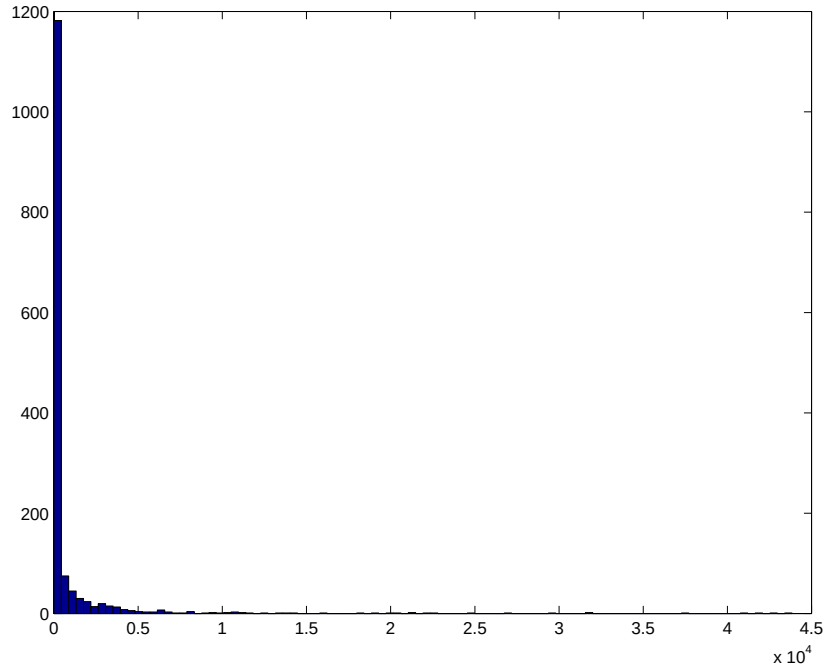


Figure 8.19: Distribution of search space sizes under default formulation

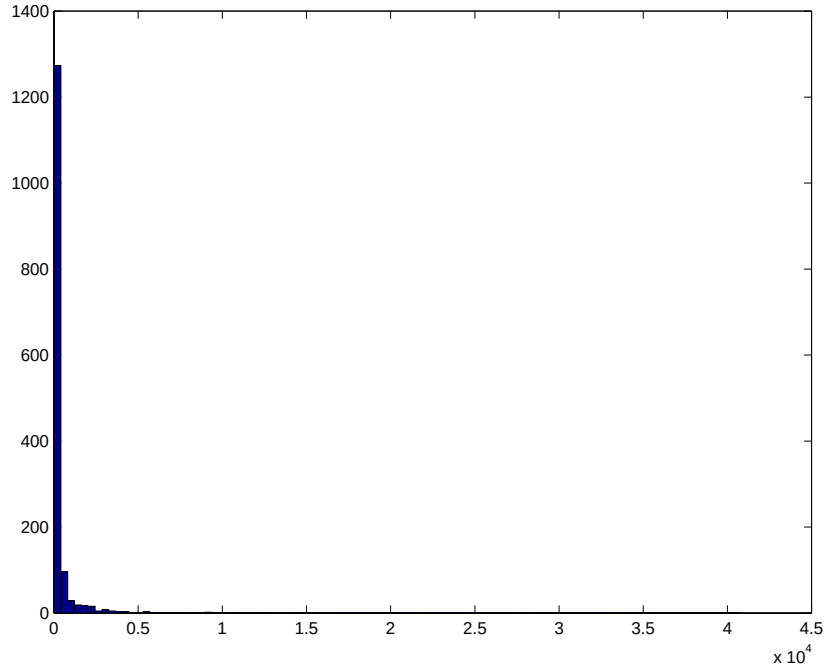


Figure 8.20: Distribution of search space sizes under POLE formulation

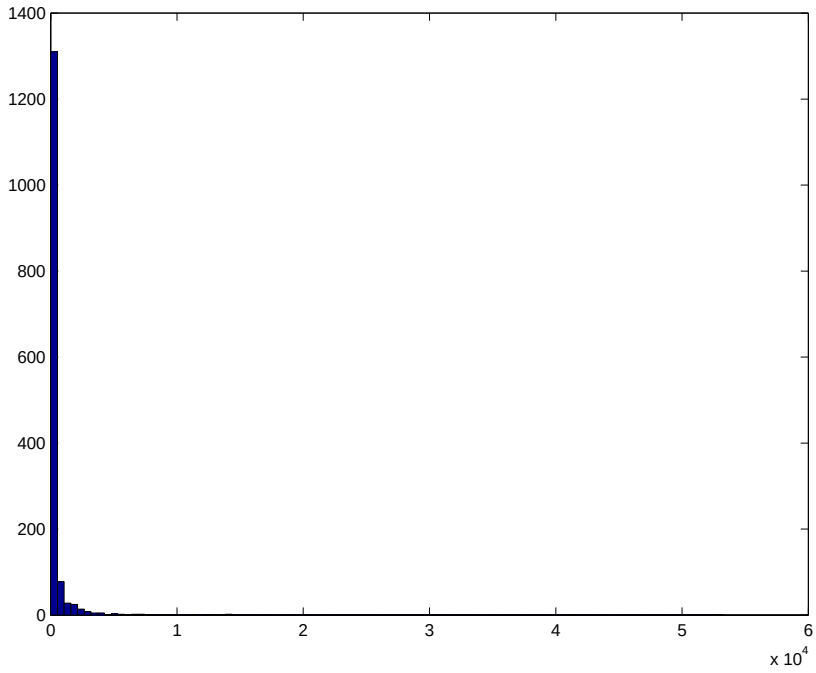


Figure 8.21: Distribution of search space sizes under BALANCE formulation

		mean	std. dev.
one constraint	default	910.09	3616.7
	POLE	371.22	1792.8
	BALANCE	381.48	2037.4
two constraints	default	201.45	1435.5
	POLE	21.76	72.7
three constraints	default	2232.0	6682.7
	POLE	192.23	341.0

Figure 8.22: Comparing search sizes of algorithms

$p_{1,2}$
- The question in my title implies a premise: that historically the United States has well afforded to be a nation of immigrants. Indeed, **it** has benefited handsomely from its good fortune as an immigrant destination.
- The question in my title implies a premise: that historically the United States has well afforded to be a nation of immigrants. Indeed, **the nation** has benefited handsomely from its good fortune as an immigrant destination.

$p_{5,1}$
- Today, however, the United States is a nation of some 264 million souls. **They** are on a continent developed beyond Lincoln's imagination.
- Today, however, the United States is a nation of some 264 million souls. **These souls** are on a continent developed beyond Lincoln's imagination.
- Today, however, the United States is a nation of some 264 million souls. **These 264 million souls** are on a continent developed beyond Lincoln's imagination.

$p_{8,2}$
- To put those numbers in perspective: prior to 1965 the period of heaviest immigration to the United States was the quarter century preceding the First World War, when some 17 million people entered the country. **This was** roughly half the total number of Europeans who migrated to the United States in the century after 1820 (along with several hundred thousand Asians).
- To put those numbers in perspective: prior to 1965 the period of heaviest immigration to the United States was the quarter century preceding the First World War, when some 17 million people entered the country. **These were** roughly half the total number of Europeans who migrated to the United States in the century after 1820 (along with several hundred thousand Asians).
- To put those numbers in perspective: prior to 1965 the period of heaviest immigration to the United States was the quarter century preceding the First World War, when some 17 million people entered the country. **These people were** roughly half the total number of Europeans who migrated to the United States in the century after 1820 (along with several hundred thousand Asians).
- To put those numbers in perspective: prior to 1965 the period of heaviest immigration to the United States was the quarter century preceding the First World War, when some 17 million people entered the country. **These 17 million people were** roughly half the total number of Europeans who migrated to the United States in the century after 1820 (along with several hundred thousand Asians).
- To put those numbers in perspective: prior to 1965 the period of heaviest immigration to the United States was the quarter century preceding the First World War, when some 17 million people entered the country. **These 17 million people who entered the country were** roughly half the total number of Europeans who migrated to the United States in the century after 1820 (along with several hundred thousand Asians).

$p_{9,3}$
- The last pre-war census, in 1910, counted about 13.5 million foreign-born people in the American population. **This was** in contrast to about 22.5 million in 1994.
- The last pre-war census, in 1910, counted about 13.5 million foreign-born people in the American population. **This census was** in contrast to about 22.5 million in 1994.

Figure 8.23: Variant referring expressions for Figure 8.8

$p_{10,1}$ aggregate to q_3, also with range $[0, 2]$. In these aggregate variables, the costs c_{ij} are the average of the costs of the component binary variables.

Mutual exclusion constraints

The IPs representing paraphrases of actual text, as well as containing constraints representing the external restrictions on the text such as length or readability, contain additional constraints, mutual exclusion conditions (MECs), specifying which paraphrases cannot be applied together (see Section 8.3.1). In order to make aggregation more likely to occur for actual text, it is useful to treat these MECs separately from the external constraints. It is clear that, if total c-sets were used in the criterion for aggregating, then aggregation would never occur between paraphrases of different sentences, as their variables would have differing coefficients in the MECs. This section shows that if the BASE IP (objective function plus external constraints) is aggregated, using the techniques of Section 8.5.3 with partial c-sets, then new MECs can be imposed on the base IP of the corresponding general formulation which will ensure that the IP solution does not propose more than one paraphrase per sentence.

The notation used in this thesis has the decision variables indexed by two indices; the first represents the (first) sentence involved in the paraphrase, and the second represents the paraphrase alternative for that sentence. The MECs then prevent alternative paraphrases for the same sentence being chosen together, using the inequalities

$$\sum_j p_{ij} \leq 1, \ \forall i$$

These inequalities induce a partitioning of the set of paraphrase variables, with partitions corresponding to sentences; define this as a BINARY-PARTITION (or B-PARTITION) of the variables.

When aggregating variables, new aggregate variables could often be combinations of binary variables from different B-partitions. These aggregate variables have been introduced to reduce the size of the search space by, in effect, 'forgetting' the ordering of the corresponding interchangeable binary solutions. That is, it doesn't matter which particular binary variables take the value 1 and which 0; after the optimal values are decided for the aggregate variables, the value of each aggregate variable can be considered to be distributed across its binary components. However, it is important that this distribution does not violate the fundamental constraint that not more than one paraphrase is chosen per sentence.

