Head-Driven Machine Translation

Deirdre Carr

M.Sc. September 1996

Head-Driven Machine Translation

A Dissertation Presented in Fulfillment of
the Requirement of the M Sc Degree

Deirdre Carr, BA Mod

School of Computer Applications
Dublin City University

Academic Supervisor Mr Andrew Way

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Masters of Science in Computer Applications, is entirely my own work and has not been taken from the work of others save to the extent that such work has been cited and acknowledged within the text of my work.

Signed: Deirdre Carr

Deirdre Carr, BA Mod

Date: 27/9/1996

Acknowledgements

Many thanks to

Andy Way for his support and guidance over the past two years Also to Josef van Genebith for providing much-needed direction and help at a critical time,

Patricia, Sophia and Miriam who have always been so generous in offering sympathy, understanding, and above all, friendship Also to Elaine and my other friends for never failing to boost my morale when things seemed almost impossible,

Gavin, Michelle, Donal, Darragh, Gary, and all the rest of the postgrads for providing coffee, company, computers and encouragement, not to mention plenty of Budweiser!

Finally, to my parents, Mary and Michael, for helping me to make my own decisions in life and for never doubting me

Abstract of "Head-Driven Machine Translation"

Deirdre Carr, BA Mod

I.D. Number: 94970700

Despite initial optimism about the feasibility of Machine Translation, it is now accepted as being an extremely different task to implement This is due in part to our lack of understanding of the human processes involved in language comprehension and production in general, and translation in particular In addition, the myriad of problems posed by ambiguities caused by structural differences, category options etc , which in most cases are resolved subconsciously by humans, have slowed down the development of a Fully Automatic, High-Quality Machine Translation System, and have convinced many people that this goal is completely unattainable

This thesis is an investigation of the suitability of Head-Driven Phrase Structure Grammar (HPSG, Pollard and Sag, 1987, 1994) for use in a transfer-based translation environment It provides an account of some of the problems tackled by such a system, as well as the reasons behind the decisions to chose HPSG and a transfer approach Moreover, some of the possible inadequacies of HPSG's current semantic framework are addressed and some potential alternatives are suggested, namely the incorporation of case grammars and semantic features to guide lexical selection in the target language The evaluation of these ideas is based on an implementation of these proposals in a system for translation between German and English, using the Attribute Logic Engine (ALE, Carpenter, 1992) for the purposes of monolingual analysis

Table of Contents

Chapter One
Introduction

1.1 Introduction

The successful implementation of a fully automatic high quality machine translation system has long been the ultimate goal of researchers in this area Combining knowledge from a broad spectrum of topic areas, including Artificial Intelligence, computational linguistics, cognitive science and psychology, the attempts to design and engineer such a system have been many and varied, ranging from first generation, direct systems (see Chapter 2) as epitomised by SYSTRAN, to ongoing work in Germany and the United States on VERBMOBIL, a system intended for translation of face-to-face dialog, and hence includes the areas of speech recognition and synthesis

Translation is accepted as being one of the most difficult areas of human cognitive activity to simulate, given the presumed complexity of the processing involved Despite this, however, research into the possibility continues There are three primary methods of machine translation, direct, transfer and interlingua (Chapter 2) While each of these exhibit their own merits and disadvantages, the transfer approach has been chosen as the main area of interest, a decision rationalised in chapter 2

One of the key ideas of machine translation has always been to use a fully expanded and highly formalised linguistic theory for the purposes of analysis and generation, in conjunction with one of the above mentioned translation strategies It is this idea which forms the basis of this dissertation, the linguistic theory in question being Head-Driven Phrase Structure Grammar (HPSG), a formalism which claims to take some psycholinguistic theories of human language processing into consideration

Furthermore, of paramount importance to this theory is the interaction of syntactic, semantic and contextual information, which act as constraints on language While earlier developments in machine translation focused almost entirely on syntax as a method of analysis, since the release of the ALPAC report in 1964, semantics and pragmatics have played an ever more significant role

Another reason for choosing HPSG is its dedication to developing a universal grammar, which would seem to be a natural progression for machine translation, as more than one language-pair could be potentially added

Obviously, to tackle any natural language in its entirety would be an enormous task, so a restricted language approach has been adopted for the purposes of clarification and implementation Bearing this in mind, the semantic component of HPSG was expanded in order to cope with ambiguities resulting from the use of words with more than one meaning For example, the verb *run* has several meanings determined by the context in which it appears

(1) John runs the shop

In this sentence, it can be translated as '*to manage*', or, in German, *fuhren* given that a shop can be described as a business and a business needs a manager, a role filled in this instance by *John* However, in the sentence

(2) John runs the race

the meaning of *run* can be understood as '*to run [a race]*' or *rennen* A form of case grammars, as presented by Charles Fillmore [Fill68], will be proposed as a logical extension of the existing meaning representation theory used in HPSG, namely situation semantics

Thus, the main content of this investigation is the suitability of Head-Driven Phrase Structure Grammar for machine translation HPSG produces a large structural representation of a sentence but, in accordance with a recommendation by Arnold et al [Arn94], it is the semantic category of the analysis representation which will be used to drive the transfer component of the system. The ultimate aim is to implement a bi-directional MT system which uses the theory of HPSG, as presented in [PoSa94], with the additional case grammar features determining target language lexical selection This grammar can then be used in the translation of a small set of English sentences into German and vice

versa, concentrating on sentences which contain ambiguous verbs similar to those in (1) and (2).

1.2 Overview

The structure of the thesis is as follows:

Chapter Two - the need for MT is ever-increasing, particularly with the breaking down of international borders in Europe and across the world. This chapter examines the reasons why MT is so necessary, and takes a look at its history, charting some of its most significant highs and lows. It also introduces the main methods of translation, in particular transfer-based approaches, as well as the linguistic problems it encounters, including category and semantic ambiguities, morphological .analysis issues, quantifier scope, prepositional phrase attachment, most of which are tackled in the system of chapter five.

Chapter Three - HPSG was developed as a psycholinguistically realistic grammar, combining some of the most successful elements of other contemporary theories in the design of a new formalism.

Chapter Four - HPSG is an extremely well-formalised constraint-based theory, whose constraints arise from the interaction of highly articulated lexical entries, universal principles and phrase structure rules (ID schemata). All of these are expressed in terms of Attribute-Value Matrices. Chapter four presents an explanation of the main constraints and features of HPSG, as well as proposing an original, augmented semantic formalism based on the use of case grammars and semantic features.

Chapter Five - This chapter presents an implementation of a machine translation system which uses the augmented HPSG grammars of German and English to analyse sentences in the source language and drive the transfer to produce the structure for generation of the equivalent sentence in the target language.

Chapter Six - The final chapter discusses the conclusions derived from the implementation in chapter five, as well as suggestions for further work and possible improvements

Chapter 2

Machine Translation

2.1 Introduction

'Unless we can develop more efficient means of communicating - sharing ideas from person to person and place to place - human progress will be inhibited

Harold Borko in 'Automated Language Processing, 1967

Quoted in [Gosh87]

"The automation of translation is one of the oldest dreams of humanity'

[MTIT94]

As early as the beginning of the 17th century, the use of mechanical devices as a method of overcoming language barriers was suggested The idea has developed considerably since then, with systems such as TAUM-MÉTÉO, LOGOS and SYSTRAN currently in everyday use Defined as the application of computers to the translation of texts and/or speech from one natural language to another ([Hutch86]), fully automatic, high-quality machine translation (MT) is a goal striven for by many In this chapter, the issues of concern for the system presented in chapter 5 will be discussed, focusing on the ways in which the chosen translation strategy (the transfer-based approach), and the formalism for the linguistic theory, (Head-Driven Phrase Structure Grammar), can aid in the resolution of these problems The system is intended for translation between German and English, and concentrates to a large extent on lexical selection problems arising from semantic ambiguity However, morphological and category ambiguity concerns will also be addressed, as well as some structurally ambiguous constructions Furthermore, the reasons for selecting a transfer method of translation will be presented by comparing this method with alternative proposals First, however, an elaboration of the reasons why MT is so necessary will be presented, as well as a very brief history of the science

2 2 The Need for Machine Translation

There is a simple reason for the popularity of MT, namely that the actual demands and requirements for translation far exceed the capacity of human translators There are any number of reasons why the demand is so high, chief amongst them being the commercial cost of human translation Political and social concerns are another important consideration in its development, as well as more general

scientific, philosophical and idealistic issues While each of these are no doubt valid reasons for continued research, the motivation behind an individual's pursuit of their goals is perhaps not as important as the end result a system capable of translating from one language to another with a minimal amount of human interaction

2 2 1 Economic Issues

In 1991, Arnold et al estimated that the European Union (then the European Community) was spending up to one million pounds per annum on translation This comprises at least 40-45% of the language costs outlayed by the EU every year ([Arn94]) More recently, the Corel Corporation conducted a survey into the usage of translation tools, the results of which indicated that those companies which did not already use some kind of MT system would be willing to pay up to £10,000 in anticipation of the reduction of overall costs in this area [Near96] Given that professional human translators are highly qualified but expensive and produce on average between four and six pages of top quality translation daily, claims by marketers of systems such as SYSTRAN that costs can be reduced by at least 40% ([SYS96]) are obviously very appealing for any company The US-based company Xerox, in fact, estimated their annual through put from the same system at sixty thousand pages every year, translating from English into five other languages ([MTIT94]) While caution should always be taken when taking on board such claims due to the damage done by exaggeration of MT's capabilities in the past, it is nonetheless true to say that for many companies, MT represents a significant reduction in translation costs

2 2 2 Socio-Political Motivation

Another important motivation is the situation in countries where more than one language is spoken (Belgium, Canada, Ireland) One option would clearly be to filter out one language in favour of another This is naturally unappealing for any speaker of the language to be eliminated, not least because of the difficulty of acquiring a new tongue However, even more important than this is the fact that the culture of a country often goes hand in hand with the language Music, art and other traditions are often inextricably bound with the tongue of a particular

civilisation Moreover, as highlighted by Arnold et al ([Arn94]), it is a human right to be allowed to communicate in one's own language A preferable solution to adopting a *lingua franca* would be machine translation, facilitating each person to keep their own language but still interact with others who speak a foreign tongue

2 2 3 General Considerations

Scientifically and philosophically, the idea of machine translation is captivating quite simply because of the difficulty involved in simulating a wide range of human cerebral activity Computer scientists, Artificial Intelligence researchers, computational and theoretical linguists, psychologists and philosophers, together and individually, have long sought for the ultimate representation of the processes involved in translation, some of which will be discussed in following chapters For now, however, it need only be mentioned as a motivation for a large number of researchers in the areas listed above

In the following section, with some understanding of the need for machine translation, and the reasons for ongoing research in the field, the highs and lows of its history will be charted, including the initial motivation, varying aspects of research over the years, and the main people involved

2.3 The History of Machine Translation

'I have wondered if it were unthinkable to design a computer which would translate when I look at an article in Russian, I say 'This is really written in English but it has been coded in some strange symbols I will now proceed to decode'

Warren Weaver, 1947, in a letter to Norbert Wiener at MIT

Machine Translation has a long and intricate history, beginning in the 17th century and continuing to the present day While the birth of machine translation is generally accepted as 1946, following a meeting between Warren Weaver of the Rockefeller Foundation and the British crystallographer, Andrew D Booth, the idea of using mechanical instruments to enable communication between all people, no matter what their mother tongue, can be traced back to the early seventeenth

century to proposals from Leibniz and Descartes In 1933, a patent for a "*Mechanical Brain*" was applied for by Georges Artsrouni, a device intended for use in railway timetables, bank accounts, commercial records, and particularly, mechanical dictionaries At approximately the same time, Petr Petrovich Smirnov-Troyanski had developed a machine which selected words and then printed them, while simultaneously translating them into another language

Nonetheless, it is Weaver and Booth who are considered the fathers if MT In mid-1949, Warren Weaver, having studied the work done by Booth and his colleagues in the time following their initial meeting in 1946, circulated a memorandum to two hundred academics, thus launching what was to become an immense research area in the following decade

Throughout the 1950's, the area of machine translation was a flurry of activity In the time between 1954 and 1960, two different schools of thought emerged, the empiricists and the theorists The lack of adequately formalised theories meant that empirical research resulted in word-for-word translations, with little or no allowances made for syntax or semantics Workers at MIT and other theorists concentrated on developing a thorough linguistic theory which could be used in MT

The science received the official seal of approval in the United States in 1960, when funding was awarded by The Committee on Science and Astronautics of the US House of Representatives However, less than four years later, in April 1964, the Automatic Language Processing Advisory Committee (ALPAC) was established to investigate the progress made in MT, and to produce a recommendation on future funding based on an evaluation of this progress Their conclusion was clear there was no system available which was capable of satisfactorily translating technical texts, nor was there any prospect of one appearing in the near future While condemned by many as biased and narrowsighted, the report had immense consequences for MT In 1963 there were ten major research groups in the United States, in 1968 there were just three

In the wake of the ALPAC report, focus on MT shifted from America to Eastern Europe and Canada The USSR saw the report as their chance to overtake the United States in this area of investigation and development The outcome was an increase in funding and the production of a large number of the world's operating MT systems For Canada, it was a solution to the problem of having more than one national language

During this time, a number of successful systems were developed, including TAUM-MÉTÉO, which has been in daily use in Canada since May 1977, translating approximately 45,000 words of weather bulletins every day ([Arn94]) Other systems designed and implemented with the aid of money from private sponsors and universities include LOGOS (Logos Corporation, US), Weidner ALPS (Utah), and SUSY (University of Saarbrucken) Work was also ongoing at Centre for Automatic Translation (CETA) in Grenoble, France, and in the Linguistic Research Center (LRC) in Texas The research in these latter two centres concentrated primarily on Chomskyan theory However, it was the installation of SYSTRAN by the Commission of the European Communities in 1975 which brought MT into a new era

While machine translation in some form is an everyday reality for many companies world-wide, research has continued and, in some cases, is still ongoing The EU-sponsored EUROTRA which began in 1982 and ended three years ago in 1993, sought to develop a transfer system capable of translation between each of the EU's official languages The Verbmobil Project is an attempt to engineer a translation device for face-to-face dialogues Initiated on the first of April 1993, Verbmobil is jointly sponsored by the German Ministry for Research and Technology (BMFT) and an industrial consortium comprising Alcatel, Daimler-Benz AG, IBM Deutschland, Philips GmbH and Siemens Aktiengesellschaft Over thirty groups in Germany and the US are currently working in the first phase of the development, which is due to end at the close of this year, 1996

However, the history of MT is really of interest when using knowledge of its successes and failures to develop a new system. There are a number of translation approaches which have been tried over the years, most notably, first generation (or direct) systems, and second generation (indirect) systems, the latter of which can be described as rule-based, linguistic knowledge approaches. For the purposes of this investigation, a derivative of this latter school of thought has been adopted, namely **a transfer-based approach.** The reasons for this are based on a comparison with direct and interlingual methods, as well as some of the more recent suggestions for MT design.

2.4 Translation Methods

Translation can be described as a mapping between the components of the source language text and their equivalent components in the target language. The focus of this section is the manner in which this mapping is implemented in the machine translation system in Chapter 5. However, before proceeding to the details of the strategy adopted, it is necessary first of all to highlight those aspects of the chosen target languages which will be considered in this implementation and the problems they might present for translation.

2.4.1 Issues of Concern

"Die stillschweigenden Abmachungen zum Verständnis der Umgangssprache sind enorm kompliziert - The silent adjustments made to understand colloquial language are enormously complicated"

Ludwig Wittgenstein, Tracatus Logico-Philosophicus, proposition 4.002

Quoted in [Gosh87]

To say that almost every sentence or phrase uttered is in some way ambiguous is no exaggeration. While this ambiguity is in most cases resolved subconsciously by humans, for a natural language processing system, it can give rise to immense problems. In context, many sentences can be easily disambiguated, but taken in isolation, their ambiguities can be more difficult to resolve. The fact that ambiguities multiply means that the potential number of interpretations for any sentence can be extremely large. Sentences and phrases can be ambiguous for any

number of reasons, including homography, polysemy, PP attachment etc In the following sections, those problematic areas tackled in the system are introduced and discusses, with a view to proposing possible solutions Obviously there are a large number of issues which will not be dealt with, as to even consider designing a system capable of resolving all those difficulties highlighted over the past few decades would be a huge undertaking Thus, the topics of category, transfer and structural ambiguities, morphology, homography and polysemy will all be reviewed

2 4 1 1 Category Ambiguities

A given word can be assigned more than one grammatical category Take, for example, the word *round* as shown in Figure 2 1 (examples adapted from [Hutch92]) Depending on the context in which it appears, round can be one of six different categories This is not of great concern when translating sentences which contain a single word exhibiting a category ambiguity, as it can be resolved using information gathered from morphological inflection, or, more commonly, by syntactic parsing

CATEGORY	SENTENCE
NOUN	Liverpool were eliminated in the first *round*
VERB	The cowboy started to *round* up the cattle
ADJECTIVE	I want to buy a *round* table
PREPOSITION	We are going on a cruise *round* the world
ADVERB	The tree measured six feet *round*
PARTICLE	If she faints, smelling salts will bring her *round*

Figure 2 1

(a) Parties never *end* on a low note

(b) The *end* of the party was as good as the beginning

In sentences (a) and (b), for example, the grammar rules should determine that in (b), the only possible option is for the word *end* to be interpreted as a noun There should be no rule allowing a construction *the* + V + PP (save where the verb has a gerundive value as in *the burning of the city* This is an issue which can be resolved by HPSG's use of verb features as introduced in chapter 4), so the rules NP + VP and *the* + N are enforced This is the case in the system in Chapter 5, where HPSG's Immediate Dominance Schemata (Section 4 5 2) impose strict constraints on the environment in which specific categories can occur The possibility of defining multiple lexical entries for any given word also aids in the resolution of category ambiguities, as, if necessary, five different entries can be specified for the word *round,* as the specification for each each will differ according to its category Hence, subcategorisation and head details can be used to decide on the correct interpretation

The main problem arises when several ambiguous words occur in the same sentence as in *Gas pump prices rose last time oil stocks fell* While this is easily understood by humans, there is at least a two-way ambiguity for each of the words in this sentence, resulting in an extremely large number of possible interpretations This obviously is a concern for a machine, and often syntactic parsing is not sufficient to resolve it 'Real-world' and contextual knowledge would seem to be the most likely solutions, but these factors have proven very difficult to incorporate in a linguistic theory As such, sentences such as the one above will, for now, not be addressed However, there are a number of constructions which can be built from the lexicon which contain more than one category ambiguity, for example, *the round rose* Again, it is HPSG's strict ID rules which analyse this correctly, in conjunction with the subcategorisation frames (Section 4 4 1 1)

2 4 1 2 Homography and Polysemy

Homography occurs when two or more words with the same spelling but different meanings are encountered, for example, *bank*, *light* and *club* As each meaning is of the same category, the ambiguity will not be spotted during syntactic analysis, so some additional measures are necessary to overcome the problem. One

solution is to completely omit unusual interpretations from the lexicon altogether as in a sublanguage approach The other is to use semantic features for disambiguation This approach is adopted in the machine translation system introduced in chapter 5, where the word *bank* has two lexical entries, the first of which is assigned the properties, *financial_institution* and *building* The second has the properties *river_side* These features can be easily built in to HPSG's attribute-value modelling domain (see Section 4 2)

A similar strategy can be applied to polysemous words, i e , words which exhibit a range of meanings which are in some way related to each other For example, the *mouth* of a river, a *branch* of a bank, *train* of thought, *flow* of ideas, etc

Of particular concern for this thesis are verbs whose sense changes according to the context in which they appear Hence, words such as *run* and *grow* (*Kim runs the race Vs Kim runs the shop, Kim grows tomatoes Vs Kim grew an inch*) which exhibit more than one meeting must be dealt with if the correct translation is to be chosen in the target language (provided, of course, that this distinction is lexicalised in the target language) This is accomplished by associating a case frame type ([Fill68]) with each of the verbs, the value of which is determined by the semantic properties of the syntactic categories with which it appears Thus, given that a race can be described as a competition, *runs* in the first sentence can be assigned a case frame value of competing Its lexical equivalent in German (rennen) has the same value, resulting in the selection of the correct translation This is discussed further in chapters 4 and 5

2 4 1.3 Transfer Ambiguity

Transfer ambiguities are an issue when a source language word is encountered which has two or more possible translations The English word *know,* for example, can be interpreted as *kennen* or *wissen* in German (*savoir* or *connaître* in French), depending on whether the 'knower' is in possession of knowledge about a fact or thing or about a person or place (*kennen/connaître* for the latter, *wissen/savoir* for the former) This phenomenon is solvable by applying syntactic

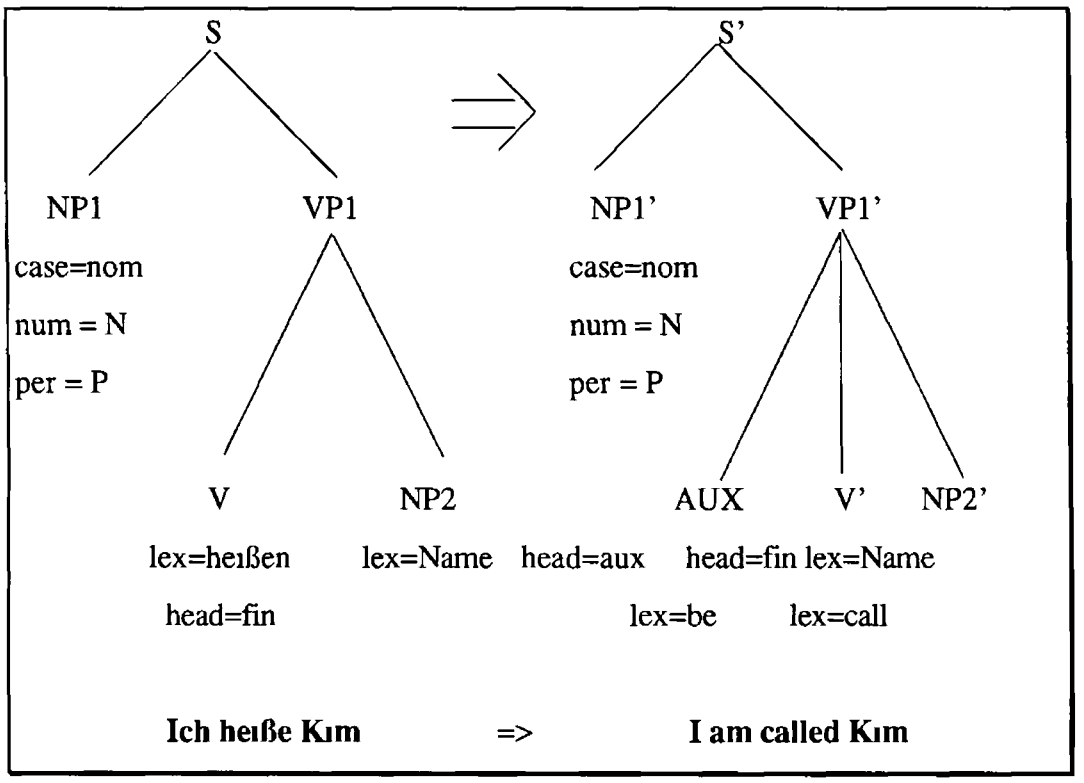


Figure 2 2

rules to determine what word should be chosen, as in the case of knowledge about a fact, an additional complementiser structure is present as in (c) and its translations, which is notably absent from (d) A number of these occur in the lexicon presented in chapter 5, and are easily dealt with by subcategorisation frames which specify the syntactic categories of the complements a word needs to form a phrase *Wissen*, for example, needs a clause, while *kennen* simply requires an accusative noun phrase It also means that translation between French and German in these cases is straightforward One example, however, where subcategorisation frames would not ensure the correct translation is in the translation of *I know the rules* to *je sais les régles* In this case, an additional content constraint could be placed on savoir's semantic component, by specifying that it can also take as its object an inanimate, common noun such as *régle* in the above example

(c) *I know that you were there* (VP -> V COMP S)

Je sais que tu étais là

Ich weiß, daß du da warst

(d) *I know Kim* (VP -> V NP)

Je connais Kim

Ich kenne Kim

There are some exceptions to this rule, however, especially when translation ambiguities occur with nouns The English word *corner* translates as *rincon* or *esquina* in Spanish depending on whether it is inside or outside The same condition can be applied to selecting the right German word for *wall*, which can be translated as *Mauer* (outside wall) or *Wand* (inside wall) The use of semantic features is the solution implemented for determining the correct option in the system presented in this thesis

Other problems arise when sentences such as *I am called Kim* are encountered, as its German translation is *ich heiße Kim* In the English utterance, there are two noun phrases, an auxiliary verb *am* and a main verb *called* In the German equivalent, however, there is only one main verb, *heißen* (see Figure 2 2) Thus, when translating, some restructuring is necessary, as well as a mechanism for selecting the correct interpretation This is done in chapter 5 by using a combination of case grammars and subcategorisation frames in accordance with the extended semantic formalism proposed for HPSG in chapter 4

2 4.1 4 Morphological Problems

There are three kinds of morphology - derivational, inflectional and compound It is only the first two of these, however, which are of particular concern for this dissertation German, for example, has a rich inflectional morphology which provides indications of subject-verb and adjective-noun agreement However, many of the language's suffixes perform multiple functions, and the MT system must make a decision as to which function it has when encountered in a sentence or phrase The ending *'-en'* can be used to indicate noun plurals, weak noun singular non-nominative, strong verb past participle, verb first or third person plural amongst other things Syntactic knowledge can aid in the decision-making process In chapter 5, a system of lexical rules is introduced whereby the information contained in a sign (such as its complements, subject, specifier,

semantic content etc) aids in the selection of the correct form. Currently, lexical rules are provided for present and past tense verbs, noun plurals, as well as adjective and determiner endings

Derivational morphology can be evidenced in the use of prefixes and suffixes of a particular kind In English, for example, the negative prefixes '*un-*' and '*non-*' can change the meaning of a adjective or a noun (*attractive -> unattractive, entity -> nonentity)* and the suffix '*-ly*' can change the category of a word from an adjective to an adverb (*careful -> carefully)* Similarly, the suffix '*-heit*' when appended to a German adjective changes the word's category to that of a noun (*schon -> Schonheit)* Again lexical rules are used to solve this problem in the implementation discussed in chapter 5, whereby the endings '*-ly*' and '*-heit*' can be correctly used To include the others, it would simply be a matter of coding a number of lexical rules Exceptions to these rules (such as *unkempt* or *only*) are dealt with by providing separate lexical entries

2 4.1 5 Structural Ambiguities

Structural ambiguities result when there is more than one way in which to produce an interpretation of a sentence in accordance with the system's grammar rules There are essentially two kinds of structural ambiguity, deep and surface structure ambiguity While neither of these problems has as yet been tackled in the system of chapter 5, the latter deserves some mention as it is proposed that the methodology adopted in this implementation can be extended so as to cater for surface structure ambiguities

Surface structure ambiguities are a consequence of a lack of knowledge about prepositional phrase and relative clause attachment

(e) *die Frau, die die Magd sah*

the woman that the maid saw

the woman that saw the maid

Sentences such as the one shown in (e) are difficult to process due to the fact that many languages permit relative clauses to be viewed as sentences which contain a 'hole' While this does not present a problem for translation from English to German, the reverse produces two translations Hence, this is a problem perhaps best dealt with at the transfer stage, where both monolingual interpretations (the first where *die Magd* is adjudged to be the subject of the relative pronoun, the second where it is the object), are translated

The prepositional phrases in *the man saw the girl in the park with the telescope* have three possible attachments, namely *the park with the telescope, the girl with the telescope,* or *the man saw with the telescope* (example taken from [Hutch86]) One possible solution to this problem is to impose co-occurence restrictions on particular verbs by extending their subcategorisation frames For instance, in the lexical entry for the word *see,* it could be specified that it can be modified by a preposition followed by something which can aid sight This information would be obtained from semantic features in the entry for *telescope* One way of implementing this would be to integrate case grammars [Fill68] into a linguistic theory such as HPSG as in chapter 5

Other options, of course, would be to build parsers which simulate human reasoning when faced with such choices The principles of Minimal Attachment and Right Association have often been suggested for this purpose ([Kimb73], [Fraz78], [Fraz79]) Prompting for human interaction at the point where such an ambiguity is encountered is another recommendation often implemented, but this rules out the possibility of a fully- automatic system. Otherwise the best choice is to either employ a best guess strategy or simply hope that the ambiguity is preserved in the target language (as is often the case, German translation of above is *Der Mann sah das Madchen in dem Park mit dem Teleskop*, or in French, *l'homme a vu la fille dans le parc avec le téléscope)*

Hence, the issues of concern for the system designed and presented in chapter 5 include category and semantic ambiguity, derivational and inflectional morphology and transfer ambiguities Anaphora resolution and quantifier scope

problems are also handled by monolingual analysis, but are currently not a main concern for translation, and as such will be discussed in the chapter focusing on HPSG (chapter 4) Bearing this in mind, we now turn to the method of translation used and why

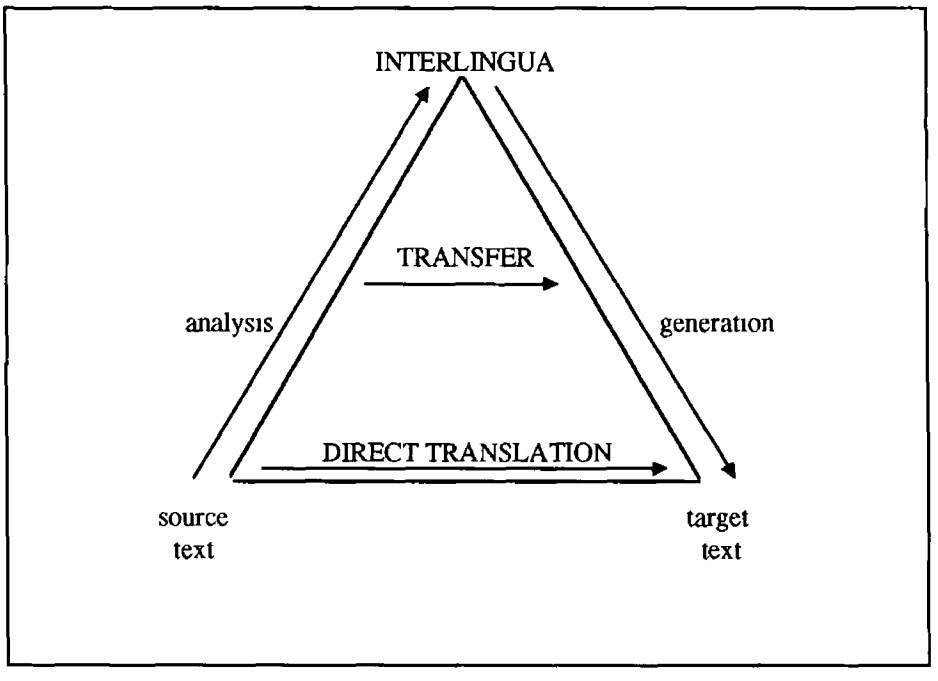


Figure 2 3

2 4 2 Transfer-based Translation

In a transfer-based system, the source language text is parsed into an internal code, named the analysis representation A bilingual component comprising a set of transfer rules converts this into the target language representation, from which the target text is generated These transfer rules can be lexical and/or structural in nature, dealing with problems exemplified by the comparison between the German structure for *Ich heiße Kim* and the corresponding English structure for *I am called Kim* (see previous section)

Examples of transfer-based methods include the aforementioned TAUM-MÉTÉO and EUROTRA, the research and development project established by the European Community in 1982 following four years of preparatory activities EUROTRA made use of a transfer module in translation systems for the then nine official languages of the EU, namely Danish, Dutch, English, German, Greek,

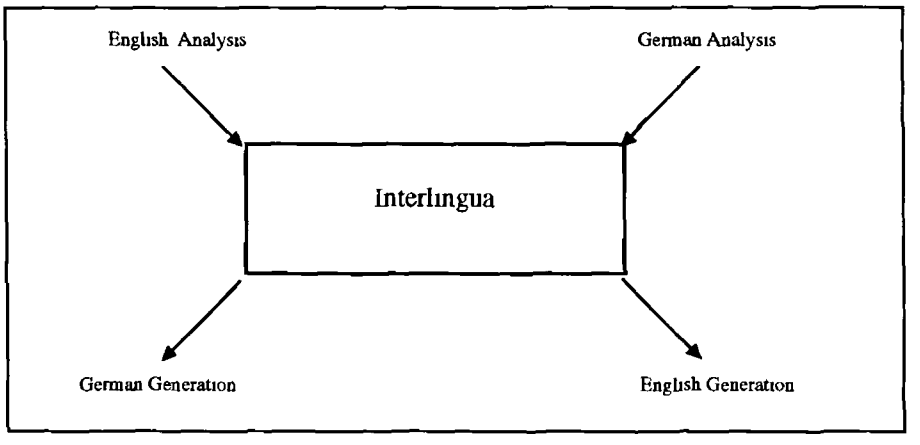


Figure 2 4

Italian, Portuguese, French and Spanish, and used contemporary linguistic theories as far as possible

The diagram in Figure 2 3 highlights the existence of two other principal methods of translation, namely direct and interlingual approaches Given that the diagram is in the shape of a pyramid, it can be seen that the more source language analysis is performed, the easier transfer becomes In fact, the need for transfer is eliminated entirely if an interlingua is implemented A successful interlingua system is reliant on the definition of a universal, language independent representation A source text can be analysed in such a way as to form this representation, from which **any** target language text can be generated

If such a language-independent representation could be designed, it would have a number of associated advantages Multilingual systems, for example, could be easily extended given that the addition of new language pairs would be relatively uncomplicated, as just two new modules need to be created (see Figures 2 4 and 2 5) It has one large disadvantage truly interlingual machine translation is almost impossible to obtain, due to the unfeasibility of creating such universal representations Take, for example, the Spanish translations *esquina* and *rincon*

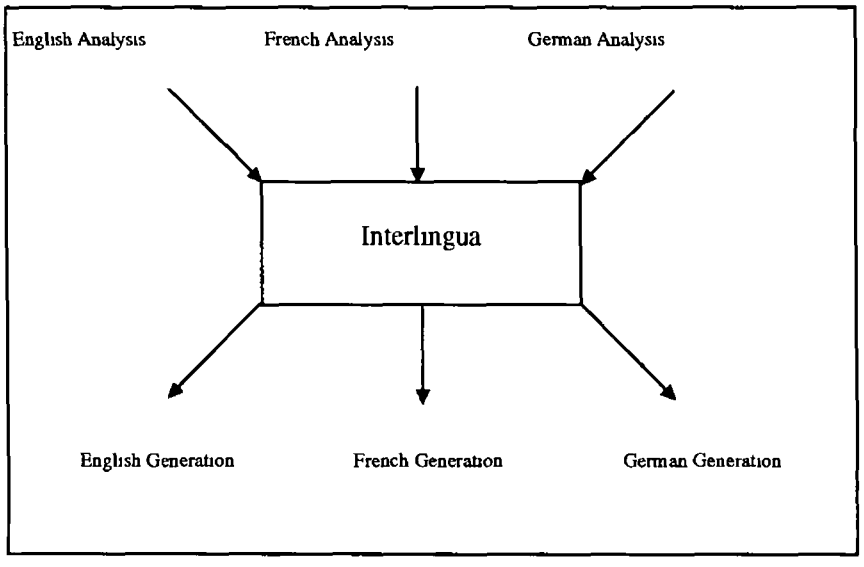


Figure 2 5

given for the English word *corner*, or the German equivalents of *wall*, *Mauer* and *Wand* The differences between these translations could be lost in an interlingua

An example of such an attempt at interlingual machine translation is the METAL system, developed at the Linguistic Research Center in Texas from 1961 onwards A system for the translation of German to English was operational by 1975, but there were many problems encountered, due mainly to the inadequacies of the universal representation Since then control of the project has been taken over by Siemens-Nixdorf (since 1980) with a number of universities throughout Europe (Germany, Spain, Belgium and Denmark) aiding in research The system was first commercialised in 1989, and research is ongoing, with all new developments being marketed by Siemens-Nixdorf and Sietec While the system has undergone radical changes since its commencement in 1961, workers on the project have continued to try to improve the quality of the interlingua

At the other end of the diagram in Figure 2 3 is the direct approach to machine translation Designed initially for a the unidirectional translation of a specific language pair, it was the first strategy adopted in an attempt to produce a working system It is critically dependent on well-developed dictionaries,

morphological analysis and text processing software Generally, there are a number of procedures involved in direct translation, including source text lookup in which all words in the input are searched for in a dictionary, some of them perhaps in the high-frequency component, in which lexical items such as *and, the* and *but* appear Other stages can be homograph and compound noun identification, idiom and preposition processing, resulting eventually in the rearrangement of words and phrases to produce the target language text The output from each of these stages is the input to the next (See Figure 2 6, taken from [Hutch92], for clarification of how translation proceeds) What this equates to is a system whose translation component is driven principally by word-for-word substitutions

