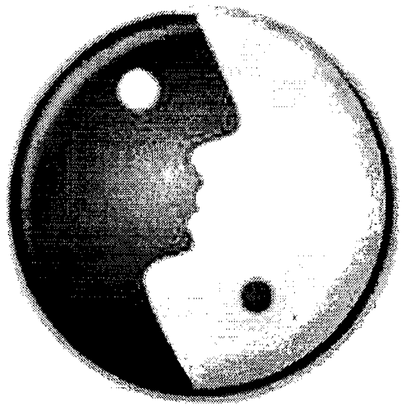
b149 9561X

Applied Linguistic Principles and Designing CALL Programmes for the ESL Classroom



Frederick Mark Muller

A dissertation submitted to meet the requirements for the degree of Magister Artium in the Faculty of the Humanities (Department of English and Classical Languages) at the University of the Free State, Bloemfontein.

Supervisor: Prof. W.J. Greyling
Date: May 2004

DECLARATION

I, Frederick Mark Muller, hereby declare that this dissertation is my own work and that it has not been submitted to another university for examination.

Frederick Mark Muller

The financial assistance of the Centre for Science Development toward this research is hereby acknowledged. Opinions expressed and conclusions arrived at are those of the author and are not necessarily to be attributed to the Centre for Science Development.

Abstract

Applied Linguistics is largely concerned with teaching English as a second language (TESL) (Cruttenden, 1994, p6). This is not a simple field. There are a number of variables, such as the personalities of the individual students and teachers involved and the approach to learning used. Computer-assisted instruction (CAI) has been used for some years in a variety of approaches and learning environments. In these the primary focus of CAI has been on providing materials for learning in methods that stimulate learning more effectively - either by providing enhanced access to texts or by providing rapid feedback to set problems. The one facet of teaching where CAI is not extensively used (except in a facilitatory role) is providing an environment in which students can practise generating texts and have these understood.

This dissertation investigates the potential of using computers to process text in such a way as to enable evaluating the cohesion and coherence of texts. It takes an interdisciplinary approach which exploits methods and insights from applied linguistics, artificial intelligence (AI) and computer-assisted language learning (CALL) to explore the potential of automating textual analysis, comparison and evaluation.

This dissertation develops the hypothesis that a dependency-based grammar can be used to generate a computerised representation of the *sense* contained in a text and that this representation is sufficient to allow contextual comparison of texts. This comparison can be used, in turn, to evaluate texts by means of comparing the representation to that of a model answer, thus providing a means of evaluating the cohesion and coherence of the text. The potential of using such a system in constructing CALL programmes and the extent to which it can assist in the process of second language acquisition (SLA) is also discussed.

Existing research studied during the writing of this dissertation included an examination of existing uses of computers in language teaching, particularly those associated with developing communicative competence. These studies pointed to a need for a utility that would enable teaching aids to evaluate texts contextually. Various methods of performing this evaluation were considered. This included the examination of a selection of grammatical systems with a view to determining their strengths in building a representation of the *sense* contained in a text. In addition, current applications using natural language processing (NLP) and AI were examined with a view to how these could be adapted or used to enable CALL programmes to evaluate coherence and cohesion in texts. Furthermore, guidelines are proposed for developing CALL programmes using this type of evaluation.

These requirements are used as a template for implementing a programme aimed at performing a contextual evaluation by means of a comparison of texts. This programme is discussed in terms of the grammatical model used as well as the implications this holds for future development.

Lastly, the implications using this kind of system in CALL programmes would have for teaching and teacher training are examined and suggestions for the future improvement and development of this sort of application are made.

The main conclusion of this dissertation is that computerised contextual evaluation of texts is possible, though with the caveat that the evaluation is limited by the extent to which world-knowledge can be represented.

Contents

1 Introduction

This dissertation attempts to identify and address some of the shortcomings of Computer Assisted Language Learning (CALL) software for the purposes of improving the effectiveness of CALL in the Second Language (SL) classroom. This involves a cross section of a number of fields, including applied linguistics, computer programming, artificial intelligence and natural language processing. Before examining some of the relevant theories and technologies that form the basis of this dissertation a brief overview will be given of some of the key fields in order to place this dissertation in the context of these fields.

1.1 Background

This section briefly describes the key influences and considerations governing the development of the application on which this dissertation is based. Each of these sections, as well as related issues, are covered in greater detail along with a literature review in section 2.

1.1.1 Applied linguistic principles

The applied linguistic principles alluded to in the title of this dissertation are heavily influenced by interactionist theories of Second Language Acquisition (SLA). Very briefly, these hypothesise that language learning takes place when both input, cognition and output occur in an environment that both encourages learners to exert and extend their abilities in the language and which provides a means of enabling learners to evaluate the accuracy of their output both grammatically and in terms of its being a meaningful contribution to an interaction. This identifies a number of necessary processes in the SLA process, such as input, cognition, output and reflection (Long, 1996; Gass, 1997; Chapelle, 1998), though the various interactionist theorists differ in the importance they attach to the different processes and the nature of the processes.