Take the example of actual text from Section 8.4. The MECs, ensuring each sentence has no more than one paraphrase applied, are as follows:

par.	c_{ij}	a_{1ij}	a_{2ij}	k-flex	opt. val.	maps to
$p_{1,1}$	24	-12	-12		0	
$p_{1,2}$	1	1	-17	$\{1\}$	1	q_1
$p_{3,1}$	4	-5	-5		0	q_2
$p_{3,2}$	3	2	-17		1	q_1
$p_{3,3}$	2	-1	17		0	
$p_{5,1}$	3	2	-17	$\{1,3\}$	1	q_1
$p_{5,2}$	1	0	17		0	
$p_{5,3}$	3	-5	12		0	
$p_{7,1}$	12	-5	-5		1	q_2
$p_{7,2}$	40	-17	-17		0	q_3
$p_{8,1}$	56	-24	-24		0	
$p_{8,2}$	3	2	-17	$\{1,3,7\}$	1	q_1
$p_{8,3}$	16	-6	-6		0	
$p_{9,1}$	4	-2	-2		0	
$p_{9,2}$	16	-8	-8		0	
$p_{9,3}$	3	2	-17	$\{1\}$	1	q_1
$p_{10,1}$	20	-17	-17		1	q_3
$p_{11,1}$	9	4	-17		1	
$p_{12,1}$	4	2	-17		1	q_1

par.	c_{ij}	a_{1ij}	a_{2ij}	range	components	opt. val.
$r_{1,1}$	24	-12	-12	1		0
$r_{3,3}$	2	-1	17	1		0
$r_{5,2}$	1	0	17	1		0
$r_{5,3}$	3	-5	12	1		0
$r_{8,1}$	56	-24	-24	1		1
$r_{8,3}$	16	-6	-6	1		0
$r_{9,1}$	4	-2	-2	1		0
$r_{9,2}$	16	-8	-8	1		0
$r_{11,1}$	9	4	-17	1		1
q_1	3	2	-17	6	$p_{1,2}$ $p_{3,2}$ $p_{5,1}$ $p_{8,2}$ $p_{9,3}$ $p_{12,1}$	6
q_2	8	-5	-5	2	$p_{3,1}$ $p_{7,1}$	1
q_3	30	-17	-17	2	$p_{7,2}$ $p_{10,1}$	1

Figure 8.24: Aggregation of variables for text of Figure 8.7: original variables p_{ij} and new aggregation variables r_{ij} and q_k

$$\begin{array}{lcllcll}
p_{1,1} & + & p_{1,2} & & & \leq & 1 \\
p_{3,1} & + & p_{3,2} & + & p_{3,3} & \leq & 1 \\
p_{5,1} & + & p_{5,2} & + & p_{5,3} & \leq & 1 \\
p_{7,1} & + & p_{7,2} & & & \leq & 1 \\
p_{8,1} & + & p_{8,2} & + & p_{8,3} & \leq & 1 \\
p_{9,1} & + & p_{9,2} & + & p_{9,3} & \leq & 1 \\
p_{10,1} & & & & & \leq & 1 \\
p_{11,1} & & & & & \leq & 1 \\
p_{12,1} & & & & & \leq & 1
\end{array}$$

This gives 9 (B-)partitions, and within each partition only one variable can be chosen to take the value 1. In the aggregate variant of the IP, there is an aggregate integer variable q_1 with range $[0, 6]$. This variable q_1 cuts across 6 partitions, representing the set of sentences {1, 3, 5, 8, 9, 12}; if any variables in one of these partitions that is not a component of q_1 has the value 1, q_1 cannot spread across any of its value into this partition. In addition, if any of the variables in these partitions has a non-zero value, this will decrease the maximum value that q_1 can take by this amount. For example, if $p_{1,1}$ is 1, the maximum value for q_1 is 5, as it can only spread its value across the set of sentences $\{3, 5, 8, 9, 12\}$.

This suggests two levels of partitions in the aggregate general IP formulation. The first is the same as for the binary IP model: binary variables representing paraphrases of a specific sentence still have conditions specifying that at most only one can be chosen per sentence. The second is a partitioning of all variables such that it takes into account that particular variables having non-zero values will reduce the maximum value for an aggregate variable; in other words, there is a maximum value for the sum of all variables in this larger partition, termed an AGGREGATE PARTITION, or A-PARTITION.

Definition 8.5.3 *A variable V_1 shares an A-partition with a variable V_2 if and only if*

- *V_1 and V_2 are derived from the same sentence, or*
- *V_1 in sentence $\mathcal{S}_1$ is linked to another sentence $\mathcal{S}_2$ via an aggregate variable (that is, there exists an aggregate variable with component binary variables from $\mathcal{S}_1$ and $\mathcal{S}_2$), and variables in $\mathcal{S}_2$ share an A-partition with V_2.*

A consequence of this definition is that no B-partition is part of more than one A-partition; an A-partition is also a partitioning of B-partitions.

Now, given the original binary IP has the (binary) variables p_{ij}, in the corresponding general IP, we will let

$$\begin{array}{rcl}
r_{ij} & = & \begin{cases} p_{ij} & \text{if } p_{ij} \text{ is } \textit{not} \text{ part of an aggregated variable} \\ 0 & \text{otherwise} \end{cases} \\
q_k & = & \text{aggregate variable } k
\end{array}$$

Then, we will define a flag variable, indicating whether two IP variables share an A-partition.

$$\mathcal{P}(V_1, V_2) \quad = \quad \begin{cases} 1 & \text{if } V_1 \text{ and } V_2 \text{ share an A-partition, where } V_i \text{ is of type } r_{ij} \text{ or } q_k \\ 0 & \text{otherwise} \end{cases}$$

Given this, we can show that it is possible to have MECs on the general IP equivalent in effect to those on the corresponding binary IP.

Proposition 8.5.4 *The conditions*

$$\sum_j r_{ij} \leq 1, \quad \forall i \tag{8.10}$$

$$\sum_{ij} r_{ij}\mathcal{P}(r_{ij}, q_k) + q_k + \sum_{l \neq k} q_l \mathcal{P}(q_l, q_k) \leq \sum_i \mathcal{P}(r_{i1}, q_k), \quad \forall k \tag{8.11}$$

guarantee a solution for the general IP which is equivalent to the MECs on the original binary IP ensuring that the paraphrases do not overlap.

Proof:

The first part is to show that the original conditions imply the new conditions; that is, that the new MECs allow the same choices of variables as the originals.

For (8.10), each of the conditions $\sum_j p_{ij} \leq 1$ implies $\sum_j r_{ij} \leq 1$, as the set $\{r_{ij} \,|\, r_{ij} = 1\}$ is a subset of $\{p_{ij} \,|\, p_{ij} = 1\}$.

For (8.11), form an A-partition of the sentences according to the aggregate IP. Then form a partition of the original MECs corresponding to the A-partitioning of the sentences. Then, take the cell from the MEC partitioning corresponding to the A-partition containing the aggregate variable q_k, which has the MECs

$$\begin{aligned} \sum_j p_{i_1,j} &\leq 1 \\ \sum_j p_{i_2,j} &\leq 1 \\ &\cdots \\ \sum_j p_{i_m,j} &\leq 1 \end{aligned}$$

Summing these together,

$$\sum_{k \in \{i_1, \ldots, i_m\}} \sum_j p_{kj} \quad \leq \quad m \tag{8.12}$$

Now, on the lefthand side of (8.12) a subset of the variables will, in mapping to the aggregate IP, aggregate together; one of these aggregate variables will be q_k (since this is the partition corresponding to the A-partition with q_k), and the others will be those

aggregate variables that share the partition with q_k, i.e. $\sum_{l \neq k} q_l \mathcal{P}(q_l, q_k)$. The remaining variables are those that are not aggregated together but which also share the A-partition with q_k, i.e. $r_{ij} \mathcal{P}(r_{ij}, q_k)$.

Therefore the lefthand side can be rewritten as

$$\sum_{ij} r_{ij} \mathcal{P}(r_{ij}, q_k) + q_k + \sum_{l \neq k} q_l \mathcal{P}(q_l, q_k)$$

For the righthand side of (8.12), m is just the number of sentences in the partition; in the corresponding A-partition, the count of sentences in the partition is given by $\sum_{ij} \mathcal{P}(r_{i1}, q_k)$.

Thus the original MECs imply the new ones.

The second part of the proof is to show that the new MECs don't allow violation of the original restrictions; that is, the aggregate IP with its new MECS gives solutions where it is possible to distribute the value of the general variables so that there is at most one paraphrase per sentence.

Assume that the aggregate IP solution has n binary r_{ij} variables equal to one, with the others zero. The condition in (8.10) means that n B-partitions in an A-partition are OCCUPIED, that is, contain a variable equal to one without doubling up, with this doubling-up prevented by $\sum_j r_{ij} \leq 1$;[10] that is, each of the corresponding n sentences has a paraphrase applied. The other B-partitions are UNOCCUPIED, that is, contain only variables equal to zero. The conditions in (8.11) then become

$$q_k + \sum_{l \neq k} q_k \mathcal{P}(q_l, q_k) \quad \leq \quad \sum_i \mathcal{P}(r_{i1}, q_k) - n \tag{8.13}$$

That is, the aggregate variables in A-partition k have, as the maximum total value to be spread across, a value equal to the number of remaining unoccupied B-partitions; that is, the aggregate variable values can be spread across the A-partition without clashes, and the corresponding sentences will not have more than one paraphrase applied. □

The two condition types (8.10) and (8.11) correspond to the two types of partition. The first is effectively the same as the binary IP conditions, leaving out the binary variables which are aggregated into a general variable. The key component of the second type is q_k: this is the general variable controlled by this condition. The other terms represent the variables in the same A-partition which affect its value. The remaining terms are just those other variables that are part of the same A-partition as q_k. The righthand side of the inequality is a count of the number of sentences involved in the A-partition; this is the set of sentences the aggregate variable q_k is able to spread across.

The effect of this proposition is to allow aggregation on just the partial c-sets of a binary IP; new MECs can then be derived which make the aggregate IP equivalent to the original binary one.

From the example of this section, summarised in Table 8.24, it can be seen that both types of condition are necessary. The inequalities are

[10] Note that they also imply that n is less than or equal to the number of B-partitions in the A-partition.

$$\sum_{ij} r_{ij} \leq 1 \tag{8.14}$$

$$\sum_{i \in \{1,3,5,7,8,9,10,12\}} \sum_{j} r_{ij} + q_1 + q_2 + q_3 \leq 8 \tag{8.15}$$

Without (8.14), it would be possible to get solutions such as $q_1 = 4$, $r_{3,1} = 1$, $r_{3,3} = 1$; without (8.15), it would be possible to have every B-partition occupied and still give q_1 a non-zero value.

Note that the solutions—vectors of decision variables—are equivalent for the binary IP and the aggregate IP: the aggregate variables in the aggregate IP can be distributed across their component variables in such a way that it is identical to the optimal solution for the binary IP. In fact, for this example there is only one way to spread the values of the aggregate variables.

8.5.5 Approximation versus Exactness

The work of the previous section has parallels to the approximate symmetry of Ellman (1993). Ellman aims to abstract away from a set of concrete states in a search space (as Joslin and Roy, 1997, do with their lifted descriptions of CSPs) to form an 'abstraction space', which he characterises in terms of permutation groups operating on states in the concrete search space. He defines an abstraction space as follows: given a group Σ of permutations that operate on states $s \in S$, then for all pairs of states s_1 and s_2 drawn from S, s_1 and s_2 lie in a common abstract state a if and only if there exists a $\sigma \in \Sigma$, such that $\sigma(s_1) = s_2$; this will yield a partition of S.