SYSTRAN, a descendant of the Georgetown Experiment, has already been referred to several times in this chapter It is one of the most successful operating systems currently in widespread use and exemplifies the direct method It was originally intended for the translation of Russian to English, although a considerable number of language pairs have been added since, including French-English, German-English, English-Spanish, English-Portuguese, and more recently, Korean-English and Chinese-English (sponsored by NAIC since 1994) Each of these is unidirectional, with separate components included for reverse translation While the system was often dismissed out of hand by academics as a crude attempt at MT, its success cannot be denied This is due principally to the large amount of work, money and expertise invested in development of language pairs, lexical entries and their equivalents over the past three decades, resulting in extremely large dictionaries, rather than to a solid underlying linguistic theory

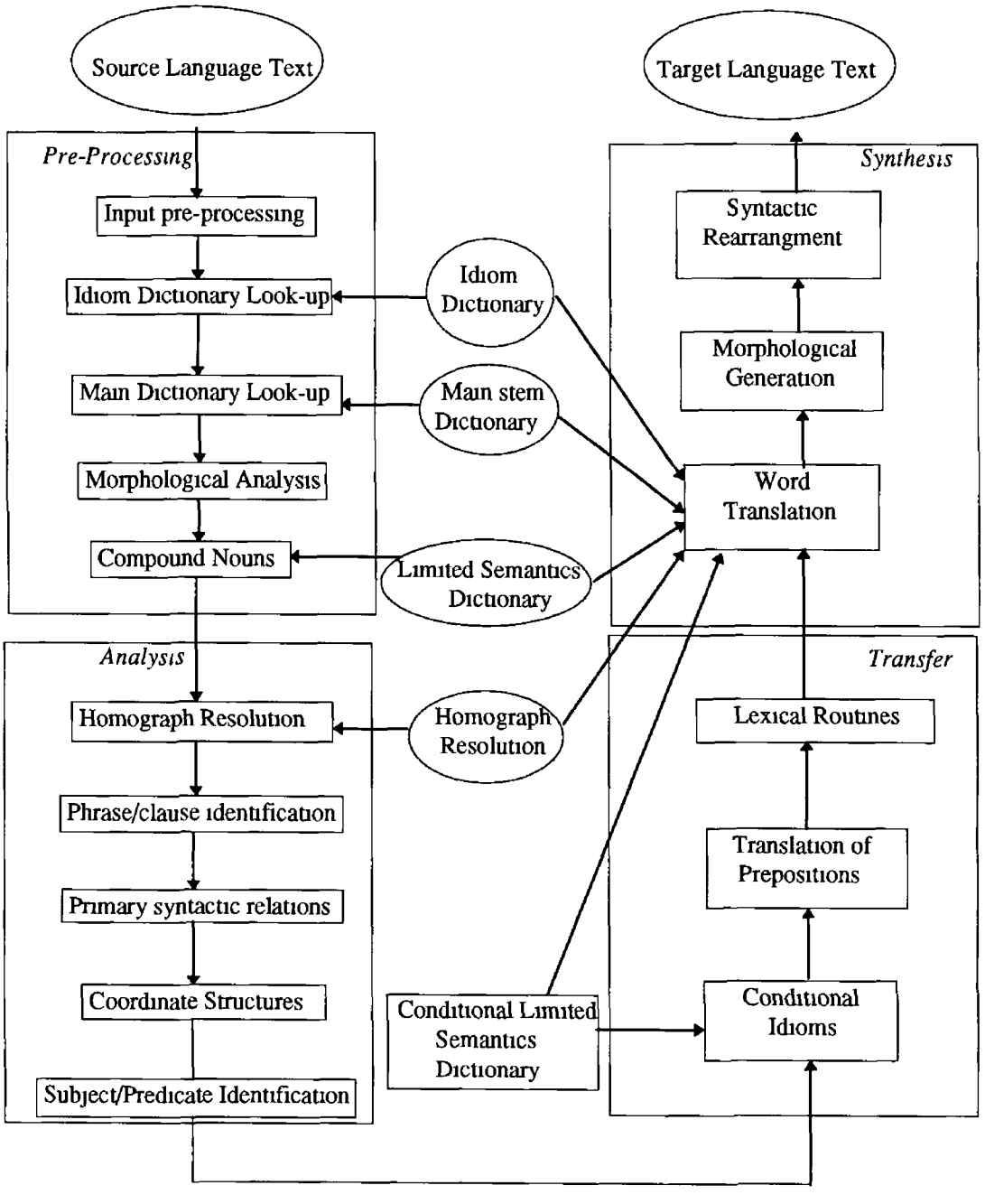

Source Language Text
Target Language Text
Pre-Processing
Input pre-processing
Idiom Dictionary Look-up
Main Dictionary Look-up
Morphological Analysis
Compound Nouns
Idiom Dictionary
Main stem Dictionary
Limited Semantics Dictionary
Synthesis
Syntactic Rearrangment
Morphological Generation
Word Translation
Analysis
Homograph Resolution
Phrase/clause identification
Primary syntactic relations
Coordinate Structures
Subject/Predicate Identification
Homograph Resolution
Conditional Limited Semantics Dictionary
Transfer
Lexical Routines
Translation of Prepositions
Conditional Idioms

Figure 2 6

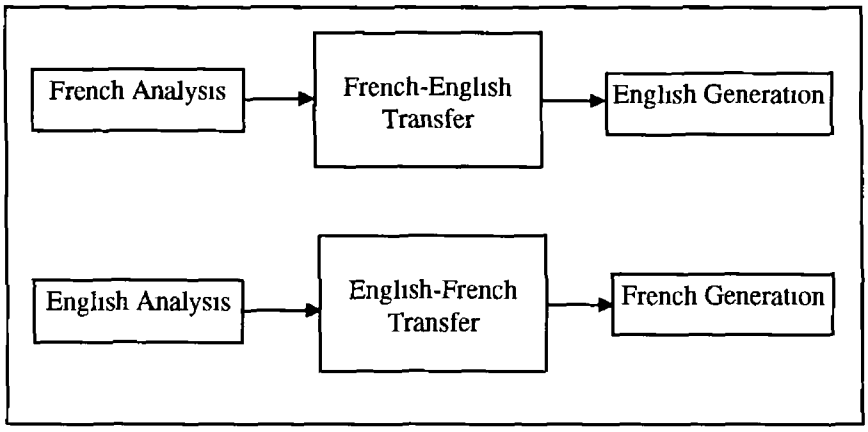


Figure 2 7

Given the methodological inadequacies of direct systems and the unfeasibility of an interlingual approach, a transfer-based methodology seems the most likely compromise While there are disadvantages to this strategy, in that the addition of new language pairs is considerably more involved that for an interlingua, these are necessary To add another language pair to a transfer-based system, a new node for the source language analysis, the target language generation and the bilingual link between the two must all be added See, for example, the diagrams in Figures 2 8 and 2 9 However, transfer-based approaches rely heavily on the analysis representation chosen The system presented here uses Head-Driven Phrase Structure Grammar as its formalism, not only for analysis, but also for generation (although this is not tackled in this implementation) HPSG's reversiblity property makes it particularly suitable for the task of translation Indeed, it is often claimed that it is this process-independence which makes HPSG in many ways a psychologically realistic theory of grammar (see Chapter 3), a goal striven for in an attempt to understand human linguistic processing, and in this case, human translation skills

The fact that HPSG is a theory which posits several, albeit integrated, levels of structure is in keeping with research into the possibility of developing *flexible* MT systems Such systems would only carry out a full analysis when absolutely necessary In the case of closely related languages, for example, such as

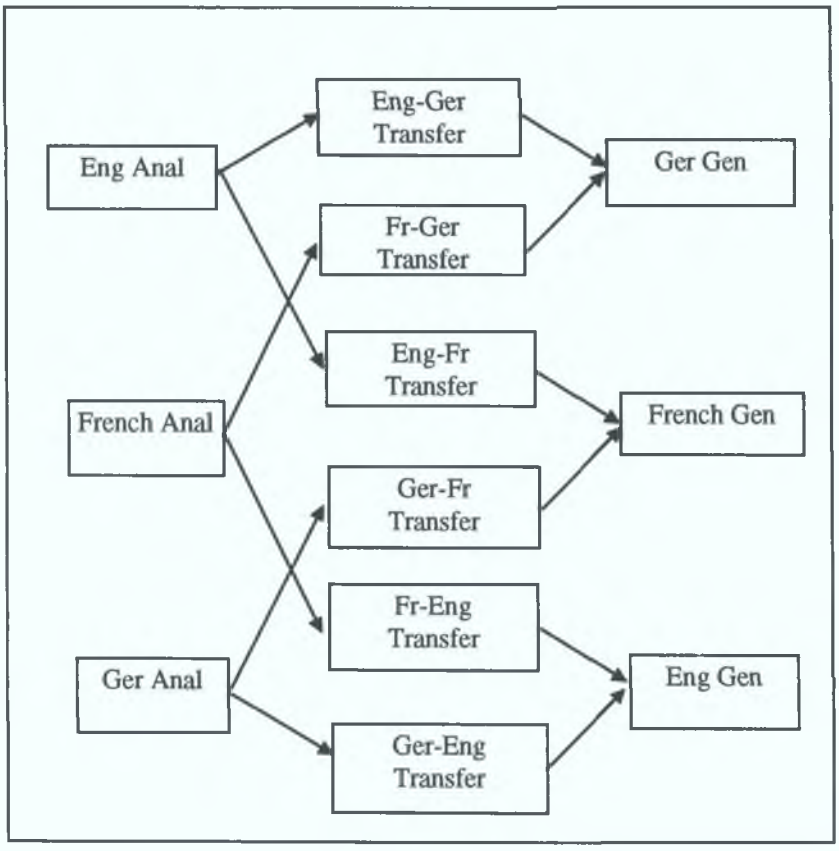


Figure 2.8

French and Spanish, the amount of analysis required could be considerably less than two languages which were members of entirely different families, e.g. English and Chinese. So, the amount of analysis carried out could be varied according to which languages were involved in the translation process. While this is not a consideration for the system of chapter 5, it is an option possibly worth evaluating in the future. An HPSG analysis representation consists of a description of the mother node and its daughters. So, for example, a simple sentence analysed according to the ID-schema which subsumes S -> NP VP, would produce a representation comprised of the details of the sentence (tense, voice, subject, complements, semantic content etc.), as well as information about the NP and VP daughters. Some of this data is superfluous for the purpose of translation, so, given that the system focuses to a large extent on semantic ambiguities, and that many of the problematic areas of translation have already been dealt with in producing the monolingual analysis (category ambiguity, derivational and inflectional morphology, quantifier scope), the system makes use of the

suggestion by Arnold et al. [Arn94] that the semantic content of the mother and some verb features are sufficient when selecting transfer equivalents.

Other recent ideas for approaches to MT include the knowledge-based method, in which the domain model is developed as far as is deemed necessary, incorporating information from several different sources. With knowledge of semantics and pragmatics, such systems are predicted to possess an ability to reason about concepts in the domain, although this ability is admittedly limited. The KANT system, developed at Carnegie Mellon University in Pittsburgh, is perhaps the best example of this approach. However, given the extreme difficulty of incorporating pragmatic and contextual knowledge to any great extent, this option was discarded here in favour of an HPSG-based system which does include some contextual constraints, but ones which are relatively easily incorporated into the formalism's feature structure representation (see chapter 4).

A recent alternative approach to MT has attempted to exclude linguistic knowledge as far as possible. This *empirical* approach is exemplified in statistical and example-based methods. The former of these depends on the notions of a language model, providing probabilities about the likelihood of specific strings of words occurring in the source language, and on a translation model. This translation model supplies additional probabilities about the occurrences of particular source-target pairs. The results of such systems, however, have been disappointing to date[1], prompting some researchers to incorporate some low level grammatical information.

Example-based approaches abandon all notion of mapping rules in favour of matching methods. A phrase in the source language is compared with the elements of a bilingual corpus of source-target pairs, and a best match algorithm is utilised in the formation of a translation template. This template is then completed by word-for-word methods. Obviously, this strategy is critically dependent on reliable data, and is thus extremely limited given that this information is available

for very few language pairs, but it is said by some to be analogous to the way in which human translators operate when making use of a bilingual dictionary

However, both of these approaches are flawed in that many of the problems of translation are logically solved using syntactic knowledge, an aspect of interpretation excluded in most systems designed according to statistical and example-based methods These include the category ambiguities and morphological issues addressed in the previous section Likewise, while the availability of resources for all aspects of natural language processing has undoubtedly increased over the past number of years, the amount of good quality sample data, on which these strategies are so dependent, is still relatively small

2.5 Conclusion

This chapter has attempted to provide a definition of MT and clarify why exactly it is needed, and hence, why ongoing research is justified Its history was reviewed briefly, concentrating on its more extreme highs and lows Before discussing the translation method chosen for the implementation of chapter 5, some of the issues to be addressed by the system were introduced, namely category, semantic and transfer ambiguity, inflectional and derivational morphology and some structural ambiguity problems The reasons behind selecting the transfer-based approach to machine translation were elaborated upon, drawing on the theoretical inadequacies of direct systems and the unfeasibility of designing an interlingua as support for the choice It was also mentioned that the representation formalism for both analysis and generation of the target language text was Head-Driven Phrase Structure Grammar In the next chapter, the underlying theories and principle of this grammar theory are presented

[1]Arnold et al quote an example in which a system designed by researchers at IBM had a 39% success rate when tested on a set of 100 short sentences [Arn94]

Chapter 3
Underlying Principles of HPSG

3.1 Introduction

' the very fact that communication is possible using natural language acquires an air of considerable mystery It is clear that we must prefer a linguistic theory whose grammars provide partial linguistic descriptions of a sort that can be flexibly integrated with non-linguistic information in a model of language processing a theory that posits different kinds of processing regimes based on a single linguistic description"

[Sag95]

The original concept behind the inception of Head-Driven Phrase Structure Grammar (HPSG) was to design a formalism which, as far as possible, used psycholinguistic evidence as its starting point In addition, its designers, Carl Pollard and Ivan Sag, proclaim it to have been a conscious effort to integrate ideas from several other linguistic formalisms, including most notably, Government-Binding Theory, Lexical-Functional Grammar, Categorial Grammar, Generalised Phrase Structure Grammar, Situation Semantics and Discourse Representation Theory [Sag93] Bearing these two considerations in mind, the linguistic theory behind the HPSG grammar formalism will be discussed in the following sections

3.2 Psycholinguistic Considerations

Evidence from recent studies by Tanenhaus and Trueswell ([TanTr95]) and McDonald et al ([McDon95]) supports the underlying theory of HPSG that language should be represented as a system of constraints which are enforced by the various components of a grammar Hence, the specification of any linguistic object is the cumulative result of the interaction of constraints arising from lexical entries, universal principles, language-specific rules and contextual factors influencing the correct interpretation of language All of these constituents, as well as the sorted feature structures used to represent them, will be discussed in considerably more detail in the following chapter For now, however, it is only important to consider their implications in the design of HPSG's theory

It is held almost without dispute that the construction of any linguistic theory which can be successfully formalised is dependent to some extent on the

incorporation of our knowledge about human linguistic processing While our understanding of this procedure is still in its infancy stages, there are several psychologically relevant ideas which played a critical role in the original conception and development of HPSG More specifically, HPSG relies on the incremental, order-independent, process-independent integration of partial grammatical information

A large amount of this grammatical information emanates from the important part played by lexical information in language processing Indeed, one of the central theories of HPSG is that many of the properties of a phrase are a direct result of its being a projection of a lexical head, a single word with a highly articulated lexical entry which expresses certain requirements (for example, subcategorisation, tense, number etc) This follows the trend of more recent grammar formalisms which place extreme importance on the lexicon (GPSG,LFG and Categorial Grammar), and is in line with the theory that human language processing is highly lexicalist in nature

3 2 1 Incremental Nature of Human Language Processing

The notion of 'echo questions' is one with which virtually all speakers of a natural language are familiar, if not by name, then certainly from experience It is a phenomenon which exemplifies the incremental nature of human language processing Take, for example, a conversation between two people, Speaker A and Speaker B An extract from the discussion in which Speaker A informs Speaker B of their intention to go to the cinema the following evening might look something like this

Speaker A John and I are going to the cinema to see "Babe" tomorrow Do you want to join us?
Speaker B You are going to see what?

It is highly likely that Speaker A will know the answer to this question long before Speaker B has finished the utterance One could perhaps even go so far as to say that, having inferred that Speaker B's intention is to repeat A's own words,

Speaker A could piece together the remainder of the utterance having simply heard the word "what" 'Echo questions' are not the only example of incrementality - everyone has no doubt completed somebody else's sentence - but it also highlights par excellence the fact that partial linguistic knowledge often suffices for processing purposes Perhaps the best way to describe incrementality is as the constant revision of *"representations on the basis of both input information and information from the discourse context"* (Marsten-Wilson, 1973, quoted in [Sag95]) Any linguistic theory which could use this fact would obviously be superior in terms of psycholinguistic adequacy to one which does not This definition also highlights a further requirement, that of information integration

3 2 2 Information Integration

The area of linguistics which continues to cause numerous problems for theoreticians, computational linguists and psycholinguists alike, is undoubtedly the integration of contextual and world knowledge, as well as speaker/hearer or writer/reader specific information, with syntactic and semantic data While it is obvious that the interpretation of even the simplest of sentences requires some amount of non-linguistic knowledge, the question of how and when such knowledge is used is still a matter of intrinsic debate

The much-quoted example of *"Kim found the book on the atom"* ([PoSa94], [Sag95], [Sag94]) illustrates perfectly how humans use information about the world to choose the correct analysis, in most cases failing to realise that there is any ambiguity present at all The reading that the book is sitting on top of the atom is instantly dismissed by humans in favour of an interpretation where Kim has found a book on the topic of atoms However, without knowledge of the size of atoms as compared with the size of books, the choice is not as clear-cut So reliant, in fact, is language processing on such factors that it is sometimes difficult to understand how any kind of communication between humans is possible at all

Despite the distinct lack of comprehension of the manner in which linguistic and non-linguistic information is combined, any model which tries in some way to simulate this aspect of human language processing certainly has a tremendous advantage over a model which neglects it entirely, and should thus be considered carefully

3 2 3 Order Independence

Examples 3 1 and 3 2 bring to light the fact that the order in which data is consulted is dependent entirely on the order of the input It cannot simply be claimed that syntactic analysis is performed first, followed by semantic interpretation and then by any other knowledge to be used in processing This is the strategy adopted in Two-Stage Models of language processing where an initial syntactic structure, determined by principles such as Minimal Attachment, Right Association or Late Closure, is constructed and only then are semantic and non-linguistic considerations taken into account in deciding on the correct analysis

(3 1) The sheep that was sleeping in the pen stood up

(3 2) The sheep in the pen had been sleeping and were about to wake up
(Examples taken from [PoSa94])

In (3 1) morphological information determines the cardinality of the sheep in question long before world knowledge comes into play in deciding between the two senses of the word *pen*, namely an enclosure or a writing implement This situation is reversed in (3 2) where the meaning of *pen* has been ascertained in advance of any conclusions made about how many sheep were involved

Hence, another consideration for any linguistic theory is to display no bias towards the order in which information is consulted in deriving an analysis of any natural language sentence or phrase

3 2 4 Process Independence

The final requirement of a linguistic theory is that it 'exhibit no bias towards any particular kind of processing activity It is known that the set of sentences which can be produced by a specific speaker is closely related to the set which they can comprehend ([PoSa94]) While it is true that language production is somewhat more inhibited than understanding, it seems obvious that a single grammar theory which can be adjusted appropriately (perhaps by reducing or imposing certain constraints) to suit the process involved, is more appealing than designing a separate theory for each process Such a unique theory could perhaps help to model how hearers who do not always strictly adhere to grammar principles can understand a larger set of utterances than those who do

While language comprehension and production are the most overt examples to cite in support of a single linguistic theory, this condition also holds true for other processes, including translation and language-games In chapters 4 and 5, the suitability of such a theory for machine translation will be discussed in more detail

3 2 5 HPSG's Satisfaction of Criteria

While the features of HPSG will be discussed in terms of their satisfaction of the prerequisites for a linguistic theory in greater detail in the following chapter, a brief outline of its adherence to the abovementioned criteria is presented here The fact that HPSG was born of the co-operation between, not just theoretical linguists, but also computational linguists and cognitive scientists, ensured that each of the requirements were important considerations in its design

HPSG is essentially a more advanced version of a context-free grammar, augmented with constraints CFGs have long been accepted as being completely process-neutral, so obviously this property is inherited by its descendants An attempt to provide evidence in favour of HPSG's satisfaction of this statement is made in chapter 5 by discussing an implementation of a machine translation system based almost entirely on the theory of HPSG as laid out in [PoSa94]

The 'computability of operations', a property of many constraint-based approaches, is determined by Stuart Shieber to ensure an incremental analysis of language This property imposes the condition that the grammatical theory be expressed as a system of logic, whose operations are *"computationally interpretable, and the solutions finitely representable, so that they can serve as the input to further computation"* ([Shieb92], quoted in [Sag95]) While HPSG is not entirely on a par with the kind of logical formalisms explored by Shieber, and hence caution in making any specific claims about HPSG's suitability for emulating incrementality must be shown, the fact remains that constraint-based theories of language show considerable promise for embodying this aspect of human language processing

HPSG's declarative nature (as opposed to the procedural nature of Government-Binding Theory) maintains the order-independent requirement placed on its linguistic theory The fact that its theory characterises merely what constraints should be imposed during analysis, and not how or in what order they should be applied, means that the grammatical components can be consulted in whatever order is deemed necessary by the process involved Each of these components is expressed in terms of a set of constraints which by definition are order-independent This characteristic of HPSG's architecture is yet another reason why its linguistic theory could be considered superior to other formalisms (Categorial Grammar, GB Theory, Dependency Grammar for example)

The nonderivationality trait of HPSG to a large extent maintains the flexible character of human language processing with respect to the integration of linguistic and non-linguistic information Contextual information is simply expressed in terms of constraints in lexical entries, as well as in the universal and language-specific principles, and can be consulted at any time during analysis The exact details of how this is formalised will be reviewed in chapter 4

Finally, one of the most critical psycholinguistic features of HPSG's linguistic theory (which has already been alluded to implicitly in this chapter) is its use of principles comprising a universal grammar This is in direct compliance

with the conventional school of thought that a linguistic theory should, as far as possible, account for linguistic knowledge One aspect of this knowledge forms one of the central points of HPSG, namely the possession of a universal grammar by all humans capable of linguistic communication The use of such constraints as the Head Feature Principle and Subcategorisation Principle (see chapter 4) represents linguistic organising principles which are believed to be common to all languages Any knowledge specific to a community of speakers of a particular language is accounted for by language-dependent rules

Having outlined the important psycholinguistically relevant requirements for a linguistic theory, it is necessary to take a look at the ideas HPSG has combined from other contemporary theories

3.3 HPSG's Debt of Honour

"HPSG was a very conscious effort to develop a precise framework for articulating hypotheses about grammar that would let one synthesise ideas from different traditions"

[Sag93]

The creators of HPSG have always gone to great lengths to acknowledge the role played by some of the most influential contemporary theories of grammar in the design of their formalism. In the following subsections, the contributions of Government-Binding Theory, Categorial Grammar, Lexical-Functional Grammar and Generalised Phrase Structure Grammar will be reviewed The significance of Discourse Representation Theory and Situation Semantics will be looked at in detail in chapter 4

3 3 1 Government-Binding Theory

While essentially members of two families which are in many ways unrelated, HPSG and GB Theory ([Haeg94], [Sell85]) do share a number of common features, not least of which is their use of highly articulated lexical entries and a parameterised universal grammar This is due, not to coincidental

overlap, but to the fact that Pollard and Sag consciously tried to incorporate those aspects of GB Theory which they considered most significant into the HPSG formalism. For example, in both theories, the number of language specific rules (or immediate dominance schemata) is greatly reduced by using a set of principles common to all languages. For now, it is important only to note these overlaps in the approaches. The details of HPSG's universal grammar will be explored fully in the following chapter, with reference made to those principles which have counterparts in GB. Similarly, the intricacies of lexical entries will be closely examined.

The chief contrast between HPSG and GB is the complete absence from the former of any notion of transformation. HPSG's nonderivational approach to language analysis removes the need to sequentially derive each level of a syntactic structure. While the removal of the need for operations such as move-α is obviously simpler in terms of formalising a theory, the primary advantage of a grammar in which a variety of information can be integrated, with no particular bias towards the order in which this information is consulted, is its adherence to the psycholinguistic requirements discussed above.

However, this poses the question of how the information normally dealt with by the move-α operation is treated in HPSG. The simplest answer is to point towards structure-sharing, a task enabled by the use of feature structures for the representation of the grammatical components. Subject and Object-equi (*Kim promised Sandy he would sleep; Kim persuaded Sandy to sleep*) and raising (*Kim seemed to sleep*) verbs are all simply and effectively accounted for by coinstantiation, where, for example, the agent of the sleeping action in the sentence *Kim persuaded Sandy to sleep* is token identical to the value of Sandy. Thus, there is no need to posit differing explanations for each of these phenomena as each is treated in exactly the same way.

Similarly, Unbounded-Dependency Constructions or wh-movement are accounted for by identifying the relationship between the gap and its filler as one

of structure-sharing, although HPSG does borrow from GB the notion that the gap position is occupied by a trace (or phonetically null constituent). Hence, the category and its properties which are missing are shared with the filler when found.

However, it is not the case that the role of transformations is entirely taken over by structure sharing. Passive constructions, for example, are dealt with by lexical rules, while the commonly-used 'subject-auxiliary' flip in English (or head movement in VSO word order, such as *tá mé* in Irish which corresponds to *I am* in English) is regarded as a consequence of the use of particular immediate dominance schemata (used to various extents in different languages).

The problem of moving the head of a verb phrase into INFL (inflected form) is eliminated completely in HPSG as the feature simply does not exist. Instead, when a verb is tensed, the value *finite* is assigned to a feature named VFORM. Likewise, a Boolean feature AUX determines whether a verb is auxiliary or not. While the arguments for this seem clear-cut, it is argued in chapter 5 that, for the purposes of machine translation, the use of the VFORM feature is insufficient as it leaves room for error.

As can be seen, the differences between HPSG and GB centre principally around the use or non-use of transformation. The similarities, however, while touched on only briefly in this chapter are of equal importance and shall be highlighted in due course, with explicit reference being made to those principles and approaches echoed in, or in some way adapted for, HPSG.

3.3.2 Lexical-Functional Grammar

LFG ([Kap82]) shares with HPSG the property of nonderivationality, as well as the motivation to design an architecture as psycholinguistically realistic as possible. Additionally, the fact that LFG is highly lexicalised, in that lexical entries contain a considerable amount of functional information, is in parallel with the underlying idea of HPSG that language processing is lexicalist in nature. In fact, HPSG's lexicalist nature can, to a large extent, be attributed to the success of this approach in LFG. Furthermore, the solutions to many linguistic phenomena

(mapping between active and passive voice for example) are more easily situated in the lexicon or in the f-structure, a set of attribute-value pairs which contain information about relationships between the parts of a sentence or utterance, for example, subject and predicate, than in the phrase structure level (c-structure)

LFG also incorporates a constraint-based approach into its formalism by using a series of equations when deriving the f-structure from the c-structure level The equations include person, number, gender, case etc and grammatical function similarly to features in HPSG in that they guarantee well-formedness

In conclusion, HPSG has adopted from LFG a highly lexicalist approach, using lexical entries and lexical rules to take over work traditionally done by transformation operations The use of constraints (arising from a series of equations) to ensure well-formedness is another aspect of LFG which is of paramount importance in HPSG

3 3.3 Generalised Phrase Structure Grammar

As with LFG, GPSG ([Benn95], [Sell85]) is nonderivational, the functions of transformations being distributed between the metarules and lexical and non-lexical immediate dominance rules

As is the case with HPSG, GPSG's grammar rules are a series of constraints on what structures are admissible in a particular language rather than rewrite rules Phrase structure trees are generated at random and checked for their legality by applying the grammar rules Principles such as Head Feature Convention and the Foot Feature Principle correspond closely to several of HPSG's universal principles and represent further constraints on structures produced in analysis In fact these principles formed the underlying ideas of their equivalents in HPSG

While GPSG treats sentences as projections of their head verbs (à la HPSG), subcategorisation is handled rather differently Verbs do not have subcategorisation frames in the sense that verbs in HPSG do Rather they have pointers to structures in which they can appear Transitive verbs, for example,

have indicators (represented by integers) which ensure that they can only be inserted into subtrees which have a noun phrase as a sister to a verb.

While GPSG does make use of universal principles, it was initially designed as a theory for English, and as such, is not as easily extended to other languages as a grammar intended for broader use. Thus, it is not entirely suitable for the purposes of machine translation, or at least not for the system presented in this thesis. GPSG has several sets of rules (lexical ID-rules, non-lexical ID-rules, metarules, expanded ID-rules) which detract from its ability to function effectively as a universal grammar. As Chomsky points out, it is a mistake to develop a grammar of English full of *"lots of rules"* and *"little riders"* that ensure correct analysis but provide nothing of sufficient generality that could lead to hypotheses about universal grammar([Sell85]).

3.3.4 Categorial Grammar

Another example of a nonderivational approach to language, Categorial Grammar's ([Wyb91], [Dowt82]) primary contribution to HPSG is its treatment of subcategorisation. CG's combinatory rules (of the type A/B B => A, A/B B/C => A/C or C\B B\A => C\A) cater for cancellation and composition. Of particular interest for HPSG is cancellation, which in effect, ensures that a sentence is completely saturated (i.e. well-formed) by specifying for each constituent its necessary complements. For example, a transitive verb is described by VP/NP. While the mechanisms used to implement cancellation in each formalism is different (the two functor symbols '\' and '/' in CG and feature structures in HPSG), the effect is exactly the same.

Nowadays, Categorial Grammar is a formalism watched closely by followers of HPSG. Ivan Sag has been known to comment on the closer relationships between ongoing work in the two areas ([Sag95]), but for the purposes of this thesis, which follows the formalism for HPSG laid out in [PoSa94], the similarities between cancellation in CG and subcategorisation in HPSG are the most important factors.

3.4 Conclusion

In this chapter the psycholinguistic requirements considered in the design of the linguistic theory of HPSG were introduced These are incrementality, integration of linguistic and non-linguistic knowledge, order and process independence, and the formalism of a universal grammar Additionally, other significant aspects of HPSG's origins were discussed by highlighting the debt it owes to other contemporary theories of grammar, most notably, Government-Binding Theory, Lexical-Functional Grammar, Generalised Phrase Structure Grammar and Categorial Grammar In the next chapter, the modelling domain and constraints of HPSG will be examined in detail, including those referred to in this chapter, namely universal grammars, ID-schemata, lexical entries and rules, and feature structures

Chapter 4
Overview of HPSG

4.1 Introduction

'HPSG is formulated in terms of constraints [which] provide partial grammatical information that can be flexibly consulted in a variety of language processing models based on the notion of incremental, on-line integration of heterogeneous types of information

[CSLI96]

As already highlighted in the previous chapter, HPSG is a formalism based on the interaction of complex order-independent constraints, rather than transformational derivations These constraints are to be found principally in the guises of lexical entries and lexical rules, Immediate Dominance Schemata and a set of Universal Principles Before reviewing each of these areas, it is necessary to introduce the notion of sorted Attribute Value Matrices, the modelling domain used for representation of HPSG constructs, as well as the unification and subsumption operations which are of crucial importance for all members of the unification-based grammar family What follows is a discussion of those aspects of HPSG considered relevant for the implementation presented in the following chapter Following the review of HPSG's modelling domain, there follows a discussion of the principal features used when describing linguistic words or phrases While not all of these are used to direct translation, they are important in determining the correct monolingual analysis, particularly syntactically As noted in chapter 2 in the section concerned with those aspects of language which will be addressed in the implementation, it is HPSG's ID-schemata and use of category and subcategorisation features which help resolve many syntactic issues before translation can proceed Hence, these merit clarification here Other topics addressed include quantifiers, although this is a limited discussion as they are not a dominant feature of the MT system of chapter 5 In fact our only real concern with quantifers is the way in which storage of determiners such as *a* and *the* is handled Nonlocal constructions have been omitted entirely as they do not figure at all in the implementation

4.2 Attribute Value Matrices

Attribute value matrices (AVMs for short), such as the one shown in Figure 4 1, are used in HPSG to describe linguistic objects The name of this data structure is

determined from its use of values to describe particular attributes or features associated with an object Features are indicated by the use of uppercase letters, while atomic values are presented simply in lowercase

4 2 1 Values

Values can be either complex or atomic (simple) depending on the attribute In the case of the former, the value of a feature is another AVM The value of the INDEX attribute in Figure 4 1, for example, is complex, associating the features NUM(BER) and PER(SON) and GEN(DER) with its description These features in turn provide an example of atomic values, where each is assigned just one value, namely *sing(ular)* and *third* and *masc*, respectively

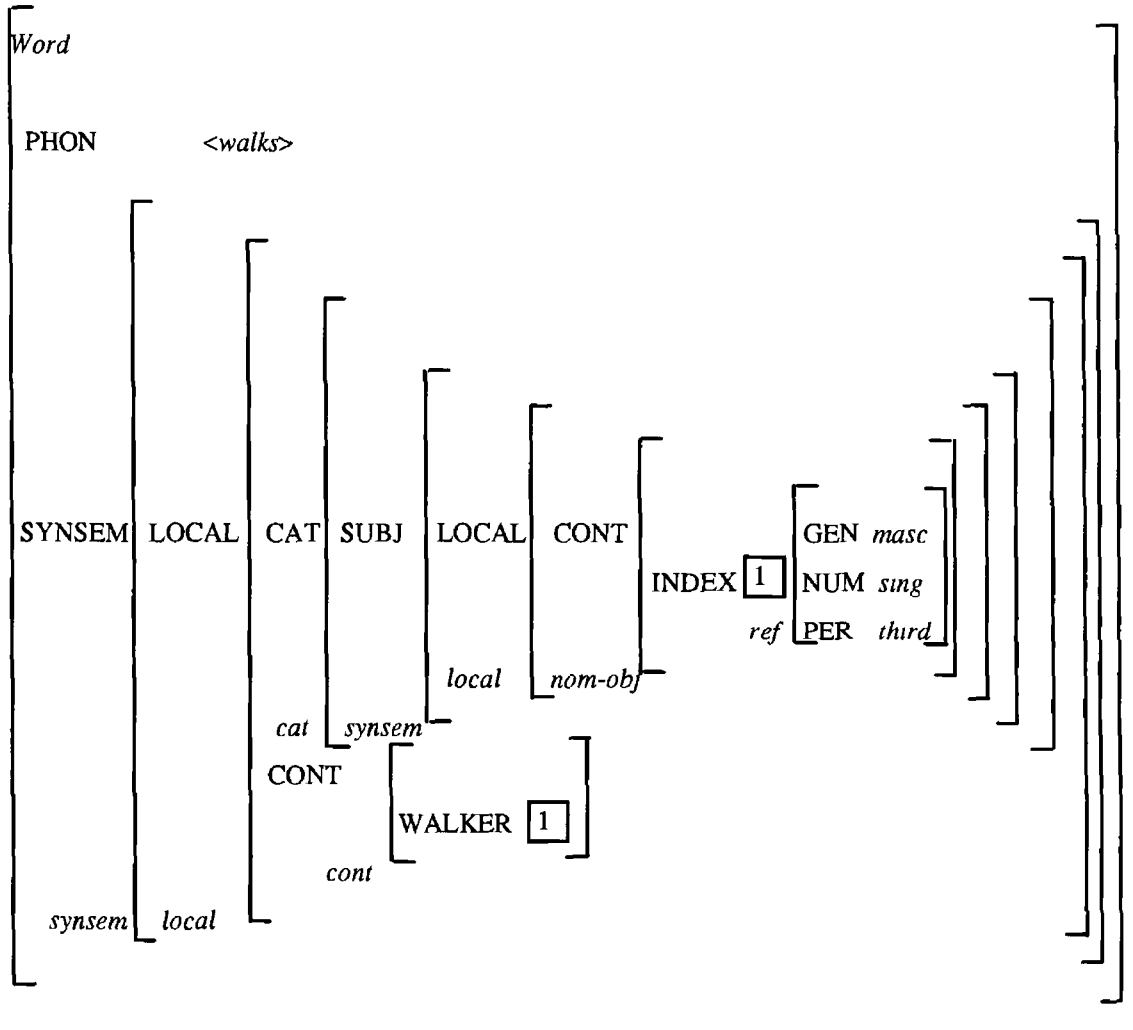


Figure 4 1

4 2 2 Sorts and Types

The italicised words on the left brackets in Figure 4 1 indicate that the AVM is sorted This implies that each attribute has a sort (also known as a type) associated with it, which constrains the attributes or atomic sorts it can take as its value In effect, sorts impose *'appropriateness conditions'* on features Returning to Figure 4 1 again, it can be seen that the value of INDEX is of type *referential* This ensures that INDEX can only be described by the attributes NUM(BER), PER(SON) and GEN(DER)