This dissertation does not attempt to choose between these hypothesis. Instead it describes the development and implementation of a computer programme intended to assist in the process of SLA. As such it provides tools that can be used to enhance a learner's interaction with texts and thus assist in the processes identified in interactionist theories as being essential to language acquisition, with particular emphasis on the processes of learner output and reflection. The importance of learner output in interactionist theories and in the application developed in this dissertation is covered in more detail in section 2.1.1.

1.1.2 Computer-assisted language learning

The effectiveness of software tools in CALL depends both on the perspective of SLA of the designers and of the teacher (cf. Kenning, 1990, p. 73).

Computers are designed to perform complex calculations and complex logic operations. As such they have an enormous potential for assisting in automating classroom activities (giving the teacher more time for actual teaching), providing student-centred resources (responding intelligently to student input) and providing immediate purposeful feedback. This has led to the development of many applications as tools for teaching language.

CALL thus includes programmes specifically designed for educational use, such as reading programmes aimed at improving reading speed and comprehension, grammar tutors aimed at practising various features of grammar (such as spelling and particular grammatical forms) and literary guides and tutorials which give structured access to literary texts of various types (Warschauer, 1996, pp. 3-20).

Kenning notes:

> *Partly because of the lack of machines and partly because of the deficiencies of the computer as an instructional medium, interest has turned towards developing the use of the computer as a resource or as a stimulus for group activities* (1990, p. 73).

Consequently CALL also includes the use of programmes which facilitate learning, such as writing tools which provide easy tools for writing and formatting text (word-processors, presentation software, email, etc), encyclopedias which provide multimedia access to information and a host of support programmes which promote learner computer capability.

1.1.3 Feedback

Feedback often ultimately determines the effectiveness of any learning activity. Good feedback will reinforce a lesson and enhance the learning experience. Poor feedback can retard learning or totally negate the effect of a lesson.

Broadly speaking, we can divide feedback into explicit and implicit feedback. Explicit feedback reports on correctness. involves some comment on the learner's response (Holland et al., 1999, p. 341). This ranges from a simple mark or comment (like 'good') to an essay detailing the learner's achievements and recommending improvements. Implicit feedback, on the other hand, does not involve any direct comment on the learner's output. Rather, the learners see the natural consequences of their language use in which their utterance is either understood as intended or misunderstood (Holland et al., 1999, p. 341). The learner is thus left to determine the effectiveness and appropriateness of his response from the nature of his respondent's communication. If the respondent's communication shows that the learner's response has been correctly interpreted and is appropriate in that context, then the respondent's communication will reflect this and in so doing provide positive reinforcement of the learner's communicative behaviour. Obviously, if the respondent's communication shows a communication gap, this will also be reflected in the communication and will, in turn, provide an incentive for the student to clarify himself - again reinforcing the desired communicative behaviour.

Developing computer models that enable implicit feedback is not easy as a basic assumption here is that the respondent is capable of understanding the communication. However, as Robinson (1991, pp. 155-167) demonstrates, implicit learning is generally more effective in language learning than explicit feedback. This makes developing systems capable of generating or enabling implicit feedback a desirable design goal in CALL software.

1.1.4 Internet

As an environment that provides easy integration of text, hypertext, pictures, video and sound, the Internet has rapidly become a powerful resource for CAI and, in particular, Computer-assisted language learning (CALL) (Warschauer, 1996, pp. 6-7). A good illustration of this is the number of sites to be found in the TEFL and TESL rings.[1] These sites provide a range of exercises and materials for use in classroom activities - including drills, cross-cultural newsletters, discussions of travels, "graffiti walls", lists and multi-user game environments (such as SMOOZE U) where students can interact with fellow students from other countries (Colburn, 1998).

While the interactional aspect of the Internet is being thoroughly explored, the aspect of online lessons is still in its infancy. This lag can be ascribed to a number of factors. Part of the cause must lie in the relatively slow speed of Internet connections which makes it unfeasible to send large files. As a result, despite recent advances in compression algorithms and scripting environments, large programmes, such as multimedia applications, are almost impossible to use online (LeLoup and Ponterio, 1998, p. 4).[2]

[1]A ring is a set of linked web-sites. These usually have similar themes or interests. In the case of the TEFL and TESL rings the common denominator is an interest in English and in learning English.

[2]While computer hardware has made substantial performance improvements since these articles were

Part of the problem must also lie in the shortage of readily available tools for building applications. In many cases, those tools that do exist use proprietary formats that prevent the applications generated by them from being accessible to people who do not have the tools themselves. These tools also suffer in that they cannot generate complex responses or make anything more than rudimentary evaluations of answers supplied by students. In practice, this means that automated responses are limited to variations on the theme of multiple-choice questions. Any application that makes an attempt to accept textual answers has to be very carefully structured to limit the range of responses to a small subset of possible responses and requires more sophisticated programming than is easily possible in authoring packages.