Exact symmetry in this notation is defined for a goal function $G : S \rightarrow \{\mathbf{true}, \mathbf{false}\}$ (the problem for Ellman being the checking of satisfiability of CSPs) with respect to Σ if for each $s \in S$, and each $\sigma \in \Sigma$, $G(\sigma(s)) = G(s)$. Inducing a partition of S using Σ will give an abstraction space A with the useful property that each abstract state a will have component concrete states $s_i \in a$ which have the same value for the goal function. That is, if any $s_i \in a$ fails to satisfy the goal function, all $s \in a$ will fail, and all can be pruned at once. Conversely, if one $s_i \in a$ succeeds, all succeed.

The two sorts of approximation Ellman describes as interesting are necessary approximations and sufficient approximations. In the former, $G \Rightarrow \tilde{G}$ (that is, G entails $\tilde{G}$). Only the first of the two consequences of exact symmetry holds here: if any $s_i \in a$ fails, all fail, and pruning can occur; but the converse is not necessarily true. Under sufficient approximation, $\tilde{G} \Rightarrow G$ (that is, $\tilde{G}$ entails G). Only the second of the two consequences of exact symmetry holds here: if any $s_i \in a$ succeeds, all succeed. These ideas of abstraction occur in other related work also, corresponding to 'TI abstraction' and 'TD abstraction' in Giunchiglia and Walsh (1992) and 'Upward Solution Property' and 'Downward Solution Property' in Knoblock *et al* (1991).

Comparing this with the work of the preceding sections, it can be seen that a partition is induced, as in Ellman's model; in the variable aggregation model this is done by taking all of the variants of a particular paraphrase instance and forming a cell, with a partition of the space resulting, as each variant is placed in a cell, with the cells being disjoint. In the aggregation model, one variant is chosen from each of these cells to represent that

	total simulations	solns (exact)	solns (agg.)	Δ (exact, agg.)
1 constraint	2251	2243	2191	52
2 constraints	220	220	208	12

Figure 8.25: Exact versus approximate formulations

paraphrase (that is, to be the abstraction of that cell or abstract state a), using either the default variant, or one of the POLE and BALANCE algorithms. However, the parallels between Ellman's approximate symmetry and the aggregation described in previous sections are not complete. There is no guaranteed entailment relationship between this choice for the cell abstraction and the components of the cell. The only situation where there is a guaranteed entailment relation is when aggregation occurs without tweaking of coefficients; then, given Proposition 8.5.4, the binary and aggregate models are equivalent, that is, there is an exact symmetry in the binary IP which is exploited to produce the aggregate IP. This lack of entailment at other times means that rather than in general producing an equivalent formulation by aggregation, the resulting formulation cannot be guaranteed to have exactly the same solution space. It can be considered as an approximation of the formulation (albeit a close one) where all variants are explicitly considered during the optimisation process. This explicit form of the problem including variants could be achieved by making the decision variables the set of all variants; for example, let

$p_{ijk} =$ 0/1 variable representing the kth variant of the jth potential paraphrase for sentence i

The variants can be partitioned in a similar way to the partitioning of paraphrases by sentence boundaries, using MECs of the form

$$\sum_k p_{ijk} \leq 1$$

However, formulating the model in this way can lead to a much larger number of variables, which (in general) increases the size of the search space; choosing one variant and using that gives a smaller model, in the same way as aggregating different binary variables into one variable produces models with smaller search spaces (Section 8.5.3).

Give that it is an approximation, it is useful to know the extent of the approximation. To investigate this, a simulation was carried out where IPs were randomly generated, using the parameters of Section 8.5.3, that included all variants of paraphrases (where these variants were defined by the flex values for each variable). Then, the corresponding aggregate model was formulated, and the search spaces compared. In terms of search space size, the results are as those of the experiment of Section 8.5.3 comparing no-aggregation and aggregation models. Here, what is of interest are the problems caused by approximation: that is, the number of times the exact formulation would produce a solution where the aggregate one, because of its approximate nature, would not. The results for one and two constraints are shown in Table 8.25.

So most of the time the approximate formulation did as well as the exact; but the exact formulation, while more likely to produce a solution, was on average 2.5 times the size of the aggregate model (that is, had 2.5 times as many decision variables), making it become

infeasibly large much more quickly than the aggregate one.

8.5.6 Heuristic Methods

Given the formulation of Reluctant Paraphrase as an IP, it is now possible to apply general approximation methods to find solutions. It is well known that, as the number of variables increases, finding exact solutions for IPs becomes intractable, and approximate, heuristic-based methods have been developed to find near-optimal solutions in much shorter times. The most well known example of this, and one which has generated much research effort in the development of heuristic methods, is the Travelling Salesperson Problem (TSP), an IP problem where the objective is to find the minimum distance travelling among a set of cities; once formulated as an IP, the TSP easily has heuristic methods applied to it. Similarly, having formulated Reluctant Paraphrase as an IP, it is easy to apply heuristic methods that are more sophisticated than those used for other paraphrase systems, as described in Section 8.1. The methods of genetic algorithms, simulated annealing and tabu search will be outlined in this section, and how they might be applied to an IP formulation of RP; the section will also discuss the relationship between these heuristic methods and the aggregation method described in Section 8.5.2.

Genetic algorithms

This type of heuristic, initially proposed by Holland (1978), is based, according to advocates such as Goldberg (1989) and Koza (1992), on the mechanics of natural selection and natural genetics, using the principle of survival of the fittest, combined with random changes to facilitate movement towards globally better solutions. A typical genetic heuristic technique will have three major components: reproduction, crossover and mutation. One type of genetic heuristic, genetic algorithms, requires potential solutions to be encoded as 'chromosomal' strings; for the binary IP formulation of RP, this falls out naturally, with the string being merely the concatenation of the decision variables, that is, the string $p_{i_1 j_1} p_{i_1 j_2} \ldots p_{i_m j_{i_m}}$. An initial population is generated at random of strings of this form, and a performance score is calculated for each of these initial strings, representing the 'fitness' of the string, which determines its likelihood of being propagated to the next generation. In terms of RP, the fitness will be the cost of applying a set of paraphrases, that is, the objective function value.[11]

The first phase of the method, reproduction, then occurs. A string's probability of reproducing is a function of its performance score, so strings that have a lower objective function value (the aim in RP being to minimise the objective function value representing textual change) are more likely to reproduce and move forward to the next generation of strings. The result is a new pool of strings which are ready to 'mate'. Strings are paired off randomly, and a crossover point determined for each pair. This might involve segmenting each string into two—for example, $p_{i_1 j_1} p_{i_1 j_2} \ldots p_{i_k j_l} | p_{i_k j_{l+1}} \ldots p_{i_m j_{i_m}}$—and then exchanging the second halves of each string in a pair. These two second halves will be the same length for each string in a pair, so that all strings in the new population will be the

[11] A more recent variant, genetic programming (Koza, 1992), which to some extent has overtaken genetic algorithms, is less appropriate here. It was developed because some problems did not naturally have a 'chromosomal' representation; consequently, genetic programming was developed to allow program induction. However, this is unnecessary here as obtaining a string representation is straightforward.

same length as those of the previous population (and still equal in length to the number of decision variables of the IP model). After this crossover stage occur any possible mutations. Each character of a string is given a probability of mutating, or changing its value from 0 to 1 or 1 to 0; typically this is a fairly low probability, so that the solution will not vary wildly within the polyhedral set. This process continues, using the set of strings that result from each generation, until a specific termination rule is invoked—say, that no change in the population of strings had occurred after a specified number of generations.

For the general IP model, genetic algorithms would work in a fairly similar way. The strings would still consist of characters corresponding to the decision variables, but these would now be general integers rather than binary ones. Reproduction would be identical, calculating a performance score using the objective function. Crossover would also be the same: since substrings are only swapped from the same position (two paired strings are divided into substrings at the same point, say $p_{i_1 j_1} p_{i_1 j_2} \ldots p_{i_k j_l} | p_{i_k j_{l+1}} \ldots p_{i_m j_{i_m}}$ as above) there will not be any problem with inconsistent ranges for the variables represented by the substrings swapped. Mutation would be slightly different, as it would be necessary to specify how the value change would occur. For example, if the decision variable p_{ij} has the value k, mutation may be defined as a change of the decision variable value to within some range of the current value ($k - \delta$ to $k + \delta$), or to any value within the range of p_{ij}, or according to some other rule.

Simulated annealing

Like genetic algorithms, the process of simulated annealing starts with randomly generated potential solutions, and then uses heuristic techniques to select better and better candidate solutions until an optimal or near-optimal one is found. The metaphor for simulated annealing comes from physics, as against the evolution metaphor for genetic algorithms, in the application of discrete optimisation (Kirkpatrick, Gelatt and Vecchi, 1983). In the annealing process for solids in physics, the aim is to reach a ground state (global optimum) while avoiding metastable states (local optima), where these are defined in terms of energy (cost). The key feature of annealing that has led to useful results in optimisation is that it is possible for particles to move to higher energy states: in terms of optimisation, it can move to a solution with a higher cost, in contrast to simple greedy algorithms such as those of STREAK and YH where this is not possible. The frequency with which these moves occur is proportional to the 'temperature'—at the start of the annealing process, when temperatures are high, moves to higher energy states are likely to occur. During the (necessarily slow) cooling process, such moves become less likely. It has been shown (van Laarhoven and Aarts, 1987; Aarts and Korst, 1989) that the true global optimum will be found if the simulated annealing proceeds through a suitable sequence of temperatures, although there is no algorithmic way of determining what this suitable sequence of temperatures is for any particular problem. It has also been shown that simulated annealing is superior to using a simple greedy algorithm (such as that of YH) even if a large initial population is randomly chosen and the best greedy outcome from all of these selected (Lundy and Mees, 1986; Johnson, Aragon and McGeoch, 1989).

Apart from the different metaphor and method of navigating the search space, simulated annealing and genetic algorithms have similar sorts of requirements. The most important is the representation of the input. Again, the IP representation suits simulated annealing: the decision variables become a string which represents the state of the annealing. This

then allows the definition of 'neighbouring' states (generally, states with one character difference between the two strings), with moves to neighbouring states being determined by the objective function value and the temperature function of the annealing process.

Tabu search

Tabu search (Glover, 1986; Glover, Taillard and de Werra, 1993) is a heuristic which aims to avoid entrapment in local optima—thus missing the global optimum—by keeping a record of solutions that have already been visited, and ensuring that these are not visited again. This is similar to the idea behind WEIVER, but it has the key difference that not all solutions are kept. In general, earlier solutions are discarded from the tabu list and more recent ones kept, working on the premise that the current search will be occurring near more recent solutions, and not near earlier ones. Some variants of tabu search have a 'short term memory' and a 'long term memory': the short term memory is the cyclic one, discarding earlier solutions, while the long term memory is used to bias a new search space when search has been restarted. Recent work using ideas of tabu search in constraint satisfaction programming (Ginsberg and McAllester, 1994; Havens, 1997) has concentrated on showing that it is possible to build a cache of NoGoods (list of solutions already tried) that can be searched in polynomial time, as exponential caches have the same problems as searching the entire solution space (as is potentially the case for WEIVER).