Sorts (which include atomic values) can stand in a hierarchical relationship to each other as shown in Figures 4 2 and 4 3 Those sorts which appear higher up in the tree are more general than those lower down These trees also introduce the notion of subtypes and supertypes In Figure 4 2, for example, *singular* and *plural* are subtypes of *number* Thus, any constraints which hold for both of these sorts need only to be specified once, namely on their most common supertype, in this case, *number*

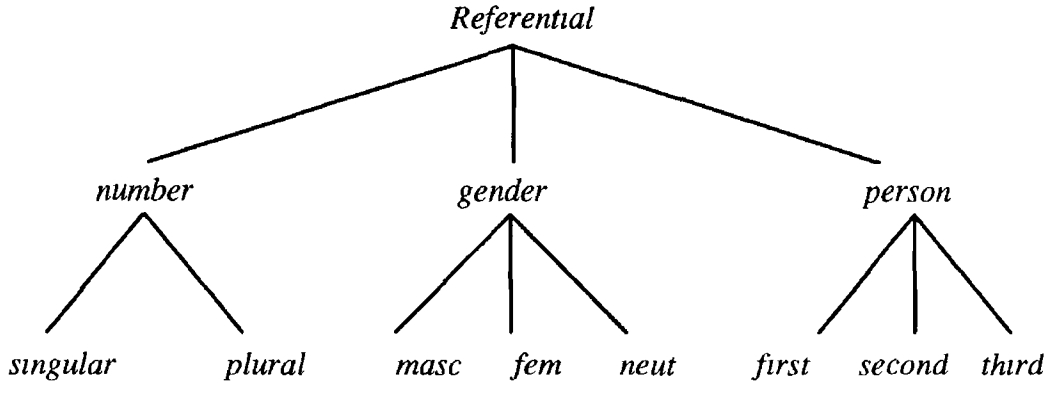


Figure 4 2

4 2 3 Feature Structures Vs AVMs

At this point, it is important to make reference to the difference between objects and their descriptions In HPSG, objects are modelled by feature structures which in turn can be described by attribute value matrices ([Rieh95]) While the latter are often referred to in terms of the former, there is in fact a very important distinction between the two Namely, feature structures are complete models of

linguistic objects, and cannot be partial in any way Moreover, they are considered to be totally well typed and sort resolved ([Rieh95]) This in essence means that any feature that can be present, must be present, and all their values must be of a maximal sort Take, for example, the feature GEN(DER) Its value cannot simply be gend, but must be one of the subtypes of gend, specifically, feminine, masculine or neuter By contrast, AVMs can be as partial as required, as well as underspecified if they are used to describe general properties of a linguistic object

At this point, a notational variation should be introduced to save confusion later in the section For the sake of clarity, sort names are often omitted when describing a HPSG linguistic object and path names are used Hence, the value of the CONTENT attribute in Figure 4 1 could also be written as SYNSEM|LOCAL|CONTENT|WALKER [1]

4 2 4 Structure Sharing

As indicated throughout chapter 3, structure sharing is one of the most important features of HPSG, in terms of eliminating the need for transformations Structure sharing is generally indicated by boxed integers, as shown in Figure 4 1

Structures marked with these boxed integers are said to be token identical (as opposed to type identical) What this means, is that it is not purely coincidental that two attributes have the same value, it is actually intended to be that way because they refer to the same object An example of what exactly structure sharing is can be seen in Figure 4 1 where the fully expanded value of the INDEX feature is preceded by [1]. This integer appears further down in the AVM as the value of the WALKER entity in the CONT(ENT) structure The values of INDEX and WALKER are said to be re-entrant or coinstantiated This property of HPSG objects is particularly important when using the operations of subsumption and unification which will be introduced briefly in the next section, before advancing to a discussion of the actual attributes and values admissible in a HPSG linguistic description

4.3 Unification and Subsumption

Current trends in generative and declarative grammars rely heavily on unification and feature matching, so before looking towards the integration of HPSG's constraints, it would seem sensible to have a clear understanding on this operation, as well as that of subsumption, on which unification is dependent

4 3 1 Subsumption

Given the definition that a category X subsumes a category Y if every piece of information in X is also contained in Y, subsumption can be compared to the subset relation in set theory Y in effect is an *extension* of X For example, (1) below subsumes (2) as (1) is more general This relationship can also be applied to hierarchical sorts of the type introduced in the previous section, where [NUM num] subsumes [NUM sing]

[NUM sing] (1)

$$\begin{bmatrix} \text{NUM} & \text{sing} \\ \text{PER} & \text{third} \end{bmatrix} \quad (2)$$

However, if two AVMs contain conflicting information (for example, [NUM sing] and [NUM plur]), then neither subsumes the other

4 3 2 Unification

Given the above definition of subsumption and extension, the unification of two categories can be described as the smallest category that extends them both, if such a category exists Otherwise, the unification of two categories is undefined (represented by $\perp$, known as bottom) An undefined unification occurs as a result of trying to unify two atoms a and b, where $a \neq b$ (*constant-constant clash)* Similarly, the result of any attempt made to unify a and b, where one of these is an atom and the other a complex value, is $\perp$ (*constant-complex clash*)

[NUM *sing*] (3)

[PER *third*] (4)

$$\begin{bmatrix} \text{NUM} & \textit{sing} \\ \text{PER} & \textit{third} \end{bmatrix} \quad (5)$$

(3) and (4) illustrate the successful unification of two categories in (5), while the results of unifying both (6) and (7) with [F [S M]] (shown in (8) and (9)) highlight the importance of structure sharing and the difference between type identity and token identity In HPSG these could only unify if their corresponding types are compatible For example, (3) and (4) are unified to form (5) because the typing system specifies that they can be joined together to partially describe an INDEX attribute

$$\begin{bmatrix} \text{F} & \boxed{1} & [\text{D} & \text{C}] \\ \text{G} & \boxed{1} & [\text{D} & \text{C}] \end{bmatrix} \quad (6)$$

$$\begin{bmatrix} \text{F} & [\text{D} & \text{C}] \\ \text{G} & [\text{D} & \text{C}] \end{bmatrix} \quad (7)$$

$$\begin{bmatrix} \text{F} & \boxed{1} & \begin{bmatrix} \text{D} & \text{C} \\ \text{S} & \text{M} \end{bmatrix} \\ \text{G} & \boxed{1} & \begin{bmatrix} \text{D} & \text{C} \\ \text{S} & \text{M} \end{bmatrix} \end{bmatrix} \quad (8)$$

$$\begin{bmatrix} F & \begin{bmatrix} D & C \\ S & M \end{bmatrix} \\ G & [D \quad C] \end{bmatrix} \qquad (9)$$

Having examined the underlying modelling domain and critical operation of HPSG's constraints, the next sections introduce each of the sources of these constraints, beginning with the lexical entries, thereby familiarising the reader with the admissible attributes and values in HPSG Also of prime importance in this section is the introduction of the notion of a multiple lexical hierarchy

4.4 HPSG's Lexical Entries

Influenced by the success achieved by the Lexical Functional Grammar, HPSG is highly lexicalised, articulating important information about a word in its dictionary entry This information includes, amongst other things, a subcategorisation frame, semantic content and contextual data The way in which this data is represented will be discussed in this section, with reference made at several points to the theory as presented in chapter 9 of [PoSa94]

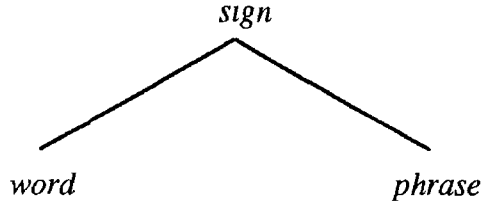


Figure 4 3

4 4 1 Signs

In HPSG a sign is defined as the most fundamental sort of an utterance An utterance can be a sentence or phrase as in *Kim walks*, or a single word, such as *Kim* or *walks* Thus, sign has two subsorts, namely word and phrase, as shown in Figure 4 3 Signs are assumed to possess a minimum of two attributes - PHON

and SYNSEM While PHON is assumed to be merely a string of phonemes, and for the purposes of this thesis will not play an important role, the SYNSEM attribute is the core of many of HPSG's essential components, containing all information about the syntax and semantics of a sign

The value of a SYNSEM attribute is a structured object of type *synsem* which constrains the values which can appear in the attributes feature structure This feature structure has two attributes, LOC(AL) and NONLOC(AL) For now, the former of these will be the focus of discussion

The LOC attribute combines information about syntax, semantics and context, hence the names of the three features which form the value of its assigned structure - CATEGORY, CONTENT and CONTEXT

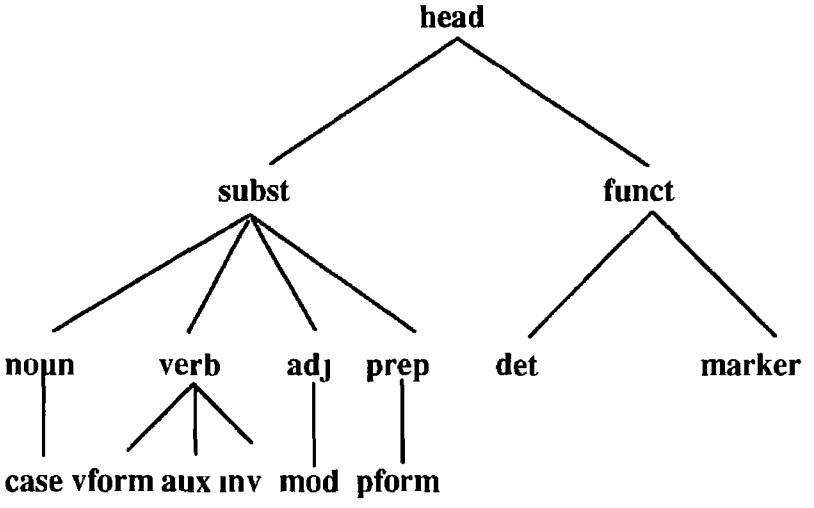


Figure 4 4

4 4 1 1 The CATEGORY Attribute

The CATEGORY value, of type *category*, is thought by many to be roughly equivalent to the traditional part of speech, or to the information contained in an X-theory category which has been stripped of all bar level details *Category* has four admissible attributes, HEAD (describes the properties of the head of a sign), SUBJ (describes the attributes of a sign's subject), SPR (describes the specifier of a sign) and COMPS (contains the details of those categories which a sign needs in order to become saturated) The last three of these are an element of HPSG III,

the formalism laid out in Chapter 9 of [PoSa94], and correspond more or less to the single valence feature of HPSG II, namely SUBCAT

The HEAD feature (see Figure 4 4) has two appropriate types, *substantive* and *functional* (abbreviated to *subst* and *funct* respectively) These essentially limit that categories a sign can take as its head Thus, the analysis of a noun phrase, *the girl,* would specify that its head is a noun whose properties correspond to those of *girl* This head would be of type *subst* which has a list of subtypes For simplicity, these will be referred to as parts of speech These include *noun* (as in the example above), *verb, adjective* and *preposition* etc The latter has two subsorts, *determiner* and *marker* (e g , complementisers) Associated with each of the subtypes of *subst* is the feature PRD (predicative) which has a Boolean value In addition, each of the subtypes has its own allowable features For example, *noun* has a CASE feature, while *verb* has a VFORM attribute, as well as the Boolean valued AUX (for auxiliary verbs such as *can, be* etc) and INV (when a clause is inverted, as in *was Kim here?)* It is some of the head features which are used to preserve information which occurs in the source language in the target language These features include *case* for nouns, *vform* for verbs, *pform* for prepositions etc

Robert Borsley ([Bors94], quoted in [PoSa94]) stated that the original SUBCAT feature of HPSG was in many ways insufficient in that it made no special provisions for access to the subject of a phrase Consequently, Pollard and Sag were prompted to take this into consideration and come up with a new idea for the representation of valence features Thus, the attributes SUBJ, COMPS and SPR were introduced to the theory Each of these takes as its value a list of *synsem* objects The SUBJ feature specifies the subject required by a lexical head, for example, most verbs have a nominative noun phrase as its subject Similarly, COMPS specifies the categories for which a word subcategorises, minus its subject (in effect the SUBCAT list of HPSG II is divided into these two separate features) The outcome of combining SUBJ and COMPS works much like cancellation in Categorial Grammar, as they express whether or not a phrase or word is fully saturated As indicated in chapter 2, the combined effect of these

subcategorisation features can aid in the resolution of category and transfer ambiguities, as well as certain structural problems The SPR attribute is a direct result of a proposition (also by Borsley) that specifiers should be considered as a grammatical relation completely separate from subjects (even though they are nonheads)

4 4 1.2 The CONTENT Attribute

Given that it is this aspect of a sign's information which is used primarily to drive transfer once syntactic problems have been resolved, its discussion is of particular significance The CONTENT feature represents a sign's role in the semantic analysis of an utterance or word The content of nouns and their projections is of the sort *nominal-object* (*nom-obj* for short), which introduces the attribute INDEX of type *index* This INDEX is analogous to a reference marker in Discourse Representation Theory ([Kamp81] or [Mahon95] for an overview of DRT) It is to these indices that semantic roles are assigned, indicating the participants in a relation or action The assignment of semantic roles is governed by Situation Semantics ([Barw83]), which views the world in terms of individuals, properties, relations and situations, whereby situations are a consequence of individuals playing specific roles in a relation For now, it will suffice to discuss the attributes and values of the CONTENT feature, for, as far as possible, this structure will be maintained in the alternative approach to semantic analysis, although some changes will be made to this in the implementation

Linguistic objects of type *nom-obj* can be divided into two subsorts, *nonpronoun (npro)* and *pronoun (pron)* The latter of these can be further partitioned into the subtypes *personal-pronoun (ppro)* and *anaphor (ana)* Lastly, objects of type *ana* can be further described by either *reflexive (refl)* or *reciprocal (recp)* As a guideline, it can be assumed that pronouns such as *she* in the sentence *she hurt herself* fall into the category of *ppro*, while the maximal sort *refl* describes *herself*

From the above example, it can be seen that *she* and *herself* actually refer to the same entity, although they have differing content values Hence, the

INDEX values of both must be structure-shared to ensure agreement This introduces the agreement features of INDEX, to wit GENDER, PERSON and NUMBER Two nominals which are thus structure-shared are said to be *coindexed* This is also a perfect example of the difference between token and type identity The fact that these structures are coindexed or token identical means that sentences such as *she hurt himself* are deemed incorrect, while *she hurt him* are allowed

There is one further feature which appears in the semantic description of nominals This RESTRICTION attribute takes as its value a set of possibly quantified states of affairs (psoas) and places a semantic restriction on an index The elements of such sets can be referential indices or psoas Generally speaking, the index arguments arise from the content of noun phrases, while psoa arguments are a result of sentential and verb phrase complements The RESTR feature in Figure 4 5, for example, states that the object in question is a house

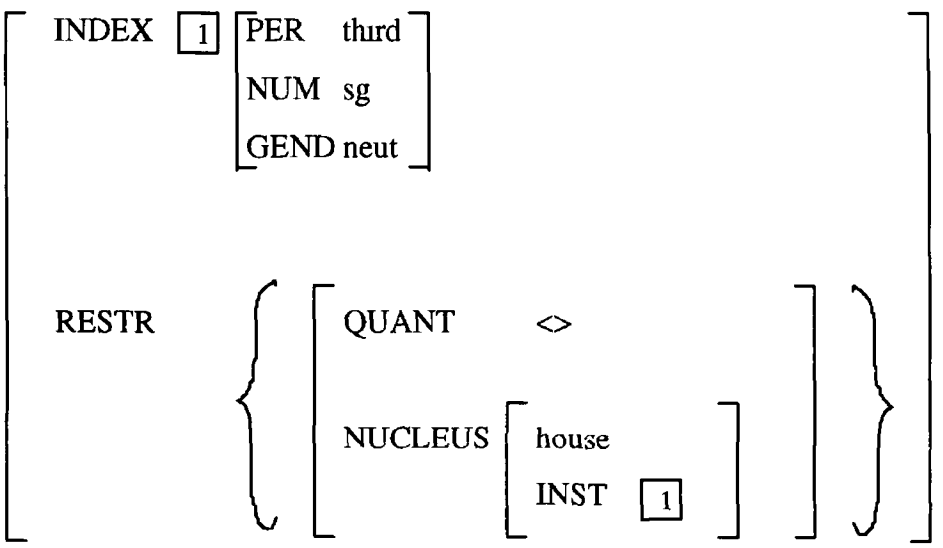


Figure 4 5

The *psoa* type introduces two further features, QUANT and NUCLEUS The former takes a possibly empty list of quantifiers as its value, and the latter contains semantic information of the type *quantifer-free psoa (qfpsoa)* The idea behind listing quantifiers in this way is to capture all aspects of scope ambiguity arising from the use of a quantifier, as in the example sentence (taken from [HPSG94]),

Every man loves some woman There are two possible interpretation for this sentence Firstly, it can be taken to mean that there is a single woman who is loved by all men The second analysis is that for each man there is a possibly different woman that he loves

The *qfpsoa* type has many subtypes, arising principally from the lexical entries Taking the verb *snore* as a starting point, *qfpsoa* can be divided into the types *property, unary-relation* and *binary-relation* *Snore,* then, is a subsort of type *unary-relation* and, in accordance with the theory of situation semantics, introduces the attribute SNORER whose value is a referential index Properties of nouns such as being a house, or being large (or whatever), are considered atomic semantic representations and introduce the attribute INST(ANCE), whose value is also of type *referential*

As already stated above, the value of the QUANT attribute is a list of quantifiers For the purposes of the implementation in chapter 5, its main significance is the way in which it deals with determiners It introduces two additional features, DETERMINER and RESTIND, the former of which can have the value *forall, exists* or *the* The second feature contains a nominal object, thus ensuring that the quantifier has a range over a property

The section entitled Sample Lexical Entries will hopefully help to make clearer the significance and role of each of these attributes and types by presenting them in some lexical entries for verbs, nominals, determiners etc Next, however, follows a brief discussion on the CONTEXT feature

4 4 1 3 The CONTEXT Attribute

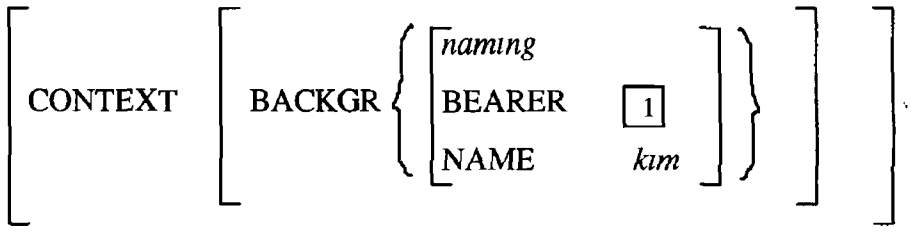


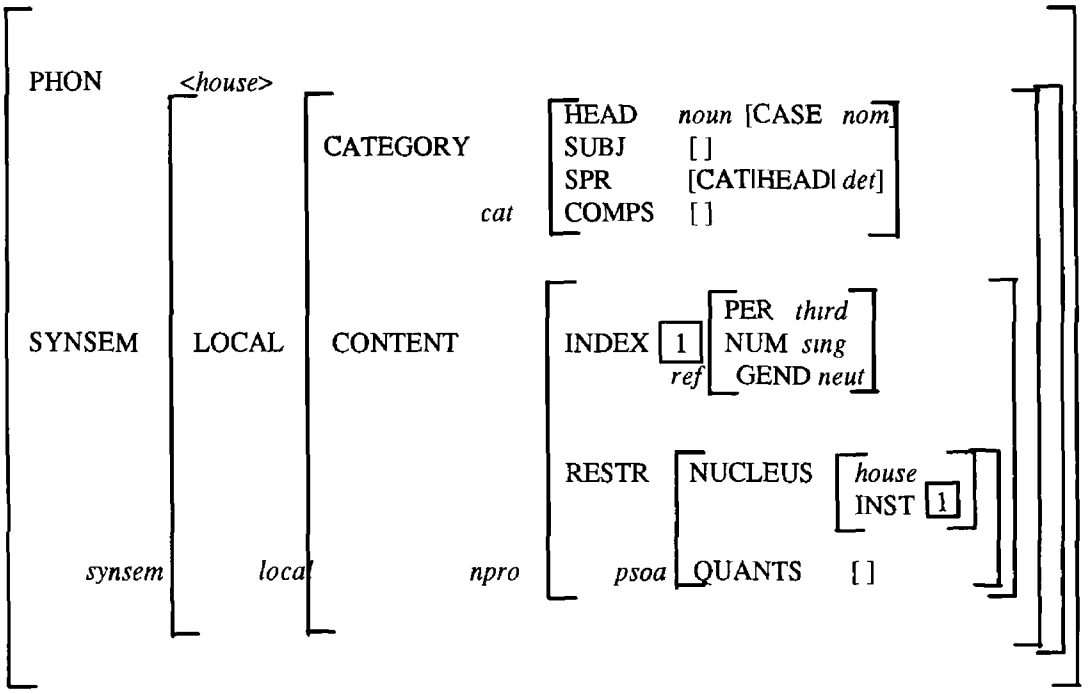
Figure 4 6

It is the CONTEXT attribute which is used in translation when any of the roles are proper nouns, as it is here that all important information about name occurs Quite simply, the CONTEXT attribute contains context-dependent linguistic information, which cannot be represented in by either CATEGORY or CONTENT It is of sort *context* and posits a single feature, BACKGROUND Like the values of the semantic feature RESTR, the value of BACKGROUND is a set of psoas which impose truth conditions on the utterance context, and deal with such linguistic phenomena as presupposition or conversational implicatures Take, for example, the context value of a proper noun *Kim* as shown in Figure 4 6 The name associated with a person is considered by HPSG to be a relation, thus introducing the NUCLEUS feature whose function is the same as in semantic CONTENT NUCLEUS is of type *naming* which introduces the features BEARER and NAME BEARER in this case has the same function as INST in the preceding section it is a referential index which identifies the bearer of the name as indicated by the value of the NAME feature This value refers to the name *Kim* and not to an individual Hence, the entire CONTEXT attribute corresponds to the presupposition that the referent (signified by the BEARER) is recognised as an individual named *Kim* in the utterance context

4 4 2 Sample Lexical Entries

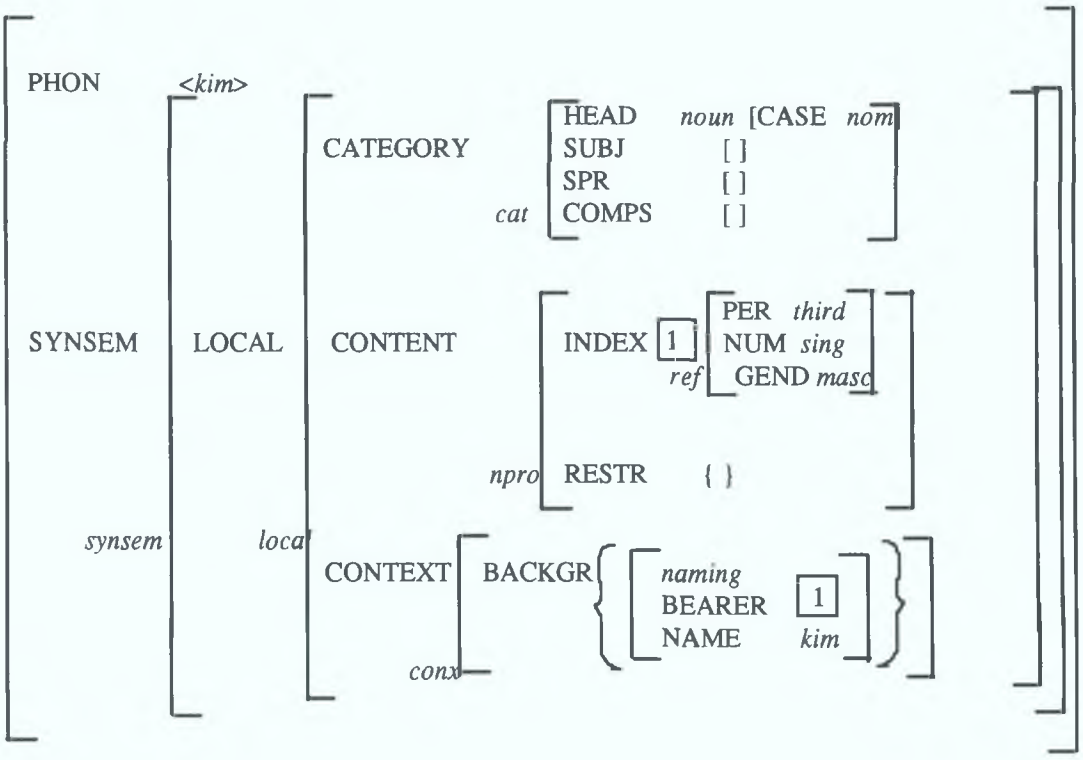
This section presents and describes a number of sample lexical entries in an attempt to clarify the role of each of the features and values discussed in the previous section Due to the variation in entries according to part of speech, nouns, verbs, adjuncts and determiners will be explained separately It is also worth noting at this point, that while the entries shown here are for the most part

fully expanded, HPSG's typing system allows for lexical items to be compactly arranged in a multiple hierarchy This greatly reduces the amount of redundant information in the lexicon as only the properties (syntactic, semantic or contextual) specific to a particular word need to be explicitly identified



4 4.2 1 Noun Entries

The above entry for the word *house* indicates that has a phonetic value of *<house>*, it head is a noun of case nominative, it has no subject or complements, and its specifier is a determiner Constructs such as *Kim's house* are dealt with by separate entries for possessives such as 's or my As this is not currently an issue for the system in chapter 5, its use will not be elaborated upon Its content value indicates an instance of a house which is co-indexed with the referent third person singular and its gender is neuter This is in accordance with the requirement stated previously that the restriction imposed by a common noun be associated with a referent in the discourse, in this case the value of INDEX The quantifier attribute corresponds to an empty list, while the context feature has been omitted altogether for the sake of clarity, as its value is also empty

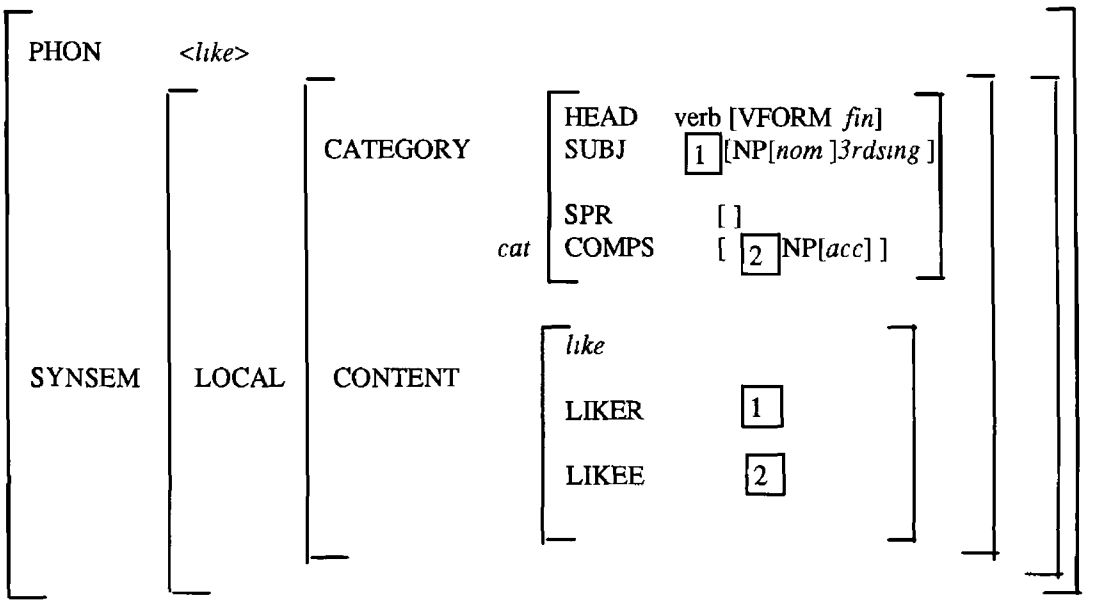


It can be assumed at this stage that the lexical entries for nearly all common nouns correspond to the example above, although the assignment of case values is generally carried out by lexical rules during analysis. Where the form of a noun changes according to case (e.g., weak nouns in German such as *Junge*), a separate entry is currently required. This redundancy could be easily overcome by implementing an additional lexical rule. However, representations of proper nouns differ somewhat, mostly in the assignment of content and context values. The RESTR value is an empty set, while its CONTEXT value is equivalent to Figure 4.5, indicating that the referent of BEARER can be identified by the name *Kim*. The full lexical entry for *Kim* can be seen above.

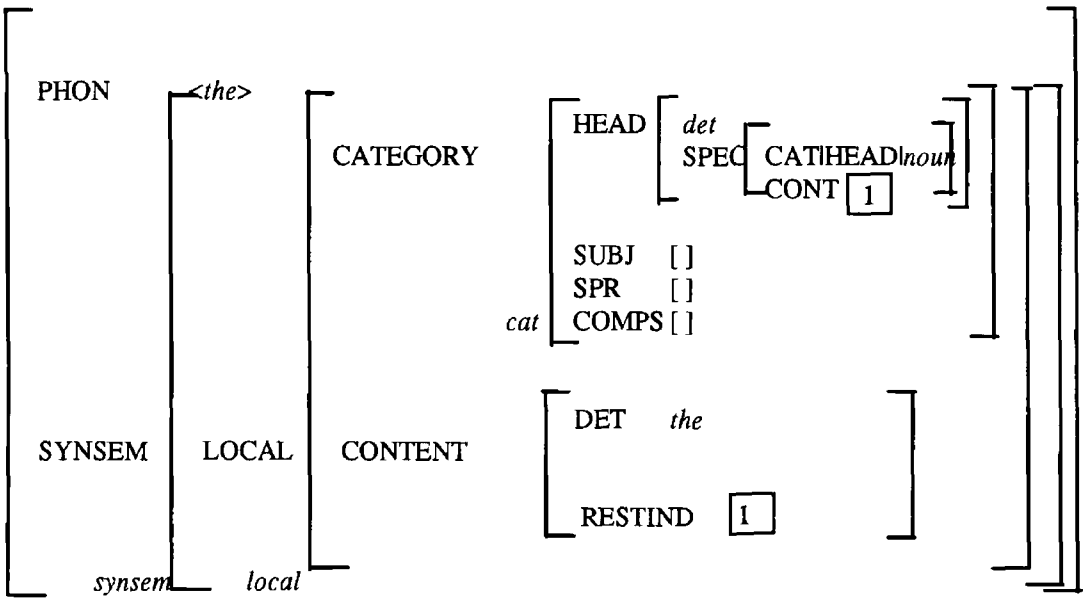
4.4.2.2 Verb Entries

The most important specifications in any verb entry are the values assigned to the SUBJ and COMPS attributes, as these will also be referenced in the semantic and contextual information. The values assigned to these will obviously depend on the kind of verb in question. For an intransitive verb, such as *snore* or *walk*, the complement list will be empty, and its subject will be a nominative noun phrase. A transitive verb, on the other hand, should have one element in its

complement list, namely an accusative noun phrase (sometimes dative in languages with strong case marking In German, for example, the transitive noun *helfen* takes a dative object), while a ditransitive verb, of which *give* is an example, will subcategorise for three noun phrases, the nominative one of which is assigned to the SUBJ feature, the remainder forming the complement list



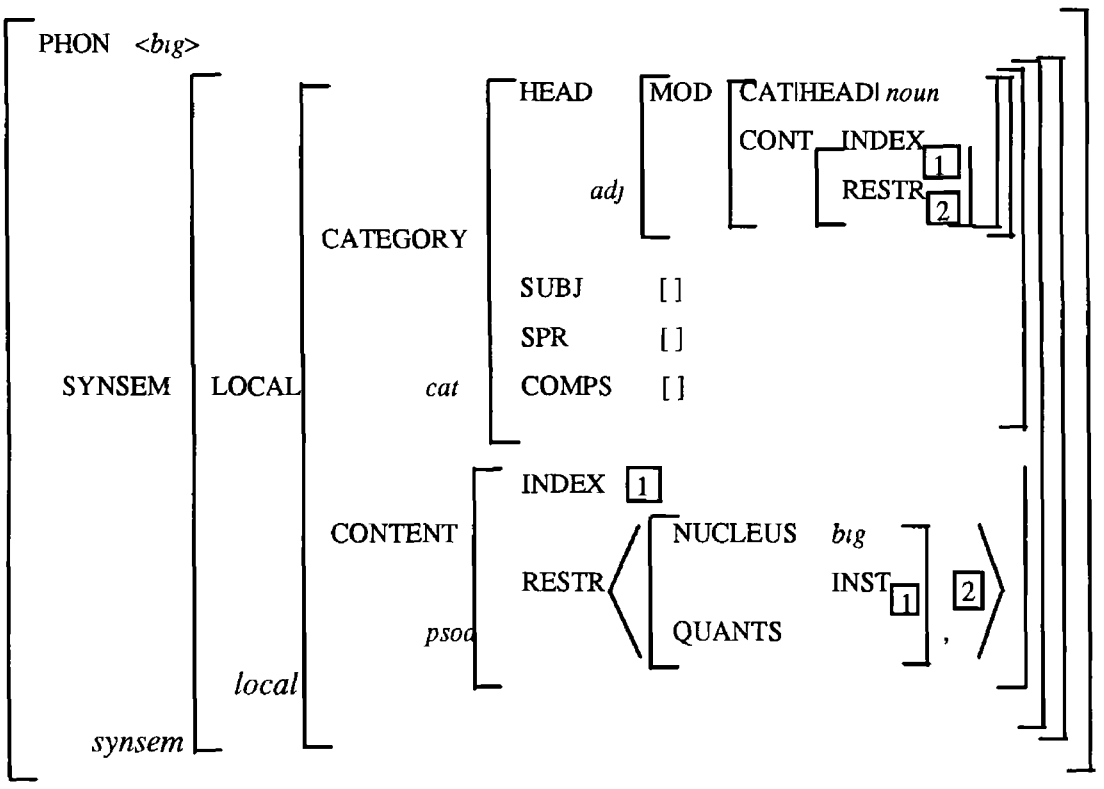
This distinction between classes of verbs is also important when assigning roles in the CONTENT part of the specification Looking at the lexical entry for *like* below, it should be observed that the value of the *liker* in the *like* relation is identical to that of the syntactic subject, while the *likee* is constantiated with the single element of the verb's complement list Auxiliary and control verbs simply pick up the semantics of their complement verb phrases (Note that in the lexical entry for *like*, the noun phrases have been abbreviated rather than including their entire synsem values)



4 4 2 3 Determiners and Adjuncts

The important thing to notice about lexical entries for determiners is the value of the entity they specify, as this restricts the way in which the determiner behaves For this purpose a feature specific to words of type *functional* is introduced, namely SPEC, the value of which is a synsem structure corresponding to a common noun This is most notable in the semantic value, where the index which is restricted by the type of the determiner (specified in DET) is reentrant with the content value of the noun A determiner can be one of three types, *forall, exists* or *the* How exactly determiners and nouns mutually restrict each other and form a noun phrase shall become clearer when the manner in which they are combined is introduced in the section on HPSG's Immediate Dominance Schemata

In addition to those features already presented, a further attribute is introduced by words of type *adjunct* This feature is given the name MOD and its value corresponds to the entity which an adjunct modifies Again, this is most important in the CONTENT value, where the proposition arising from the use of



the adjunct must be combined with the semantic information contained in the noun Hence, there are two elements in the set value of RESTR, the second of which is identical to the content of the noun specified by the MOD feature This notion can be understood more clearly by examining the lexical entry for the word *big* An actual instance of this combination of propositions can be seen later in the chapter

Now that the principal features of HPSG have been introduced, and the use of these attributes and values have been shown in sample lexical entries, it is necessary to look at the way in which these entries can be combined into phrases and sentences This is done via two more sets of constraints which, in conjunction with the constraints imposed by the lexical specifications of the elements of a phrasal sign, serve to ensure well-formedness

4.5 HPSG's Universal Grammar

A candidate phrase is well-formed if it satisfies all the principles of grammar, both universal and language-specific [PoSa94] Phrases in HPSG have an additional DAUGHTERS feature (abbreviated as DTRS) whose value is of sort *constituent-structure* This value represents the immediate constituent structure of the phrase *Con-struc* has several subsorts which depend on the kinds of daughters that appear in them. The one which is mostly used is the sort *headed-structure (head-struc)* which is appropriate for all headed structure Attributes which can appear in a structure of this type include HEAD-DAUGHTER (HEAD-DTR), SUBJECT-DAUGHTER (SUBJ-DTR), COMPLEMENT-DAUGHTERS (COMP-DTRS), ADJUNCT-DAUGHTER (ADJ-DTR), FILLER-DAUGHTER (FILLER-DTR), SPECIFIER-DAUGHER (SPR-DTR) and MARKER-DAUGHTER (MARK-DTR) It is, however, only the first three which always appear The remainder of the daughters are used mostly in more specific subsorts of head-struc