This rigid structure makes for applications which are suitable for practising or testing a fairly basic level of proficiency in a second language, but does not allow learners much opportunity to experiment with the language or to practise its use outside of the scope of the exercises. It is desirable to foster an environment in which learners may "attempt to use target language forms that may stretch his or her competence" (Chapelle, 1998, p. 27) and, what is more, expect to have their responses comprehended (Chapelle, 1998, p. 27).

This brings us to the last problem associated with the growth of online learning. Even though the Internet has become commonplace, it is still a new technology. Relatively few language teachers are prepared to embrace new technologies.[3] Of these, most do not have much knowledge of the limitations and possibilities of the Internet environment and fewer still have the knowledge needed to develop applications that use the environment. This is inevitable and unlikely to change much in the foreseeable future.[4]

While a basic understanding of the possibilities and use of the Internet is not difficult to gain, few practising teachers have the time to do the study and experimentation that will enable them to use the medium effectively. This trend was evident in the relative scarcity

written, the Internet infrastructure depends largely on the capability of public phone lines to transmit data. In some countries broadband and satellite provide faster access than analogue phone cables allow but these are the exception rather than the rule. As a result, the amount of data that can be contained in web-pages has not substantially changed since these articles were written.

[3]From his experience in developing CAI applications using authoring packages and training teachers in using these packages, Arisland concludes that access to authoring tools, both the software itself and actual training in using the tools seems to have little or no impact on whether potential authors actually become Computer Assisted Learning (CAL) courseware authors.(Arisland, 1994, http://www.ifi.uio.no/ ftp/publications/others/KOArisland-1.html)

[4]Christochevsky observes that "each new technical facility gives good results only when a new generation of teachers is ready and eager to use the technical facility" (Christochevsky, 1997, http://www.media.uwe.ac.uk/masoud/cal-97/papers/christ.htm), implying a long wait before CAI becomes commonplace.

of teachers who were capable of producing their own small applications in the 1980's when programming languages and computer systems were considerably simpler than is the case today (cf. Higgins, 1985) and is apparent today (cf. Levy, 1997, p. 3).

To add to the problem, the developments of the Internet that increasingly hold the hope that distributed multimedia applications will soon be viable, also mean added complexity in the Internet protocols. As a result, effective Internet programming is rapidly becoming more complex. A fine illustration of this complexity can be seen in the programming languages that have proliferated in the last few years. Java, Python, Tcl, Perl, Visual Basic and Javascript all make for more flexible web-pages but have the combined problems of decreased portability.[5]

In spite of these problems the Internet is a fertile source of materials and lessons for Teaching English as a Second Language (TESL). Given the fact that the cost of setting up multimedia computer laboratories, capable of running software on DVD and CD, is prohibitively expensive for most schools in South Africa, the possibilities inherent in using an intranet comprising a number of smaller computers is well worth investigating.[6]

1.1.5 Should linguists programme?

There is a commonly held view that "linguists should not write *[syntax-directed]* programs" (Koster, 1991, p. 1). This is partly because linguists do not have the same skills as professional software developers, partly because a fascination with programming prevents them exercising their own specialities and partly because a grammar encapsulated as a computer programme is not accessible to other linguists. A theoretical grammatical model can be implemented and updated as desired in any existing or future programming language - provided the grammar is accessible. Higgins agrees that programmes written by teachers (other than those who are coincidentally trained programmers) will

> *run slowly, waste memory, be inelegantly laid out and will not be "bug free". Moreover no individual teacher will be able to produce material in the quantity*

[5]Visual Basic is only available on Microsoft platforms. Java, Python, Perl and Tcl are cross-platform to a greater or lesser degree but require special add-ons or environments which are not always readily accessible. Javascript is limited to a few major web-browsers. Almost all of these also have differences and incompatibilities in their implementations on the different platforms and between versions on the same platform.

[6]An example of these are the TuXlabs set up by the Shuttleworth foundation (`http://www.shuttleworth.org`). These consist of recycled computers running off a single terminal server using Open-Source software.

needed to satisfy the demands of a body of learners (Higgins, 1985, p. 73).[7]

This is not, as Higgins goes on to state, a sufficient reason to discourage teachers from developing Computer Assisted Instruction (CAI) tools. Programming helps teachers clarify their own notions of how English grammar may appear to a learner (Higgins, 1985, p. 75) and gives teachers the opportunity to deepen perceptions, to make discoveries, and to learn techniques which may turn out to be applicable in places where there is no electricity, let alone computing equipment (Higgins, 1985, p. 76).