Tabu search can be viewed as a metaheuristic (Ignizio and Cavalier, 1994): it can be used as a heuristic to guide other solution methods, either exact (such as branch-and-bound) or heuristic (such as the genetic algorithms and simulated annealing described earlier). As such, it could be integrated with these search methods to increase search efficiency.

Aggregation and heuristic searches

As noted above in the discussion of tabu search, it is possible to integrate heuristics of different types. The aggregation of variables described in this chapter is a different class of method for improving search, and can be combined with types of heuristics similar to those described above, or with techniques from other fields, such as that of Beale (1997).

The search methods described in this section, genetic algorithms and simulated annealing, are ones where the search time is reduced by applying heuristics to the process of moving around the solution space: moving towards higher fitness / lower energy states, occasionally introducing randomness to prevent being trapped in local minima, and so on. However, the models themselves are the same size, in terms of number of inputs and outputs, as the exact models they are finding approximate solutions for; it is only the search space that is navigated differently. The number of inputs—the variables for which the problem is being solved—is the same for the exact and approximate alternatives. Aggregation, however, reduces the number of variables, making the search space smaller, which also has the effect of making the search quicker, as demonstrated in Section 8.5.3; efficient navigation techniques can then be added to make the search quicker still. In doing this, aggregation has similarities to the methods of principal components analysis and factor analysis. Factor analysis has been applied to problems in language by Biber (1988) who used it to analyse the variation between styles of writing. He investigated 67 linguistic features (variables) such as types of tense, pronouns particles and word length, and used factor analysis to extract from these seven 'factors' (new group variables) which represent

the original 67 to some extent. These seven factors do not by default have any inherent meaning, although Biber provides interpretations for them: for example, Factor 1, from analysing its components and their weighting, appears to "represent a dimension marking high information density and exact information content versus affective, interactive and generalized content" (Biber, 1988: 107). In general, it is used to draw out underlying factors behind a set of variables; in other words, replacing an initial set of variables by a smaller set of different variables. However, it is generally only applicable when the variables are naturally motivated by underlying factors, such as those described by Biber for writing styles:

> Each factor [the small set of group variables derived from the originals] represents some area of the original data that can be summarized or generalized. That is, each factor represents an area of high shared variance in the data, a grouping of linguistic features that occur with a high frequency. [*ibid*: 79]

For example, if passives and nominalisations are highly correlated, they could be replaced by a single factor. The factors are linear combinations of the original variables; to derive them, it is necessary to calculate a correlation matrix of all variables. Aggregation of variables, on the other hand, can be applied when this is not the case, but where the variables can in some sense be naturally grouped together, as is the case for paraphrases tending to have similar coefficients. In addition, aggregation does not require calculating a matrix of correlations between variables; rather, it can be determined using the simple polynomial algorithms of Figures 8.16 and 8.17.

8.6 Summary

Only very simple methods for finding a good set of paraphrases have been used in other systems which perform some kind of textual rewriting. This has generally been satisfactory where the constraints have been positively correlated, and where either the number of choices of alternative paraphrases is small (as occurs when constraining text at the sentence level) or the system is satisfied with a choice found through only a short search.

In RP, this is not the case. The constraints are not necessarily correlated positively, the constraints apply over the whole text, resulting in a large number of possible paraphrases and hence many alternative texts, and the aim is to find a good (minimal change) solution.

This chapter has shown how Reluctant Paraphrase can be represented in an IP framework: the paraphrases are represented by decision variables, the constraints can be expressed by polynomial constraint conditions, and an objective function can be constructed to enable the selection of a good solution. Using this type of model allows search techniques such as branch-and-bound to be used which do not have the problems that using earlier algorithms for RP would have had.

The chapter then outlined a technique for removing symmetries from the model, which, as demonstrated by experimental data, improves the search for an optimal solution. Finally, it described how other heuristic search methods can also be applied to further improve the search for an optimal solution.

Chapter 9

Conclusion

This fairly brief conclusion consists of two parts. The first is an outline of the major results of the thesis; the second is a list of ideas for possible future work following from the work of the thesis.

9.1 Results

Framework

At the most general level, the contribution of this thesis is the task framework, Reluctant Paraphrasing (RP). The task—taking a text and making it conform to some imposed constraints, while keeping as close to the original as possible—is one that occurs frequently in practice, but unlike language generation and language checking has not been modelled. This thesis has developed a framework for integrating the various aspects of the task: collecting a set of paraphrases, representing them formally, looking at their effects on the text, and choosing from among them the subset that best enables the text to fit the constraints while causing minimal 'damage'.

Paraphrase collection

A useful byproduct of the thesis is the collection of paraphrases of Chapter 4. Other collections are more narrowly focussed, such as the sports-oriented information-adding paraphrases of Robin (1994), or the nominalisation transformations, used to explore issues in Transformational Generative Grammar, of Lees (1963). This collection is a general one, bringing together a range of textual mappings whose common feature is that the mapping alternatives are considered interchangeable.

Paraphrase definition

More importantly, the collection leads to a refinement of the notion of paraphrase, and what it means for one textual construction to be substitutable for another. This in turn leads to a semantic model for describing the effects of, or damage caused by, substituting one construction for another. This semantic model is a coarse one, using only notions of truth-conditional meaning, possible worlds semantics and information packaging, and

the approximate realisation of these in part of speech and syntactic structure. However, this sort of semantic model suits the intended purpose, a rough quantification used in the Integer Programming framework of Chapter 8.

Paraphrasing in S-TAG

A major component of the thesis is the expression of the collected paraphrases within a formalism. The formalism selected was Synchronous Tree Adjoining Grammar (S-TAG), as it is a well-defined, mathematically restricted one whose properties have already been explored to some extent, and which has already been used in applications in machine translation and syntax-semantics mappings. The thesis shows that paraphrases can be represented by two constructions, structural mapping pairs obeying certain conditions and generating functions; and that given this representation of paraphrase, structures under S-TAG remain well-formed.

Generalising S-TAG

Arising from this representation of paraphrase in S-TAG is the most significant aspect of the thesis. The structural complexity of paraphrase, and approaches to dealing with it, leads to a realisation that a number of problems with S-TAG relating to derivation and isomorphism can be solved by generalising the formalism. The central concept retained under this generalisation is the mapping between context free derivation structures such that an isomorphism exists, ensuring that the weak language preservation property holds. The generalisations relate in one case to mapping between object-level formalisms with differing generative capacities, the instance in this thesis being between standard TAG and MCTAG; this generalisation solves problems such as are encountered in the representation of coordination. In the other case, they relate to using a grammar at the meta-level to capture structure, and specify isomorphism based on the structure so defined; this generalisation solves problems that are encountered in unbounded movement of constituents, such as relative clauses in paraphrase, or clitics in machine translation. In this second case, moreover, this generalisation is achieved without increasing the generative capacity of the resulting object level formalism beyond that of TAG. The formal proofs of these generalisations led to a realisation that they could provide natural solutions to outstanding problems in TAG-based formalisms, including syntactic coordination, and theoretical difficulties connected with machine translation such as clitic climbing in romance languages.

Integer Programming

The final major component of the thesis is in the formulation of RP within an Integer Programming framework. This allows the expression of the components of RP—the set of paraphrases that can be applied to a text, the constraints to be satisfied, and a desire to minimise the effects of the paraphrase—in a way that deals naturally with issues such as conflicting constraints and the consequent need for efficient search space navigation. The thesis provided a model showing how paraphrases can be represented as decision variables; how constraints can be represented as constraint equations, so that they satisfy the definition of Integer Programming through linearity or some other property; and how the semantic model developed earlier in the thesis can lead to a quantified objective function.

Variable aggregation

Further, since Integer Programming is well known to have much scope for efficiency in formulations, the thesis investigated a number of formulations arising from the concept of symmetry breaking in artificial intelligence. This led to a technique of variable aggregation, with several variant formulations for the RP framework; through a number of simulation experiments, the thesis determined the best of a range of these formulations.

Overall

This thesis has drawn on a range of disciplines, bringing together in varying degrees work in formal grammars, mathematical programming, semantics and artificial intelligence, within the novel framework of Reluctant Paraphrase; and this bringing together of disciplines has provided an environment that led to the contributions outlined above. Further work that could potentially follow from this is described in the next section.

9.2 Future Work

Implementation

Foremost of the future work is the building of a complete implementation of the theory presented in this thesis. One of the reasons for selecting both the S-TAG formalism for the paraphrase representation, and the Integer Programming formalism for embodying Reluctant Paraphrase generally, is that there are large-scale, easily available systems for carrying out each part. For the former, there is the XTAG system (XTAG, 1995), which produces TAG parse trees from a broad-coverage grammar. For the latter, there is commercially available software for solving Integer Programming models. In the intermediate stage would be the selection from a 'paraphrase library', consisting of S-TAG tree pairs, with possible paraphrases for the original input text. A schematic diagram of such a system might look something like Figure 9.1.

In order for such a system to be tractable on larger-scale text, a number of sources of combinatorial explosion would have to be contained. Two predominant sources occur in the parsing, and in the solving of the Integer Programming model.

One possible approach for speeding up the parsing is to use SuperTAGs instead of carrying out full parsing. Such an approach is used in Srinivas (1997), where the ideas of Chandrasekar and Srinivas (1997) on sentence splitting are implemented using SuperTAG. The SuperTAGs obtained in the almost-parse of one or more sentences would still need to be combined in order to match with the S-TAG structural mapping pairs, given the structural complexity of paraphrases discussed in Section 6.2.2; exactly how this would be done is a research question.

Approaches for speeding up the Integer Programming solution include those discussed in Chapter 8: the technique of variable aggregation presented in Section 8.5.2, combined with the heuristic navigation techniques of Section 8.5.6.

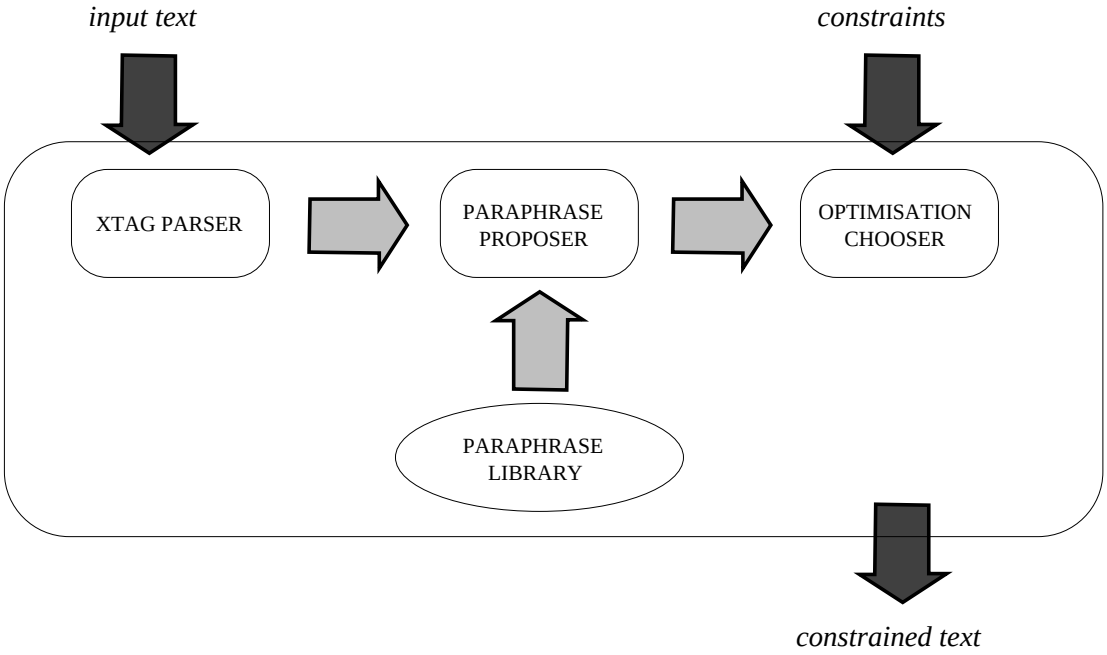


Figure 9.1: Schematic diagram for implementation

Formal grammar

In terms of the formal grammar aspect of the thesis, future work here could potentially involve extending the formal results, developing new results that would be useful in showing that an implementation would be feasible at a theoretical level, and extending to other areas the ideas behind the generalising of the S-TAG definition.