Before discussing HPSG's Immediate Dominance Schemata (which correspond for the most part to grammar rules), it is necessary firstly to acquaint the reader with some of the universal principles which must be adhered to when employing a grammar These principles ensure that the grammar of a phrase is truly projected from its lexical head While there are quite a number of these principles, those relating to nonlocal structures will not be discussed here as they are not used in the implementation in chapter 5 However, a full listing of all of HPSG's universal principles can be found in the Appendix B 1

4 5 1 Universal Principles

The principles which are of importance for this particular discussion include the Head Feature Principle, the Valence (or Subcategorisation) Principle, the Quantifier Inheritance Principle, and the Semantics and Context Consistency Principles Each of these will now be discussed in turn, presenting their formal definitions along with an explanation of their functions Where relevant, the linguistic theory from which the principle was 'borrowed' or adapted will also be

indicated, continuing the expression of HPSG's debt of honour begun in chapter 3

4 5.1 1 Head Feature and Valence Principles

Two of the most important principles of HPSG are the Head Feature and the Subcategorisation Principles (HFP and SubcatP respectively) The combined effect of these two principles is comparable to GB's Projection Principle in that they adopt basic X-bar syntax The former of these, the HFP, is formulated in such a way as to ensure that phrases really are projections of their head constituents For translation purposes, this is particularly important as in the case of sentences, it relays information about tense, voice etc , details which must be preserved in the target language if the correct translation is to be produced In the case of noun phrase, it ensures that the mother inherits case, gender and number details from the head noun The HFP guarantees that verb phrases and sentences are verbal, noun phrases are nominal, prepositional phrases are headed by prepositions, and so on The principle is formally expressed as follows

HEAD FEATURE PRINCIPLE (HFP)
The HEAD value of any headed phrase is structure-shared with the HEAD value of the head daughter

What this states is that the head of a phrase is token identical to its head daughter So, for example, the head of a sentence like *Kim likes Sandy* inherits all the details of the verb *likes*, while a noun phrase such as *the book* is assigned its case value according to the corresponding value on the specification for *book*

As already indicated, subcategorisation is useful in solving many of the syntactic issues of concern for the system to be presented The original subcategorisation principle which stated that the SUBCAT value of a head daughter is the concatenation of the phrase's SUBCAT list with the list of SYNSEM values of the complement daughters, has been replaced by the Valence

Principle due to the additional SUBJ feature The formulation of this principle is as follows

VALENCE PRINCIPLE In a headed phrase, for each valence feature F, the F value of the head daughter is the concatenation of the phrase's F value with the list of SYNSEM values of the F-DTRS value

The valence features indicated are SUBJ, COMPS and SPR What this does is equivalent to cancellation in Categorial Grammar, as it 'checks off' [PoSa94] the subcategorisation requirements of a lexical head as specified in its dictionary entry For a verb phrase, for example, it makes sure that its subject and the correct number of complements are present, for a noun phrase containing a common noun, it checks for the presence of a determiner, as stipulated by the noun's SPR attribute This is also similar to completeness and coherence checks in Lexical-Functional Grammar

4 5 1 2 Quantifier Inheritance Principle

While quantifiers are generally an issue worth discussing in detail, in its current form, they are not particularly an aspect addressed in the implementation Hence, their only interest here is for storage of the determiners *the* and *a* However, one example of how quantifiers are retrieved is shown to illustrate the function of the quantifier inheritance principle

An assumption made in HPSG's use of Cooper Storage is that every quantifier starts out in storage, and its final scope depends on the node at which it is retrieved and the order in which that retrieval takes place with respect to other quantifiers retrieved at the same node In other words, quantifiers are passed from constituents to their mothers according to the Quantifier Inheritance Principle, as specified below

The QSTORE value is passed from daughter to mother up along a tree-like path until an element thereof is retrieved, at which point it is passed to the

QRETR attribute This can perhaps be clarified somewhat by looking at the tree diagram in Figure 4 7 (Tree notation is adopted simply as a way of highlighting the idea of 'passing')

QUANTIFIER INHERITANCE PRINCIPLE

In a headed phrase, the RETRIEVED (QRETR) value is a list whose set of elements forms a subset of the union of the QUANTIFIER-STOREs (QSTORE) of the daughters, and is nonempty only if the CONTENT of the semantic head is of sort *psoa*, and the QSTORE value is the relative completement of the RETRIEVED value

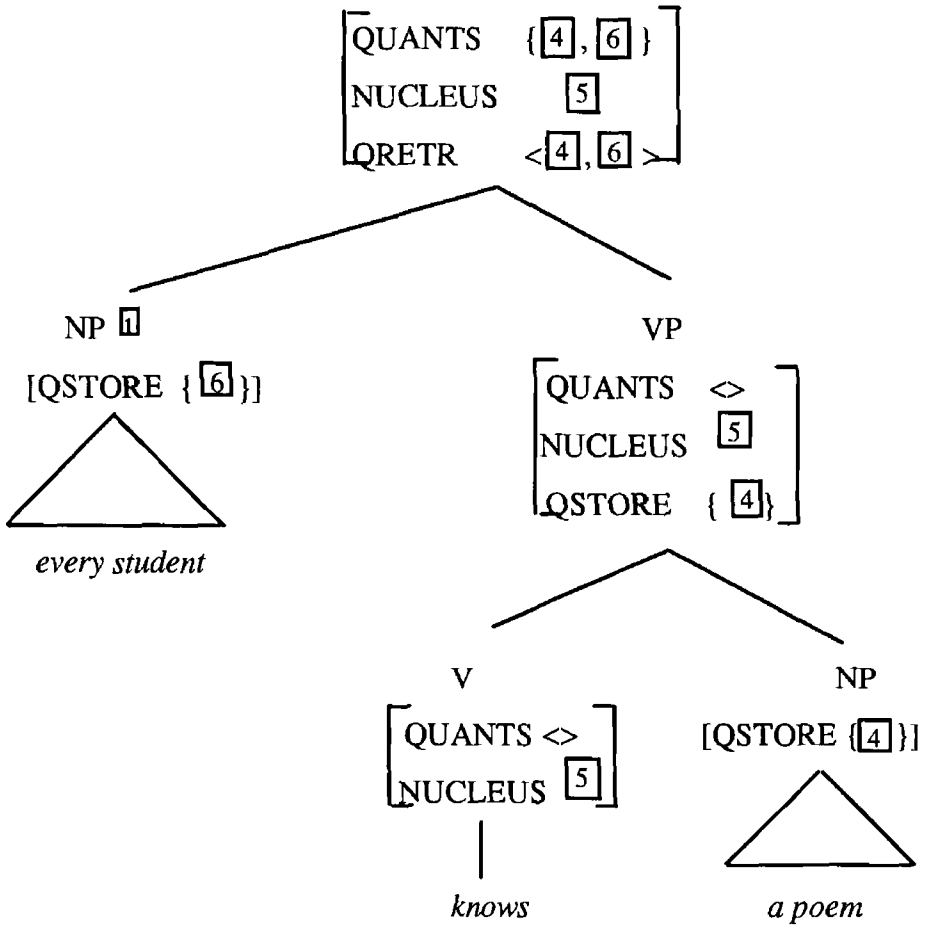


Figure 4 7

Starting at the bottom node and working upwards, the value of QSTORE is ($\exists x$|{poem(x)}), corresponding to the boxed integer, 4, and is passed upwards

through the tree until it is eventually retrieved at the top node Also retrieved at this node is the quantifier corresponding to (∀x|{student(x)}), which began in storage at the bottom left-most node Hence the meaning of the sentence is considered to be equal to the proposition (∃x2|{poem(x)}) (∀x1|{student(x)}) know(x1,x2), which reads *there exists some entity x2 such that x2 is a poem which every x1 such that x1 is a student knows*

4 5 1 3 The Semantics and Contextual Consistency Principles

These two principles combined take care of the consistency of the semantic and contextual information of a phrase The Contextual Consistency Principle is quite simply expressed as

PRINCIPLE OF CONTEXTUAL CONSISTENCY
The CONTEXT | BACKGROUND value of a given phrase is the union of the CONTEXT | BACKGROUND values of the daughters

In other words, the CONTEXT value of a sentence such as *Kim walks* would simply be the CONTEXT value of *Kim*, as this value for the verb *walks* is an empty list This is significant for chapter 5 in that it ensures the correct passing of proper noun information to the mother node, hence guaranteeing that the correct information is extracted in translation

The Semantics Principle, however, is somewhat more complicated, being derived from the definition of the Quantification Inheritance Principle, as well as the more general rule that the CONTENT value of a mother is token-identical to that of the head daughter, except where the non-head daughter is an adjunct In the latter case, the CONTENT value comes from the adjunct daughter The full specification for the semantics principle is shown below This is possibly the most important principle for this thesis, as it ensures the correct semantic information is available for the transfer component of the translation system in the following chapter

As already indicated the semantic head is the adjunct daughter if any, and the head daughter otherwise If the content attribute is not of type *psoa* then the assignment of semantic information is easy in that the phrase's QRETR value is an empty list and its content is structure shared with that of its semantic head On the other hand, however, if the value is of type *psoa*, the Quantifier Inheritance Principle is invoked (part (b) of Semantics Principle) The nucleus of the sentence is still unified with that of the semantic head, but the resolution of the semantics of the quantifiers is somewhat more involved The value of the QUANTS attribute is the result of combining its QRETR value with the QUANTS value of its semantic head

THE SEMANTICS PRINCIPLE

(a) In a headed phrase, the QRETR value is a list whose set of elements is disjoint from the QSTORE value set, and the union of those two sets is the union of the QSTORE values of the daughters,

(b) If the semantic head's SYNSEM | LOCAL | CONTENT value is of sort *psoa*, then the SYNSEM | LOCAL | CONTENT | NUCLEUS value is token-identical with that of the semantic head, and the SYNSEM | LOCAL | CONTENT | QUANTS value is the concatenation of the QRETR value and the semantic head's SYNSEM | LOCAL | CONTENT | QUANTS value, otherwise the QRETR value is the empty list, and the SYNSEM | LOCAL | CONTENT value is token identical with that of the semantic head

Now that some of the most critical universal principles for the implementation in chapter 5 have been introduced, the obvious topic to follow is the discussion and clarification of HPSG's ID-schema

4 5 2 Immediate Dominance Schemata

To take the definition given in [PoSa94], Immediate Dominance Schemata in HPSG are a small, universally available set of disjunctive constraints on the immediate constituency of phrases, from among which each language makes a selection They are universal in that they provide partial information about a

universal set of phrase types and, in effect, correspond to descriptions of the families of phrases permissible in a particular grammar Not all languages make equal use of a schema Schema 3, for example, would be used far more in Verb Subject Order languages such as Irish or Welsh, than in languages where this is used only for inverted clauses in questions ID-schemata can also be referred to as phrase structure rules without linear precedence constraints

Another universal principle, which was omitted from the discussion in the previous section on the grounds that its introduction fitted more easily into this topic area, is the Immediate Dominance Principle There is little need to explain it as it simply states

IMMEDIATE DOMINANCE PRINCIPLE Every headed phrase must satisfy at least one of the ID schemata

There are six ID schemata in HPSG, each of which will now be presented with detailed examples to illustrate their function

4 5 2 1 SCHEMA 1 (HEAD-SUBJECT SCHEMA)

Schema 1, also know as the head-subject schema, licences fully saturated phrases with a phrasal head daughter and one other daughter, a subject-daughter As a result of the Valence Principle, the SYNSEM value of this daughter will be token identical to the SUBJ value of the head daughter This schema subsumes the standard rewrite rule S -> NP VP, and is formally expressed as

(SCHEMA 1) a phrase with DTRS value of sort *head-subj-struc* in which the HEAD- DTR is a phrasal sign

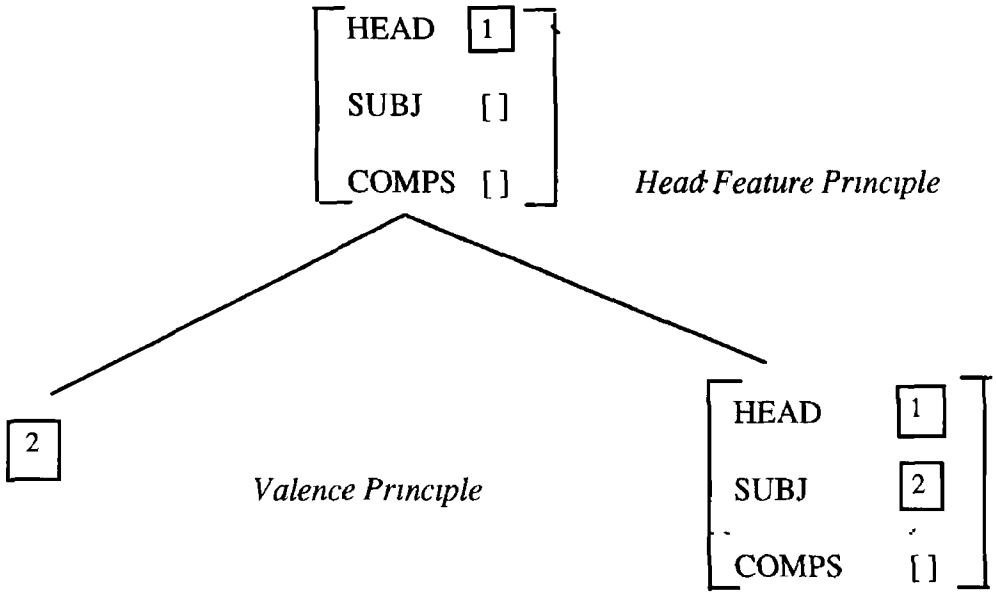


Figure 4 8

The sort *head-subj-struc* is a subsort of *head-struc* It ensures that the SUBJ-DTR is a list of length one and its COMP-DTRS feature is an empty list Figure 4 8 shows how this schema works, and the effects of the principles involved are also indicated An example of a sign governed by this schema is *Kim snores* or *Kim likes the book*

4 5 2 2 SCHEMA 2 (HEAD-COMPLEMENT SCHEMA)

Schema 2 can be considered as a rule which subsumes those rules which generally expand a phrase as a lexical head with its non-subject complements In other words, rules such as VP -> V, VP -> V NP and VP -> V NP NP are replaced by the head-complement schema Stated formally, the head-complement schema is

(SCHEMA 2) a phrase with DTRS value of sort *head-comp-struc* in which the HEAD-DTR is a lexical sign

So, schema 2 licenses those phrases whose subcategorisation requirements have all been satisfied with the exception of its subject, as in Figure 4 9 A phrase governed by this schema might be *likes sandy*, or *gives Sandy the book*

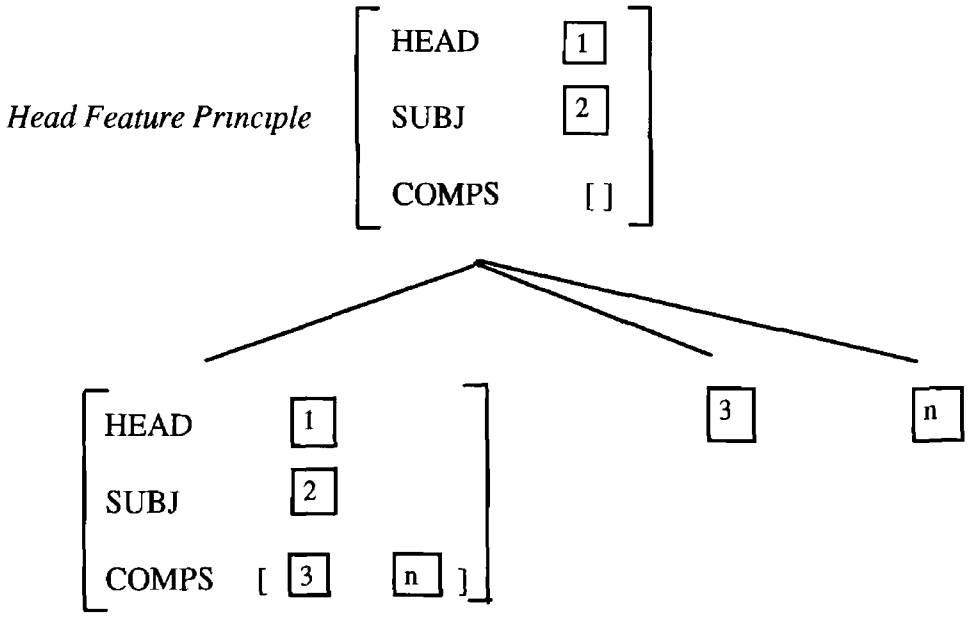


Figure 4 9

4 5 2 3 SCHEMA 3 (HEAD-SUBJECT-COMPLEMENT SCHEMA)

Schema 3 licences phrases in which all complements of a lexical head, including the subject, are realised as its sisters A new subsort of *head-struc* is introduced by the schema, namely *head-subj-comp-struc*, whose SUBJ-DTR is list of length one and whose COMP-DTRS value is a list, possibly empty

(SCHEMA 3) A [SUBJ <>] phrase with DTRS value of sort *head-subj-comp-struc* in which the head daughter is a lexical sign

The licenses clausal structures in free constituent order languages (Warlpiri), and English 'subject-auxiliary flip' constructions It can also be used in verb subject order languages, for example, Welsh and Irish The tree produced by the schema for the sentence *can Kim go?* is shown in Figure 4 10 Note that some abbreviations are used here, particularly in the specification for the verb *can* - this simply states that it is an finite, auxiliary verb in an inverted position

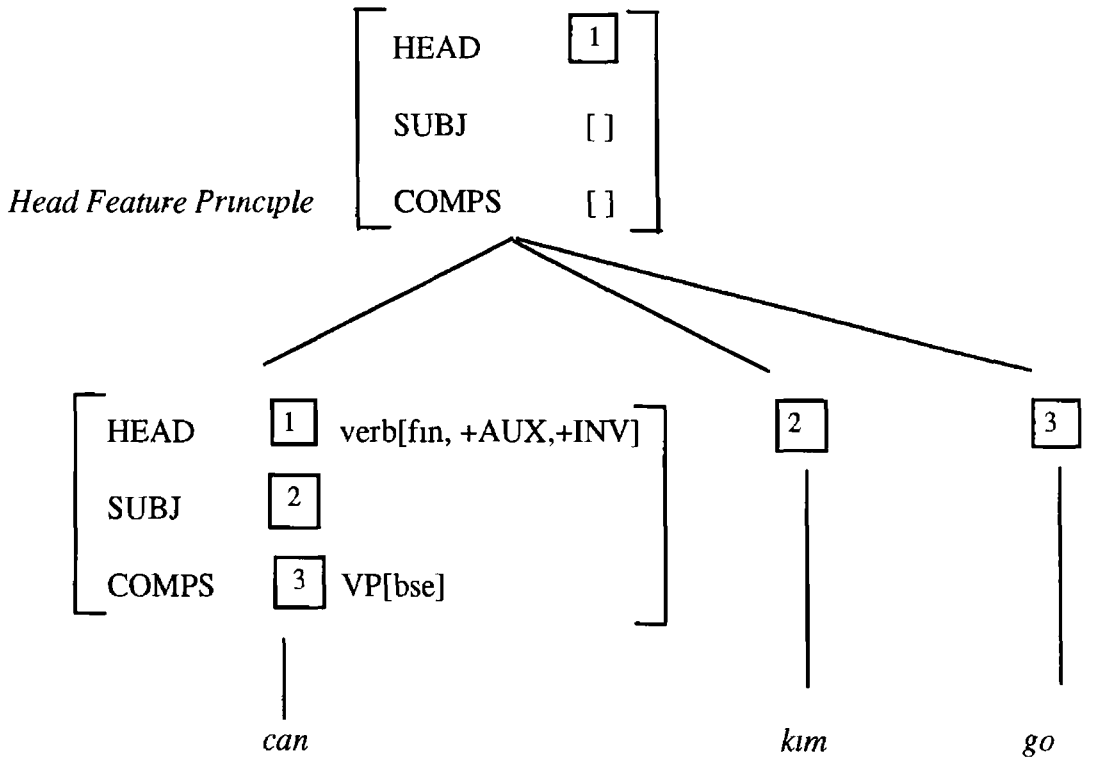


Figure 4 10

4 5 2 4 SCHEMA 4 (SPECIFIER-HEAD SCHEMA)

This schema is responsible for ensuring that a specifier and that entity which it specifies mutually restrict each other This is accomplished by enforcing reentrancy to hold between the value of the lexical head and the value of the SPEC attribute on the specifier daughter This can be written in X'-theory notation as

X" -> [1] Y" [SPEC [2]], [2] X'[SPR <[1] >]

where X" is a specifier-saturated phrase ([SPR <>]), and X' is a phrase with a missing specifier ([SPR<Y">]) The schema can be formalised as

(SCHEMA 4) a phrase whose HEAD-DTR is a lexical head, the SPR value of which is structure shared with the phrase's non-head daughter, SPR-DTR

The schema subsumes general determiner noun rules for noun phrases amongst other things Its function can perhaps be made clearer by looking at Figure 4 11 below, bearing in mind that it allows phrases such as *the book* or *a girl*

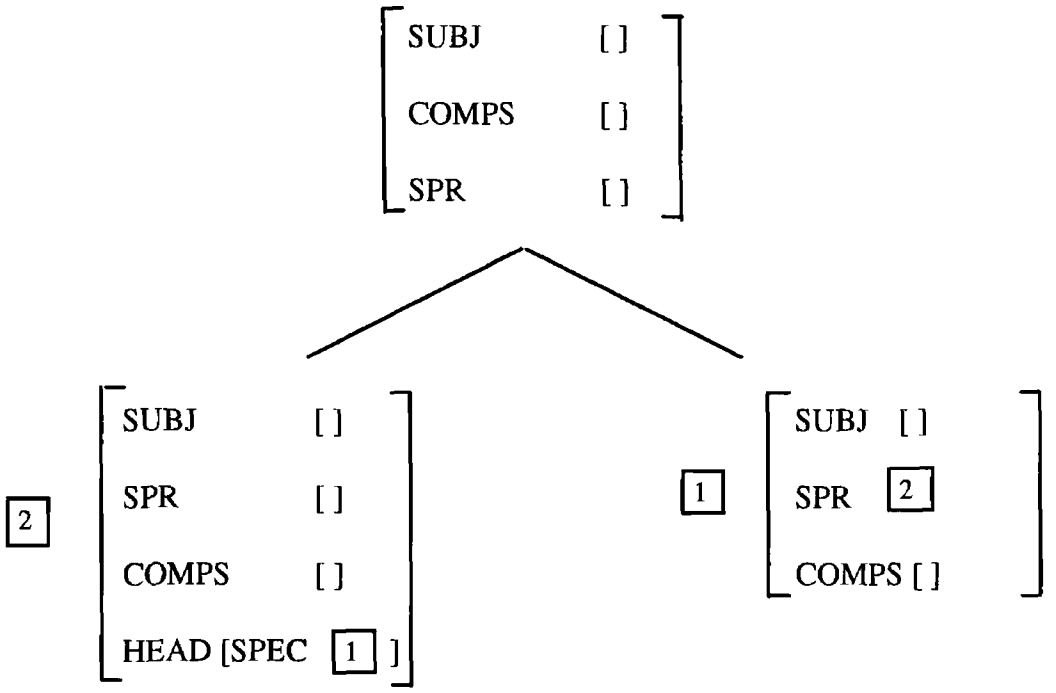


Figure 4 11

4 5 2.5 SCHEMA 5 (ADJUNCT-HEAD SCHEMA)

Schema 5 introduces yet another subsort of type *head-struc* In this case the type is *head-adjunct-struc*, and the appropriateness conditions imposed on a structure of this type guarantee that the content value of the mother is token identical to that of the adjunct daughter As already indicated in the section on the lexical entries of adjuncts, the content of the phrase's head daughter is incorporated into that of the adjunct daughter The stance taken by HPSG with regard to adjuncts is similar to that of Categorial Grammar[2] Thus, an adjunct and the head it selects can combine in accordance with the head-adjunct schema as specified below Its exact function can be seen in Figure 4 12

(SCHEMA 5) a phrase with DTRS value of sort *head-adjunct-struc*, such that the MOD value of the adjunct daughter is token-identical to the SYNSEM value of the head daughter

[2]In Categorial grammar, adjuncts are treated as functions which take heads as arguments

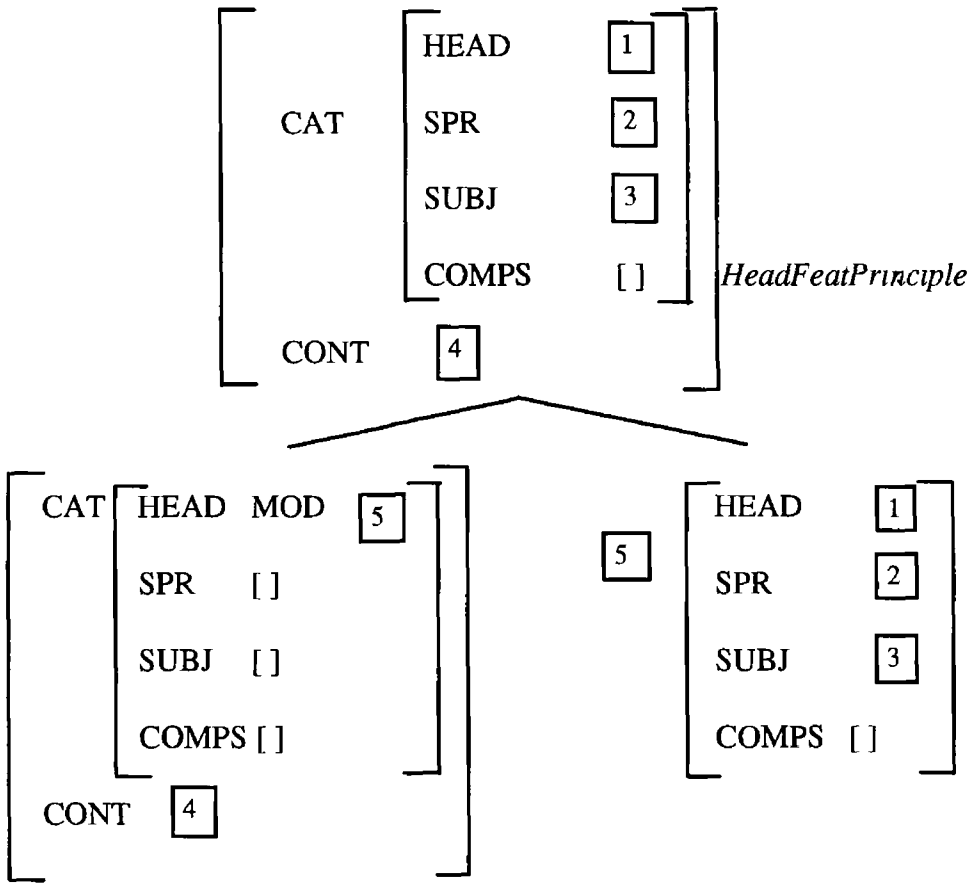


Figure 4 12

4 5 2.6 SCHEMA 6 (HEAD-MARKER-SCHEMA)

A feature which was overlooked in previous discussion in this chapter is now introduced The attribute MARKING is associated with sign's whose category is functional as opposed to substantive The distinction between substantive and functional is determined on the grounds of semantic content In the case of functional words, such as *that, for, than* etc , their meaning is purely logical in nature, i e their function is entirely syntactic, their semantics making no contribution to the overall meaning of a phrase Constituents which contain a marker daughter inherit their MARKING value from that daughter The SPEC value of a funtional word is constantiated with the SYNSEM value of the head sign with which it combines in order to form aphrase Markers do in fact resemble heads in that they select the phrases they mark The complementiser *that*, for example, requires a sentence whose head is a finite or base verb

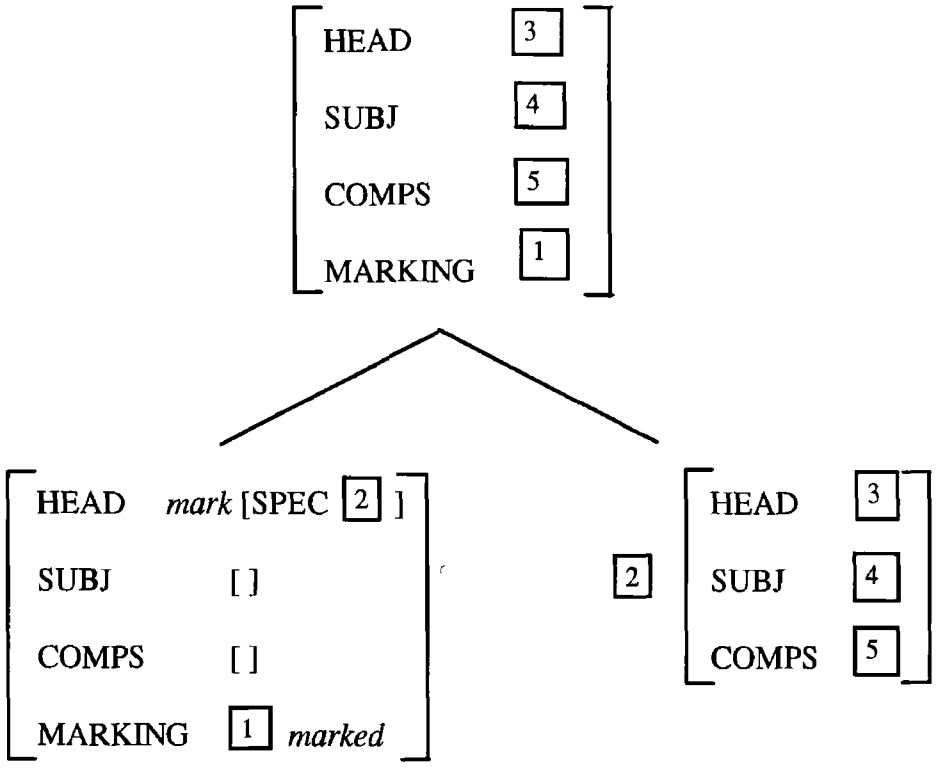


Figure 4 13

The head-marker schema adds another element to the list of subsorts of type *head-struc* This time, the type is known as *head-marker-struc* The way in which a marker combines with the necessary sign is governed by this schema which is formally stated below The schema licenses phrases of the type shown in Figure 4 13 , as in the phrase *that Kim was sick* which might occur in a sentence *Sandy knew that Kim was sick*

(SCHEMA 6) a phrase with DTRS value of sort *head-marker-struc* whose marker daughter is a marker with its MARKING value token-identical to that of the mother

There is one final ID-schema, but this deals with nonlocal constructs, and as has already been mentioned, for the purposes of this investigation, this is

currently not a factor However, it can be found in the Appendix B 2 along with a full listing of all the schemata and the universal principles For now, however, the discussion of the principal components of HPSG - the AVM modelling domain, unification and subsumption, highly articulated lexical entries, universal principles and ID schemata, is completed and the focus now turns to semantics

4.6 An Augmented Semantic Framework for HPSG

As already mentioned previously in the chapter, HPSG uses Situation Semantics for the purpose of meaning representation While this is sufficient for monolingual analysis, the purpose for which HPSG will be used in this thesis, namely machine translation, requires something more explicit Before expanding the reasons for this, and introducing the alternative method of semantic representation proposed, the principal theories behind Situation Semantics will be explained

4 6 1 Situation Semantics

Developed by Jon Barwise and Jon Perry [Barw83], situation semantics seeks to model the world according to the individuals, properties, relations and situations which exist in it It is from the interaction of constraints which hold between different kinds of conditions that the meaning of language arises The classic example of a constraint is, to quote the old proverb, *there is no smoke without fire* In other words, smoke means fire People who comprehend these constraints exploit them in order to convey meaning In [PoSa87] linguistic meaning is defined as relations of a certain kind which exist between different types of situations This definition places semantic interpretation in the real world

It is the *scheme of individuation* proposed in [Barw83] which determines the way in which the world is split into its comprehensible parts It is this scheme which accounts for objects in the world and uniformities across events and states It is thus the situation composed of individuals playing roles in a relation which forms the meaning of a straightforward, declarative sentence As has been shown already, in HPSG this is represented by the CONT attribute, whose value can be

of type *nom-obj* (individuals), *unary-reln, binary-reln, ternary-reln* etc (relations), or *atomic-prop* (property)

Every relation has an associated arity indicating the number of individuals or properties that can participate in them. A unary relation, for example, has an arity of one The ways in which individuals and properties can participate in a relation are defined by roles An *n*-arity relation thus has *n* number of roles The way in which roles are defined depends entirely on the relation in which they are involved In the sentence *Kim walks,* for example, Kim is the doer of the walking action In *Kim dies,* however, Kim has no input whatsoever into the event, as death is something which happens to him. Kim in this case is an *experiencer* In the sentence *Kim is a dog,* the canine properties can be attributed to Kim purely by virtue of his being an instance of a dog This is known as a *sortal* property The important point to note at this stage, however, is that every relation has its own list of associated roles In the lexical entry given for the verb *like* in section 4 4 2 2, for example, the roles were *liker* and *likee*

It is the proposal of this thesis, however, that these roles are too specific for the purposes of translation attempted in chapter 5, as it would require different rules for each verb Given that it is the semantics of the mother which is used to drive transfer, this is very important Given that the verb *like* has the associated roles *liker* and *likee,* while *see* has the roles *seer* and *seen,* for these two verbs alone, two different rules would be required to interpret their semantics during transfer However, if the roles agent and patient were used for both, this number is obviously reduced Also, the absence of any notion of semantic features is a downfall, given their significance in distinguishing between alternate interpretations As indicated in chapter 2, when discussing some of the problems encountered by MT, semantic features are often the simplest way to overcome these problems However, not all of the theories of situation semantics should be abandoned The representation of states of affairs (introduced in section 4 4 1 2) could be very useful in MT, simply because it can be preserved in the target language, and is guaranteed to present an accurate meaning analysis, up to a point It is this failure of situation semantics to fully present an accurate meaning

representation that has prompted the author to develop an augmented version to be used in the implementation in chapter 5 This version incorporates the notion of states of affairs with some of the ideas from *case grammars* [Fill68] Hence it is necessary to have some understanding of this latter theory for the representation of meaning before proceeding to chapter 5

4 6 2 Case Grammars

The important proposal made in connection with case grammars is that *case form* (such as noun and article endings) is merely an expression in a particular language of the underlying *case function* The sentences below, shown in English, Russian and German, help to illustrate this point (Examples taken from [Char76])

(1) Peter hit the boy
(2) Peter udaril molodoycika
(3) Peter schlug den Jungen

In each of these *Peter* is the agent of the hitting action, while *the boy* is the patient However, the case function of the latter is expressed by three different case forms While *the boy* occurs in the postverbal position in each of the sentences, in English which has a relatively impoverished morphology, it is this position which determines case, in Russian the case form is identified by the genitive noun ending, while in German, it is the accusative noun ending which performs this role However, if the order of the words is changed around somewhat, as in sentences (4)-(6), the meaning is preserved in the German and Russian translations However, in English, given that case form is represented by the sentential position of nouns, the participants have swapped roles *The boy* is now the agent of the action, while *Peter* has become the patient These examples justify Fillmore's proposal for differentiating between case form and function in order to specify a semantic structure

(4) The boy hit Peter
(5) Molodoycika udaril Petr

(6) Den Jungen schlug Peter

Fillmore's reasoning for the insufficiency of the traditional roles subject and object for specifying such a semantic structure are also worth noting[3]. The sample sentences below, also taken from [Char76], highlight the fact that, for the same relation, the case form, *subject,* can be filled by three different case functions, agent, instrument and experiencer.