Just as teachers do not generally have sufficient knowledge of computer programming to be able to design effective programmes, very few programmers have sufficient knowledge of language teaching to enable them to develop applications that can be used effectively in a language learning environment (Warschauer, 1996, p. 6). As Vincent (1985, p. 80) points out, computer scientists are largely mathematically orientated, which makes communication with specialists in the fields of language, literature or linguistics more difficult. Consequently collaboration between these becomes much harder. She argues that not only should teachers learn about computing, including programming, but computer programmers and systems analysts need to become more linguistically and pedagogically aware. It may also be desirable for teachers to programme when professional programmers prove unsatisfactory mediators of teacher designs or when stepwise experimentation is desired in the writing process (Rope, 1985, p. 67). In particular, teachers need to be able to relate CALL to current thinking on language learning and to be better informed of how computers can be used to develop language skills (Kenning, 1990, p. 74).

1.2 Identifying the problem

From the background summary given in section 1.1.4, we can identify two primary problems with developing applications for language teaching using the Internet as medium. The first is the necessity for tools that enable teachers to construct lessons. This is more than just the primitive requirements of a hypertext or multimedia editor. It involves facilitating all aspects of lesson construction. This includes presentation (the domain of current hypertext editors) as well as testing and generating feedback (the interactive elements of the lesson). Interactive elements inevitably involve a fair amount of programming - the complexity of

[7]The application developed here suffers from all these faults. In its defence, it is intended to demonstrate potential. If this potential is sufficiently proved to exist, then a version can be developed that does not have these problems.

which will be proportional to the complexity of the responses that the lesson allows or requires.

The second problem follows from the first. The near disappearance of the question-answer dialogue, for example, must not make us overlook the fact that many of the 'new' programmes ask for a response which is then assessed in terms of right and wrong (Kenning, 1990, p. 67).

Some aspects of language learning lend themselves well to a multiple-choice style of testing. Aspects that can be tested this way include knowledge of grammatical forms - at least as far as the recognition of correct and incorrect usage is concerned - and comprehension of passages. Any learning activity that exercises a learner's ability to produce unique texts (and in particular comprehensible texts) cannot be tested in this way. This is something that cannot yet be done in a stand alone application. The closest applications to this in current use are natural language interfaces (discussed in 2.3.4 on page 50) and automated essay evaluation (discussed in section 2.3.2 on page 43).

1.3 Goals of research

The second problem identified in section 1.2 comprises the primary focus of this dissertation - an examination of the potential for software to evaluate texts in terms of coherence and cohesion, and the problems involved in developing applications capable of doing this. The dissertation includes sample applications developed in the course of this study which illustrate the difficulties involved and provide pointers to possible solutions to the problem of evaluating texts contextually in the future.

The approach to contextual evaluation used in this dissertation is to develop an application that can, at least minimally, compare the content of texts and highlight differences in those texts. That is, the application is intended to generate a comparison of the meanings of the texts rather than an analysis of the grammaticality of the texts. The method proposed for doing this is to generate a dynamic network representing the relationships between concepts as defined by the sentences of the texts and arrange these in a structure that enables mathematical set operations (such as difference, intersection and union) to be performed on the networks. The results of these operations can then be interpreted to provide insight into the differences and similarities of the texts.

This will provide a method of evaluating texts larger than a single sentence without severely limiting the range of possible sentences used in responses. This addresses a distinct need as it would enable students using the system to test the appropriateness of their responses and self-correct when errors or omissions are detected (cf.Chapelle, 1998, pp. 22-34).

An ideal computerised language teacher should be able to

> *understand a user's spoken input and evaluate it not just for correctness but also for appropriateness* (Warschauer, 1996, p. 6).

This is still an unrealistic expectation of computerised learning aids (cf. Chapelle, 1997); however, the programme developed in this dissertation will hopefully be a step toward realising this ideal as it should enable a means of testing the appropriateness of written texts, even if this is only by comparison with a model answer provided by the teacher. Warschauer's comment was made with an interactive system in mind in which a dialogue occurs between learner and computer that should ideally resemble a dialogue between human participants at talk. In this context there are a number of variables (such as register and social context) that make it hard to construct an application that can cater for all of these. Computerised lessons that only allow essay or paragraph style texts fall well short of the ideal of lessons that permit unlimited learner text production, but, as with speech recognition systems, programmes can be developed which materially benefit a SL student provided that the limitations of the technology are understood and systems are designed that work around those limitations (Ehsani and Knodt, 1998, p. 47).