In the first category here, extensions could include those where the generalisations of S-TAG potentially lead to natural ways of dealing with coordination and machine translation between languages represented in formalisms with differing generative capacities, as may be the case for, say, English and Spanish; modelling of such applications would need to be more fully explored, to see how well the new generalisation of S-TAG fits. Further extensions could involve investigating transductions between more radically different grammars, for example, between TAG and UVG-DL (Unordered Vector Grammar with Dominance Links, from Rambow, 1994)—for mapping from syntax to a semantic representation that allows a better treatment of quantifiers than previous models (Rambow and Satta, 1996)—or between TAG and DTG (Rambow *et al*, 1995)—for mapping from English to a language with free word order phenomena such as German.

Recent work in dependency grammar (Kahane *et al*, 1998) suggests that results from the thesis could be applied there as well. In the thesis the idea of meta-level grammars was explored (specifically, the definition of a grammar for TAG derivation structures) along with the effects of generative power on the resulting formalism. In Kahane *et al*, 'lifting rules' are proposed to deal with problems giving formal results in non-projective dependency grammar; one objection at the paper presentation was that the rules could be arbitrary. Defining lifting rules by a grammar, and investigating the effects on the dependency grammar, would enable a precise characterisation of the effects.

Other results based on the generalisations of S-TAG could also be useful. A major one would be related to parsability results: whether, because of the multiplicity of possible paraphrases for a given construction, the transduction can be carried out in polynomial time. Rambow and Satta (1996) show this for synchronised UVG-DLs; however, this has not been shown even for synchronised standard TAGs, much less for the generalised ones presented in this thesis. In the case the transduction cannot be carried out in polynomial time, the sort of constraint-based techniques for navigating a search space described here may be able to be usefully applied internally to S-TAG.

Semantic model

The semantic model of this thesis, from Section 4.3, is only a coarse one, and could be developed much further. Two possibilities using TAG-related formalisms, to fit with the paraphrase representation, are modelling the semantics with S-TAG, and introducing a specific clausal relation representation with D-TAG (Cristea and Webber, 1997). For the former, it has already been noted that one of the original uses of S-TAG was for syntax-semantic mapping (Shieber and Schabes, 1990). Thus, instead of the semantics being defined only by a function from the syntax representation to a vector of reals (Section 7.3.2), it would have an explicit representation which could be used to produce the objective function of Chapter 8. However, this would require a semantic grammar, which is beyond the scope of this thesis. The second possibility, using D-TAG, would allow explicit modelling of the change in relations—as discussed in Section 4.2.3, for example, splitting a sentence often removes explicit relations, and these relations are what D-TAG describes. D-TAG also allows a uniform treatment of relations within the sentence and between sentences.

Integer Programming model

Future work here could involve either changes to the model, or further investigation of the current one. One potential change is the relaxation of the fundamental Reluctant Paraphrase assumption that the starting text is ideal and that any alteration should therefore be treated as 'damage' to the text. Some alterations may actually be improvements: this could be evaluated by psycholinguistic tests, showing, for example, that a clausal construction is better than its original nominal equivalents. This could be reflected in the objective function of the model of Chapter 8; the coefficients would merely no longer have the same sign, which does not affect the underlying Integer Programming model in any way.

The other area of future work would be to extend the investigation of the Integer Programming model; for example, by building a better model of sensitivity analysis. Although it is well known that Integer Programming does not come with methods of sensitivity analysis that arise naturally from the Integer Programming models themselves (Williams, 1995), it would be useful to explore whether there are any ways of producing general sensitivity results in a manner somewhat analogous to Linear Programming.

Appendix A

The XTAG Naming Scheme

Most of this material is taken from Appendix B of XTAG (1995), and is just to allow a quick reference for tree types referred to in the thesis.

A.1 General Naming Principles

Each tree begins with either an α (alpha) or β (beta) symbol, indicating whether it is an initial tree or an auxiliary tree, respectively. Following an α or a β the name may additionally contain one of:

I	imperative
E	ergative
N{0, 1, 2}	relative clause position
G	NP gerund
D	Determiner gerund
pW{0, 1, 2}	wh-PP extraction {position}
W{0, 1, 2}	wh-NP extraction {position}

Numbers are assigned according to the position of the argument in the declarative tree, as follows:

0	subject position
1	first argument (i.e. direct object)
2	second argument (i.e. indirect object)

The body of the name consists of a string of the following components, which correspond to the leaves of the tree. The anchor of the tree is indicated by capitalising the part of speech corresponding to the anchor; if there is more than one anchor, all corresponding parts of speech are capitalised. These are:

s	sentence
a	adjective
arb	adverb
be	*be*
x	phrasal category
d	determiner
v	verb
lv	light verb
conj	conjunction
comp	complementiser
it	*it*
n	noun
p	preposition
pl	particle
by	*by*
neg	negation

Tree families are named according to the basic declarative tree structure of the family, but with a T as the first character instead of an α or β.

A.2 Trees Used in this Thesis

αnx0Vnx1	verb with noun phrase argument (transitive verb)
αnx0Vpnx1	verb with prepositional phrase argument
αnx0Vnx1pnx2	verb with noun phrase and prepositional phrase arguments
αnx0Ax1	adjective small clause
αGnx0V	NP gerund as intransitive verb
αnx1Vbynx0	passive form of transitive verb with *by* clause
αNXdxN	noun phrase with determiner
αNXN	noun phrase without determiner
αDXD	determiner
αvxPnx	prepositional phrase
αsPUs	inter-sentence punctuation
βN0nx0Vnx1	subject relative clause form of transitive verb
βN0nx0Ax1	subject relative clause form of adjective small clause
βVvx	auxiliary verb
βAn	adjective
βARBvx	verbal adverb
βCOMPs	complementiser
βsPUs	inter-sentence punctuation
βnxPUnxPU	appositive noun phrase punctuation

Bibliography

[1] Aarts, E. and J. Korst. 1989. *Simulated Annealing and Boltzmann Machines.* John Wiley. New York, NY.

[2] Abeillé, Anne and Yves Schabes. 1989. Parsing Idioms in Tree Adjoining Grammars. *Fourth Conference of the European Chapter of the Association for Computational Linguistics.*

[3] Abeillé, Anne, Yves Schabes and Aravind Joshi. 1990. Using Lexicalized TAGs for Machine Translation. *Proceedings of the 13th International Conference on Computational Linguistics*, 1–6.

[4] Abeillé, Anne. 1992. Synchronous TAGs and French Pronominal Clitics. *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, 60–66.

[5] Allerton, D. 1982. *Valency and the English verb.* Academic Press. London, UK.

[6] Allwood, Jens, Lars-Gunnar Andersson and Östen Dahl. 1977. *Logic in Linguistics.* Cambridge University Press. Cambridge, UK.

[7] Almqvist, Ingrid and Anna Sagvall Hein. 1996. Defining Scania Swedish—a controlled language for truck maintenance. *Proceedings of the First International Workshop on Controlled Language Applications*, 159–165.

[8] Barthe, Kathleen. 1996. EUROCASTLE—A User's Experience with Prototype AECMA SE Checkers. *Proceedings of the First International Workshop on Controlled Language Applications*, 42–63.

[9] Bazaraa, M., H. Sherali and C. Shetty. 1993. *Nonlinear Programming: Theory and Algorithms.* John Wiley. New York, NY.

[10] Beale, Stephen. 1997. Using Branch-and-Bound with Constraint Satisfaction in Optimization Problems. *Proceedings of the 1997 Conference of the American Association for Artificial Intelligence*, 209–214.

[11] Bell, Allan. 1991. *The Language of News Media.* Blackwell. Oxford, UK.

[12] Benhamou, B. 1994. Study of symmetry in Constraint Satisfaction Problems. In *Principles and Practice of Constraint Programming.*

[13] Benhamou, B. and L. Sais. 1992. Theoretical Study of Symmetries in Propositional Calculus and Applications. In Kapur, D. (ed.) *Automated Deduction: 11th International Conference on Automated Deduction*, 281–294. Springer-Verlag. New York, NY.

[14] Berwick, R. and A. Weinberg. 1984. *The Grammatical Basis of Linguistic Performance.* MIT Press. Cambridge, MA.

[15] Biber, Douglas. 1988. *Variation Across Speech and Writing.* Cambridge University Press. Cambridge, UK.

[16] Bleam, Tonia. 1994. Clitic Climbing and The Power of Tree Adjoining Grammar. *Symposium on Tree Adjoining Grammar.*

[17] Bradac, James, Roger Desmond and Johnny Murdock. 1977. Diversity and Density: Lexically Determined Evaluative and Informational Consequences of Linguistic Complexity. *Communication Monographs* 44, 273–283.

[18] Bruce, Bertram, Andee Rubin and Kathleen Starr. 1981. Why Readability Formulas Fail. *IEEE Transactions on Professional Communication*, 24(1), 50–52.

[19] Candito, Marie-Hélène. 1996. A principle-based hierarchical representation of ltags. *Proceedings of the 16th International Conference on Computational Linguistics*, 194–199.

[20] Candito, Marie-Hélène and Sylvain Kahane. 1998. Can the TAG derivation tree represent a semantic graph? An answer in the light of Meaning-Text Theory. *Fourth International Workshop on Tree Adjoining Grammars and Related Frameworks*, 21–24.

[21] Carroll, John. 1993. *Practical Unification-based Parsing of Natural Language.* University of Cambridge Computer Laboratory.

[22] Chandrasekar, Raman. 1994. *A Hybrid Approach to Machine Translation using Man Machine Communication.* PhD thesis, University of Bombay.

[23] Chandrasekar, Raman, Christy Doran and B. Srinivas. 1996. Motivations and Methods for Text Simplification. *Proceedings of the 16th International Conference on Computational Linguistics*, 1041–1044.

[24] Chandrasekar, Raman and B. Srinivas. 1997. Automatic Induction of Rules for Text Simplification. *Knowledge-Based Systems*, 10, 183–190.