(7) Peter broke the window with the hammer
(8) The hammer broke the window
(9) The window broke

Hence, the notion of case frames was developed (not to be confused with Minsky's frames for knowledge representation). Each verb has an associated frame which lists the cases by which it should or can be accompanied. A list of some of the cases proposed by Fillmore, and, for the most part, used in the implementation in chapter 5, are shown in Figure 4.14. While the compilation of a definitive list of cases or roles is an ongoing debate in the area of linguistics, it is generally accepted that agent, patient and instrument at least, are common to most formalisms using the notion of roles. An example frame is shown in Figure 4.15. One criticism which has often been levied at this approach is that it does not account for subtle differences between verbs such as those appearing in this sample frame. However, it does account for the large array of similarities between them and greatly reduces the complexity of declarations for computational purposes, a consideration which is particularly important when working within the boundaries of a grammar formalism such as HPSG, which imposes a rather heavy load on computation. However, this issue will be addressed further in the following chapter. Nonetheless, this issue should be dealt with. One suggestion is to include semantic features, an argument which was not expressed in Fillmore's essay. This will obviously necessitate the subdivision of some frames, but still

[3] Although it should also be mentioned that more recently this has not particularly been an option in most linguistic theories, the roles of subject and object being primarily restricted to syntactic analysis

maintains a connection between those which are similar in nature The alternative to the frame in Figure 4 15 is shown in Figure 4 16, where the frame has been split up into two separate tables, the first of which caters for sentences which contain one of the verbs *kill, hurt, hit,* while the second is responsible for extracting the semantic information from sentences which include *smash* and *break* Note the inclusion of the features *breakable* and *animate* which were absent before There is, of course, no limit on the number of features which can be included, but it is generally just those which are deemed necessary for correct interpretation by experimental evidence While the exact manner in which the new semantic formalism can be worked into HPSG's linguistic theory will be shown in the next chapter, the general idea is that each verb has an associated frame with a number of slots Each slot specifies a semantically motivated relation between the verb's action and one of its necessary complements As indicated, the relation is one of agent, patient etc In chapter five, a number of frame types are introduced and the implementation of each of these stipulates the properties of those entities which can fill its slots For example, a verb of the user-defined *managing* frame type, requires that its patient slot or role have the property *business* associated with it Optional slots for sentences such as *the man saw the girl with the telescope* can also be included and thus filled if they occur Default values are also possible as in *the dog bit the man with his teeth*

The assignment of roles governed by a principle action or verb is an idea used extensively in linguistic analysis and artificial intelligence Schank's conceptual dependencies, for example, make use of a principle action to determine the functionality of participants in that action ([Sch72]) Schank and Abelson's SAM program for answering questions about short stories also made use of the slot-filling technique and Minsky's frames, *"a data structure for representing a stereotypical situation like going to a child's birthday party"* ([Mins81]) These also use semantic features to limit the entities which can take on the roles Simmons's tree system ([Char76]) represents sentences in terms of the relation between the concept proposed by the main verb and the participants

Primitive	Example	Meaning
AGENT	JOHN loves Mary	causative agent, instigator of event
PATIENT	John loves MARY	target of some action, entity which undergoes the effect of an action
INSTRUMENT	Jon killed Tom with a GUN	immediate physical cause of an action
RESULT	Tom made a PLANE	entity which came into existence because of action
GOAL	Mary went to PARIS	target location of some movement
SOURCE	Tom came from DUBLIN to Paris	the place from which something moves
EXPERIENCER	TOM died	

Figure 4 14

Case-frame 1

Verbs kill, hit, smash, break, hurt
sem-agent = syn-subject if (animate)(active voice)
else = by <agent> (passive voice)
sem-experiencer = syn-object (active voice)
else = syn-subject (passive voice)
sem-instrument = syn-subject (if not animate)
else = with <instrument>

Figure 4 15

Case-frame 2

Verbs kill, hurt, hit
sem_agent = syn_subject if (animate)(active voice)
else = by <agent> (passive voice)
sem-patient = syn-object if (animate)(active voice)
else = syn-subject (passive voice)
sem-instrument = syn-subject (if not animate)
else = with <instrument>

Case-frame 3

Verbs smash, break
sem-agent = syn-subject if (animate)(active voice)
else = by <agent> (passive voice)
sem-patient = syn-object if (breakable)
else = syn-subject (passive voice)
sem-instrument = syn-subject (if not animate)
else = with <instrument>

Figure 4 16

4.7 Conclusion

The focus of this chapter has been the introduction of the principle features of HPSG, concentrating on those components from which its constraints arise, specifically, the lexical entries, universal principles and ID-schemata Moreover, while acknowledging the advantageous ideas of situation semantics, an extended version of Fillmore's case grammars was presented with the intention of incorporating this into the already existing semantic analysis constituent of HPSG In the next chapter, the MT system designed using this semantically augmented version of HPSG is discussed, highlighting in particular, those areas of translation for which this formalism is suited Also introduced is the ALE platform, the Prolog environment intended for the implementation of HPSG grammars It is in this environment that the system was designed

Chapter 5
Implementation

5.1 Introduction

The aim of the implementation was to test the suitability of HPSG for the purposes of Machine Translation However, its situation semantics component was deemed insufficient for this task, so an augmented version was composed, extending the theory with some of the ideas proposed by Charles Fillmore in his essay "Case for Case" [Fil168]

The system consists of two grammars, for German and English, each of which was designed on the Attribute Logic Engine (ALE) [Carp94], an environment initially designed for the development of HPSG grammars[4] An interface to the system allows the user to specify the source language A monolingual analysis of a sentence in this language is then output by ALE This in turn is reduced so that only a number of the features of the lexical head, including, most importantly, its semantic content, are passed to the transfer stage This is in accordance with the recommendation by Arnold et al that multidimensional formalisms be used to perform an in-depth analysis of the source language text, before limiting this information to guide the translation phase [Arn94]

Thus, the content of this chapter is concerned primarily with the details of how exactly this is achieved The details of the ALE platform must first be explained, before presenting some examples of lexical entries and phrase structure rules and illustrating how exactly these are formalised This leads into the discussion of the transfer rules which map the reduced HPSG structure for the source language into its equivalent in the target language Unfortunately, in its current state, no generation component has been implemented for the MT system, but its effectiveness can nonetheless be proven by comparing the structure produced after transfer with a monolingual structure for the same text in the target language

[4] The environment is now also capable of running grammars designed according to the formalisms of LFG and Categorial Grammar, DCGs, PATR-II, as well as Prolog, Prolog-II and LOGIN programs

5.2 Attribute Logic Engine

Designed by Bob Carpenter and Gerald Penn of Carnegie Mellon University, ALE is a system which integrates the use of phrase structure rules with constraint logic programming. Typed feature structures are expressed as terms. It is these typed feature structures which can be used to implement the type inheritance and appropriateness conditions so essential to HPSG grammar specifications, as introduced in the previous chapter. Phrase Structure rules (ID-schemata) are expressed similarly to DCG rules in that definite clause procedural attachments are allowed. These are particularly useful when ensuring the satisfaction of HPSG's universal principles (examples of their use will be shown later in the chapter). Lexical entries are developed as rewrite rules with the provision of lexical rules to enable inferences to be made about inflectional and derivational morphology, as well as greatly reducing redundancy in the system's dictionary component. Similarly, the use of macros can aid in the organisation of descriptions, particularly when a description occurs frequently throughout the grammar.

An aspect of the system which will not be discussed at any great length here, but which nonetheless is important enough to mention, is the method of parsing used in monolingual analysis. ALE employs a bottom-up active chart parser which has been specially modified to suit the requirements of attribute-value grammars in Prolog. While rules are evaluated from left to right, the chart is filled using a combination of depth- and breadth-first searches. The parser moves through the string from right to left and records all inactive edges which can possibly be generated beginning from the current left-hand position in the string in the chart. The example quoted in the ALE User's Guide [Carp94] to illustrate the procedural development of the processing is the sentence *The kid ran yesterday*. Firstly, all lexical entries for the word *yesterday* are extracted from the lexicon and entered in the chart as inactive edges. For each of these edges, the appropriate rules are also 'fired' according to the bottom-up rule of parsing. However, no active edges are recorded at this stage. The advantage of proceeding in this way is that when an active edge is eventually proposed by the bottom-up

rules, every inactive edge which it may need is already to be found in the chart Another benefit of this approach is the fact that active edges can be represented as dynamic structures In conclusion, while the overall strategy is bottom-up, breadth-first in that the string of words is processed incrementally, nevertheless, lexical entries, the bottom-up rule and active edges are all evaluated in a depth-first fashion The parser terminates once all inactive edges which can possibly be derived from the lexical entries and grammar rules have been found

5 2 1 Features Structures and Types

Given HPSG's use of strong typing, it follows that any system using this grammar formalism must be able to deal with this Fundamental to the use of ALE is an understanding of the phenomena known as *inheritance-based polymorphism*, one of the basic tools of object-oriented programming This implies that types are arranged in a hierarchy, whereby subtypes inherit all constraints imposed on their supertypes

There are three conditions which must be imposed on type specifications in ALE Firstly, only direct subtype relationships should be explicitly expressed, as typing is understood to be transitive in nature Hence if the atom b is a subtype of a, and c is in turn a subtype of b, it follows that c is also a subtype of a One constraint on this transitivity is that it must be anti-symmetric, i e a situation in which two distinct types are specified, each of which is a subtype of the other, is not allowed Thus, the inheritance hierarchy is said to form a partial order

The second condition which should be obeyed is the necessity to declare a most general type named bot Every type is a subtype of bot, and if its declaration is omitted, all other types will be considered undeclared by the ALE compiler The bot declaration in the English grammar of this system is partly described by

```
bot sub [boolean, case, cat, cont, conx, gend,
grammartype, head, ind, list, loc, marking,
mod_synsem, name, nonloc, nonlocll, num, per, pform,
```

qfpsoa, sem_det, set, sign, special, tense, vform, voice]

The function of each of these types will be discussed as they are encountered in lexical specifications Common to each of them, however, is the fact that they are the top-most subtypes of the grammar, and hence must be declared as direct subtypes of bot

The final condition imposed on the typing system is that the hierarchies be *bounded complete* In other words, every pair of types which share a common subtype must have a single, most general subtype This can perhaps be more clearly understood by examining an example violating this condition, and the way in which the declaration in this example can be altered to ensure its adherence to the rule (see Figure 5 1)

```
bot sub [a,b]                bot sub [a,b]
     a sub [c,d]                  a sub [e]
          c sub []                     e sub [c,d]
          d sub []                     c sub []
     b sub [c,d]                       d sub []
                                  b sub [e]
```

Figure 5 1

The problem with the first declaration (on the left of Figure 5 1) is that, while a and b have two common subtypes, neither of these fill the role of the most general subtype, as d is not a subtype of c and, vice versa However, by simply adding an extra type, e, in the second declaration, this problem is easily resolved as e is now the most general subtype of a and b

5 2 1 1 Feature Structures

It should be clear by now that feature structures in HPSG consist of two pieces of information, a type and a collection of feature-value pairs Thus, a further operation must be carried out by the type system, namely, it must guarantee

feature appropriateness In other words, each type includes a specification of the features for which it is defined, as well as the types of the values these features can take Another important consideration is that appropriateness conditions are also part of the inheritance scheme, in that whatever features and values are appropriate for a type, they will also be appropriate for all of its subtypes

As an example of a declaration, Figure 5 2 shows the reduced specification for a list of signs as encoded in the type system of this implementation This indicates that `list_sign` has two subtypes, namely `e_list` and `ne_list_sign` From this it can be inferred that an empty list has no associated features, as none are declared either directly or indirectly (by `list`) Neither does it have any subtypes, implying that it is an atomic type A nonempty list of signs, on the other hand, introduces two features, `hd` and `tl`, which correspond to the head and tail notation of Prolog lists `Hd` is of type `sign`, while the value of `tl` is a `list_sign` For now, it is important only to note that the subtypes of `sign` are `phrase` and `word` There are no subtypes for a nonempty list of signs

```
list_sign sub [e_list, ne_list_sign]
     e_list sub []
     ne_list_sign sub []
          intro [hd sign,
                 tl list_sign]
```

Figure 5 2

An example of how exactly the inheritance of features operates is shown in Figure 5 3 Here, a sign has two subtypes `phrase` and `word`, each of which inherit the features, `synsem`, `qretr`, `qstore`, introduced by `sign`

```
sign sub [phrase, word]
     intro [synsem synsem,
            qretr set_quant,
            qstore list_quant]
     phrase sub []
     word sub []
```

Figure 5 3

Features structures are required to be totally well-typed This implies that if a feature is defined for a particular feature structure of a given type, then that type must be appropriate for the feature Hence, a declaration such as

```
list_sign
HD   a
TL   bot
```

violates this condition as `list_sign` does not have any associated features, only `ne_list_sign` does Moreover, the value of the feature must not violate the appropriateness conditions imposed on it by the declaration Hence, the above example would be incorrect even if `list_sign` were changed, as the `TL` feature must have a value of type `list_sign`, and not `bot`

Before continuing with the next section, there are two more important restrictions which should be mentioned ALE maintains an acyclicity requirement which forbids type declarations in which a type is assigned a value which is of the same or of a more specific type Thus, specifications such as

```
gender sub [masc,fem]
       intro [woman fem,
              man masc]
```

are not admissible

The second restriction is slightly more complicated It is comparable to the bounded completeness condition on the inheritance hierarchy and is solved in much the same way Basically, every feature must be introduced at a unique, most general type, disallowing declarations like the one below

```
verb sub [trans, intrans]
        trans sub []
           intro [agent ref,
              patient ref]
        intrans sub []
            intro [agent ref]
```

This problem is resolved by introducing the feature `agent` at the `verb` level as in

```
verb sub [trans,intrans]
             intro [agent ref]
            trans sub []
              intro [patient ref]
            intrans sub []
```

A full representation of the inheritance hierarchies used in the implementation presented here can be seen in Appendix A

5 2.1 2 Macros

Macros are a useful way of defining information which is used repeatedly throughout the grammar In the lexicon in particular, the specification of lengthy paths can make entries difficult to read and understand, and hence nearly impossible to expand or develop in any way A partial specification for the verb *walks*, for example, is

```
walks ---> cat (head verb,
                  subj [cat head noun])
```

A simpler way of expressing this would be to define a macro for a noun phrase (the value of the subject feature above) This can be done as follows

```
np macro
cat (head noun,
     subj [])
```

A macro is called by placing the character '@' in front of its name Thus, whenever a call is made to the noun phrase macro (`@np`), `cat (head noun,subj [])` is substituted

Macros can contain calls to other macros, although macros cannot be recursive in that they cannot call themselves

5 2.2 The Lexicon in ALE

Having introduced the structure of lexical entries in HPSG in the previous chapter, it should not now be necessary to fully explain the nature of the lexicon in ALE However, some sample entries from both the German and the English grammars will be shown in order to reinforce the idea of the use of typing and feature structures in ALE, as well as macros Another facility provided for by ALE is a series of lexical rules which greatly reduce redundancy in the dictionary These too will be discussed in the following sections Moreover, the realisation of the proposed semantic augmentations will also be presented via examples, highlighting the use of semantic features as a way of distinguishing between semantically ambiguous words in preparation for the transfer phase of translation

5 2 2 1 Some Sample Lexical Entries

In ALE, lexical entries are specified as rewrite rules, whereby the right-hand side of the ---> is substituted for the word on the left-hand side when that word is encountered in the input string Thus the BNF notation for lexical entries is

```
<lex_entry>  = <word> ---> <desc>
```

A simple specification for the intransitive verb *walks* was shown in the previous section Read declaratively, this says that the word *walks* has as its lexical category a verb whose subject is a noun phrase As already indicated, macros are of vital importance in terms of organisation when describing words, as will be seen shortly when more detailed entries are shown as examples

5 2.2 1 1 Nominal Entries

Shown in Figure 5 4 is the lexical entry for the proper noun *kim* as it appears in both the German and English grammars In order to fully understand it, the macro `@np` must first be expanded If in doubt about the significance of any of the entry's features, chapter 4 should be consulted

```
kim --->
   word,
   synsem ((@ np((per third,num sg,gen masc),
              ([human,animate])),@ mod(none)),
           loc (cont (index Ind,
                      restr e_set,
                      properties Props),
                conx backgr (elt (nucleus (naming,
                                  bearer Ind,
                                  name kim),
                             quants []),
                     elts e_set)),
          @ no_slash),
        qstore e_set
```

Figure 5 4

The np macro is expanded according to

```
np(Ind,Props) macro
loc ((cat) ((head noun,spr [],subj [],comps [],
             marking unmarked,
      (cont) (index Ind,properties Props)
```

It is here that the first additional feature required by the new semantic formalism is introduced for the first time This does not form part of ALE's original type and attribute declaration component, but has been added by way of aiding the investigation into the suitability of using case grammars and semantic features for translation, one of the focal points of this thesis This feature is `Props` and represents the properties associated with a nominal object Notice that in the call to `@np` by *Kim*'s lexical entry, the properties are specified as human and animate Of course, this is just a general assumption made about individuals named *Kim* It could easily be the case that *Kim* is a dog, and in this case, the properties would change, perhaps to canine and animate

Other than that, the entry for *Kim* specifies that it is a lexical sign (indicated by `word` as opposed to `phrase`), whose `synsem, local` value is that of a noun phrase, `ie`, its `head` feature has the value `noun`, while its `subject, specifier` and `complements` attributes are all empty Its `index` is a referent to a third person, singular, masculine individual who is

associated with the name *Kim* courtesy of a naming relation It has no quantifiers, nor does it impose any semantic restrictions Note that the index and property values are instantiated by those values passed to the np macro The @noslash macro indicates that there is no nonlocal structure to be considered All of this information in completely in accordance with the specifications and conditions for lexical entries as laid out in the previous chapter

The entry for *book* in Figure 5 5 is somewhat different to that of the proper noun Again, it is a lexical sign, but it calls the macro @nbar (after X'-theory), which is expanded as follows

```
nbar(Ind,Props) macro
loc ((cat) (head noun,spr [_],subj [],comps [],
            marking unmarked),
      (cont) (index Ind,properties Props))
```

```
book --->
word,
synsem ((@ mod(none),
          @ nbar((per third,num sg,gen neut),
          ([inanimate,readable,pages])),
          @ spr([(@ detp(_))]),
          @ cont((npro,
                  index Ind,
                  restr (elt (nucleus (book,
instance Ind),
                                      quants []),
                              elts e_set),
                  properties Props)),
          loc conx backgr e_set,
          @ no_slash)),
    qstore e_set
```

Figure 5 5

The principal difference between this macro and that for noun phrases is that the value of the SPR feature is not an empty list, rather it is specified by two further calls to the macros @spr and @detp These macros are respectively

```
spr(Spr) macro
      loc ((cat) spr Spr)
```

```
detp(Cont) macro
loc ((cat) (head det,spr [],subj [],comps [],
            marking unmarked),
      cont Cont)
```

Other than this, the entry for *book* should be relatively straightforward, merely indicating that it is third person, singular and neuter and it has the properties of being inanimate, readable and of having pages These properties have been chosen in accordance with the distinguishing characteristics as specified by their definitions in [Collins] and [Oxford] It has no contextual information associated with it Prolog variables are used to indicate structure sharing (note the use of the variable `Ind`) The `elt` and `elts` (which stand for element and elements, respectively) features which appear as values of `restr` are to sets what the features `hd` and `tl` are to lists

Many of the nouns which appear in the lexicon were chosen completely at random. Others which exhibit particular properties were necessary to test the functionality of verbal entries These will be indicated where necessary

5 2 2 1.2 Verb Entries

It is in this section that the remainder of the new semantics will be introduced as it is the verbs which *"provide templates within which the remainder of the sentence can be understood"* (Charles Fillmore, quoted in [Barr81]) In addition, some extra features beyond those proposed for verb entries in [PoSa94], have been added to the specifications for the purposes of translation

The first additional feature has been added as part of the lexical sign's syntactic specification, although its function is entirely semantic in nature This is the CASEFRAME attribute This feature can take a number of values, each of which **describe the context in which the verb can occur**, i e the properties of its surrounding noun phrases Take, for example, the first lexical entry shown for *run* in Figure 5 6 The value of its `caseframe` feature is `competing`, so it can appear in a sentence such as *Kim ran the race* This is not particularly relevant

as it stands, but when used in conjunction with a lexical rule which extracts its caseframe type, it can be effective Its main task is to impose the appropriate constraints on the verb's environment, in order to discriminate between this meaning and that of the second entry for *run*, whose frame type is `managing` as in the construction *Kim runs the shop* While this is not especially significant for monolingual analysis, for translation it is crucial, as each of these meanings has a different lexical equivalent in German The former meaning translates as *rennen*, while the latter is *fuhren* The exact constraints imposed can be seen in the discussion on lexical rules in the following section

Note also that the specific roles played by the verb's subject and complements (if any) have been replaced by the more generic roles of agent, patient, instrument, goal etc, also in concurrence with Fillmore's proposals as introduced in section 4 6 2 Moreover, the properties associated with these roles are listed as part of the verb's semantics The notion of relations and states of affairs introduced by situation semantics, however, is maintained

The other new features are TNS and VOICE Pollard and Sag actually argued that these attributes can be replaced by the VFORM feature, but as this gives no details about tense (it merely says whether the verb is finite, gerundive etc), it is insufficient for translation purposes Similarly, while the VFORM attribute can take a passive value, all other information about the verb's tense and form are lost Note that in the entries shown here, the values of each of these features is unspecified This is because the tense and voice can only be identified from the actual input word in context The values are thus filled in by the lexical rules

Thus, verb entries tell us

(1) what they require in order to become saturated (subj, comps and spr features),

(2) their tense, voice and verb form,

(3) whether or not they are inverted or auxiliary

(4) their caseframe type

(5) the name of the relation (state of affairs) in which they are involved

(6) the semantic roles assigned to their complements and subject

```
run --->
   word,
   synsem (loc ((cat) (head (verb,
                       mod none,
                       vform bse,
                       aux minus,
                       inv minus,
                       caseframe competing,
                       tns notense,
                       voice novoice),
                       spr [(@comp)],
                       subj [(@ np(Ind1,Props)
                              @ case(nom))],
                       comps [(@ np(Ind2,Props2),
                                @case(acc))]
                       marking unmarked),
                  cont (nucleus (run,
                                 agent Ind1,
                                 agentprops  Props,
                                 patient Ind2,
                                 patientprops Props2),
                        quants []),
                  conx backgr e_set),
          @ no_slash),
     qstore e_set
```

Figure 5 6

5 2.2 1 3 Determiners

What is most significant about the specification of determiners in the German grammar is that there is only one entry for both indefinite and definite articles, namely *das* and *ein* Given that the form of a determiner in German depends greatly on case, gender and number, choosing the correct form can be rather a difficult task (see Figure 5 7 for full listing of German articles) However, this task is accomplished rather efficiently by a series of four lexical rules with procedural attachments for both indefinite and definite determiners These will be explained at greater length in the next section For now, the discussion is limited to the details of their lexical entries

CASE	MASC/DEF/INDEF	FEM/DEF/INDEF	NEUT/DEF/INDEF	PLUR
NOM	der/ein	die/eine	das/ein	die
ACC	den/einen	die/eine	das/ein	die
GEN	des/eines	der/einer	des/eines	der
DAT	dem/einem	der/einer	dem/einem	den

Figure 5 7

What the specification in Figure 5 8 essentially tells us, is that das is a sign whose lexical head is a determiner which specifies a noun with content Cont It has no complements or subject and it is unmarked Its content is referenced by the variable GQ The information supplied by the content structure is that *das* is a determiner of type the and its restrictive index is structure shared with the content of the noun it specifies (Cont) Its entry contains no contextual or nonlocal information However, the qstore feature is worthy of discussion here as it has not yet been encountered, save for where its value is an empty set The first element of the set is constantiated with the content of the determiner, as identified by GQ The set has no further elements

```
das --->
   word,
   synsem (loc ((cat) (head (det,
                             spec ((@ cont(Cont),
                                    @ nbar( _,_))),
                             specific def),
                       spr [],
                       subj [],
                       comps [],
                       marking unmarked),
                 cont (GQ,
                       det  the,
                       restind  Cont),
                 conx backgr e_set),
           @ no_slash),
   qstore (elt GQ,
           elts e_set)
```

Figure 5 8

CASE	MASC/DEF/INDEF	FEM/DEF/INDEF	NEUT/DEF/INDEF	PLUR
NOM	-e/-er	-e/-e	-e/-es	-en
ACC	-en/-en	-e/-e	-e/-es	-en
GEN	-en/-en	-en/-en	-en/-en	-en
DAT	-en/-en	-en/-en	-en/-en	-en

Figure 5 9

The presence of an additional feature in the head declaration should also be observed The attribute `specific` takes a value `def` or `indef`, according to whether the article is definite or indefinite This is to ensure that the correct form is chosen when using the lexical rules It is also important for determining adjective endings

5 2 2 1 4 Adjectives

Adjuncts are treated similarly to determiners in that it is the lexical rules which select the correct inflected form according to case, gender and number A full listing of the possible adjective endings for German can be seen in Figure 5 9

```
red --->
  word,
  synsem (loc (cat (head (adj,
                          prd minus,
                          mod (pre_mod_synsem,
                               @ nbar(Ind,Props),
                               @ restr(Restrs))),
                    spr [],
                    subj [],
                    comps [],
                    marking unmarked),
              cont (index Ind,
                    restr (elt (nucleus (red,

instance Ind),
                                quants []),
                           elts Restrs),
                    properties Props),
              conx backgr e_set),
          @ no_slash),
  qstore e_set
```

Figure 5 10

Obviously this is not a consideration for English as it does not have any adjective-modifer agreement

The entry for the adjective *red* is shown in Figure 5 10 and supplies the information that it is a word that modifies a noun with index Ind, properties Props and which imposes a semantic restriction Restr on the phrase in which it occurs It has empty values for the subj, comps and spr features and it is unmarked Its content value imposes two restrictions on its phrase, namely Restr and a property of *redness* Its index and additional property information is taken from that of the noun it modifies This specification is typical in every way of HPSG's theory for adjuncts, in that no modifications have been made to its semantic content (only in so far as the properties of the modified entity are added but this is a consequence of the noun's semantics rather than the adjective's), nor are there any additional features

All other lexical entries can be derived from these specifications Obviously, different features are used depending on the part of speech in question For the specification of prepositions, for example, the feature `pform` is necessary Quantifiers are declared in much the same way as the determiner entry shown, but the value of its `det` attribute is changed according to its type (`forall, exists, the`) Pronouns differ slightly from nouns in that their semantic type is `pron` as opposed to `npro`, and their contextual features point to the content index, purely for the purposes of identifying the referent for anaphora resolution Differences in verbs occur where the `caseframe` type changes according to the meaning of the verb This can be seen more clearly in the discussion on lexical rules which follows

5.2.3 Lexical Rules

From the examples already discussed with reference to verbs, determiners, nouns and adjectives, it is clear that the main function of lexical rules is to reduce redundancies in the lexicon These rules can be applied sequentially to their own output or the output from other rules to a depth determined by the user Thus, the

nominal *receiver* can be derived from the verb *receive*, and the plural *receivers* is then derivable from the nominal The idea behind specifying the depth is to limit the number of rules which can be applied to a category

However, the function of lexical rules in this implementation is twofold Not only are they used for derivational and inflectional morphology, but also for ensuring that verbs occur in the right context It was mentioned in the previous section that the caseframe feature of verbs is what distinguishes them from each other in terms of their accompanying subjects and complements It is in the lexical rules, however, that this feature is used for the purpose for which it is intended, i e to aid in the selection of a verb's translation

Firstly, however, the format of the lexical rules should be explained, taking some of the morphological rules for German as examples

5 2 3 1 Morphological Analysis

The lexical rule for the formation of present tense, third person singular verbs is shown in Figure 5 11 Following the name of the rule, the first half of the declaration is essentially a general lexical entry with variable names used for all values, except the verb form and tns and voice which are uninstantiated This is termed the *input category* The second half is the *output category* and its details are changed somewhat Most notably, the value of `tns` is now present, while `voice` has been assigned the value active and `vform` is finite Both of these descriptions may contain macros and disjunctions

However, these values of `tns`, `vform` and `voice` are changed only if the condition specified in the procedural attachment at the bottom of the rule In this example, the procedure `pres_3s_act` (shorthand for present, third person singular active and shown in Figure 5 12), takes as its input the value of the subject attribute in both the input and output categories If the subject in the latter is a noun phrase of the form third person, singular, nominative case, then the value true is returned and the value assignments are confirmed in the output category

From here, it is necessary to find the correct morphological pattern From Figure 5 11 it can be seen that these patterns are composed of variables, sequences and lists Hence, the rewrite rule can be expressed via the use of atoms (as in `geben`), lists, or sequences such as `(X,oe,Y,en)` Thus, the latter example could be interpreted as a verb containing the letters `oe` between an arbitrary number of letters signified by `X` and `Y`, but ending in en The verb *moegen* fits this description The value of `X`, in this case, would be `m` and `g` would be assigned to `Y` The morphological component of the rule for the formation of third person, single verb forms is shown in Figure 5 12

The way in which lexical rules are consulted is actually quite simple Every lexical entry is checked to see if its category unifies with the input category of a lexical rule If it does, then a new word is created which satisfies the output category In the cases of multiple solutions, every option is generated, leading to multiple lexical entries It is only then that the lexical entry of the input word is passed

```
pres_3s lex_rule
  (word,
   synsem (loc ((cat) (head (verb,
                             vform bse,
                             aux minus,
                             mod Mod,
                             inv Inv,
                             caseframe CG,
                             tns _,
                             voice _),
                        spr Spr,
                        subj [Subj],
                        comps Comps,
                        marking Marking),
                  cont Cont,
           conx Conx),
         @ noslash,
  qstore Qstore)
  **>
  (word,
   synsem (loc ((cat) (head (verb,
                             vform fin,
                             aux minus,
                             mod Mod,
                             inv Inv,
                             caseframe CG,
                             tns pres,
                             voice active),
                        spr Spr,
                        subj [NewSubj],
                        comps Comps,
                        marking Marking),
                   cont Cont,
            conx Conx),
          @ noslash
   qstore Qstore)
  if  pres_3s_act(Subj,NewSubj)
  morphs
    (X,oe,Y,en) becomes (X,a,Y),
    (X,geben) becomes (X,gibt),
    (X,nehmen) becomes (X,nimmt),
    (X,essen) becomes (X,isst),
    (X,a,Y,en) becomes (X,ae,Y,t),
    (X,ieten) becomes (X,ietet),
    (X,en) becomes (X,t)
```

Figure 5 11

```
if  pres_3s_act(Subj,NewSubj)
  morphs
     (X,y) becomes (X,i,e,s),
      X becomes (X,s)

pres_3s_act((NP,@ np(_,_)),(NP,@
np((per third,num sg),_),@ case(nom))) if
  ',true
pres_3s_act(X,X) if
  true
```

Figure 5 12

through the morphological analyser to produce the correct output This word is matched against each of the patterns on the left-hand side of the morphological rules, until a match is found Once found, the corresponding changes are made Taking the verb *moegen* as an example again, when it is unified with the left-hand side of the first rule, `oe` is changed to `a` and `[m,a,g]` is the resulting output The most general rule `X becomes X` is always last, as the rules are tested sequentially and this would otherwise satisfy all words input

Another point worth noting is that even those values which are not changed by the lexical rule (such as `Cont, Qstore` etc) must be explicitly mentioned Otherwise each input and output categories would have unconstrained values for the unmentioned features

The operation of rules dealing with articles, adjuncts and noun derivations and plurals is very similar However, as a further example, the rule for determining the form of neuter indefinite articles in the nominative or accusative case is shown in Figure 5 13 The main difference here is that the value of the `specific` attribute of the input category must be `indefinite` and the type of determiner is `exists` The procedural attachment `neut_nom_acc` takes as its input the Spec value of each category and tests to see if it is neuter, nominative or accusative, before returning a value of true or false If the value returned is false, then the next rule is tried until eventually one of them succeeds or it is determined that there is no specification for the stem of the word

```
neuter_nom_acc_indef_article lex_rule
 (word,
   synsem (loc ((cat) (head (det,
                             spec Spec,case Case,
                    specific indef),
                       spr [],
                       subj [],
                       comps [],
                       marking unmarked),
                cont (det  exists,
                      restind  Cont),
          conx Conx),
            @ no_slash),
 qstore QStore,
 qretr QRetr)
**>
(word,
   synsem (loc ((cat) (head (det,
                             spec NewSpec,
                    case Case,
                       specific indef),
                       spr [],
                       subj [],
                       comps [],
                       marking unmarked),
                cont (det  exists,
                      restind  Cont),
          conx Conx),
            @ no_slash),
  qstore QStore,
  qretr QRetr)
if neut_nom_acc(Spec,NewSpec),masc_nom(Spec,NewSpec)
morphs
X becomes X
neut_nom_acc((NBAR, @ nbar(_,_)),(NBAR,@
nbar((num sg,gen neut),_), @ case(nom),@ case(acc)))
if
   ',true
neut_nom_acc(X,X) if
   true
```

Figure 5 13

CASE FRAME	PROPERTIES	EXAMPLE
animaction	animate agent	*snore, walk, run, schnarchen*
going	animate agent, inanimate, building goal	*enter, go, gehen*
competing	animate agent competition goal	*run,enter,rennen, teilnehmen an*
managing	animate, human agent business patient	*run, manage, fuhren*
humeating	human agent food, edible patient	*eat, essen*
animeating	animal agent food, edible patient	*eat fressen*

Figure 5 14

5 2 3 2 Case Frames

The principal difference between rules intended for morphological analysis and semantic testing is the complete absence of any kind of rewrite rules The only features not assigned variable values are the case frame and content attributes The former of these is named according to the properties which appear in the procedural attachment `memeberchk` The complete list of case frames and their associated properties (selected purely for their relevance to the verbs included in the test data) are shown in Figure 5 14

In the example in Figure 5 15, the case frame is of type competing, This implies that the agent of this action (exemplified by the words *run* or *enter*) must be animate and the patient or goal must be a competition of some sort Obviously, both of the words mentioned above have more than one associated case frame The verb *run* can also mean to manage a business so the required properties (and hence the tests for the `memberchk` predicate) would change accordingly For the purposes of translation, it is imperative that different verb senses be easily told

apart This method helps to accomplish this via the use of semantic features and the type of case frame assigned to a lexical entry It also means that the duplication of lexical entries with the same spelling but different senses is eliminated as the word needs only to be represented once in the dictionary, as its

```
compete_frame lex_rule
  (word,
   synsem (loc ((cat) (head (verb,
                             vform bse,
                             aux minus,
                             mod Mod,
                             inv Inv,
                         caseframe _),
                         tns Tense,
                         voice Voice),
                         spr Spr,
                         subj ([(@ np(Ind1,Props1))]),
                         comps ([(@ np(Ind2,Props2))]),
                         marking Marking),
                 cont _),
           @ noslash,
   qstore QStore)
  **>
  (word,
   synsem (loc ((cat) (head (verb,
                             vform bse,
                             aux minus,
                             mod Mod,
                             inv Inv,
                             caseframe competing,
                         tns Tense,
                         voice Voice),
                         spr Spr,
                         subj ([(@ np(Ind1,Props1))]),
                         comps ([(@ np(Ind2,Props2))]),
                         marking Marking),
                 cont (nucleus (_,
                                agent Ind1,
                                agentprops Props1,
                                patient Ind2,

patientprops Props2))),
          @ noslash,
   qstore Qstore)

 if  memberchk(animate,Props1),
   \+memberchk(inanimate,Props1),
     memberchk(competition,Props2)
```

Figure 5 15

sense is entirely dependent on the case frame value assigned to it Of course for words such as *let* which have different senses because of the syntactic nature of their complements (to allow and to rent out), this problem of information redundancy is still an issue

5 2 4 Universal Grammar

Before discussing the format of ID-Schemata in ALE, it is necessary first of all to introduce the way in which HPSG's universal principles are dealt with, as these are of crucial importance in the successful interpretation of the rules

```
% head_feature_principle(Mother,Head-Daughter)

head_feature_principle(synsem loc (cat) head X,
                       synsem loc (cat) head X) if
      true


% marking_principle(Mother,Head-Daughter)

marking_principle(synsem loc (cat) marking X,
                  synsem loc (cat) marking X) if
      true
```

Figure 5 16

5 2 4 1 Universal Principles

The way in which the universal principles are implemented is similar to standard Prolog The value returned by the `head-feature_principle` (Figure 5 16), for example, is true if the value of the `head` feature in its first argument unifies with that of the second argument Similarly, the `marking_principle` ensures that the value of the mother's marking feature is identical to that of its head daughter

The definitions of the `context-consistency` and `semantic` principles are somewhat more complicated, however (Figure 5 17) The `conx_consistency _rule` takes as its arguments the Mother and Daughters of a phrase What this ensures is that the value of `conx·backgr` on the mother node is initiated to that of the head daughter This is achieved by

calling the procedure backgrs_of which functions as follows The first clause is a boundary

```
% conx_consistency_rule(Mother,Dtrs)

conx_consistency_principle((synsem loc conx backgr MBa
ckgr),Dtrs) if
   backgrs_of(Dtrs,e_set,MBackgr)


backgrs_of([],MBackgr,MBackgr) if true
backgrs_of([(synsem loc conx backgr e_set)|DRest],Accu
m,MBackgr) if
    backgrs_of(DRest,Accum,MBackgr),
    !
backgrs_of([(synsem loc conx backgr DBackgr)|DRest],Ac
cum,MBackgr) if
    union(Accum,DBackgr,NewAccum),
    backgrs_of(DRest,NewAccum,MBackgr)
```