Bearing this in mind, the first requirement is that the system must be able to process texts that are not represented in a predefined database. It is computationally unfeasible to have a representation of grammar that covers the gamut of human interaction, or even of grammatical usage in the intended target of the project - that of written monologues. This is especially true when, as Chomsky points out

> *The normal use of language is innovative in the sense that much of what we say in the course of normal language use is entirely new, not a repetition of anything that we have heard before, and not even similar in pattern - in any useful sense of the terms 'similar' and 'pattern' - to sentences or discourse that we have heard in the past (Chomsky, 1972, p. 12).*

Therefore, the programme must be flexible enough that it can intelligently respond to unrecognised grammatical usage and still accurately evaluate the content of these usages. In part this means that the programme must be able to learn from input text - building its knowledge database dynamically rather than relying on a pre-built knowledge base. This involves freeing the application as far as possible from the necessity of "understanding" content (in the sense of having a predefined knowledge base that provides a context for the concepts defined in the text).

A second requirement is that the programme must also be reasonably fast. This is especially important with regard to the learning requirement of the programme, as speed rapidly decreases as the sizes of databases associated with the programme increase. If knowledge is represented as relationships between words, then there will be a logarithmic increase in the size of the database as new words are added and, in particular, as the relationships between those words are added. A large internal knowledge base rapidly affects performance (Ritchie, 1987, pp. 225-256) as will be shown in some of the experimental systems developed during the course of this project.

Ultimately, this dissertation aims to increase the range of tools available to teachers for developing interactive applications for the ESL classroom. Any application using techniques developed or described in this dissertation would require an easily accessible interface. Input of lesson material and development of lesson activities ideally should not require any more computer knowledge than that needed to operate a word-processor.

Interface design is an essential part of effective educational design as a well-designed interface, in addition to making the software enjoyable to use, can promote different kinds of competence (cf. Plass, 1998, pp. 35-45). One place this type of interface is readily accessible is in the form of web-pages. As such, the application developed in this dissertation is designed to be capable of being used from inside a web-site which provides all content associated with lessons.

1.4 Research methodology

Research for this dissertation has consisted of four main processes. The first has been examining papers and applications in a variety of related fields in the hope that a synthesis of these disciplines would yield a viable application. These fields include Natural Language Processing (NLP) and Computational Linguistics (CL),[8] Artificial Intelligence (AI) programming and knowledge representation,[9] CALL,[10] linguistics (especially with regard to

[8]discussed in section 2.3.3 on page 46.
[9]discussed in section 2.3.1 on page 43.
[10]discussed in section 2.1.1 on page 15.

theories of grammar and text linguistics) and applied linguistics.[11]

There has been considerable development in language processing in the last few decades. Out of this have come a range of grammatical systems which have, to a greater or lesser degree, been found to be useful as models for computerised processing of language. The intended use of most of these is to handle database queries in reasonably natural language usage. As a result, many of these grammars cover a relatively small subset of language as it pertains to handling database queries.

The second process thus involved examining, applying and comparing these systems in an attempt to find an existing system of grammar that is computationally inexpensive and easy to represent and implement. Research for this has largely been experimental in nature.

The third process has been development and experimentation. Different methods of analysis were tried and discarded until a system was developed that generated relationship networks which adequately reflect the content of sample texts. These are described in more detail in the course of this dissertation.

Various grammatical systems were tried in this process. This involved considerable testing of sentence-level processing using a set of test sentences designed to represent a range of grammatical structures. Later in the development process, sample texts, consisting of learner-generated essays, were processed and the resulting networks examined and compared to determine if the output adequately reflected the content of the texts and if that sort of representation could be compared automatically.

Finally, the practical use of the system is discussed from an Applied Linguistic perspective, illustrating the system's use in ESL teaching and in developing ESL teaching applications. Guidelines for further development are made and an examination is made of the extent to which the programme can be used in a real-world teaching environment.

1.5 The case for modular design

1.5.1 Design philosophies

The design of CALL programmes is inevitably influenced by the design philosophies common to the popular operating systems that they run on. In the past, when mainframes and terminals were the order of the day, programmes where developed that shared resources and enabled, even encouraged, interaction between students (such as the PLATO system described by Woolley (1994, pp. 5-7)). More recently the commercial programming com-

[11]discussed in section 2.5.1 on page 57

munity has been dominated by the operating systems provided by Apple Macintosh and Microsoft. In these circles there is a tendency to design programmes that are complete resources in themselves.

This development philosophy applies to educational software as well. Developers are not able to make assumptions about the software contained on the system they design for - that is, with minor exceptions, they cannot assume that particular software will be present on these systems. Inevitably their applications have to duplicate large sections of programming to cater for the cases where these are not available. This adversely affects both the cost and time of developing educational software. It is more productive if tools can share resources - something that can only happen if those resources are both readily available and common in some widely accessible form on all instances of an operating system.

A good example of this can be seen in the design of the ubiquitous word-processor. Not only is the basic functionality of typing a document duplicated in every implementation, but the add-on features are also duplicated. Every commercial word-processor comes complete with dictionary, thesaurus and grammar-checking tools. These are not integral to the word-processing programme, often take up a considerable amount of storage capacity, are often based on software licensed from third-party producers and yet are consistently duplicated across every competing word-processor implementation.