[25] Cherry, Lorinda. 1981. *Writing Tools—The STYLE and DICTION Programs.* Bell Labs Computing Science Technical Report No. 91.

[26] Chomsky, Noam. 1957. *Syntactic Structures.* Mouton. The Hague, The Netherlands.

[27] Chomsky, Noam. 1965. *Aspects of the Theory of Syntax.* MIT Press. Cambridge, MA.

[28] Chomsky, Noam. 1981. *Lectures in Government and Binding.* Studies in Generative Grammar 9. Foris. Dordrecht, The Netherlands.

[29] Clémencin, Grégoire. 1996. Integration of a CL-checker in an Operational SGML Authoring Environment: Methodological and Technical Issues. *Proceedings of the First International Workshop on Controlled Language Applications*, 32–41.

[30] Coleman, E. 1962. Improving Comprehensibility by Shortening Sentences. *Journal of Applied Psychology*, 46, 131–134.

[31] Crawford, J., M. Ginsberg, E. Luks and A. Roy. 1996. Symmetry Breaking Predicates for Search Problems. *Proceedings of the 1996 Conference on Knowledge Representation*

[32] Cristea, Dan and Bonnie Webber. 1997. Expectations in Incremental Discourse Processing. *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, 88–95.

[33] Crystal, David and Derek Davy. 1969. *Investigating English Style.* Longman Group UK Limited. Harlow, England.

[34] Dale, Edgar and Jeanne Chall. 1948. A formula for predicting readability. *Educational Research Bulletin*, 27, 11–20 and 37–54.

[35] Dale, Robert. 1988. *Generating Referring Expressions in a Domain of Objects and Processes.* PhD thesis. University of Edinburgh.

[36] Davison, Alison, Robert Kantor, Jean Hannah, Gabriella Hermon, Richard Lutz and Robet Salzillo. 1980. *Limitations of Readability Formulas in Guiding Adaptations of Texts.* BBN Technical Report no. 162.

[37] de Beaugrande, R. and W. V. Dressler. 1981. *Introduction to Text Linguistics.* Longman. New York, NY.

[38] Delin, Judy. 1989. *Cleft Constructions in Discourse.* PhD thesis, University of Edinburgh.

[39] Delin, Judy and Jon Oberlander. 1995. Syntactic constraints on discourse structure: the case of *it*-clefts. *Linguistics*, 33(3).

[40] DiMarco, Chrysanne and Graeme Hirst. 1993. A Computational Theory of Goal-Directed Syntax. *Computational Linguistics*, 19(3), 451–499.

[41] DiMarco, Chrysanne, Graeme Hirst and Manfred Stede. 1993. The semantic and stylistic differentiation of synonyms and near-synonyms. *Proceedings of the AAAI Spring Symposium on Building Lexicons for Machine Translation.*

[42] Doran, Christy, D. Egedi, B.A. Hockey, B. Srinivas and M. Zaidel. 1994. XTAG System—A Wide Coverage Grammar of English. *Proceedings of COLING94*, 922–928.

[43] Douglas, Shona and Matthew Hurst. 1996. Controlled Language Support for Perkins Approved Clear English (PACE). *Proceedings of the First International Workshop on Controlled Language Applications*, 93–105.

[44] Dras, Mark and Mike Johnson. 1996. Death and Lightness: Using a Demographic Model to Find Support Verbs. *Proceedings of the 5th International Conference on the Cognitive Science of Natural Language Processing*, Dublin, Ireland.

[45] Dras, Mark. 1997a. Representing Paraphrases Using S-TAGs. *Proceedings of the 35th Meeting of the Association for Computational Linguistics*, 516–518.

[46] Dras, Mark. 1997b. Reluctant Paraphrase: Textual Restructuring under an Optimisation Model. *Proceedings of PACLING97*, 98–104.

[47] Dras, Mark. 1998. Search in Constraint-Based Paraphrasing. *Natural Language Processing and Industrial Applications (NLP+IA98)*, 213–219.

[48] Drury, Alinda. 1985. Evaluating Readability. *IEEE Transactions on Professional Communication*, 28(4), 11–14.

[49] Duffy, Thomas. 1985. Readability Formulas: What's the Use? In Duffy, T. and R. Waller (eds) *Designing Usable Texts.* Academic Press. Orlando, FL.

[50] Duffy, Thomas and P. Kabance. 1982. Testing a readable writing approach to text revision. *Journal of Educational Psychology*, 74, 733–748.

[51] Elhadad, Michael. 1992. *Using Argumentation to Control Lexical Choice: A Functional Unification Implementation.* Ph.D. thesis, Graduate School of Arts and Sciences, Columbia University.

[52] Ellman, Thomas. 1993. Abstraction via Approximate Symmetry. *Proceedings of the 1993 International Joint Conference on Artificial Intelligence*, 916–921.

[53] Emonds, J. 1976. *A Transformational Approach to Syntax.* Academic Press. New York, NY.

[54] Entin, E. and George Klare. 1978. Some Inter-relationships of Readability, Cloze, and Multiple-choice Scores on a Reading Comprehension Test. *Journal of Reading Behaviour*, 10, 417–436.

[55] Evans, Roger, Gerald Gazdar and David Weir. 1995. Encoding Lexicalized Tree Adjoining Grammars with a Nonmonotonic Inheritance Hierarchy. *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, 77–84.

[56] Fairclough, Norman. 1992. *Discourse and Social Change.* Polity Press. Cambridge, UK.

[57] Farrington, Gordon. 1996. AECMA Simplified English: an Overview of the International Aircraft Maintenance Language. *Proceedings of the First International Workshop on Controlled Language Applications*, 1–21.

[58] Flesch, Rudolf. 1943. *Marks of readable style: A study in adult education.* Bureau of Publications, Teachers College, Columbia University. New York, NY.

[59] Foucault, Michel. 1972. *The Archaeology of Knowledge.* Tavistock Publications. London, UK.

[60] Fowler, Roger, B. Hodge, G. Kress and T. Trew. 1979. *Language and Control.* Routledge. London, UK.

[61] Frank, Robert. 1992. *Syntactic Locality and Tree Adjoining Grammar: Grammatical, Acquisition and Processing Perspectives.* PhD thesis, University of Pennsylvania.

[62] Frank, Robert and K. Vijay-Shanker. 1998. TAG derivation as monotonic c-command. *Fourth International Workshop on Tree Adjoining Grammars and Related Frameworks*, 46–49.

[63] Frase, Lawrence. 1981. Ethics of Imperfect Measures. *IEEE Transactions on Professional Communication*, 24(1), 48–50.

[64] Frazier, L. and J. Fodor. 1979. The sausage machine: a new two-stage parsing model. *Cognition*, 6, 191–325.

[65] Fry, Edward. 1965. *Teaching faster reading: A manual.* Cambridge University Press. Cambridge, UK.

[66] Gabriel, Richard. 1988. Deliberate Writing. In David McDonald and Leonard Bolc (eds.) *Natural Language Generation Systems.* Springer-Verlag. New York, NY.

[67] Gates, A. and W. MacGintie. 1965. *Gates-MacGintie reading tests, Survey D.* Teachers College Press. New York, NY.

[68] Gazdar, Gerald, Ewan Klein, Geoffrey Pullum and Ivan Sag. 1985. Generalized Phrase Structure Grammar. Harvard University Press. Cambridge, MA.

[69] Gécseg, Ferenc and Marcus Steinby. 1984. *Tree Automata.* Akadémiai Kiadó. Budapest, Hungary.

[70] Gécseg, Ferenc and Marcus Steinby. 1996. Tree Languages. In Grzegorz Rozenberg and Arto Salomaa (eds.) *Handbook of Formal Languages, Vol 3.* Springer-Verlag. Berlin, Germany.

[71] Ginsberg, M. and D. McAllester. 1994. GSAT and Dynamic Backtracking. *Proceedings of the 1994 Workshop on Principles and Practice of Constraint Programming.*

[72] Giuchiglia, F. and T. Walsh. 1992. A Theory of Abstraction. *Artificial Intelligence*, 57, 323–389.

[73] Givón, Talmy. 1979. *On understanding grammar.* Academic Press. New York, NY.

[74] Glover, F. 1986. Future Paths for Integer Programming and Links to Artificial Intelligence. em Computers and Operations Research, 13, 533–549.

[75] Glover, F., E. Taillard and D. de Werra. 1993. A User's Guide to Tabu Search. In F. Glover and D. de Werra (eds.) *Annals of Operations Research: Tabu Search*, 41.

[76] Goldberg, David. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley.

[77] Goldberg, E. and N. Driedger. 1994. Using Natural Language Processing to Produce Weather Forecasts. *IEEE Expert, April 1994.*

[78] Gomory, R. 1960. *An Algorithm for the Mixed Integer Problem.* RAND Corporation Research Memorandum RM-2597. Santa Monica, CA.

[79] Gorn, S. 1962. Processors for Infinite Codes of Shannon-Fanno type. *Symp. Math. Theory of Automata.*

[80] Grefenstette, Greg and Simone Teufel. 1995. Corpus-based Method for Automatic Identification of Support Verbs for Nominalizations. *Proceedings of EACL95.*

[81] Halliday, Michael. 1973. *Explorations in the Functions of Language.* Edward Arnold. London, UK.

[82] Halliday, Michael. 1978. *Language as a Social Semiotic.* Edward Arnold. London, UK.

[83] Halliday, Michael. 1985a. *An introduction to functional grammar.* Edward Arnold. London, UK.

[84] Halliday, Michael. 1985b. *Spoken and Written Language.* Oxford University Press. Oxford, UK.

[85] Harris, Zellig. 1957. "Co-occurrence and transformation in linguistic structure". *Language*, 33, 293–340.

[86] Havens, William. 1997. NoGood Caching for MultiAgent Backtrack Search. *Proceedings of the 1997 Conference of the American Association for Artifical Intelligence*

[87] Hawkins, John. 1994. *A performance theory of order and constituency.* Cambridge University Press. Cambridge, UK.

[88] Hayes, Phil, Steve Maxwell and Linda Schmandt. 1996. Controlled English Advantages for Translated and Original English Documents. *Proceedings of the First International Workshop on Controlled Language Applications*, 84–92.

[89] Heidorn, George, Karen Jensen, Lance Miller, R. Bird and Martin Chodorow. 1982. The EPISTLE Text-Critiquing System. *IBM Systems Journal*, 21(3), 305–326.

[90] Heritage, John. 1984. *Garfinkel and Ethnomethodology.* Polity Press.

[91] Holland, John. 1975. *Adaptation in Natural and Artificial Systems.* University of Michigan Press.

[92] Holmback, Heather, Serena Shubert and Jan Spyridakis. 1996. Issues in Conducting Empirical Evaluations of Controlled Language. *Proceedings of the First International Workshop on Controlled Language Applications*, 166–177.

[93] Hopcroft, John and Jeffrey Ullman. 1979. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley. Reading, MA.

[94] Hovy, E.H. 1988. *Generating Natural Language under Pragmatic Constraints.* Lawrence Erlbaum Associates. Hillsdale, NJ.