Figure 5 17

clause and simply returns the value of MBackgr (Mother Background) as that of the accumulated context if the list of daughters is empty The second clause contains a recursive call to backgrs_of which is carried out if the background value of the head of the list of daughters is the empty set The accumulated value remains unchanged, as does that of MBackgr The tail of the list of daughters is the first argument of the recursive call

The final clause caters for the final scenario in which the head daughter on the list of daughters contains some contextual information This is joined to the contextual information already accumulated in Accum via a union predicate A recursive call to backgrs_of with this NewAccum value and the rest of the list of daughters completes the procedure

The semantics_principle takes in the Qstore, Qretr and Content values of the mother, as well as the qstore and content of the semantic head, and a list of the other daughters It combines the qstore values of the semantic head and the other daughters to return DQStore The predicate which performs this operation, qstores_of, functions in exactly the same way

as the procedure which joins the contextual backgrounds in the conx_consistency_ principle This value is then passed to another procedure, semp_act, along with the content of the mother

```
% semantics_principle(Mother, SemHead,Other_Dtrs)
%-----------------------------------------
semantics_principle((qstore MQStore,qretr MQRetr,synse
m loc cont X),
(SHead,synsem loc cont Y),ODtrs) if
  qstores_of([SHead|ODtrs],e_set,DQStore),
  semp_act(Y,MQRetr,MQStore,X,DQStore)

semp_act((psoa,nucleus Nucl,

quants SQuants),MQRetr,MQStore,(nucleus Nucl,

quants MQuants),DQStore) if
   ',set_sublist(MQRetr,DQStore,MQStore),
   append(MQRetr,SQuants,MQuants).
semp_act(Cont,[],QStore,Cont,QStore) if true

qstores_of([],QStores,QStores) if
   true
qstores_of([(qstore e_set)|Dtrs],Accum,QStores) if
   qstores_of(Dtrs,Accum,QStores),
   '
qstores_of([(qstore DQStore)|Dtrs],Accum,QStores) if
    union(Accum,DQStore,NewAccum),
    qstores_of(Dtrs,NewAccum,QStores)

set_sublist([],Set,Set) if
   true
set_sublist([X|Subs],Set,RestSet) if
   set_select(X,Set,Rest),
   set_sublist(Subs,Rest,RestSet)


set_select(X,(elt X,elts Xs),Xs) if
  true
set_select(Member,(elt X,elts Xs),(elt X,elts Rest))
if
               set_select(Member,Xs,Rest)
```

Figure 5 18

and the semantic head, the qstore and qretr values for the mother, and the quants value of the semantic daughter The value returned is the quants

feature for the mother The semantic content of the daughter is shared with that of the mother, according to the semantics principle, and the value of the mothers qretr feature is appended to the quants value of the semantic daughter to return the phrase's quants value

These principles are all used as goals in the phrase structure rules of HPSG

5 2 4 2 ID Schemata

```
% Schema 1

subject_head rule
(phrase,Mother,synsem loc (cat) (spr Spr,subj [],
comps []))
===>
cat> (SubjDtr),
goal> (sign_to_synsem(SubjDtr,SubjSynsem)),
cat>
(HeadDtr,synsem loc (cat) (spr Spr,subj [SubjSynsem],c
omps []))
goal> (head_feature_principle(Mother,HeadDtr),
       semantics_principle(Mother,HeadDtr,[SubjDtr]),
       marking_principle(Mother,HeadDtr),
       conx_consistency_principle(Mother,
                                  [SubjDtr,HeadDtr]))
sign_to_synsem(synsem Synsem,Synsem) if
     true
```

Figure 5 19

The descriptions of ID schemata in ALE consist of the specifications of the attribute-value structure values of categories, as well as the goals to be solved The left-hand side of a '===>' operator corresponds to the mother category while the body of the rule describes its daughters and any conditions on the rules which need to be satisfied (Head Feature Principle, Subcategorisation Principle etc)

Taking Schema 1 (the Subject-Head Rule, shown in Figure 5 19) as an example, the exact procedure involved in applying a HPSG rule can be explained Firstly, the name of the rule is specified, followed by the description of the

mother, in this case, a phrase which is fully satisfied (all complements and subject accounted for) The first category in the main body of the rule is the `SubjDtr` which is then passed to the goal `sign_to_synsem/2` This returns the value of the subject daughter's `synsem` feature The second category is the phrase's head daughter The value of its subject feature is unified with that of the phrase's subject daughter according to the schema as specified in section 4 5 2 1 The goals of this category correspond to the fulfilment of the head feature principle, the semantics principle, the marking principle and the context consistency principle

The other schemata are all formulated in exactly the same way, invoking the appropriate conditions, and passing the correct daughters A full listing of the schemata can be found in the Appendix B 2

5 2 5 Sample Monolingual Representations

As the output from ALE is generally very long, even for the simplest of sentences, only two examples will be shown here The format of output is generally the same for most sentences so this should suffice for the purposes of clarification

5 2 5 1 *Kim snores*

The simplest sentence to use is obviously one involving an intransitive verb with a proper noun agent The monolingual analysis for *Kim snores* is shown in Figure 5 20 The first part of the output simply iterates the input string with numbers between each of the words for use in conjunction with the chart parser The second part is the structure of the category of the string Only the mother structure is output, the daughters being considered superfluous once analysis has been carried out

The representation tells us that the input string *kim snores* is a phrase whose quantifer retrieved and stored values are empty Its lexical head is a finite verb, noninverted, nonauxiliary verb in the present tense, active voice, whose case frame is of type `actionintrans`, indicating that the agent of the action should

be animate It has no complements or subject and is thus saturated It is unmarked and its spr value is uninstantiated as there are no complementisers present in the input This, however, raises an interesting point Information which is not directly specified in lexical entries is nonetheless included when they are expanded or when they occur in a sentence or phrase This is in keeping with the condition that HPSG representation structures be totally well-typed

The content value indicates that the head verb is of type *snore*, which, according to its type specification, has two associated features, namely agent and agentprops The agent value is a referent for a singular, third person, masculine entity, whose property's value specifies that he is human, in keeping with the requirements imposed on the verb's roles by its case frame type There is no nonlocal information associated with the phrase, but its context value specifies that the referent of the agent is an individual bearing the name *kim*

5 2 5 2 *Kim eats the bread*

In this example, shown in Figure 5 21, only the content and quantifier retrieved values are shown, as the remainder of the analysis is very similar to that of *kim snores* The latter feature has a non-empty value, which corresponds to the determiner type, the, and its restrictive index which accounts for the content of the noun over which its scope ranges, in this case, *bread*

```
STRING
0 kim 1 snores 2
CATEGORY
phrase
QRETR e_list
QSTORE e_set
SYNSEM synsem
       LOC loc
           CAT cat
               COMPS e_list
               HEAD verb
                    AUX minus  CASEFRAME actionintrans
                    INV minus MOD none
                    PRD boolean TNS pres
                    VFORM fin VOICE active
               MARKING unmarked
               SPR ne_list_synsem
                   HD synsem
                      LOC loc
                          CAT cat
                              COMPS list_synsem HEAD head
                              MARKING comp SPR list_synsem
                              SUBJ list_synsem
                          CONT cont CONX conx
                                          BACKGR set_psoa
                      NONLOC nonloc
                             INHERITED nonloc1
                                       SLASH set_loc
                             TO_BIND nonloc1
                                     SLASH set_loc
                      TL e_list
               SUBJ e_list
           CONT psoa
                NUCLEUS snore
                        AGENT [0] ref
                              GEN masc NUM sg PER third
                        AGENTPROPS ne_list_special
                                   HD hume TL e_list
                QUANTS e_list
           CONX conx
                BACKGR ne_set_psoa
                       ELT psoa
                           NUCLEUS naming
                                   BEARER [0]
                                   NAME kim
                           QUANTS e_list
                       ELTS e_set
      NONLOC nonloc
             INHERITED nonloc1
                       SLASH e_set
             TO_BIND nonloc1
                     SLASH set_loc
```

Figure 5 20

```
QRETR ne_list_quant
      HD [2] quant
         DET the
         RESTIND npro
                 INDEX [0] ref
                       GEN neut
                       NUM sg
                       PER third
                 PROPERTIES [1] ne_list_special
                            HD food
                            TL e_list
                 RESTR ne_set_psoa
                       ELT psoa
                           NUCLEUS bread
                                   INSTANCE [0]
                           QUANTS e_list
                       ELTS e_set
       TL e_list
CONT psoa
     NUCLEUS eat
             AGENT [3] ref
                       GEN masc
                       NUM sg
                       PER third
             AGENTPROPS ne_list_special
                        HD hume
                        TL ne_list_special
                           HD animate
                           TL e_list
             PATIENT [0]
             PATIENTPROPS [1]
     QUANTS ne_list_quant
            HD [2]
            TL e_list
```

Figure 5.21

The content of the phrase is again a parametrised state of affairs, the relation of which is of type *eat* The details of the agent are exactly the same as those of the sentence *Kim snores*, but the patient referent and properties are structure shared with the values of the determiner's restrictive index in the quantifier retrieved value The restriction that the patient properties include edible and food is enforced by the fact that the head of the sentence, eat, has been assigned a case frame of type humeating (not shown here)

It is this structure which is passed through the reduction process and then on to the transfer phase

```
CATEGORY  phrase

SYNSEM  LOCAL  CAT  HEAD  verb,
                          CASEFRAME actionintrans,
                          TNS pres,
                          VOICE active,
                    CONT NUCLEUS snore,
                                AGENT [0] GEN masc,
                                          PER sg,
                                          NUM third
                               AGENTPROPS  [human],
                          QUANTS e_list,
                    CONX BACKGR naming,
                                BEARER  [0],
                                NAME  kim
```

Figure 5 22

5.3 Transfer

'One way to use such a [multidimensional] structure would be just to take the value of the SEM attribute for the mother sign in the output of analysis, and input this value to transfer

[Arn94]

Before being passed to the transfer component of the system, the structure produced by ALE's monolingual analysis is first reduced so that only the semantic features of the head of a sentence, along with tense, voice and case frame details are used The details of how exactly this is done are not particularly important as it is a tedious process, consisting primarily of moving step-by-step through the initial structure and extracting those features necessary for translation However, an example of the kind of structure which is passed to transfer is shown in Figure 5 22 Note that all code presented here was not part of ALE's original design, but has been added for the purposes of extending the system to deal with translation

In this system, transfer is based almost entirely on use of the semantic information derived from situation semantics and case frames It is this data which guides the selection of the components of the equivalent target language phrase What follows is a discussion of how exactly this is done, working through some of the transfer and lookup rules There will also be some examples of the kind of output achieved from the system, and as there is no generation component in its current form, these structures will be compared with the reduced monolingual structures of the target language It should also be noted at this point, that while the discussion focuses mainly on translation from English to German, the system is reversible in that it is also capable of translating from German to English

5 3 1 Beginning Translation

Translation is begun by calling the procedure `translate/0` within the ALE platform, having compiled the appropriate grammar This predicate prompts the user to enter the source language, followed by the sentence in Prolog list format (as in [kim,snores]) From here the procedure `rec/2` is called This takes in the string of input words and the language to be translated from, produces a complete analysis of the phrase (which is output to a file for further analysis if required,

such as investigation of quantification), and then calls `reduce/2` to effectively strip the analysis of that information deemed superfluous for the purposes of translation The value returned by this procedure is the structure which guides transfer This contains the mother's semantic and contextual information, as well as details of tense, verb form, case frame and voice

This value is used in calling the procedure `extract_trans/2` which returns the structure of the target language This procedure determines the nature of the verb in question e g , whether it is intransitive, transitive, ditransitive etc This information is contained in the name of the case frame which has been assigned to the verb Verbs of type `actionintrans`, for example, are by definition intransitive, while the verb *like*, of type `liking`, is transitive Once this issue has been resolved, the appropriate transfer procedure is called To clarify how exactly translation proceeds from this point, the example of *kim snores* will be used again to trace the steps taken to produce the target language structure

5 3 2 Transfer of *kim snores*

The verb *snore* has been determined as intransitive by virtue of having been assigned `actionintrans` as the value of its `caseframe` type This value, along with the tense and semantic details, are accessible throughout the transfer process Transfer is begun by

```
transferintrans(Cont,Conx,Caseframe,TenseDetails,Lang)
 -
% Begins transfer of intransitive verb, passing through the
values
% of its content, context, tense and voice, as well as the
source
% language
transferintransverb(Lang,Cont,Caseframe,
                            Details,Verb),
transferagent(Cont,Conx,Lang,Details,_,
                Nountype,Nounphrase),
arrange(TenseDetails,Details,Nounphrase,Verb,Nountype)
```

This calls three procedures, the first of which deals with the translation of the verb itself The second translates the agent of the action and the final procedure outputs the target language information to the screen

5 3 2 1 transferintransverb/5

Before discussing the exact details of this procedure, it is important to understand the internal representation of the feature structure, as this is critical to the way in which the structure is manipulated, and hence to how transfer proceeds Figure 5 23 shows the internal representation of the feature structure in Figure 5 22 The words on the left-hand side of each hyphen '-' is the name of the attribute, while the right-hand side represents its value Where the value is a feature structure, its type appears outside the brackets For example `cont-psoa(nucleus-snore( ))),quants-e_list)` indicates the the feature `cont` has a value of type `psoa`, the value of which is `nucleus-snore( )),quants-e_list` There is an important procedure `derefnew/3` which takes in such a structure and returns the name of the attribute and its value separately The built-in Prolog predicate '= ' is then used to split the type from the actual value This process is continued until the desired attribute and value are found

```
transferintransverb(Lang,Cont,Caseframe,Details,
                    Translation) -
% Finds verb type and looks this up in bilingual dictionary
% Also returns details of agent for purposes of identification
later
% in translation process
   verbdetails(Cont,Verb,Details1),
   extractagentdetails(Details1,Details),
   Fs= [Verb|Caseframe],
   lookup(Lang,Fs,Translation)

verbdetails(Cont,Verb) -
% Strips content of content feature structure until nucleus is
% found, then returns name of verb
    callderef(Cont,FS),
    getverb(FS,NewFS),
    NewFS= [Verb|_]
```

```
 synsem(
head-verb(caseframe-actionintrans,tns-pres,
                  vform-fin,voice-active),
cont-psoa(nucleus-snore(agent-ref1(gen-masc,
                                   num-sg,
                                   per-third),
                        agentprops-list_special
                                   (hd-human,
                                    tl-ne_list_special
                                        (hd-animate,
                                         tl-e_list))),
          quants-e_list),
conx-conx(backgr-ne_set_psoa(elt-psoa(nucleus-naming
                                        (bearer-ref1,
                                         name-kim),
                                    quants-e_list),
                             elts-e_set)))
```

Figure 5 23

The procedure `transferintrans/5` is quite simple It passes the verb's content value to the procedure `callderef/2` which carries out the above procedure until the feature `nucleus` is found, and the type of its value, in this instance `snore`, is taken as the name of the verb and combined with its case frame type This structure `snore(actionintrans)` is then searched for in the bilingual lookup dictionary, where it is matched with the first argument of the rule

```
lookup(snore(actionintrans),schnarchen(actionintrans))
```

Hence, the value `schnarchen(actionintrans)` is returned as the German translation of *snore*

5 3 2 2 transferagent/7

Once the translation of the verb has been found, it is necessary to translate its associated roles In this case, there is only one role, namely agent Before beginning to translate, it must be ascertained if the agent is a common or proper noun There are three clauses in `transferagent/7` which determine this fact

```
transferagent(Cont,Conx,_,_,Arg2,proper,Noun) -
% Calls strip_cont to extract value of quants feature  If
% equal to an empty list, then agent is a proper noun,
% the details of which can be found in the context feature
    strip_cont(Cont,_,Type),
    equal(Type,e_list),
    transferproperagent(Conx,Arg2,Noun)
```

The first clause calls a procedure `strip_cont/3` which recursively calls `derefnew/3` and `'= '` until it finds the feature quants and returns its value If this value is equal to an empty list, then the agent is a proper noun or a pronoun (as a common noun must be quantified), the details of which are to be found in the sentence's context attribute

```
transferagent(Cont,Conx,_,Details,Arg2,
                proper,Noun) -
% strips content until feature quants is found  If its
% value is nonempty, the index of the quantifiers
% restrictive index is compared with the details of the
% agent as obtained when transferring the verb  If they
% do not match, the agent is deemed to be a proper noun
    strip_cont(Cont,_,Type),
    findindex(Type,Index),
    \+compareindices(Details,Index),
    Arg2=Type,
    transferproperagent(Conx,_,Noun)
```

The second clause deals with the scenario where the value of the quants feature is nonempty The value of the specified quantifier's restrictive index is then extracted and compared with the index of the agent, as determined when translating the verb itself If this value does not match, then it is assumed to be the second argument e g , patient, goal etc In this case, there is no patient, but in other constructions this would be relevant The agent is thus taken to be a proper noun and `transferproperagent/3` is called

```
transferagent(Cont,_,Lang,Details,Arg2,
                common,[TransDet,TransNoun]) -
% extracts quants value and compares to index of agent
% If it matches, the determiner and noun are extracted
% and looked up in the bilingual dictionary
    strip_cont(Cont,_,Type),
    findindex(Type,Index),
    compareindices(Details,Index),
    transfercommonagent(Type,Arg2,Determiner,Noun),
    lookup(Lang,Noun,TransNoun),
    lookup(Lang,Determiner,TransDet)
```

The third and final clause is invoked when the quants value is nonempty and its first element matches the index of the agent `transfercommonagent/5` is called to extract the details of the determiner and common noun, and these values are then looked up in the dictionary

5 3.2.2.1 transferproperagent/3

```
transferproperagent(Conx,Arg2,Noun) -
% extracts the name of the proper agent
   extractname(Conx,Arg2,Noun)
```

The procedure `transferproperagent/3` is responsible purely for extracting the name of the proper agent The vast majority of proper nouns have no translations, although any exceptions to this can be dealt with by overriding the default mechanism by specifying the appropriate equivalencies in the target language (e g , Munich -> Munchen) The procedure `extractname/3` simply strips the context value until the name of the agent is found

5 3 2 2 2 transferpronoun/3

```
transferpronoun(Conx,Arg2,Pronoun) -
% extracts information necessary to generate correct
% translation of pronoun
     extractpronoun(Conx,Arg2,Pronoun)
```

The task assigned to the procedure `transferpronoun/3` is to retrieve the necessary information to select the correct pronoun in the target language Case will be dictated by the role of the nominal in question, so it is only the referential index and the nucleus type which needs to be found at this stage of transfer This procedure strips away all superfluous nucleus information using `derefnew` and

' ='

5 3 2.2.3 transfercommonagent/4

```
transfercommonagent(Quants,_,
                    Determiner1,CNoun) -
   extractquant(Quants,Type,List),
   equal(Type,e_list),
   assign_noun_det(List, Determiner1,CNoun)
transfercommonagent(Quants,Arg2,
                    Determiner1,CNoun) -
   extractquant(Quants,Type,List),
   \+equal(Type,e_list),
   Arg2=Type,
   assign_noun_det(List,Determiner1,CNoun)
```

The procedure `transfercommonagent/4` has two clauses which simply determine whether there are any elements left on the quants set after the agent has been extracted If there is (as in the second clause), this is assigned as the value of the `Arg2` variable The noun and determiner are found by `assign_noun_det/3` which, yet again, is a method of dereferencing the structure until the specified attributes and values are found These are returned to the `transferagent` procedure and looked up in the dictionary This technique could be problematic for sentences with embedded clauses, but this could be dealt with if extended to cater for such concerns

For sentences which have more than one associated role (as in *run, enter, give* etc), the translation method of the additional constituents is much the same as that of the agent Any value remaining in the `context` and `content` features after the extraction of the agent are passed through a similar series of procedures to find the patient or goal etc Again, any values still remaining will be assigned to an appropriate third role This continues until all elements of the sentence have been dealt with, ensuring that everything is translated

5 3 3 The Use of Case Frames in Transfer

The example of *kim snores* is not particularly useful in demonstrating the advantage of exploiting case frame information during transfer, as *snore* has only one translation in each of the two target languages However, if the sentence *Kim runs the shop* had been input instead, the head of the sentence would have been

assigned the case frame value managing Hence, the input to the lookup stage is run(managing) The entries in the dictionary include

```
lookup(run(competing),rennen(competing))
lookup(run(managing),fuehren(managing))
lookup(run(actionintrans),laufen(actionintrans))
```

Thus, the sentence would be translated as *Kim fuhrt das Geschaft,* as opposed to **Kim rennt das Geschaft* or **Kim lauft das Geschaft* The correct interpretation for *Kim runs the race* is also chosen

5.4 Assessment of Results

While, as yet, no generator exists for this system, the accurateness of translation can be assessed by comparing the structures of the target language produced by transfer with those produced after reduction of monolingual analysis Three examples of different constructs have been taken, the simple *Kim snores*, *Kim runs the shop* and *He is called Kim* The latter is shown by virtue of the problems it poses in terms of structural transfer, while the other two examples show the use of case frames

5 4 1 Comparison of *kim snores*

There is very little to be said about the two structures, except to say that they are both exactly the same Figure 5 24 shows the structure after transfer of *Kim snores*, which is identical to that of the reduced monolingual analysis (before transfer for translation to English) of the sentence *Kim schnarcht* The tense and agent details in both are the same, as is the `nucleus` type

5.4 2 Comparison of *kim runs the shop* and *kim fuhrt das Geschaft*

The analysis of *Kim fuhrt das Geschaft* is shown in Figure 5 25 The structure produced by transfer for *Kim runs the shop* is not shown as it is the same in every way, the correct translation of *run* having been chosen by the lookup dictionary Similarly, the two analyses of *kim runs the race* and *kim rennt das Rennen* are also the same and are shown in Figure 5 26 If this latter sentence were to be

translated from German to English, there would be a category ambiguity to be resolved as the word *rennen* can be both a verb and a noun (in written text this can be distinguished by spelling, i.e. *Rennen Vs rennen*). This issue is easily dealt with by the monolingual syntactic analysis, specifically by the ID-schemata invoked and the subcategorisation frames specified for each input word. Only one possible category could be assigned to each occurrence of the word.

```
CATEGORY: phrase
SYNSEM: LOCAL: CAT: HEAD: verb,
                          CASEFRAME:actionintrans,
                          TNS:pres,
                          VOICE:active,
                CONT:NUCLEUS:schnarchen,
                                AGENT:[0] GEN:masc,
                                          PER:sg,
                                          NUM:third
                                AGENTPROPS: [human, animate],
                          QUANTS:e_list,
                CONX:BACKGR:NUCLEUS:naming,
                                          BEARER: [0],
                                          NAME: kim
                                QUANTS:e_list
```

Figure 5.24

5.4.3 Comparison of *He is called Kim* and *er heisst Kim*

The differences between these two sentences are structural rather than semantic, so case frames alone will not suffice in chosing the correct translation. Hence, a different method of transfer is required. The structure for the English sentence *He is called Kim* is shown in Figure 5.27. There is a lexical entry for *be* whose list of complements include a past participle (as in *called, given, told* etc.). When this complement corresponds to the word *called*, the caseframe value assigned to the

structure is naming When this is encountered during transfer, *am* and *called* are taken collectively and translated together as *heissen*, producing the structure in

```
SYNSEM LOCAL CAT HEAD verb,
                      CASEFRAME managing,TNS pres,
                      VOICE active,
               CONT NUCLEUS fuhen,
                         AGENT [0] GEN masc,
                                   PER sg,NUM third
                         AGENTPROPS [human, animate],
                         PATIENT [1] GEN neut
                                     PER sg,
                                     NUM third
                         PATIENTPROPS [2] [business,building],
                   QUANTS ne_list_quant,
                         HD quant
                            DET the,
                            RESTIND npro
                                     INDEX [1]
                                     PROPERTIES [2]
                                     RESTR ne_set_psoa
                                       ELT psoa
                                    NUCLEUS geschaeft
                                                 INST [1]
                                       ELTS e_set
                         TL e_list
                CONX BACKGR NUCLEUS naming,
                                     BEARER [0],
                                     NAME kim
                              QUANTS e_list
```

Figure 5 25

```
SYNSEM LOCAL CAT HEAD verb,
                     CASEFRAME competing,TNS pres,
                     VOICE active,
               CONT NUCLEUS rennen,
                          AGENT [0] GEN masc,
                                    PER sg,
                                    NUM third
                          AGENTPROPS [human, animate],
                          PATIENT [1] GEN neut
                                      PER sg,
                                      NUM third
                          PATIENTPROPS [2] [competition],
                    QUANTS ne_list_quant,
                          HD quant
                              DET the,
                              RESTIND npro
                                        INDEX [1]
                                        PROPERTIES [2]
                                        RESTR ne_set_psoa
                                          ELT psoa
                                            NUCLEUS rennen
                                                    INST [1]
                                          ELTS e_set
                          TL e_list
                    CONX BACKGR NUCLEUS naming,
                                        BEARER [0],
                                        NAME kim
                               QUANTS e_list
```

Figure 5 26

```
CATEGORY phrase
SYNSEM LOCAL CAT HEAD verb,
                     CASEFRAME naming,
                     TNS pres,
                     VOICE active,
               CONT NUCLEUS is,
                          AGENT [0] GEN masc,
                                    PER sg,
                                    NUM third
                          AGENTPROPS [1] [human],
                          PROP CONT NUCLEUS call,
                                         BEARER [0]
                                         BEARERPROPS [1]
                                    QUANTS e_list
                    QUANTS e_list
               CONX BACKGR ELT NUCLEUS he
                                         INST [0],
                                    QUANTS e_list,
                               ELTS NUCLEUS naming
                                         BEARER [0],
                                         NAME kim
                                    QUANTS e_list
```

Figure 5 27

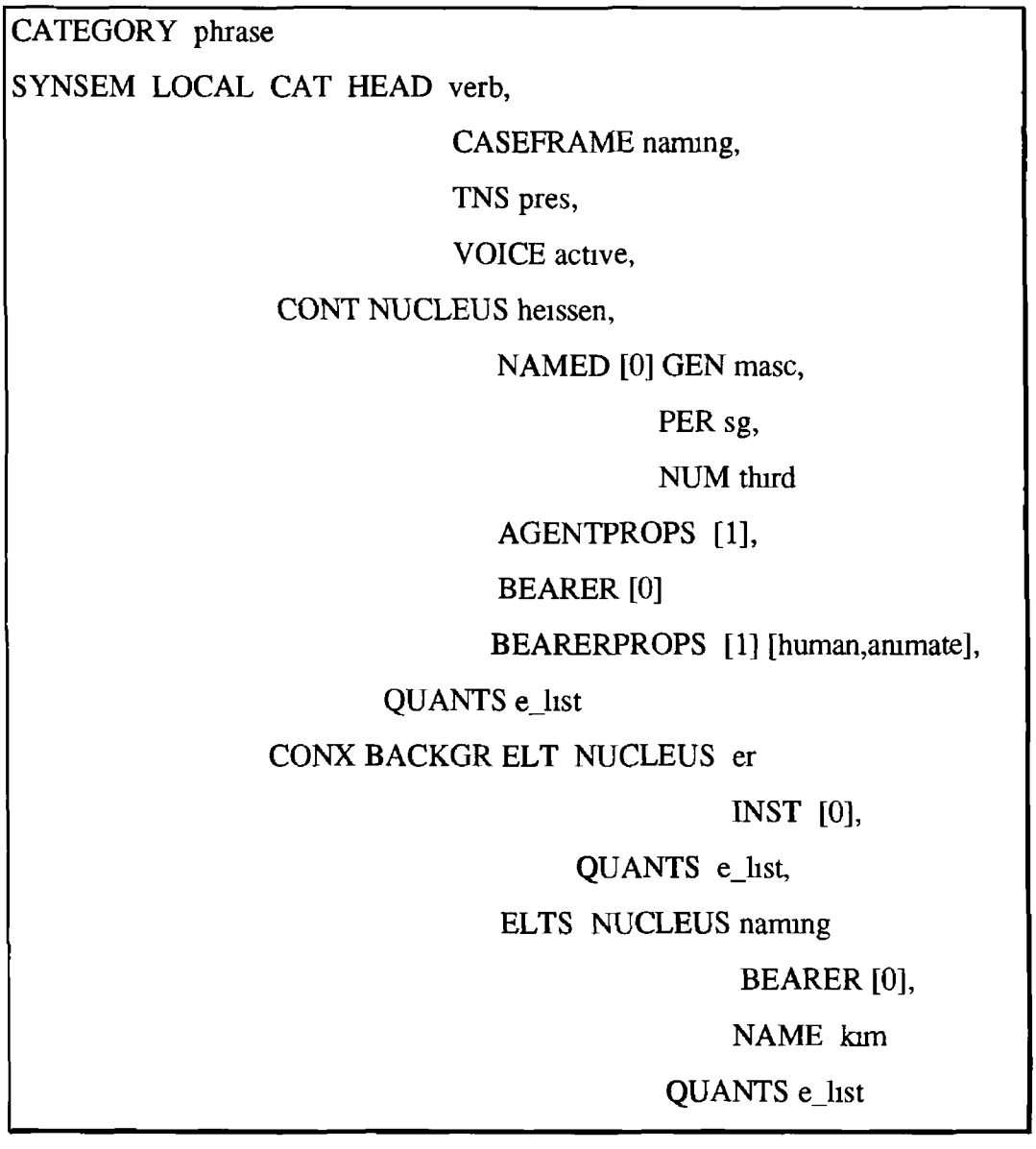


Figure 5 28

Figure 5 28 The same structure is produced when the German sentence *er heisst Kim* is analysed and reduced, suggesting that the correct output should be produced if a generation component were available Note that the system does not make allowances for the inclusion of the feature C_INDICES which would allow the use of pronouns referring to the speaker, such as *I, me, we* or *us* To make this extension, it would be simply a matter of including the attribute and its type in the type declaration segment of the grammar, and specifying it and its value for all such pronouns

For each of the test sentences used in translating, both from English and German, the results were found to be the same, in that the structures produced after transfer and by monolingual analysis were equivalent A full list of the lexical items used in this implementation can be found in the Appendix C 1

5.5 Evaluation

While the focus of the machine translation system implemented has been the differentiation between semantically ambiguous verbs via the use of head-driven transfer, and a combination of situation semantics and case grammars, there are other issues addressed in chapter 2 which are also dealt with

The problems of derivational and inflectional morphology have been abstracted away from the translation process entirely and are dealt with by the monolingual analysis component Information provided by morphology is preserved to ensure correct generation in the target language via the values of the appropriate features, for example, TNS, VOICE, PER, NUM etc These values are assigned by the lexical rules

As already indicated, category ambiguity in the form of words which can be assigned a number of syntactic categories is effectively treated by subcategorisation and HPSG's ID-schemata The problem of words of the same category but with different semantics is resolved by multiple lexical entries (for nouns and adjuncts) and case frame types (for verbs) Transfer ambiguities arising from such issues have been discussed to some extent in previous sections

Anaphora and the use of pronouns are treated by the structure sharing of referential indices as can be seen in the example in Figure 5 28 for the sentence *he is called Kim* The sentences *she likes Sandy, he reads the book* can also be translated, but, while *he reads the book that sandy gave him* can be analysed by ALE, at this stage it cannot be translated, as there are no transfer rules Given that the correct source language representation is produced by ALE, however, the

task of creating transfer rules for such constructions should not be that difficult and could be a possible extension to the system

While the sample sentences shown here have been straightforward in terms of syntactic structure, other constructions are also possible Determiner-adjective-noun phrases, for example, are treated in much the same way as agent and patient translation If the list of restrictions of a daughter (remember that the semantics of a noun phrase containing an adjective is represented by including the property of the adjective, as well as the index of the entity it modifies, in the list of restrictions imposed on a sentence by that noun phrase), is nonempty after having translated its first element, the process is repeated, ensuring that everything present in the input sentence is included in transfer This process also applies to prepositional phrases which may occur at the end of a verb phrase as in *Kim runs the race in the Olympics*

Other cases which have not been tackled as yet include head switching and preposition selection As indicated in chapter 4, head movement is dealt with by Schema 3, the head-subject-complement schema This correctly analyses sentences such as *can kim go?* The semantic analysis of this sentence is exactly the same as for *kim can go*, the difference between the two being the value of the INV feature While the value of INV in the latter sentences is '-', in the inverted sentence, the value is changed to '+' Obviously, for this to be correctly translated, some provision needs to be made for passing this value when it is '+' This idea is in keeping with Arnold et al's suggestion for using multidimensional analyses to different extents depending on the translation task [Arn94] Thus, when reducing the analysis representation structure before transfer, this value could be tested, and the structure changed accordingly (inclusion of INV feature if value is '+', exclusion otherwise) The translation of the sentence's components would proceed in the same way as for an uninverted sentence, but knowledge of a positive INV value would be necessary for generation (in order to generate *kann Kim gehen?* as opposed to *Kim kann gehen* This is an obvious extension to the current system

Preposition selection is another issue which is of particular relevance for German When translating from the English sentence *Kim enters the shop* to the German equivalent *Kim geht in das Geschaft*, the inclusion of an additional prepositional phrase absent from the English sentence is necessary This is accomplished by the specification of the complement list for the German verb *gehen* (to go) Where this takes on the meaning of *to enter* as in the above example, its complement list contains an element whose head feature has a `pform` (preposition form) of `in` However, if the `pform` value were um, as in the sentence *es geht um das Spiel (it is about the game)*, the meaning would change to *about sth* (as in a book, a report, a conversation etc)

In essence, while the test set is fairly limited, the system can deal quite easily with any of the constructions created from the lexicon The test set could be extended simply by following the format of the existing lexical entries, and perhaps including information about nonlocal structures in the universal component of the grammar

5.6 Conclusion

This chapter presented the system which was implemented in order to find some evidence for the theories presented in chapters three and four, namely that HPSG in an augmented form, is highly suitable for the purposes of machine translation given that the correct structures in the target language were produced for the set of sentences tested The details of monolingual analysis, carried out in the Attribute Logic Engine environment, were presented, in particular the formulation of lexical entries, universal principles, phrase structure rules and lexical rules The way in which transfer progresses when dealing with semantically ambiguous verbs was charted, as well as the translation of the verbs and their associated roles While no generator exists to produce an actual sentence in the target language, the structures produced by transfer compared favorably with those resulting after reduction was performed on the monolingual analysis of the correct sentence in the target language A brief discussion of the MT problems dealt with by the

system was also included The next chapter provides a summary of the conclusions which can be drawn from the implementation, as well as any possible improvements which could be carried out

Chapter 6
Conclusions

6.1 Introduction

Machine Translation has long been accepted as being a more difficult task than was initially thought, due principally to the intricate nature of language and our current lack of understanding of human language processing The idea of using a contemporary linguistic theory within an existing approach to MT is not a new one, but it is in this area that a large amount of the ongoing MT research is being carried out It is this idea which was taken as the basis of this thesis, choosing HPSG as the linguistic theory, and the transfer strategy as the method of translation The aim of this chapter is to draw the conclusions from the theories proposed in chapters two, three and four, together with the evidence from the implementation presented in chapter five, whether this supports or contradicts these theoretical proposals

6.2 Discussion of Proposals and Implementation

The core of the implementation in chapter five was to take a HPSG analysis of the source language input and produce the corresponding structure for use in generation of the target language However, as it stands, the HPSG formalism was deemed insufficient in some key areas, most notably in the declaration of semantic content The use of a combination of the theory's existing meaning representation formalism and a form of Fillmore's case grammars was proposed as a way of overcoming this problem. Moreover, the addition of several new features for syntactic interpretation was also suggested, namely TENSE and VOICE

The augmented semantic formalism proved quite efficient in choosing between conflicting translations for verbs such as *run, enter, eat* and *make* In each scenario tested, the correct case frame type was assigned ensuring the correct translation This succeeded even in those situations where accompanying nouns were also considered ambiguous, as in *Kim runs the bank*, where both meanings of *bank* were tried to make sense of the sentence (as in *Kim runs the river bank* and *Kim runs the money bank*) The use of additional features also proved essential in preserving information absent from the VFORM attribute,

namely tense and voice details. The other constructions and issues dealt with were outlined in the previous chapter, and include preposition selection, head-movement, morphological analysis and category ambiguities amongst other things. For the most part, the transfer rules are reversible, although those which deal with structural changes have a different form depending on the source and target languages. However, there are a number of improvements which could be made to develop its efficiency and power.

6.3 Future Work

The fact that the structure must be thoroughly constructed after transfer is, to a large extent, a wasted effort, given that there exist two complete lexicons, one for each of the languages involved. However, in its current form, it is possible to have only one of these compiled and operable in ALE at any given time, so all the data is not available at any one time. If this could be overcome, and both lexicons could be simultaneously compiled and accessed, the lookup dictionary could find the appropriate words in the target language lexicon and use these specifications to create its structure to be used in generation. An investigation into the feasibility of devising a methodology to do this, and hence increase the robustness and effectiveness of the system, is an idea currently under consideration. In such a system, the bilingual lookup component would consist of procedures to search in the target language lexicon for the appropriate words and thus find their entire HPSG representations. Therefore, the semantic features of the source language would be used only to aid in selection of the full lexical specification and not to control translation completely. Whether or not this would truly increase the efficiency of the system is a question which can only be answered after implementation, but obviously this approach would make considerable use of the fact that HPSG is a reversible theory, using the same lexical specifications for both analysis and generation, helping to create a truly bi-directional system.