The rise of OpenSource software is making a more productive alternative visible to the general programming community. The model adopted by the UNIX programming community has for years been one of interdependency - a model necessitated by the design of UNIX systems, in an era when storage capacity was at a premium, as multi-user systems in which a number of users could be working on one system simultaneously. If every user required unique copies of software with duplicated utilities, this would have made storage capacity prohibitively expensive.[12] The UNIX philosophy, as summarised by Mike Gancarz, is one of minimalism. Small is beautiful, with each programme designed to do only one thing and do it well (Gancarz, 1995, p. 4). For example, every UNIX system comes equipped with a spelling programme. This programme is used by any software needing a spell-checker because the interface is known and designed to be accessible from other programmes. Similarly, this can be replaced by third-party software, and third-parties exist who specialise in producing both spelling-checkers and custom dictionaries. The only proviso is that the software uses the standard common command-line interface which enables

[12]This problem is still an issue today even though the storage capacity of a small personal computer is many times greater than that of the mainframes on which UNIX originally ran. A single application and its associated utilities, such as the Corel Draw Graphics Suite, can easily eat up more than a Gigabyte of space.

it to be run interactively from other programmes. In this way a number of text-editing systems can exist concurrently, yet the functionality of this aspect of the system is provided by one utility. Furthermore, should a development occur which allows for an improvement in this particular function, there is only one part of the system that needs to be updated. The rest of the text-editing features can utilise this development without the necessity of purchasing new copies of the software.

The tools used to type this dissertation are another good example of this philosophy. These consist of a number of independent programmes, maintained by different developers, designed to work together. A few of these are: the interface used to type the dissertation (LyX, available from `http://www.lyx.org`), a bibliographic database (BibTeX, available from CTAN at `http://www.ctan.org`), the document formatting programme (LaTeX, an extremely flexible set of utilities capable of formatting almost any type of document, from text to mathematical formulae to music and also available from CTAN), Aspell (a freely distributed version of the spelling programme described earlier) and XFig (a drawing application, available at `http://www.xfig.org`), to name but a few from the list of independent software used. In each of these cases alternatives exist.

Ideally educational software should be able to have a similar level of inter-dependency and shared resources.

1.5.2 Internet use

One place where the viability of having standardised functionality in competing applications is visible is the general development of the Internet. Its popularity has been dependent on the availability of the necessary programmes that interface with it. In an almost unique case in the commercial programming environment, these resources were both readily and cheaply available from its earliest days. Initially this was in the form of the *Mosaic* web-browser which was free and supported on Microsoft, Macintosh and Unix. Later *Mosaic*'s successor, the *Netscape* web-browser, was also freely available for academic use and ran on a wide variety of operating systems. Currently, there are a number of free options as all major browser manufacturers have opted to allow free access to their browsers. These include Microsoft's *Internet Explorer* (available from `http://www.microsoft.com`), which has been free for Microsoft customers for some time, *Opera*, a lightweight, fast browser produced by Opera Software (available from `http://www.opera.com/`) and *Netscape* (available from `http://www.netscape.com`). Netscape has released the source-code of their browser under the GNU public license with the result that a number of versions of this browser now exist (see `http://www.mozilla.org` for the "official" version[13]).

Just as importantly, the software needed by web-servers to serve the documents is also freely available and has been since the inception of the Internet. The most popular example of these is the *Apache* web-server (cf. `http://www.apache.org/`) which is used by 56% of the world's web-sites (Netcraft, 2003). Naturally, commercial versions are also available, often offering performance benefits in terms of speed and capacity to handle extreme loads.

For all the range of software manufacturers, both of Internet-browsers and of web-servers, their software remains able to access the same documents. In order to qualify as a web-browser or server, the applications have to comply with the standard maintained by the World-Wide Web Consortium (W3C at `http://www.w3c.org`). This guarantees that any HTML hypertext[14] document will be readable in approximately the same form regardless of which software or OS one uses to access the document.

One of the results of this availability is that many applications make use of hypertext connections for access to online resources (many products are now released with service contracts permitting Internet downloads of updates for a limited period after purchase). Similarly, many use HTML hypertext for providing documentation and resources locally on the assumption that computers will have access to a hypertext reader - proof that if common resources can be provided, or if a common interface is readily available which makes the construction of common resources possible, these will be used.

1.5.3 Plug-in architecture

In this spirit, Koedinger and Ritter (1996, pp. 315-347) proposed a "plug-in" architecture for intelligent tutoring systems with the primary purpose of allowing incorporation of commercial, off-the-shelf software into a tutoring environment. In addition to allowing incorporation of existing software, this also has the benefit of enabling applications to use novel components for its own purposes regardless of the original intent of the software (Ritter et al., 1998, pp. 554-563).