[95] Huckin, Thomas, Elizabeth Curtin and Debra Graham. 1986. Prescriptive Linguistics and Plain English: The Case of 'Whiz-deletions'. *Visible Language*, 20(2), 174–187.

[96] Ignizio, James and Tom Cavalier. 1994. *Linear Programming.* Prentice-Hall. Englewood Cliffs, NJ.

[97] Inui, Kentaro, Takenobu Tokunaga and Hozumi Tanaka. 1992. Text Revision: A Model and its Implementation. In Robert Dale, Eduard Hovy, Dietmar Rösner and Oliviero Stock (eds.) *Aspects of Automated Natural Language Generation: Proceedings of the Sixth International Workshop on NLG.* Springer-Verlag. New York, NY.

[98] Ionesco, Eugène. 1954. *Amédée ou Comment s'en Débarrasser.* In "Théâtre d'Eugène Ionesco Volume I". Gallimard. Paris, France.

[99] Janssen, Gerd, Gerhard Mark and Bernd Dobbert. 1996. Simplified German—A practical approach to documentation and translation. *Proceedings of the First International Workshop on Controlled Language Applications*, 150–158.

[100] Jelinek, Fred, J. Lafferty, D. Magerman, R. Mercer, A. ratnaparkhi and S. Roukos. 1994. Decision tree parsing using a hidden derivation model. *ARPA Workshop on Human Language Technology*, 260–265.

[101] Johnson, D., C. Aragon and L. McGeoch. 1989. Optimization by Simulated Annealing: An Experimental Evaluation: Graph Partitioning. *Operations Research*, 37, 865–892.

[102] Jordan, Michael. 1993. Towards an Understanding of Mature Writing: Analyzing and Paraphrasing Complex Noun Phrases. *Technostyle*, 11(2), 39–72.

[103] Jordan, Michael. 1994. Toward Plain Language: A Guide to Paraphrasing Complex Noun Phrases. *The Journal of Technical Writing and Communication*, 24(1), 77–96.

[104] Joshi, Aravind, Leon Levy and Masako Takahashi. 1975. Tree Adjunct Grammars *Journal of Computer and System Sciences*, 10, 136–163.

[105] Joshi, Aravind. 1985. How much context-sensitivity is necessary for characterizing structural descriptions—Tree Adjoining Grammars. In David Dowty, Lauri Karttunen and Arnold Zwicky (eds.), *Natural Language Processing—Theoretical, Computational and Psychological Perspectives.* Cambrdge University Press. New York, NY.

[106] Joshi, Aravind. 1987. An Introduction to Tree Adjoining Grammars. In Alexis Manaster-Ramer (ed.), *Mathematics of Language.* John Benjamins. Amsterdam, The Netherlands.

[107] Joshi, Aravind. 1990. Phrase Structure and Intonational Phrases: Comments on the papers by Marcus and Steedman. In G. Altmann (ed.), *Computational and Cognitive Models of Speech.* MIT Press. Cambridge, MA.

[108] Joshi, Aravind and Yves Schabes. 1991. Fixed and flexible phrase structure: Coordination in Tree Adjoining Grammar. *Proceedings of the DARPA Workshop on Spoken Language Systems.*

[109] Joshi, Aravind, K. Vijay-Shanker and David Weir. 1991. The convergence of mildly context-sensitive grammatical formalisms. In Peter Sells, Stuart Shieber and Tom Warsow (eds.), *Foundational Issues in Natural Language Processing.* MIT Press. Cambridge, MA.

[110] Joshi, Aravind and Yves Schabes. 1996. Tree-Adjoining Grammars. In Grzegorz Rozenberg and Arto Salomaa (eds.), *Handbook of Formal Languages, Vol 3*, 69–123. Springer-Verlag. New York, NY.

[111] Joslin, David and Amitabha Roy. 1997. Exploiting Symmetry in Lifted CSPs. *Proceedings of the 1997 Conference of the American Association for Artificial Intelligence*, 197–202.

[112] Kahane, Sylvain, Alexis Nasr and Owen Rambow. 1998. Pseudo-Projectivity: A Polynomially Parsable Non-Projective Dependency Grammar. *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics*, 646–652.

[113] Kane, Thomas S. 1983. *The Oxford Guide to Writing.* Oxford University Press, New York, NY.

[114] Kaplan, Ron and Joan Bresnan. 1981. Lexical-functional grammar: A formal system for grammatical representation. In Joan Bresnan (ed.), *The Mental Representation of Grammatical Relations.* MIT Press. Cambridge, MA.

[115] Karmarkar, N. 1984. A New Polynomial-Time Algorithm for Linear Programming. *Combinatorica*, 4, 373–395.

[116] Kay, Martin. 1979. Functional Grammar. *Proceedings of the 5th Annual Meeting of the Berkeley Linguistic Society.*

[117] Kern, R. 1979. *Usefulness of readability formulas for achieving Army readability objectives: research and state-of-the-art applied to the Army's problem.* Technical Advisory Service, US Army Research Institute. Fort Benjamin Harrison, IN.

[118] Khachian, L. G. 1979. A Polynomial Algorithm in Linear Programming. *Soviet Mathematics Doklady*, 20, 191–194.

[119] Kimball, J. 1973. Seven principles of surface structure parsing in natural language. *Cognition*, 2, 15–47.

[120] Kincaid, J. Peter, R. Fishburne, P. Rogers and B. Chissom. 1975. *Derivation of new readability formulas (Automated Readability Index, Fog Count, and Flesch Reading Ease Formula) for Navy enlisted personnel.* Navy Research Branch Report 8-75. Millington, TN.

[121] Kincaid, J. Peter, James Aagard, John O'Hara and Larry Cottrell. 1981. Computer Readability Editing System. *IEEE Transactions on Professional Communication*, 24(1), 38–41.

[122] Kieras, David. 1990. *The Computerized Comprehensibility System Maintainer's Guide.* University of Michigan Technical Report no. 33.

[123] Kirkpatrick, S., C. Gelatt, and M. Vecchi. 1983. Optimization by Simulated Annealing. *Science*, 200, 671–680.

[124] Klare, George. 1974–75. Assessing Readability. *Reading Research Quarterly*, Number 1, 1974–1975, 62–102.

[125] Klare, George. 1979. *Readability Standards for Army-wide Publications.* Evaluation Report 79-1, US Army Administrative Centre. Fort Benjamin Harrison, IN.

[126] Klee, V. and G. Minty. 1972. How Good is the Simplex Algorithm? In O. Shisha (ed.), *Inequalities III*, 159–175. Academic Press. New York, NY.

[127] Klein, Sheldon. 1965a. Automatic paraphrasing in essay format. *Mechanical translation*, 8(4): 68–83.

[128] Klein, Sheldon. 1965b. Control of style with a generative grammar. *Language*, 41(4), 619–631.

[129] Knott, Alistair. 1996. *A Data-Driven Methodology for Motivating a Set of Coherence Relations.* PhD thesis, University of Edinburgh.

[130] Knoblock, C., J. Tennenberg and Q. Yang. 1991. Characterizing Abstraction Hierarchies for Planning. *Proceedings of the 1991 National Conference on Artificial Intelligence*

[131] Koza, John. 1992. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press. Cambridge, MA.

[132] Kroch, Anthony. 1987. Unbounded dependencies and subjacency in a tree adjoining grammar. In Alexis Manaster-Ramer (ed.), *Mathematics of Language.* John Benjamins. Amsterdam, The Netherlands.

[133] Kroch, Anthony. 1989. Asymmetries in long distance extraction in a tree adjoining grammar. In Mark Baltin and Anthony Kroch (eds.), *Alternative Conceptions of Phrase Structure.* University of Chicago Press. Chicago, IL.

[134] Kulick, Seth. 1998. Clitic climbing in Romance: "Restructuring", causatives, and object-control verbs. *Fourth International Workshop on Tree Adjoining Grammars and Related Frameworks*, 88–91.

[135] Lees, Robert. 1960. *The Grammar of English Nominalizations.* Indiana University Research Centre in Anthropology, Folklore and Linguistics.

[136] Leibniz, G. 1952. *Discourse on Metaphysics.* Lucas, P. and L. Grint (trans.) University Press. Manchester, UK.

[137] Lenke, Nils. 1994. Anticipating the Reader's Problems and the Automatic Generation of Paraphrases. *Proceedings of COLING 1994*, 319–323.

[138] Lorge, Irving. 1939. Predicting reading difficulty of selections for children. *The Elementary English Review*, 16, 229–233.

[139] Lorge, Irving. 1948. The Lorge and Flesch readability formulae: A correction. *School and Society*, 67, 141–142.

[140] Lundy, M. and A. Mees. 1986. Convergence of an Annealing Algorithm. *Mathematical Programming*, 34, 111-124.

[141] Lyons, John. 1977. *Semantics, Vol. 1.* Cambridge University Press. Cambridge, UK.

[142] Maclane, Saunders. 1971. *Categories for the working mathematician.* Springer-Verlag. New York, NY.

[143] Malnikjær, Kirsten. 1991. *The Linguistics Encyclopedia.* Routledge. New York, NY.

[144] Makkai, Adam. 1977. "The passing of the syntactic age: A first look at the ecology of the English verb *take*". In *Linguistics at the crossroads* (ed. Adam Makkai *et al*) 79–104. Liviana, Padova.

[145] McArthur, Tom. 1991. The pedigree of plain English. *English Today: The International Review of the English Language*, 27(3), 13–19.

[146] McCall, William and Lelah Crabbs. 1961. *Standard test lessons in reading, 3rd edition.* Bureau of Publications, Teachers College, Columbia University. New York, NY.

[147] McKeown, Kathleen. 1985. *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text.* Cambridge University Press. Cambridge, UK.

[148] Mel'čuk, Igor and Nikolaj Pertsov. 1987. *Surface syntax of English: a formal model within the meaning-text framework.* John Benjamins. Amsterdam, The Netherlands.

[149] Mel'čuk, Igor. 1988. *Dependency Syntax: Theory and Practice.* State University of NY Press. Albany, NY.

[150] Meteer, Marie, David McDonald, Scot Anderson, David Forster, Linda Gay, Alison Huettner and Penelope Sibun. 1987. *Mumble-86: Design and Implementation.* Technical Report COINS 87-87a. University of Massachusetts, MA.

[151] Meteer, Marie W. 1991. The Implications of Revisions for Natural Language Generation. In Cécile Paris, William Swartout and William Mann (eds.), *Natural Language Generation in Artificial Intelligence and Computational Linguistics*, 155–177. Kluwer Academic Publishers. Norwell, MA.

[152] Mönnich, Uwe. 1998. TAGs M-Constructed. *Fourth International Workshop on Tree Adjoining Grammars and Related Frameworks*, 108–111.

[153] Nyberg, Eric and Teruko Mitamura. 1996. Controlled Language and Knowledge-Based Machine Translation: Principles and Practice. *Proceedings of the First International Workshop on Controlled Language Applications*, 74–83.

[154] Orwell, George. 1946. Politics and the English Language. In *George Orwell: Collected Essays*, 353–367. Secker and Warburg. London, UK.