This brings forth the problem of the lack of a generation component. The fact that the most important information about semantics, tense, agreement etc. is present means that generation should be simply a question of reversing the

analysis procedure and should not pose any great problems in terms of developing the system. The completion of the system in terms of developing a generation component is certainly an aim at this stage

In its current form, the lexicon is quite small having been developed merely as a means of testing the proposed theories With the ever-increasing availability of resources in terms of on-line dictionaries and text corpus, the options for extending a lexicon are vast One possible method could be to take one such on-line resource and implement a strategy for automatically creating HPSG representations The feasibility of such a plan is already being researched in several centres Researchers at the University of Lancaster are attempting to create a British National Corpus containing 100M words which have been tagged with grammatical information Obviously this tagging would have to be extremely accurate in order for translation to succeed, and it could be a difficult task, but it is undoubtedly an option worth considering

One proposal is to use WordNet®, an on-line lexical reference system whose design is inspired by current psycholinguistic theories of human lexical memory ([Word96]) English nouns, verbs, adjectives and adverbs are organized into synonym sets (*synset*), each representing one underlying lexical concept Different relations link the synonym sets There are two kinds of relations lexical and semantic The former exist between word forms, while the latter hold between word meanings Each synset consists of synonymous words (e g , *couch, sofa)* or collocations (such as *bank account, bank draft*) and pointers indicating the relations between this and other synsets These relations can include hyponymy, antonymy, entailment etc Of great significance for translation is the fact that a word or collocation can be found in more than one synset and more than one part of speech (category and semantic ambiguity) Nouns in Wordnet are organised into hierarchies, whereby every noun synset is governed by at least one of the relational hierarchies in Figure 6 1 Thus semantic attributes can be inherited by a word from a entity higher up in the network Similarly, polysemous verbs are connected so that the senses of the verb *make* in *make a good salary* and *make 29 points in the game* are grouped together, as the meanings are similar, although not

{ entity, (something having concrete existence, living or nonliving) }
{ psychological_feature, (a feature of the mental life of a living organism) }
{ abstraction, (a concept formed by extracting common features from examples) }
{ location, space,#p (a point or extent in space) }
{ shape, form, (the spatial arrangement of something as distinct from its substance) }
{ state, (the way something is with respect to its main attributes, "the current state of knowledge", "his state of health", "in a weak financial state") }
{ event, (something that happens at a given place and time) }
{ act, human_action, human_activity, (something that people do or cause to happen) }
{ group, grouping, (any number of entities (members) considered as a unit) }
{ possession, (anything owned or possessed) }
{ phenomenon, (any state or process known through the senses rather than by intuition or reasoning) }

Figure 6 1

identical (example taken from [Word96]) This corresponds to the grouping of verbs according to their case frame types and could possibly be exploited in future developments of the system presented in this thesis

WordNet was developed by the Cognitive Science Laboratory at Princeton University under the direction of Professor George A Miller (Principal Investigator) As of 30 June 1996 WordNet contained 120,401 different entries (different ASCII strings) organized into 96,757 lexicalized concepts (synsets) by 134,864 semantic relations (labeled links between synsets) Given that the primary goal of Wordnet has been identified by its creators as the development of lexical resources for natural language research ([Word96]), it would seem that it is ideally suited for work in conjunction with a lexically-driven machine translation system. Moreover, the fact that all the information contained in WordNet is stored in a Prolog database makes it particularly appropriate for the system presented here as there is no programming language clash Hence, extending this system would be a question of manipulating the entries in WordNet in such a way as to generate HPSG representations from them The exact manner in which this could

Sense 1
bank, tip laterally -- of boats and aircraft
=> tip, cause to tip, cause to tilt

Sense 2
bank, enclose with a bank -- as of roads
=> enclose, enclose, shut in

Sense 3
bank, have an account, keep money -- do business with a bank
=> transact, deal -- do business

Sense 4
bank -- be in the banking business
=> work, do work -- be employed

Sense 5
deposit, bank -- put into a bank account
give -- transfer possession of something concrete or abstract to somebody,
=>"I gave her my money",
"can you give me lessons?"

Sense 6
bank, cover with ashes -- of fires, to control the rate of burning
=> cover -- provide with a covering

Collocations
bank on
bankroll
bankrupt
enclose with a bank

Figure 6 2

be done is a question yet to be answered but one that may be worth investigating in an attempt to develop a more robust system

The information supplied by WordNet could prove invaluable in resolving a number of MT's associated problems, particularly those of structure, semantic and category ambiguities As an example of the data it can supply, the word *bank* was input to the system. On request, it returned the collocations and senses of the verb *bank*, and the same relations for the corresponding noun Other options included finding any existing antonyms, synonyms, attributes and the frequency with which the word in used in the English language (rated on a polysemy scale of 0-10), amongst others The values returned for the verb are shown in Figure 6 2, while an edited version of the results of the search on the noun can be seen in Figure 6 2 (there were over 50 different collocations returned) Given that there are attributes available for each of these meanings, if HPSG representations for each of these could be generated, an extremely large lexicon would ensue This could perhaps be reduced by limiting the generated lexicon to those words with a polysemy count over a certain value Obviously some of the senses returned for

Sense 1
bank, side -- sloping land (especially the slope beside a body of water);
=> slope, incline, side -- an elevated geological formation

Sense 2
a financial institution that accepts deposits and channels the money into lending activities
=> financial institution, financial organization

Sense 3
bank -- a long ridge or pile; "a huge bank of earth".
ridge -- a long narrow natural elevation or striation.

Sense 4
bank -- an arrangement of similar objects in a row or in tiers; "he operated a bank of switches".
=> array

Sense 5
bank -- a supply or stock held in reserve esp for future emergency use; "the Red Cross has a blood bank for emergencies".
=> reserve, backlog, stockpile resource,

Sense 6
bank -- the funds held by a gambling house or the dealer in some gambling games; "he tried to break the bank at Monte Carlo".
=> funds, finances, monetary

Sense 7
bank, cant, camber -- a slope in the turn of a road or track;
=>slope, incline, side

Sense 8
savings bank, coin bank, money box, bank a container (usually with a slot in the top) for keeping money at home
=> container.

Sense 9
bank, bank building -- a building in which commercial banking is transacted; "the bank is on the corner of Nassau and Witherspoon".
=> depository, deposit, repository -- a place where things can be deposited for safekeeping.

Collocations

bank	bank account	bank bill	bank building
bank card	bank charter	bank check	bank clerk
bank draft	bank holiday	bank note	bank of england
bank rate	bank statement	bank vault	bankbook
banker	banker's draft	banking	banknote
bankrupt	blood bank	bundesbank	central bank

Figure 6.3

bank in Figures 6.2 and 6.3 would be more commonly used than others, some occuring only in very specific contexts, e.g. Sense 6 of bank in Figure 6.3 - funds held by gambling house. Such senses could maybe be omitted entirely, creating a sublanguage approach.

While such a system would not necessarily result in a very fast system, it would certainly be complete and robust in many ways, given its large coverage

and WordNet's ability to determine varying meanings and senses from the information available This option is in many ways considerably more appealing than a system which operats quickly, but whose results are unreliable Obviously the lack of a similar system for German could be a problem for translation Perhaps some kind of lookup system could be used to generate the corresponding information for German While this could limit the implementation to functioning as a unidirectional system, it could prove to be a vital starting point for generation of HPSG representations from an extensive on-line lexical resource

6.4 Conclusion

This thesis presented a discussion of machine translation, HPSG and the way in which the latter could be used to implement the former Chapter one introduced the main focus points of the thesis, namely the use of a Head-Driven Phrase Structure Grammar with an augmented semantic framework of case grammars and semantic features for the purposes of analysis in a transfer-based machine translation system for German and English, before summarising the history of machine translation in terms of its highs and lows in chapter two The approach adopted in the system developed for this thesis, namely transfer, was also presented, highlighting some of the problems which might be encountered and, in most cases, suggesting a possible solution This led to the introduction of the HPSG formalism in chapter three Its psychological validity and the debt of honour it owes to many of linguistics' most successful and influential contemporary theories were outlined, highlighting the importance of order and process independence, the integration of information from several different sources (syntax, semantics, context etc) and the incremental nature of language processing

Chapter Four presented a discussion of the main components of HPSG, namely its highly lexicalised nature, the universal principles and Immediate Dominance schemata, and the interaction of all three to impose constraints on language An attempt to clarify the way in which these could be used in the implementation was also made These components were then implemented in

chapter five in conjunction with the Attribute Logic Engine developed by Bob Carpenter and Gerald Penn A grammar for English designed by the latter was taken as the basic building block from which to write an extended grammar for English and German, augmenting the semantic interpretation component with the use of case grammars A transfer-based machine translation system was then outlined, taking sentences containing semantically ambiguous verbs as examples

Finally a number of suggestions as to the way in which this system could be developed were presented, concentrating primarily on the advantages of the WordNet system, and the way in which its features could be seen as beneficial for use with Head-Driven Phrase Structure Grammar

References

[All87] Allen, James *Natural Language Understanding* Benjamin Cummings, Menlo Park, 1987

[Arn94] Arnold, Doug, Balkan, Lorna, Meijer, Siety, Humphreys, R L and Louisa Sadler *Machine Translation An Introductory Guide* NCC Blackwell, Manchester, 1994

[Bal87] Ballard, Bruce W and Mark A Jones Computational Linguistics *Encyclopaedia of Artificial Intelligence* (Stuart C Shapiro, ed) 133-51 Wiley, New York, 1987

[Barr81] Barr, Avron and Edward A Feigenbaum *The Handbook of Artificial Intelligence,* 1 William Kaufmann, Palo Alto, 1981

[Barw83] Barwise, Jon and John Perry *Situations and Attitudes* MIT Press, Cambridge, Mass, 1983

[Benn95] Bennett, Paul *A Course in Generalised Phrase Structure Grammar* UCL Press in association with the Centre for Computational Linguistics, London, 1995

[Borg91] Borgida, Alexander *Principles of Semantic Networks Explorations in the Representation of Knowledge* (John Sowa, ed) Morgan Kaufmann, CA, 1991

[Bos94] Bos, Johan, Mastenbroek, Elsbeth, McGlashan, Scott, Millies, Sebastian, and Manfred Pinkal *The VERBMOBIL Semantic Formalism* Technical Report VM-Report 6, Universitat des Saarlandes, 1994

[Bos94b] Bos, Johann *Presupposition as Anaphora in the VERBMOBIL Semantic Formalism* Technical Report VM-Report 25, Universitat des Saarlandes, 1994

[Brat90] Bratko, Ivan *Prolog Programming for Artificial Intelligence, Second Edition* Addison-Wesley,MA, 1990

[Car94] Caroli, Folker, Nubel, Rita, Ripplinger, Babs, and Joerg Schutz *Transfer in VERBMOBIL* Technical Report VM-Report 11, Universitat des Saarlandes, 1994

[Carp94] Carpenter, Bob and Gerald Penn *ALE 2 0 User's Guide* Carnegie Mellon University Laboratory for Computational Linguistics Technical Report, Pittsburgh, 1994 *http //macduff andrew cmu edu/ale/*

[Carp95] Carpenter, Bob *Interview published in Ta! the Dutch students' magazine for Computational Linguistics*, volume 3, number 1, 1995 *http //wwwots let ruu nl/Ta/HOME-english html*

[Char76] Charniak, Eugene and Yorick Wilks *Computational Semantics* North-Holland Publishing Company, 1976

[Char85] Charniak, Eugene and Drew V McDermott *Introduction to Artificial Intelligence* Addison-Wesley, Reading, MA, 1985

[Clock94] Clocksin, William F and Christopher S Mellish *Programming in Prolog, Fourth Edition* Springer-Verlag, Berlin, 1994

[Coll95] Collins, Bróna and Pádraig Cunningham A Methodology for Example Based Machine Translation *Proceedings of the Fourth International Conference on Cognitive Science of Natural Language Processing* 1-10, Dublin City University, 4-7 July 1995

[Collins] *Collins English Dictionary* 3rd edition Harper Collins, Glasgow, 1994

[CollinsGer] Ternell, Peter *Collins German-English, English-German Dictionary*, 2nd ed Harper-Collins, London, New York, Stuttgart, E Klett Verlag, 1991

[Cope95] Copestake, Anne, Flickinger, Dan, Malouf, Rob, Riehemann, Susanne and Ivan A Sag Translation using Minimal Recursion Semantics *Proceedings of the Sixth International Conference on Theoretical and Methodological Issues in Machine Translation*, Leuven, 1995

[CoQu69] Collins, A & M R Quillian Retrieval time from semantic memory *Journal of Verbal Learning & Verbal Behaviour*, **8** 240-47, 1968

[Daim95] Daimler Benz *Verbmobil A computer that comprehends speech and translation* Daimler-Benz News, October 27, 1995

[Dean95] Dean, Thomas, Allen, James and Yiannis Aloimonus *Artificial Intelligence Theory and Practice* Benjamin/Cummings Publishing Company, CA, 1995

[Dowt79] David R Dowty *Word Meaning and Montague Grammar* Reidel, Dordrecht, 1979

[Dowt82] David R Dowty Grammatical relations and Montague grammar *The Nature of Syntactic Representation* (Pauline Jacobson and Geoffrey Pullum, eds) 79-130 Reidel, Dordrecht, 1982

[Duden] Scholze, Werner, Sykes, John Bradbury, Dudenredaktion, Oxford University Press, Dictionary Department, German Section *The Oxford Duden German Dictionary German-English, English-German* Clarendon Press, New York, Oxford University Press, Oxford, 1990

[Euro96] Centre for Computational Linguistics, Faculty of Arts, Katholieke Universiteit Leuven EUROTRA *http //www ccl/kuleuven ac be/about/EUROTRA html*

[EyKe90] Eysenck, Michael and Mark T Keane *Cognitive Psychology A Students Handbook* Lawrence Erlbaum Associates, Sussex, 1990

[Fill68] Fillmore, Charles J The Case for Case *Universals in Linguistic Theory* (Emmon Bach and Robert Harms, eds) 1-88, Holt, Rinehart & Winston, New York, 1968

[Fod74] Fodor, J A Bever, T G and M F Garrett *The Psychology of Language* McGraw-Hill Book Company, 1974

[Fran95] Frank, Anette and Uwe Reyle *Principle Based Semantics for HPSG* Institute for Computational Linguistics, University of Stuttgart, 1995

[Fraz78] Frazier, Lyn and Janet Dean Fodor The Sausage Machine a New Two-Stage Model of the Parser *Cognition International Journal of Cognitive Psychology, Vol VI* 197-224 Elsevier Sequoia S A 1978

[Fraz79] Frazier, Lyn *On Comprehending Sentences Syntactic Parsing Strategies* Ph D Thesis University of Connecticut, Indiana University Linguistics Club, 1979

[Gazd89] Gazdar, Gerald and Christopher Mellish *Natural Language Processing in PROLOG* Addison-Wesley, England, 1989

[Gosh87] Goshawke, Walter, Kelly, Ian D K , J David Wigg *Computer Translation of Natural Language* Sigma Press, Wilmslow, Halsted Press, New York, 1987

[Haeg94] Haegeman, Liliane M V *Introduction to Government and Binding Theory*, 2nd edition Blackwell, Oxford, UK, Cambridge, Mass, USA, 1994

[Hal88] Halvorsen, P K Projections and Semantic Description in Lexical Functional Grammar *Proceedings of the International Conference on 5th Generation Computer Systems* 1116-22, 1988

[Hall78] Halle, Morris, Bresnan, Joan and George A Miller *Linguistic Theory and Psychological Reality* MIT Press, 1978

[Hein95] Heine, Julia E , and Karsten L Worm *Semantic Phenomena for German with Examples* Technical Report VM-Report 86, Universität des Saarlandes, 1995

[Heiz94] Heizmann, Suzanne *Human Strategies in Translation and Interpreting - what MT can Learn from Translators* Technical Report VM-Report 43, Universitat Hildesheim, 1994

[Hend75] Hendrix, Gary G Expanding the utility of semantic networks through partitioning *IJCAI-75* 115-21, 1975

[HoUl79] Hopcroft, J E & J D Ullman *Introduction to Automata Theory Languages and Computation* Addison-Wesley, Reading, MA, 1979

[Hutch86] Hutchins, W J *Machine Translation Past Present and Future* Ellis Horwood Limited, Chichester, England, 1986

[Hutch92] Hutchins, William John and Harold L Somers *An Introduction to Machine Translation* Academic Press, London, 1992

[KaFo63] Katz, Jerrold J and Jerry A Fodor The structure of a semantic theory *Language*, 39, volume 3 170-210, 1963

[Kamp81] Kamp, H A Theory of Truth and Semantic Representation *Formal Methods in the Study of Language* (Groenendijk, Jeroen, A G, Janssen, T M V and Martin B J Stokhof eds), Mathematical Centre Tract 135, Amsterdam 277-322 Reprinted in *Truth, Representation and Information*, GRASS Series no 2, Dordrecht, 227-322, 1981

[Kap82] Kaplan, Ronald M And Joan Bresnan Lexical-Functional Grammar A Formal System for Grammatical Representation *The Mental Representation of Grammatical Relations*, (Joan Bresnan, ed) MIT Press, MA, 1982

[Kay85] Kay, Martin Parsing in Functional Unification Grammar *Natural Language Parsing* (David R Dowty, Lauri Kartunnen and Arnold M Zwicky, eds) 251-78 Cambridge University Press, Cambridge, 1985

[Kean94] Keane, Mark *Lecture Notes on AI and Natural Language Processing*, presented at Trinity College, Dublin, Michaelmas and Hilary Terms, 1993/4

[Kimb73] Kimball, John Seven Principles of Surface Structure Parsing in Natural Language *Cognition International Journal of Cognitive Psychology, Vol II*, pp15-47 Mounton & Co, N V, 1973

[King87] King, Margaret, *Machine Translation Today* Edinburgh University Press, 1987

[Klein95] Klein, Ewan *CSLI and Verbmobil An Interview with Dan Flickinger* Elsnews, September 1995

[John83] Johnson-Laird, P *Mental Models* Harvard University Press, Cambridge, MA, 1983

[John88] Johnson-Laird, P *The Computer and the Mind* Harvard University Press, Cambridge, MA, 1988

[Lehn88] Lehnert, Wendy G Knowledge-based natural language understanding *Exploring Artificial Intelligence Survey Talks from the National Conferences on Artificial Intelligence* (Howard E Shrobe, ed) 83-131 Morgan Kaufmann Publishers, Sand Mateo, CA, 1988

[LuSt93] Luger, George F & William A Stubblefield *Artificial Intelligence Structures and Strategies for Complex Problem Solving* Benjamin Cummings Publishing Company, Inc, CA, 1993

[McC86] McCord, M C Design of a Prolog-Based Machine Translation System *Proceedings of the 3rd International Logic Programming Conference* 359-74, 1986

[McDon] McDonald, M , N J Pearlmutter, and M S Seidenberg In press The Lexical Nature of Syntactic Ambiguity Resolution *Psychological Review* Quoted in [Sag95]

[Mahon95] Mahon, Elaine and Andy Way *An Overview of Discourse Representation Theory* Working Paper CA-2195, Dublin City University, 1995

[Mai94] Maier, Elisabeth, Scott McGlashan *Semantic and Dialogue Processing in the VERBMOBIL Spoken Dialogue Translation System* Technical Report, VM-Report 51, Universitat des Saarlandes, 1994

[Mal95] Malouf, Robert Implementing HPSG Grammars in ALE Slide presentation, *http //hpsg stanford edu/hpsg/papers/html*

[Man95] Mandelblit, Nili Beyond Lexical Semantics Mapping and Blending of Conceptual and Linguistic Structures in Machine Translation *Proceedings of the Fourth International Conference on Cognitive Science of Natural Language Processing* 1-12, Dublin City University, 4-7 July 1995

[Math95] Matheson, Colin Course Notes on HPSG in ALE Centre for Cognitive Science, University of Edinburgh, 1995 *http //www ltg hcrc ed ac uk/projects/ledtools/ale-hpsg/index html*

[Mins81] Minsky, Marvin A framework for representing knowledge *Mind Design* (John Haugeland, ed) 95-128 MIT Press, Cambridge, 1981

[MTIT94] Dong, Zhen Dong, *Machine Translation Computers have a role in language translation* IT The Next Wave, Business Times, September 26, 1994

[Near96] Neary, Orlagh Corel Corporation Translation Tools Survey *Software Localisation The Quarterly Newsletter of the LRC, Volume 1 Number 1*, July 1996 *http //lrc ucd ie*

[Nir87] Nirenburg, Sergei *Machine Translation Theoretical and Methodological Issues* Cambridge University Press, 1987

[Nobl88] Noble, H M *Natural Language Processing* Blackwell Scientific Publications, Edinburgh, 1988

[Noord96] van Noord, Gerthan, Dorrepaal, Joke, van der Eijk, Pim, Florenza, Maria, Ruessink, Herbert and Louis des Tombe An Overview of MiMo2 *http //grid let rug nl/~vannoord/papers/mt/node1 html*

[Oxford] Simpson, John Andrew ad Edmund S C Weiner *The Oxford English Dictionary*, Second edition Clarendon, Oxford, 1989

[PeSh86] Pereira, Fernando C N and Stuart M Shieber *An Introduction to Unification-based Approaches to Grammar* CSLI Lecture Notes no 4, Chicago University Press, Chicago, 1986

[Poll88] Pollard, Carl and Ivan Sag *Information-based Syntax and Semantics, Volume I Fundamentals* CSLI Lecture Notes no 13, Chicago University Press, Chicago, 1988

[PoSa94] Pollard Carl and Sag Ivan A *Head-Driven Phrase Structure Grammar* The University of Chicago Press, Chicago, 1994

[QuSch94] Quantz, Joachim J. and Birte Schmitz. *Knowledge-Based Disambiguation for Machine Translation*. Technical Report VM-Report 44, Technische Universität Berlin, 1994.

[Reape94] Reape, Mike. *Lecture Notes on Computational Linguistics*, presented at Trinity College, Dublin, Hilary Term 1994.

[Rieh95] Riehemann, Susanne. *The HPSG Formalism*. 1995 *http://hpsg.stanford.edu/~sr*

[RiTh84] Ritchie, Graeme D. and Henry S. Thompson. Natural Language Processing. *Artificial Intelligence: Tools, Techniques and Applications* (Tim O' Shea and Marc Eisenstadt, eds.):358-88. Harper & Row, New York, 1984.

[Sag93] Sag, Ivan A. *Interview published in the EACL special of Ta!, the Dutch students' magazine for computational linguistics*, volume 2, number 2, Summer 1993. *http://wwwots.let.ruu.nl/Ta/HOME-english.html.*

[Sag94] Sag, Ivan A. *HPSG: Background and Basics*. Manuscript, Stanford University, 1994.

[Sag95] Sag, Ivan A. *Taking Performance Seriously*. Manuscript, Stanford University, April 1995.

[Sch72] Roger C. Schank. Conceptual Dependancy: a theory of natural language understanding. *Cognitive Psychology*, **3**: 552-631, 1972.

[Shieb92] Shieber, Stuart M. *Constraint-based Grammar Formalisms: Parsing and Type Inference for Natural and Computer Languages*. MIT Press, Cambridge, Mass, 1995.

[Sell85] Sells, Peter. *Lectures on Contemporary Syntactic Theories*. CSLI Lecture Notes: No.3. CSLI, Stanford, 1985.

[Shar95] Sharp, Randall and Oliver Streiter. Applications in Multilingual Machine Translation. *Proceedings of PAP95, The Practical Application of Prolog*, Paris, 3-6 April, 1995.

[Stan96] Center for the Study of Language and Information Head-Driven Phrase Structure Grammar Precision Grammars for Human Languages *http //hpsg stanford edu/*

[SYS96] SYSTRAN Software Inc The History and Development of SYSTRAN, 1996 *http //systranmt com/*

[TanTr] Tanenhaus, M and J Trueswell Sentence Comprehension *Handbook of Perception and Cognition Vol II Speech and Language* (J Miller and P Eimas,eds) Academic Press, New York , In Press Quoted in [Sag95]

[Thur91] Thurmair, Gregor, Recent Developments in Machine Translation *Computers and the Humanities,* **25** 115-28, 1991

[Uszk86] Hans Uszkoreit *Word-Order and Constituent Structure in German* CSLI Lecture Notes no 8, Chicago University Press, Chicago, 1986

[Verb94] University of Saarbrucken The Verbmobil Project Introduction to a Vision *http //www dfki uni-sb de/verbmobil/*

[Wino72] Terry Winograd *Understanding Natural Language* Academic Press, New York, 1972

[Wood81] William A Woods Procedural semantics as a theory of meaning *Elements of Discourse Understanding* (Aravind K Joshi, Bonnie Lynn Webber and Ivan Sag, eds) 300-34 Cambridge University Press, Cambridge, 1981

[Word96] Cognitive Science Laboratory, Princeton University WordNet A Lexical Database for English *http //www cogsci princeton edu/~wn/index html*

[Wyb91] Wybraniec-Skardowska, Urszula *Theory of Language Syntax Categorial Approach* Kluwer Academic Publishers, Dordrecht, Boston, 1991

[Yng58] A programming language for mechanical translation *Mechanical Translation,* **5**, 25-41, 1958

Appendix A HPSG's Hierarchical Typing System

A.1 Type Hierarchy in ALE

```
bot cons bot
bot sub
[boolean,case,cat,cont,conx,gend,grammartype,head,ind,list,loc,
        marking,mod_synsem,name,nonloc,nonloc1,num,per,pform,qfpsoa,
        sem_det,set,sign,special,tense,vform,voice,definite]


list sub
[e_list,ne_list,list_sign,list_synsem,list_quant,list_special]

set sub [e_set,ne_set,set_psoa,set_loc,set_npro,set_quant,set_ref]

e_list sub []

e_set sub []

ne_list sub [ne_list_sign,
ne_list_synsem,ne_list_quant,ne_list_special]
       intro [hd bot,
              tl list]

ne_set sub [ne_set_psoa, ne_set_loc, ne_set_npro, ne_set_ref,
          ne_set_quant]
       intro [elt bot,
              elts set]

list_sign sub [e_list,ne_list_sign]

list_synsem sub [e_list,ne_list_synsem]

list_quant sub [e_list,ne_list_quant]

set_psoa sub [e_set,ne_set_psoa]

set_loc sub [e_set,ne_set_loc]

set_ref sub [e_set,ne_set_ref]

set_quant sub [e_set,ne_set_quant]

set_npro sub [e_set,ne_set_npro]

ne_list_sign sub []
       intro [hd sign,
              tl list_sign]

ne_list_synsem sub []
       intro [hd synsem,
              tl list_synsem]

ne_list_quant sub []
       intro [hd quant,
              tl list_quant]

ne_set_psoa sub []
```

```
       intro [elt psoa,
              elts set_psoa]

ne_set_loc sub []
           intro [elt loc,
                  elts set_loc]

ne_set_npro sub []
          intro [elt npro, elts set_npro]

ne_set_quant sub []
           intro [elt quant, elts set_quant]

ne_set_ref sub []
         intro [elt ref, elts set_ref]


sign sub [phrase,word]
     intro [synsem synsem,
          qstore set_quant,
          qretr list_quant]
  phrase sub []
  word sub []

mod_synsem sub [synsem,none]
  synsem sub [pre_mod_synsem, post_mod_synsem]
         intro [loc loc,
                nonloc nonloc]
     pre_mod_synsem sub []
     post_mod_synsem sub []
  none sub []

loc sub []
     intro [cat cat,
            cont cont,
          conx conx]

cat sub []
     intro [head head,
            spr  list_synsem,
            subj list_synsem,
            comps list_synsem,
            marking marking]

cont sub [nom_obj, quant, psoa]



% ----------------------- nom_obj ----------------------------------
--

nom_obj sub [npro,pron]
        intro [index ind,
               restr set_psoa,
               properties list_special]
      npro sub []
      pron sub [ana,ppro]
           ppro sub []
           ana sub [recp,refl]
               recp sub []
             refl sub []

% ---------------------- quant -----------------------------------
```

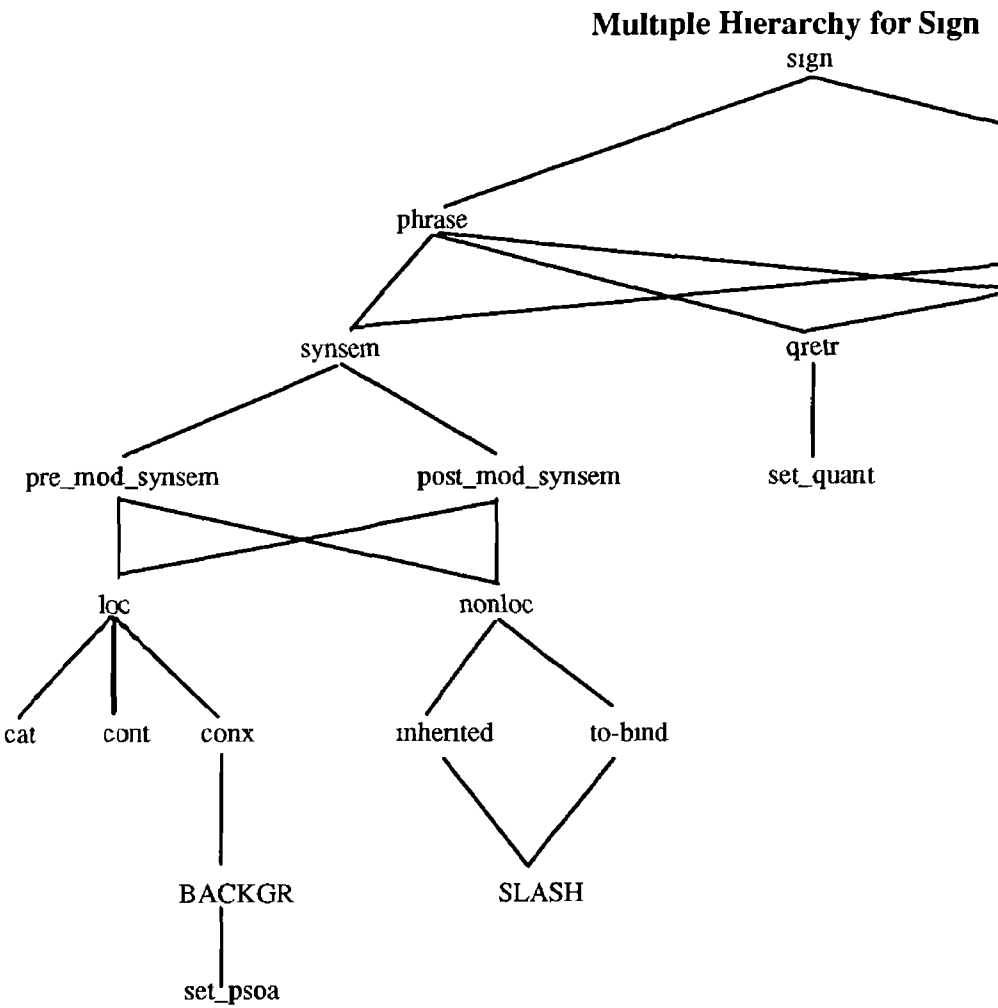
```
quant sub []
       intro [det sem_det,
              restind nom_obj]

sem_det sub [forall,exists,the]
    forall sub []
    exists sub []
    the sub []

%  ---------------------- ind   -------------------------------------
  ind sub [it, there, ref]
      intro [gen gend,
             num num,
             per per]
    it sub []
    there sub []
    ref sub []

%props sub [ref1]
%      intro [special list_special]
%      ref1 sub []

list_special sub [e_list,ne_list_special]

ne_list_special sub []
       intro [hd special,
            tl list_special]

special sub
[hume,animate,inanimate,nonhume,readable,travel,public,pages,
          building,competition,business,animal,food]
       hume sub []
       animate sub []
       inanimate sub []
       nonhume sub []
       readable sub []
       travel sub []
       public sub []
       pages sub []
       building sub []
       competition sub []
       business sub []
       animal sub []
       food sub []

gend sub [masc,fem,neut]
   masc sub []
   fem sub []
   neut sub []

num sub [sg,pl]
   sg sub []
   pl sub []

per sub [first,second,third]
    first sub []
    second sub []
    third sub []

% ---------------------- psoa ---------------------------------

psoa sub []
     intro [quants list_quant, nucleus  qfpsoa]
```

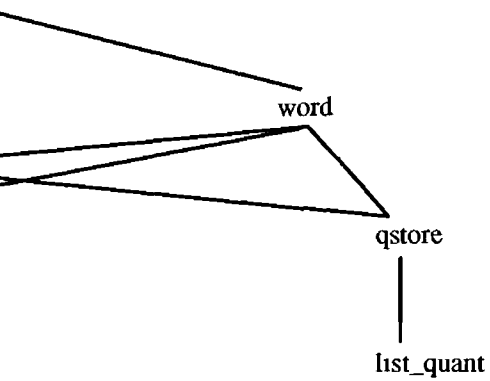
```
%---------- qfpsoa def taken from hpsg pl in ALE ----------------

 qfpsoa sub [atomic_prop,relational,naming]
  naming sub [] intro [bearer ref,name name]

  name sub [kim,sandy]
    kim sub []
    sandy sub []

   atomic_prop sub [buch,rot,gluecklich,bus,tisch,haus,rennen,laden,
                    kuh,brot]
             intro [instance ref]
      buch sub []
      rot sub []
      gluecklich sub []
      bus sub []
      tisch sub []
      haus sub []
      rennen sub []
      laden sub []
      kuh sub []
      brot sub []


relational sub [unary_prop,transitive_prop,control_qfpsoa]
      intro [agent ref,
             agentprops list_special]

unary_prop sub [schnarken,rennen]

control_qfpsoa sub [duerfen,versuchen,lassen]
      intro [soa_arg psoa]

duerfen sub []
versuchen sub []
lassen sub []

transitive_prop sub [binary_prop,ternary_prop]
          intro [patient ref,
                 patientprops list_special]

 binary_prop sub[schlagen,moegen,gehen,teilnehmen,vermieten,essen,
                 fressen]

 ternary_prop sub [geben,verkaufen]
        intro [arg3 ref,
               arg3props list_special]
schnarken sub []
schlagen sub []
moegen sub []
gehen sub []
teilnehmen sub []
vermieten sub []
essen sub []
fressen sub []
geben sub []
verkaufen sub []

conx sub []
     intro [backgr set_psoa]

% ---------------------------------------------------------------------

head sub [subst,func]
```

```
subst sub [noun,verb,adj,prep,reltvz]
        intro [prd boolean,
               mod mod_synsem]
    noun sub []
       intro [case case]
    verb sub []
       intro [vform vform,
              inv boolean,
              aux boolean,
            caseframe grammartype,
              tns tense,
              voice voice]
    adj sub []
    prep sub []
       intro [pform pform]

pform sub [in,auf,nach,an]
        in sub []
      auf sub []
      nach sub []
      an sub []
reltvz sub []

tense sub [notense,pres,past,future,pastperf]
  notense sub []
  pres sub []
  past sub []
  future sub []
  pastperf sub []

voice sub [novoice,active,passive]
  novoice sub []
  active sub []
  passive sub []

case sub [nom,acc,genat,dat]
   nom sub []
   acc sub []
   genat sub []
   dat sub []

vform sub [bse, fin, ger, inf, pas, prp, psp]
    bse sub []
    fin sub []
    ger sub []
    inf sub []
    pas sub []
    prp sub []
    psp sub []

boolean sub [plus,minus]
   plus sub []
   minus sub []

grammartype sub [liking,hitting,going,competing,giving,intransitive,
            managing,allowing,renting,eating]
   liking sub []
   hitting sub []
   going sub []
   competing sub []
   giving sub []
   intransitive sub []
   managing sub []
```