The application developed here is developed specifically with this principle of building "plug-in" software in mind. As such the programme developed here does not have an

[13]Official in the sense that this branch of the code has a number of Netscape programmers working on it.

[14]"Hypertext" is a class of text allowing explicit linkages inside text. It does not refer to a particular variant *per se*. HTML hypertext (for Hypertext Markup Language) is a subset of SGML (for Standard Generalised Markup Language) - an international standard (ISO 8879) for the definition of device-independent, system-independent methods of representing texts in electronic form (Sperberg-McQueen and Burnard, 1994). Any reference to "hypertext" in this document refers to the HTML variant used to format Internet documents.

extensive interface. Output is sent to the standard output device (usually the screen). This has the benefit that the output can be redirected or read by other programmes. Similarly, output can be saved by redirecting the output to a file.[15]

Additionally, in keeping with the UNIX philosophy of programmes, namely "make each program do one thing well" (Gancarz, 1995, p. 19), the programme has been limited to the processes of analysis. Interpretation of the resulting networks and output is left for later programmes. Sample programmes illustrating some of the ways this output can be used are provided but these are brief. They are intended purely to illustrate the use of the application rather than as functional educational tools in their own right.

[15]In a UNIX-like environment this is done using the '>' character as in

```
perl analise.pl > [output_file]
```

or

```
perl analise.pl [input_file] > [output_file]
```

2 Existing research

As stated earlier, the primary aim of this research is to provide a tool that can be used to improve the automated analysis of student responses in CAI software for the ESL classroom. As such, it is intended to fill a gap in current computer applications and provide a number of unique benefits for both learners and teachers. This statement raises some questions that need to be answered before the programme itself can be discussed. The first of these is what the gaps in current computer programmes are. The second is what benefits can be achieved by filling these gaps - both for learners and for teachers.

2.1 Computer-assisted language learning

Pica (1997, p. 54) categorises approaches to SLA on the basis of their interface with teaching: Some SLA research *coexists* with L2 teaching while having little if any intellectual interface. Other SLA research *collaborates* with L2 teaching when teachers and researchers work together toward similar goals within the classroom and the educational environment. A third type of SLA research (that most significant for CALL design according to Chapelle (1998, p. 22)) *complements* L2 instruction.

The particular approach to teaching that initiated this research is the communicative approach to language teaching. One of the key requirements for this approach is realism. With this in mind language practitioners tend to use authentic materials in the classroom, or they simulate real-life problems in learning tasks so that authentic learning may occur (Darian, 2001, pp. 2-9; Dumitrescu, 2000, pp. 20-22). Language in the classroom should be as close as possible to language used by first language speakers outside the classroom as language learning is aimed at a goal that lies outside the classroom (Weideman, 1988, p. 105). As a result the goals of this dissertation are heavily influenced by interactionist research into the process of SLA. The process of SLA and the effectiveness of CAI in promoting SLA is discussed in more detail in this section.

2.1.1 Requirements for learning

In the communicative use of CALL, John Underwood proposes a series of "Premises for 'Communicative' CALL". According to Underwood, Communicative CALL:

- *focuses more on using forms rather than on the forms themselves;*

- *teaches grammar implicitly rather than explicitly;*
- *allows and encourages students to generate original utterances rather than just manipulate prefabricated language;*
- *does not judge and evaluate everything the students do nor reward them with congratulatory messages, lights, or bells;*
- *avoids telling students they are wrong and is flexible to a variety of student responses;*
- *uses the target language exclusively and creates an environment in which using the target language feels natural, both on and off the screen; and*
- *will never try to do anything that a book can do just as well* (Underwood, 1984, p. 52).

Stevens (1989, pp. 31-43) also contends that all CALL courseware and activities should build on intrinsic motivation and should foster interactivity–both learner-computer and learner-learner.

Given the shortage of data and the relative newness of theories of SLA, it is not easy to determine how a particular application will aid the learning process. We can, however, distinguish a few key components that influence SLA. The first of these has to be the input learners receive in the second language which, according to Krashen, promotes language acquisition if it is both comprehensible and contains linguistic material which is new to the learner (Krashen, 1982). This is one area where computers are particularly powerful as they have the capability to bring learners into contact with other humans in a more dynamic way than other media such as books or videos (Hubbard, 1996, p. 21). This means that learners are exposed to texts that have immediate personal relevance, whether this be in the form of email or chatroom/discussion-list interactions - a motivational factor lacking in books and videos.

A second key component may well be the process of interaction with second language input. The model illustrated in Figure 1 is a simplified version of the one outlined by Gass (1997). It summarises a consensus view among interactionist SLA researchers (Chapelle, 1998, p. 23). INPUT at the left of Figure 1 refers to the target language that the learner is exposed to. Much of the target language input is beyond learners' capabilities to understand. Only that which is APPERCEIVED has the potential to be acquired. An important consideration in designing instructional material, thus, may be to include features that prompt learners to notice important aspects of the language.