[155] Palmer, Martha, Owen Rambow and Alexis Nasr. 1998. Rapid Prototyping of Domain-Specific Machine Translation Systems. *AMTA-98*, Langhorne, PA.

[156] Partee, Barbara, Alice ter Meulen and Robert Wall. 1993. *Mathematical Methods in Linguistics.* Kluwer Academic Publishers. Dordrecht, The Netherlands.

[157] Payette, Julie and Graeme Hirst. 1992. An Intelligent Computer-Assistant for Stylistic Instruction. *Computers and the Humanities*, 26, 87–102.

[158] Pêcheux, Michel. 1982. *Language, Semiotics and Ideology.* Macmillan. London, UK.

[159] Perfetti, Charles. 1969. Lexical Density and Phrase Structure Depth as Variables in Sentence Retention. *Journal of Verbal Learning and Verbal Behavior*, 8, 719–724.

[160] Peters, P. and R. Ritchie. 1973. On the generative power of transformational grammars. *Information Sciences*, 6, 49–83.

[161] Plung, Daniel. 1981. Readability Formulas and Technical Communication. *IEEE Transactions on Professional Communication*, 24(1), 52–54.

[162] Pollard, Carl and Ivan Sag. 1987. *Information-Based Syntax and Semantics. Vol I: Fundamentals.* CSLI, Stanford University.

[163] Pollard, Carl and Ivan Sag. 1994. *Head-Driven Phrase Structure Grammar.* University of Chicago Press. Chicago, IL.

[164] Powell, Kenneth. 1981. Readability Guides are Helpful If . . . . *IEEE Transactions on Professional Communication*, 24(1), 43–45.

[165] Pullum, Geoffrey and Gerald Gazdar. 1982. Natural languages and context-free languages. *Linguistics and Philosophy*, 4, 471–504.

[166] Quirk, Randolph, Sidney Greenbaum, Geoffret Leech and Jan Svartnik. 1985. *A Comprehensive Grammar of English.* Longman Group Ltd. London, UK.

[167] Rajasekaran, Sanguthevar and Shibu Yooseph. 1995. TAL recognition in $\mathcal{O}(M(n^2))$ time. *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, 166–173.

[168] Rambow, Owen. 1994. Formal and Computational Aspects of Natural Language Syntax. PhD thesis. University of Pennsylvania.

[169] Rambow, Owen and Giorgio Satta. 1994. *A Two-Dimensional Hierarchy for Parallel Rewriting Systems.* University of Pennsylvania Technical Report IRCS94-02.

[170] Rambow, Owen and Aravind Joshi. 1995. A Formal Look at Dependency Grammars and Phrase-Structure Grammars, with Special Consideration of Word-Order Phenomena. In Leo Wanner (ed.), *Current Issues in Meaning-Text Theory.* Pinter. London, UK.

[171] Rambow, Owen, K. Vijay-Shanker and David Weir. 1995. D-Tree Grammars. *Proceedings of the 33rd Meeting of the Association for Computational Linguistics*, 151–158.

[172] Rambow, Owen and Giorgio Satta. 1996. Synchronous Models of Language. *Proceedings of the 34th Meeting of the Association for Computational Linguistics*, 116–123.

[173] Redish, Janice. 1981. Understanding the Limitations of Readability Formulas. *IEEE Transactions on Professional Communication*, 24(1), 46–48.

[174] Richardson, Steve and Lisa Braden-Harder. 1988. The Experience of Developing a Large-Scale Natural Language Text Processing System: CRITIQUE. *Proceedings of the 2nd Conference on Applied NLP*, 195–202.

[175] Robin, Jacques. 1994. *Revision-Based Generation of Natural Language Summaries Providing Historical Background.* Columbia University Technical Report CUCS-034-94.

[176] Rogers, James. 1994. Capturing CFLs with Tree Adjoining Grammars. *Proceedings of the 32nd Meeting of the Association for Computational Linguistics*, 155–162.

[177] Rogers, James. 1998. On defining TALs with logical constraints. *Fourth International Workshop on Tree Adjoining Grammars and Related Frameworks*, 151–154.

[178] Salomaa, Arto. 1973. *Formal Languages.* Academic Press. Orlando, FL.

[179] Sampson, Geoffrey and Robin Haigh. 1988. Why are long sentences longer than short ones? In Kytö, Merja, Ossi Ihalainen and Matti Rissanen (eds.) *Corpus Linguistics, Hard and Soft: Proceedings of the 8th International Conference on English Language Research on Computerized Corpora.* Amsterdam, The Netherlands.

[180] Sarkar, Anoop and Aravind Joshi. 1996. Coordination in TAG: Formalization and Implementation. *Proceedings of the 16th International Conference on Computational Linguistics*, 610–615.

[181] Sarkar, Anoop. 1997. Separating Dependency from Constituency in a Tree Rewriting System. *Proceedings of the 5th Meeting on Mathematics of Language.*

[182] Satta, Giorgio. 1994. Tree adjoining grammar parsing and boolean matrix multiplication. *Computational Linguistics*, 20(1), 91–124.

[183] Schabes, Yves, Anne Abeillé and Aravind Joshi. 1988. Parsing Strategies with 'Lexicalized' Grammars: Application to Tree Adjoining Grammars. *Proceedings of the 12th International Conference on Computational Linguistics*, 578–583.

[184] Schabes, Yves. 1990. *Mathematical and Computational Aspects of Lexicalized Grammars.* PhD thesis, University of Pennsylvania.

[185] Schabes, Yves and Aravind Joshi. 1991. Parsing with lexicalized tree adjoining grammar. In M. Tomita (ed.), *Current Issues in Parsing Technologies.* Kluwer Academic Publishers. Norwell, MA.

[186] Schabes, Yves and Stuart Shieber. 1992. An Alternative Conception of Tree-Adjoining Derivation. *Proceedings of the 30th Annual Meeting of the Association for Computational Linguistics*, 167–176.

[187] Schachtl, Stefanie. 1996. Requirements for Controlled German in Industrial Applications. *Proceedings of the First International Workshop on Controlled Language Applications*, 143–149.

[188] Scott, Donia and Clarisse Sieckenius de Souza. 1990. Getting the Message Across in RST-based Text Generation. In Robert Dale, Chris Mellish and Michael Zock (eds.) *Current Research in NLG.* Academic Press. London, UK.

[189] Shieber, Stuart. 1985. Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8, 333–343.

[190] Shieber, Stuart and Yves Schabes. 1990. Synchronous Tree-Adjoining Grammars. *Proceedings of the 13th International Conference on Computational Linguistics*, 253–258.

[191] Shieber, Stuart. 1994. Restricting the Weak-Generative Capacity of Synchronous Tree-Adjoining Grammars. *Computational Intelligence*, 10(4), 371–386.

[192] Srinivas, B. 1997. *Complexity of Lexical Descriptions and its Relevance to Partial Parsing.* PhD thesis, University of Pennsylvania.

[193] Stabler, E. 1992. *The Logical Approach to Syntax.* MIT Press. Cambridge, MA.

[194] Steedman, Mark. 1985. Dependency and coordination in the grammar of Dutch and English. *Language*, 61, 523–568.

[195] Steedman, Mark. 1991. Structure and Intonation. *Language*, 67, 260–296.

[196] Strong, W. 1973. *Sentence Combining: A Composing Book.* Random House. New York, NY.

[197] Strunk, William and E. B. White. 1979. *The Elements of Style, 3rd edition.* MacMillan Publishing Co. New York, NY.

[198] Tarski, A. 1935. Der Wahrheitsbegriff in den formalisierten Sprachen. *Studia Philosophica*, 1, 261–405. English translation in Tarski (1956).

[199] Tarski, A. 1956. *Logic, Semantics and Metamathematics.* Oxford University Press. Oxford, UK.

[200] Thatcher, J. 1967. Characterizing derivation trees of context-free grammars through a generalization of finite automata theory. *Journal of Computer and System Sciences*, 1, 317–322.

[201] Vallduví, Enric. 1993. *Information Packaging: A Survey.* University of Edinburgh Technical Report HCRC/RP-44.

[202] Vallduví, Enric and Elisabet Engdahl. 1996. The linguistic realization of information packaging. *Linguistics*, 34, 459–519.

[203] van der Eijk, Pim, Michiel de Koning and Gert van der Steen. 1996. Controlled Language Correction and Translation. *Proceedings of the First International Workshop on Controlled Language Applications*, 64–73.

[204] van Laarhoven, P. and E. Aarts. 1987. *Simulated Annealing: Theory and Applications.* D. Reidel. Dordrecht, The Netherlands.

[205] Vervalin, Charles. 1980. Checked Your Fog Index Lately?. *IEEE Transactions on Professional Communication*, 23(2), 87–88.

[206] Vijay-Shanker, K. 1987. *A Study of Tree Adjoining Grammars.* PhD thesis, University of Pennsylvania.

[207] Vijay-Shanker, K. and Yves Schabes. 1992. Structure Sharing in Lexicalised Tree Adjoining Grammar. *Proceedings of the 15th International Conference on Computational Linguistics*, 205–211.

[208] Vinay, J-P and J. Darbelnet. 1958. *Stylistique comparée du français et de l'anglais.* Didier.

[209] Walters, Robert. 1991. *Categories and Computer Science.* Carslaw Publications. Sydney, Australia.

[210] Weir, David. 1988. *Characterizing Mildly Context-Sensitive Grammar Formalisms.* PhD thesis, University of Pennsylvania.

[211] Wierzbicka, Anna. 1982. "Why Can You *Have a Drink* When You Can't **Have an Eat*?" *Language*, 58(4), 753-799.

[212] Williams, H. P. 1995. *Model Building in Mathematical Programming.* John Wiley. Chichester, UK.

[213] Williams, Joseph M. 1990. *Style: Ten Lessons in Clarity and Grace.* The University of Chicago Press Ltd. Chicago, IL.

[214] Winston, Wayne. 1991. *Operations Research Applications and Algorithms.* PWS-Kent. Boston, MA.

[215] Wojcik, Richard and Heather Holmback. 1996. Getting a Controlled Language Off the Ground at Boeing. *Proceedings of the First International Workshop on Controlled Language Applications*, 22–31.

[216] Wright, Patricia. 1985. Editing: Policies and Processes. In T. Duffy and R. Waller (eds.) *Designing Usable Texts.* Academic Press. Orlando, FL.

[217] Xerox Corporation. 1988. *Xerox Publishing Standards: A Manual of Style and Design.* Watson-Guptill Publications. New York, NY.

[218] Xia, Fei, Martha Palmer, K. Vijay-Shanker and Joseph Rosenzweig. 1998. Consistent Grammar Development Using Partial-Tree Descriptions for Lexicalised Tree-Adjoining Grammars. *Fourth International Workshop on Tree Adjoining Grammars and Related Frameworks*, 180–183.

[219] XTAG. 1995. *A Lexicalized Tree Adjoining Grammar for English.* Technical Report IRCS95-03, University of Pennsylvania.

[220] Yngve, V. 1960. A model and an hypothesis for language structure. *Proceedings of the American Philosophical Society*, 104, 444–466.