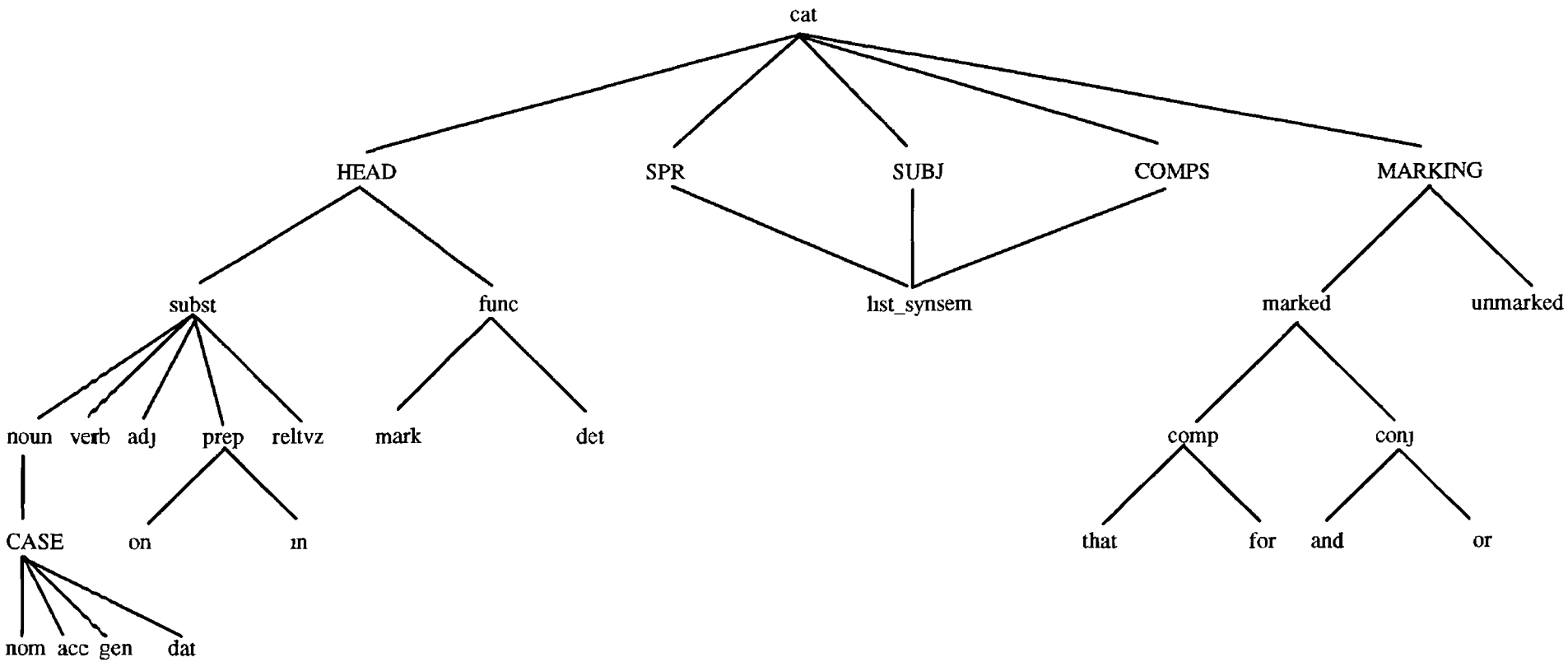
```
   allowing sub []
   renting sub []
   eating sub []

func sub [mark, det]
        intro [spec  synsem]
   mark sub []
   det sub []
     intro [specific definite]

definite sub [indef,def]
   indef sub []
   def sub []

marking sub [marked, unmarked]
   marked sub [comp,conj]
       comp sub [dass,fuer]
            dass sub []
            fuer  sub []
       conj sub [und,oder]
            und sub []
            oder sub []
   unmarked sub []


% ----------------------- nonloc -------------------------------------
-

nonloc sub []
       intro [inherited nonloc1,
              to_bind nonloc1]
nonloc1 sub []
        intro [slash set_loc]
```

A 2 Multiple Inheritance Hierarchies

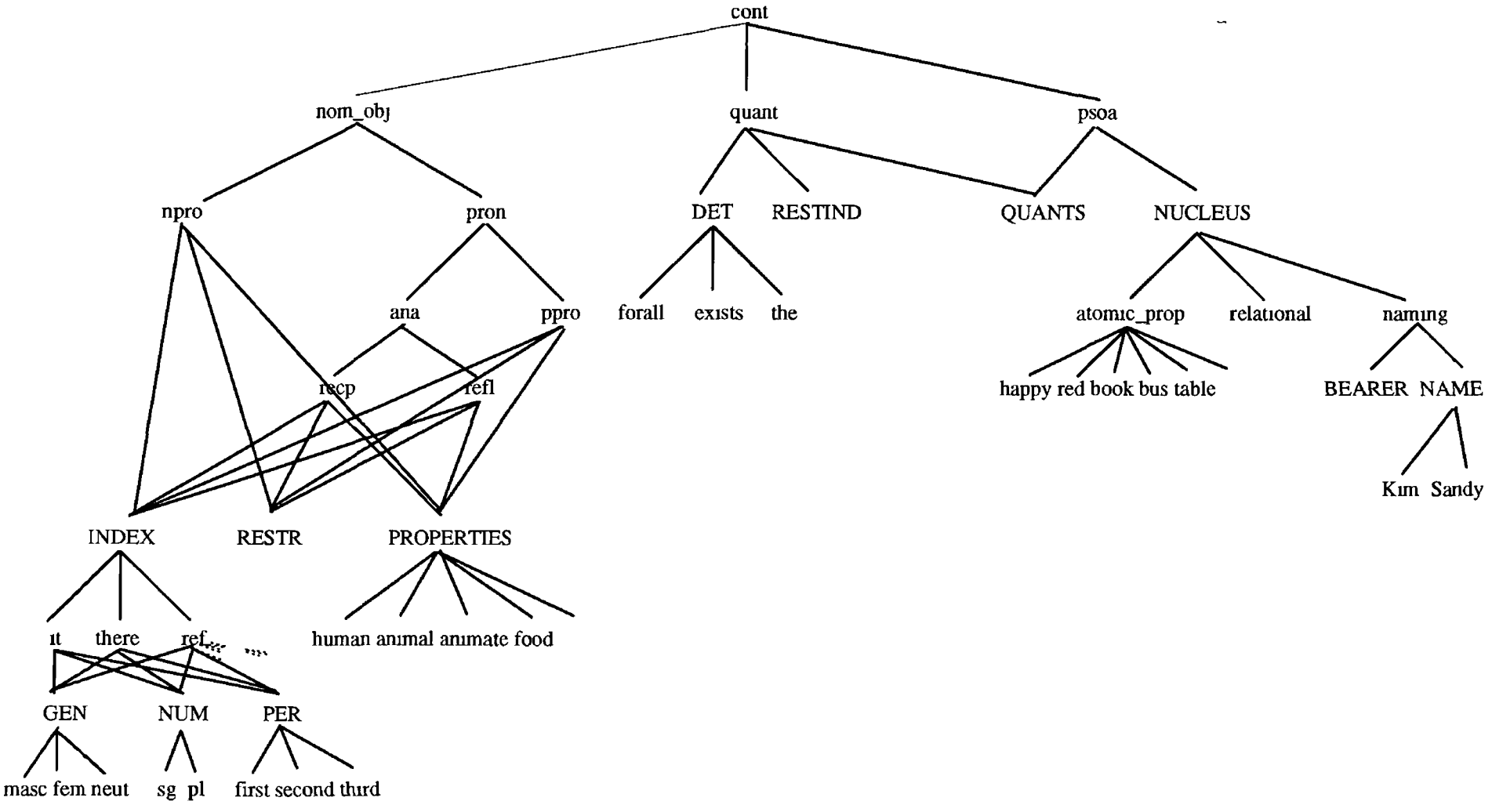



word
qstore
list_quant

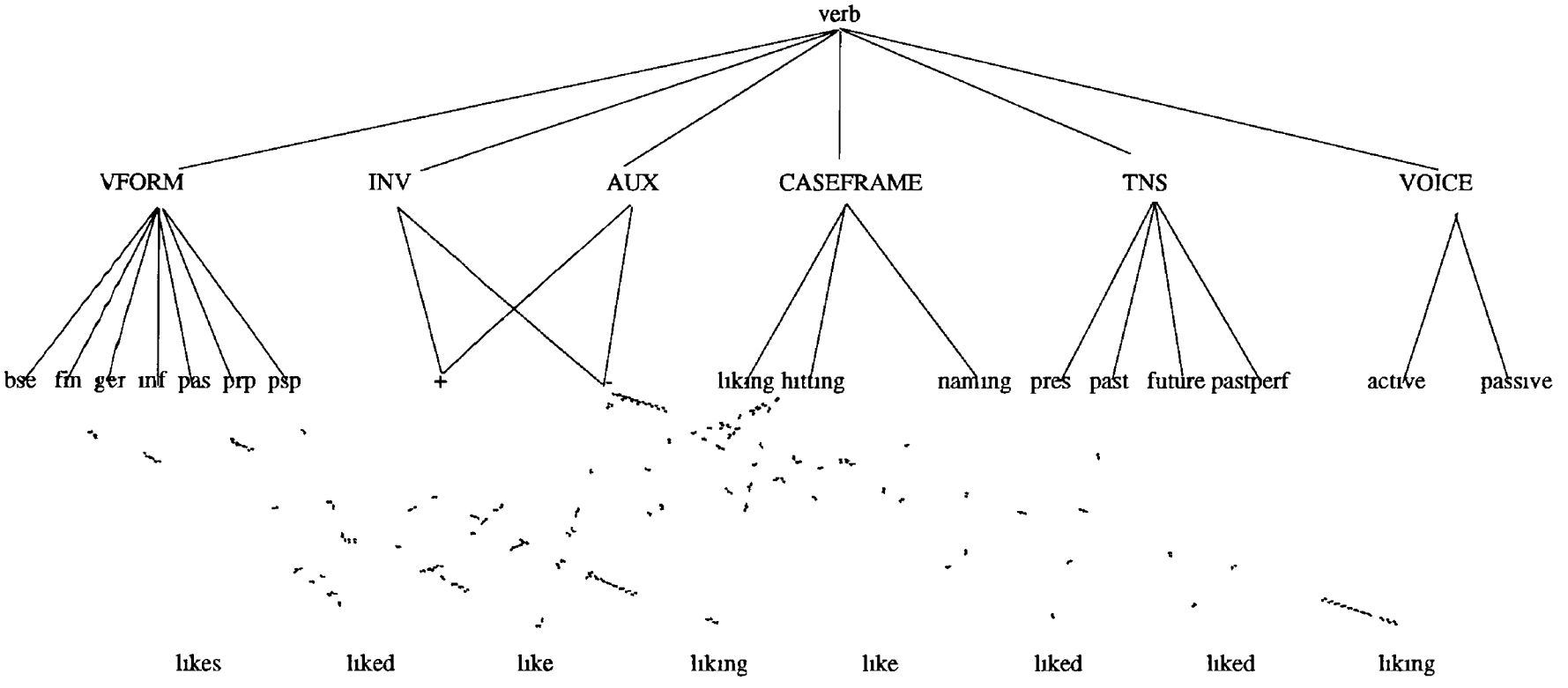
Multiple Hierarchy for Category



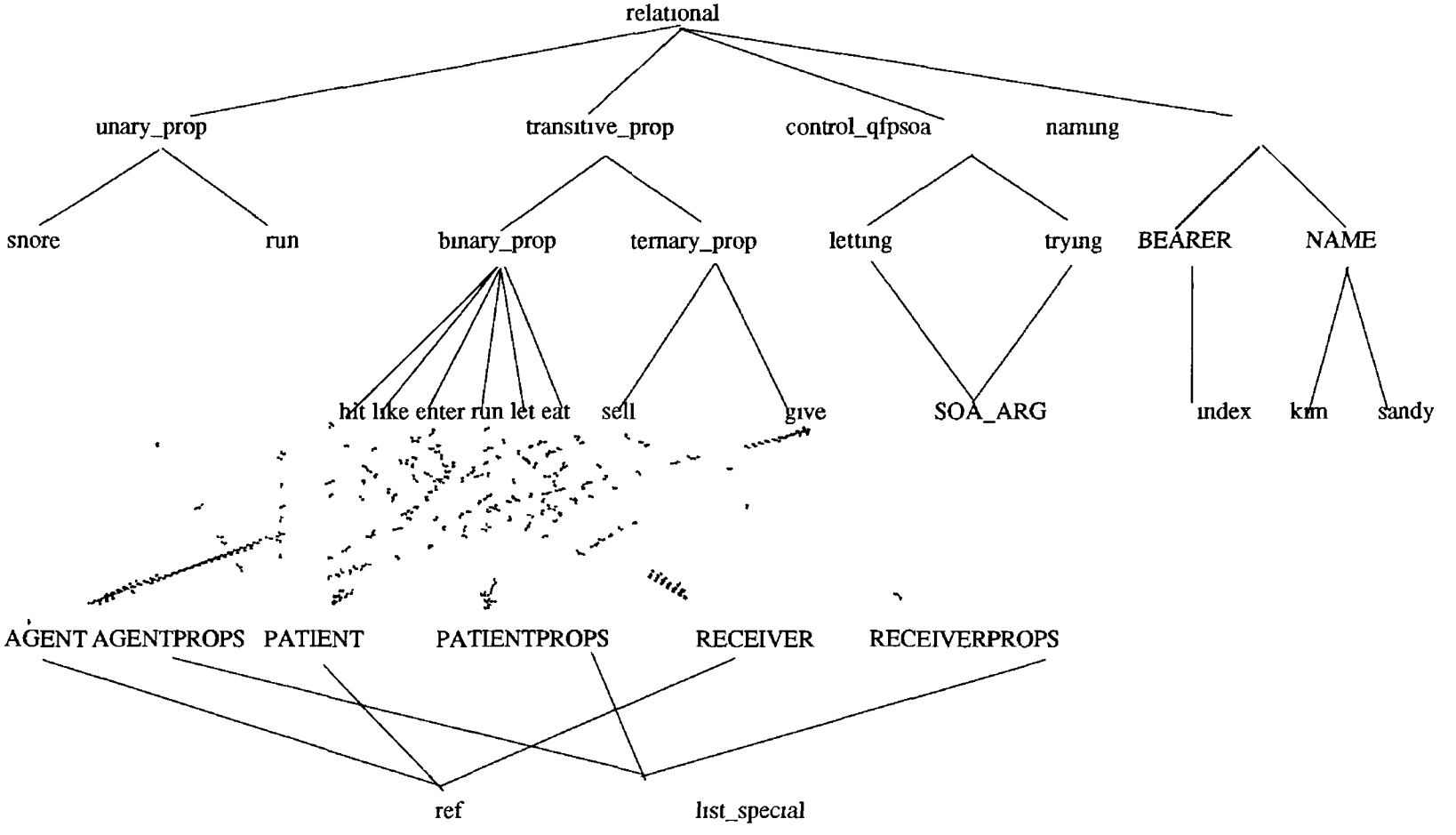
Multiple Hierarchy for Cont



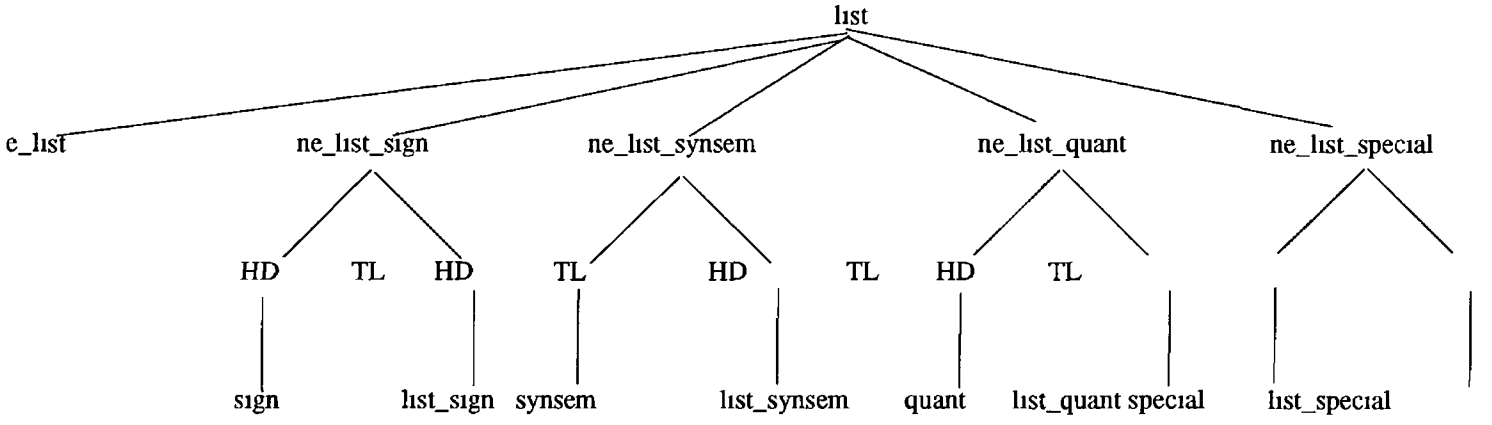
Multiple Hierarchy for Verb

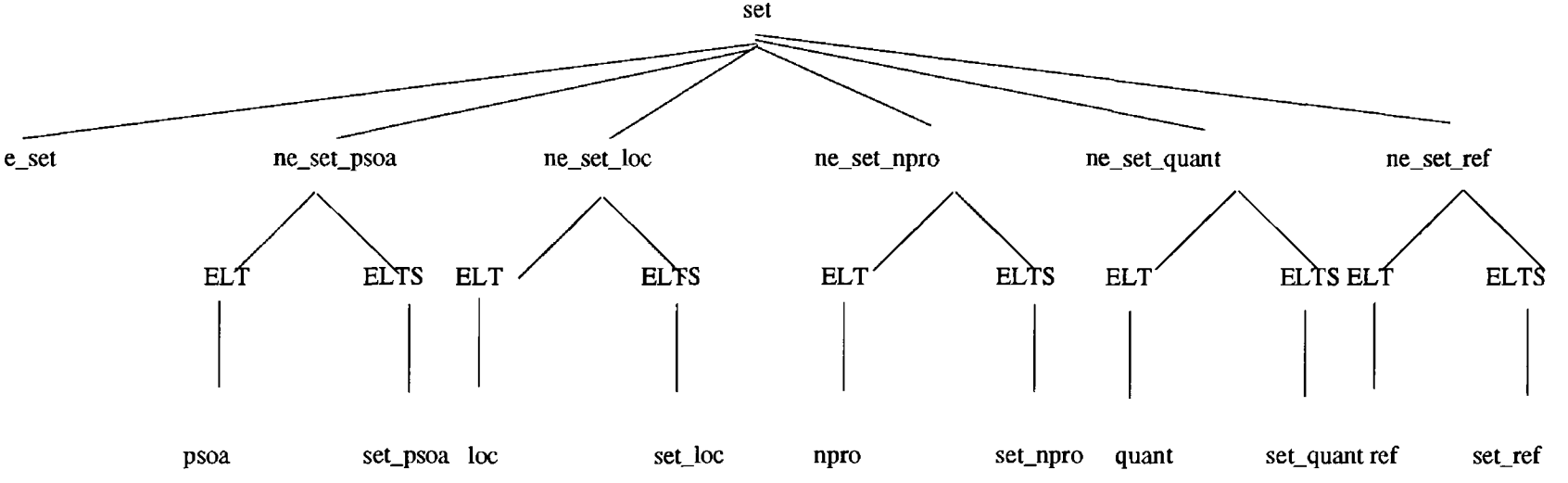


Multiple Hierarchy for Relational



Hierarchies for List and Set Types





Appendix B HPSG's Universal Grammar

B 1 Universal Principles

1 Head Feature Principle (HFP)
The HEAD value of any headed phrase is structure-shared with the HEAD value of the head daughter

2 Valence Principle
In a headed phrase, for each valence feature F, the F value of the head daughter is the concatenation of the phrase's F value with the list of SYNSEM values of the F-DTRS value

3 Quantifier Inheritance Principle
In a headed phrase, the RETRIEVED (QRETR) value is a list whose set of elements forms a subset of the union of the QUANTIFIER-STOREs (QSTORE) of the daughters, and is nonempty only if the CONTENT of the semantic head is of sort *psoa*, and the QSTORE value is the relative completement of the RETRIEVED value

4 Principle Of Contextual Consistency
The CONTEXT | BACKGROUND value of a given phrase is the union of the CONTEXT | BACKGROUND values of the daughters

5. The Semantics Principle
(a) In a headed phrase, the QRETR value is a list whose set of elements is disjoint from the QSTORE value set, and the union of those two sets is the union of the QSTORE values of the daughters,
(b) If the semantic head's SYNSEM | LOCAL | CONTENT value is of sort *psoa*, then the SYNSEM | LOCAL | CONTENT | NUCLEUS value is token-identical with that of the semantic head, and the SYNSEM | LOCAL | CONTENT | QUANTS value is the concatenation of the QRETR value and the semantic head's SYNSEM | LOCAL | CONTENT | QUANTS value, otherwise the QRETR value is the empty list, and the SYNSEM | LOCAL | CONTENT value is token identical with that of the semantic head

6 Immediate Dominance Principle
Every headed phrase must satisfy at least one of the *ID schemata*

7 The Marking Principle
In a headed phrase, the MARKING value is token-identical with that of the MARKER-DAUGHTER if any, and with that of the HEAD-DAUGHTER otherwise

8 The SPEC Principle
In a headed phrase whose nonhead daughter (either the MARKER-DAUGHTER or SUBJECT-DAUGHTER) has a SYNSEM|LOCAL|CATEGORY|HEAD value of

sort *functional*, the SPEC value of that value must be token-identical with the phrase's DAUGHTERS|HEAD-DAUGHTER|SYNSEM value

9 The NONLOCAL FEAURE Principle

In a headed phrase, for each nonlocal feature F=SLASH, QUE or REL, the value of SYNSEM|NONLOCAL|INHERITED|F is the set difference of the union of the values on all the daughters and the vale of SYNSEM|NONLOCAL|TO-BIND|F on the HEAD-DAUGHTER

B 2 Immediate Dominance Schemata

Head-Subject Schema
(SCHEMA 1) a phrase with DTRS value of sort *head-subj-struc* in which the HEAD-DTR is a phrasal sign

Head-Complement Schema
(SCHEMA 2) a phrase with DTRS value of sort *head-comp-struc* in which the HEAD-DTR is a lexical sign

Head-Subject-Complement Schema
(SCHEMA 3) A [SUBJ <>] phrase with DTRS value of sort *head-subj-comp-struc* in which the head daughter is a lexical sign

Head-Specifier Schema
(SCHEMA 4) a phrase whose HEAD-DTR is a lexical head, the SPR value of which is structure shared with the phrase's non-head daughter, SPR-DTR

Head-Adjunt Schema
(SCHEMA 5) a phrase with DTRS value of sort *head-adjunct-struc*, such that the MOD value of the adjunct daughter is token-identical to the SYNSEM value of the head daughter

Head-Marker Schema
(SCHEMA 6) a phrase with DTRS value of sort *head-marker-struc* whose marker daughter is a marker with its MARKING value token-identical to that of the mother

Head-Filler Schema
The DAUGHTERS value is an object of sort *head-filler-struc* whose HEAD-DAUGHTER |SYNSEM |LOCAL |CATEGORY value satisfies the description [HEAD *verb*[VFORM *finite*, SUBJ <>, COMPS <>]], whose HEAD-DAUGHTER |SYNSEM | NONLOCAL | INHERITED |SLASH value contains an element token-identical to the FILLER_DAUGHTER |SYNSEM |LOCAL value, and whose HEAD-DAUGHTER |SYNSEM |NONLOCAL |TO-BIND |SLASH value contains only that element

Appendix C The Lexicon

C 1 Lexical Entries

```
kim --->
   word,
   synsem ((@ np((per third,num sg,gen masc),
           ([hume,animate])),@ mod(none)),
            loc (cont (index Ind,
                       restr e_set,
                       properties list_special),
              conx backgr (elt (nucleus (naming,
                                bearer Ind,
                                name kim),
                         quants []),
                       elts e_set)),
            @ no_slash),
   qstore e_set

sandy --->
   word,
   synsem ((@ np((per third,num sg,gen fem),
           ([hume,animate])),@ mod(none)),
           loc (cont (index Ind,
                      restr e_set,
                  properties list_special),
             conx backgr (elt (nucleus (naming,
                               bearer Ind,
                               name sandy),
                         quants []),
                      elts e_set)),
           @ no_slash),
   qstore e_set

moegen --->
   word,
   synsem (loc ((cat) (head (verb,
                    mod none,
                             vform bse,
                             aux minus,
                             inv minus,
                    caseframe liking,
                    tns tense,
                    voice voice),
                       spr [(@ comp)],
                       subj [(@ np(Ind1,Props1),
                              @ case(nom))]
                       comps [(@ np(Ind2,Props2),
                                @ case(acc))],
                       marking unmarked),
                 cont (nucleus (moegen,
                                agent Ind1,
                                agentprops Props1,
```

```
                                patient Ind2,
                                patientprops Props2),
                        quants []),
            conx backgr e_set),
              @ no_slash),
     qstore e_set


teilnehmen --->
   word,
   synsem (loc ((cat) (head (verb,
                    mod none,
                             vform bse,
                             aux minus,
                             inv minus,
                    caseframe competing,
                    tns notense,
                    voice novoice),
                       spr [(@ comp)],
                       subj [(@ np(Ind1,Props1),
                                @ case(nom))],
                       comps [(@ pp(an),
                                 @comps([(@ np(Ind2,Props2),
                                           @ case(dat))]))],
                       marking unmarked),
                cont (nucleus (teilnehmen,
                                 agent Ind1,
                                 agentprops Props1,
                                 patient Ind2,
                                 patientprops Props2),
                        quants []),
             conx backgr e_set),
             @ no_slash),
     qstore e_set

geben --->
   word,
   synsem (loc ((cat) (head (verb,
                    mod none,
                             vform bse,
                             aux minus,
                             inv minus,
                    caseframe grammartype,
                    tns notense,
                    voice novoice),
                       spr [(@ comp)],
                       subj [(@ np(Ind1,Props1),
                                @ case(nom))],
                       comps [(@ np(Ind2,Props2),
                                 @ case(dat)),
                                (@ np(Ind3,Props3),
                                 @ case(acc))],
                       marking unmarked),
                cont (nucleus (geben,
```

```
                                agent Ind1,
                                agentprops Props1,
                                patient Ind2,
                                patientprops Props2,
                                arg3 Ind3,
                                arg3props Props3),
                        quants []),
            conx backgr e_set),
              @ no_slash),
    qstore e_set

schnarchen --->
    word,
    synsem (loc ((cat) (head (verb,
                              mod none,
                              vform bse,
                              aux minus,
                              inv minus,
                              caseframe actionintrans,
                              tns _,
                              voice _),
                         spr [(@ comp)],
                         subj [(@ np(Ind,Props),
                                  @ case(nom)],
                         comps [],
                         marking unmarked),
                cont (nucleus (schnarken,
                               agent Ind,
                               agentprops Props),
                      quants []),
                conx backgr e_set),
             @ no_slash),
    qstore e_set

gehen --->
    word,
    synsem (loc ((cat) (head (verb,
                     mod none,
                              vform bse,
                              aux minus,
                              inv minus,
                              caseframe going,
                              tns _,
                              voice _),
                         spr [(@ comp)],
                         subj [(@ np(Ind1,Props1),
                                  @ case(nom))],
                         comps [(@ pp(in),
                                   @comps([(@ np(Ind2,Props2),
                                            @ case(dat))])],
                         marking unmarked),
                cont.(nucleus (gehen,
                               agent Ind1,
                               agentprops Props1,
```

```
                                goal Ind2,
                                goalprops Props2),
                        quants []),
                conx backgr e_set),
                @ no_slash),
     qstore e_set

schlagen --->
   word,
   synsem (loc ((cat) (head (verb,
                    mod none,
                              vform bse,
                              aux minus,
                              inv minus,
                              caseframe hitting,
                              tns tense,
                              voice voice),
                         spr [(@ comp)],
                         subj [(@ np(Ind1,Props1),
                                  @ case(nom))],
                         comps [(@ np(Ind2,Props2),
                                   @ case(acc))],
                         marking unmarked),
                cont (nucleus (schlagen,
                               agent Ind1,
                               agentprops Props1,
                               patient Ind2,
                               patientprops Props2),
                      quants []),
                cont backgr e_set),
           @ no_slash),
     qstore e_set


vermieten --->
   word,
   synsem (loc ((cat) (head (verb,
                    mod none,
                              vform bse,
                              aux minus,
                              inv minus,
                              caseframe renting,
                              tns tense,
                              voice voice),
                         spr [(@ comp)],
                         subj [(@ np(Ind1,Props1),
                                  @ case(nom))],
                         comps [(@ np(Ind2,Props2),
                                   @ case(acc))],
                         marking unmarked),
                cont (nucleus (vermieten,
                               agent Ind1,
                               agentprops Props1,
                               patient Ind2,
```

```
                                patientprops Props2),
                        quants []),
                    conx backgr e_set),
                @ no_slash),
     qstore e_set

lassen --->
   word,
   synsem (loc ((cat) (head (verb,
                    mod none,
                            vform bse,
                            aux minus,
                            inv minus,
                            caseframe allowing,
                            tns tense,
                            voice voice),
                        spr [(@ comp)],
                        subj [(@ np(Ind1,Props1),
                                @ case(nom))],
                        comps [(@ np(Ind2,Props2),
                                 @ case(acc),
                                (@ vp(Prop),
                                 @ head(vform bse),
                                 @ subj(
                                  @ np(Ind2,Props2)))],
                        marking unmarked),
                cont (nucleus (lassen,
                                 agent Ind1,
                           agentprops Props1,
                                 soa_arg Prop),
                       quants []),
                conx backgr e_set),
            @ no_slash),
     qstore e_set

essen --->
   word,
   synsem (loc ((cat) (head (verb,
                    mod none,
                            vform bse,
                            aux minus,
                            inv minus,
                    caseframe humeating,
                    tns tense,
                    voice voice),
                       spr [(@ comp)],
                       subj [(@ np(Ind1,Props1),
                               @ case(nom))],
                       comps [(@ np(Ind2,Props2),
                                @ case(acc))],
                       marking unmarked),
                cont (nucleus:(essen,
                                 agent Ind1,
                                 agentprops Props1,
```

```
                                patient Ind2,
                                patientprops Props2),
                        quants []),
                   conx backgr e_set),
              @ no_slash),
     qstore e_set

fressen --->
   word,
   synsem (loc ((cat) (head (verb,
                     mod none,
                           vform bse,
                           aux minus,
                           inv minus,
                           caseframe animeating,
                           tns tense,
                           voice voice),
                        spr [(@ comp)],
                        subj [(@ np(Ind1,Props1),
                                 @ case(nom))],
                        comps [(@ np(Ind2,Props2),
                                  @ case(acc))],
                        marking unmarked),
                  cont (nucleus (fressen,
                                 agent Ind1,
                              agentprops Props1,
                                 patient Ind2,
                                 patientprops Props2),
                        quants []),
                  conx backgr e_set),
              @ no_slash),
     qstore e_set



ein --->
   word,
   synsem (loc ((cat) (head (det,
                                 spec ((@ cont(Cont),@
nbar((num sg),_))),
                     specific indef),
                        spr [],
                        subj [],
                        comps [],
                        marking unmarked),
                   cont (GQ,
                            det  exists,
                           restind  Cont),
                  conx backgr e_set),
              @ no_slash),
   qstore (elt·GQ,
        elts e_set)
```

```
das --->
   word,
   synsem:(loc:((cat):(head:(det,
                             spec:((@ cont(Cont),
                          @ nbar( _,_))),
                    specific:def),
                       spr:[],
                       subj:[],
                       comps:[],
                       marking:unmarked),
                 cont:(GQ,
                 det: the,
                       restind: Cont),
           conx:backgr:e_set),
             @ no_slash),
   qstore:(elt:GQ,
           elts:e_set).
buch --->
    word,
    synsem:((@ mod(none),
             @ nbar((per:third,num:sg,gen:neut),
                 ([inanimate,readable,pages])),
             @ spr([(@ detp(_))]),
             loc:(cont:(npro,
                  index:Ind,
                        restr:(elt:(nucleus:(buch,

instance:Ind),
                                    quants:[]),
                                 elts:e_set),
                  properties:_),
               conx:backgr:e_set),
             @ no_slash)),
   qstore:e_set.



brot --->
    word,
    synsem:((@ mod(none),
             @ nbar((per:third,num:sg,gen:neut),
                 ([food])),
             @ spr([(@ detp(_))]),
             loc:(cont:(npro,
                  index:Ind,
                        restr:(elt:(nucleus:(brot,

instance:Ind),
                                    quants:[]),
                                 elts:e_set),
                  properties:_),
               conx:backgr:e_set),
             @ no_slash)),
   qstore:e_set.
```

```
kuh --->
    word,
    synsem ((@ mod(none),
             @ nbar((per third,num sg,gen fem),
                ([animal])),
             @ spr([(@ detp(_))]),
             loc (cont (npro,
                  index Ind,
                        restr (elt (nucleus (kuh,

instance Ind),
                                    quants []),
                                 elts e_set),
                     properties _),
               conx backgr e_set),
             @ no_slash)),
   qstore e_set


bus --->
    word,
    synsem ((@ mod(none),
             @ nbar((per third,num sg,gen masc),
              ([inanimate,travel,public])),
             @ spr([(@ detp(_))]),
             @ cont((npro,
               index Ind,
                     restr (elt (nucleus (bus,
                                          instance Ind),
                                    quants []),
                               elts e_set),
               properties _)),
             loc conx backgr e_set,
             @ no_slash)),
   qstore e_set



tisch --->
    word,
    synsem ((@ mod(none),
             @ nbar((per third,num sg,gen masc),
([inanimate])),
             @ spr([(@ detp(_))]),
             @ cont((npro,
               index Ind,
                     restr (elt (nucleus (tisch,
                                          instance Ind),
                                    quants []),
                               elts e_set),
               properties _)),
            loc conx backgr e_set,
            @ no_slash)),
   qstore e_set
```

```
haus --->
    word,
    synsem ((@ mod(none),
             @ nbar((per third,num sg,gen neut),
([inanimate,building])),
             @ spr([(@ detp(_))]),
             @ cont((npro,
               index Ind,
                     restr (elt (nucleus (haus,
                                          instance Ind),
                                 quants []),
                           elts e_set),
              properties _)),
          loc conx backgr e_set,
            @ no_slash)),
   qstore e_set

rot --->
  word,
  synsem (loc (cat (head (adj,
                          prd minus,
                          mod (pre_mod_synsem,
                               @ nbar(Ind,Props),
                               @ restr(Restrs), @
case(_))),
                   spr [],          % Not quite right
degree sprs
                   subj [],
                   comps [],
                   marking unmarked),
               cont (index Ind,
                     restr (elt (nucleus (rot,
                                          instance Ind),
                                 quants []),
                           elts Restrs),
              properties Props),
             conx backgr e_set),
             @ no_slash),
   qstore e_set
gluecklich --->
  word,
  synsem (loc (cat (head (adj,
                          prd minus,
                          mod (pre_mod_synsem,
                               @ nbar(Ind,Props),
                               @ restr(Restrs), @
case(_))),
                   spr [],
                   subj [],
                   comps [],
                   marking unmarked),
               cont (index Ind,
                     restr (elt (nucleus (gluecklich,
                                          instance Ind),
```

```
                                    quants []),
                                elts Restrs),
                    properties Props),
                conx backgr e_set),
                 @ no_slash),
    qstore e_set


in --->
    word,
    synsem (loc (cat (head (prep,
                      pform in,
                      prd minus,
                      mod none),
                    spr [],
                    subj [],
                    comps [(@ np(Ind,_),@ restr(Restr),@
case(acc),
                          @ building)],
                    marking unmarked),
              cont (index Ind, restr Restr),
              conx backgr e_set),
                @ no_slash),
    qstore e_set


an --->
     word,
     synsem (loc (cat (head (prep,
                       pform an,
                       prd minus,
                       mod none),
                     spr [],
                     subj [],
                     comps [(@ np(Ind,Props),@ restr(Restr),
                          @case(_))],
                     marking unmarked),
               cont (index Ind, restr Restr),
               conx backgr e_set),
                 @ no_slash),
     qstore e_set
```

C 2 Lexical Rules

1 pres_3s lexical rule
2 pres_non3s lexical rule
3 past_3s lexical rule
4 past_non3s lexical rule
5 plur_noun lexical rule
6 passive lexical rule
7 neuter_nom_acc_indef lexical rule
8 masc_genat_indef lexical rule
9 dative_indef lexical rule
10 accus_indef lexical rule
11 fem_nom_acc_indef lexical rule
12 fem_gen_dat_indef lexical rule
13 fem_nom_acc_def lexical rule
14 fem_gen_dat_def lexical rule
15 masc_acc_def lexical rule
16 genat_def lexical rule
17 dat_def lexical rule
18 neut_nom_acc_def lexical rule
19 liking_frame lexical rule
20 naming_frame lexical rule
21 hitting_frame lexical rule
22 actionintrans lexical rule
23 humeating lexical rule
24 animeating lexical rule
25 managing_frame lexical rule
26 competingframe lexical rule
27 goingframe lexical rule
28 namingframe lexical rule
29 sing_nom_def_adj lexical rule
30 sing_acc_gen_dat_adj lexical rule
31 plur_adj lexical rule
32 sing_fem_acc_adj lexical rule
33 masc_nom_indef_adj lexical rule
34 neut_nom_acc_indef_adj lexical rule