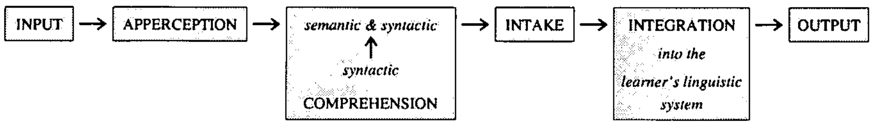


Figure 1: Basic components in the SLA process in interactionist research (Chapelle, 1998, p. 23).

COMPREHENSION represents the hypothesis that understanding of the semantic content of a message can be accomplished either with or without any comprehension of the syntax. When comprehension takes place through a combination of semantic and syntactic processing, the linguistic characteristics of the input can become INTAKE. In other words, comprehended language that holds the potential for developing the learners' linguistic system.

INTEGRATION consists of the processes using or holding the intake in short-term memory to influence the development of the linguistic system which, in turn, affects the L2 OUTPUT, which is the observable result of the process. This is considered an important contributor to linguistic development in two ways. Producing linguistic output forces learners to use the syntactic system and therefore develops this aspect of their ability. It also elicits subsequent input from interlocutors, some of which may contain indications of problems with the learners' output which will result in the learners noticing aspects of the linguistic form, making new hypotheses, and producing more output (Chapelle, 1998, p. 23). This process, referred to as *negotiation of meaning,* is believed to facilitate L2 development (Larsen-Freeman and Long (1991, p. 144), Long (1996, p. 413-468)).

This process is one particular aspect of text presentation in which computers can be particularly powerful, providing a range of methods of accessing a particular text and information about the text. These means include audio, graphic and textual media. In addition to this, related information can be linked in the appropriate context, enhancing the process of negotiating meaning (cf. Plass, 1998). This process includes the following in the case of reading skills: processing of prerequisite knowledge; paying attention to and selecting relevant information; building internal connections (i.e. reorganising the new information in short term memory into a coherent form); and building external connections (i.e. integrating new information with the existing prerequisite knowledge into the learners mental model) (Plass et al., 1998). Each of these steps on the road to understanding a text can benefit from multimedia enhancement.

Understanding and interpreting do not, however, make for competence in communicating. Therefore we have a third key component, comprehensible output. This is also the one element that is most often missing in typical classroom settings, "language classrooms and immersion classrooms being no exceptions" (Swain, 1985, p. 252). Comprehensible output has two implications. The first of these is that there will be the presence of someone capable of comprehending the output. (Chapelle notes that it may well be important for learners to have an audience for their linguistic output so that they attempt to use the language to construct meanings for communication rather than solely for practice (Chapelle, 1998, p. 23).) Secondly, that the other party will produce an appropriate response demonstrating that the output has been correctly understood. Failure at this level should lead to remedial discourse acts - the student would be made aware of the fact that the output had not been understood and would have to rephrase the output until the student was confident that the output had been understood. This coincidentally brings the process of learning close to the ideal learning process proposed by Larsen-Freeman and Long in which they state:

> *Modification of the interactional structure of conversation or of written discourse during reading ... is a [good] candidate for a necessary (not sufficient) condition for acquisition. The role it plays in negotiation for meaning helps to make input comprehensible while still containing unknown elements, and, hence, potential intake for acquisition* (Larsen-Freeman and Long, 1991, p. 144).

This poses problems for course design in that a learner's competence changes dramatically during the process of SLA. In order for texts to form comprehensible input students have to have a significant vocabulary covering most of the words used in the text. Similarly, they cannot be expected to produce much by way of comprehensible output unless they have a minimal vocabulary sufficient to cover most of what they want to communicate. Courses need to cater to the changing characteristics of learners as they progress in acquiring a second language. For example, initially a course may need to focus more on acquisition of necessary vocabulary than on acquisition of grammatical forms. Vocabulary can allow the students access to comprehensible input which, in turn, will allow them to become aware of grammatical forms. Later in the acquisition process, once students have a reasonable vocabulary, the process may shift to better understanding and use of the grammatical forms of the target language. Later still it may shift to understanding and using the second language in different contexts of talk.

As learners have different capabilities and aptitudes, they will inevitably progress through these phases at different rates. This implies that an ideal course should be learner driven. The learner's level of competence should decide the direction of the course and the level of the texts and communication to which he or she is exposed (cf. Chapelle, 1998, p. 29). There are two ways of achieving this. One method is by constant evaluation (overtly by testing or covertly by, for example, a facilitator's assessment of the learner's capabilities). The second is to allow the learner to decide the direction of study. In a CALL context, this would be achieved, for example, by allowing the learner to decide who he or she communicated with over the Internet, or by allowing access to texts covering a range of topics and requiring a range of levels of proficiency.

As long as the learners are motivated to acquire new information, they will inevitably attempt texts requiring a greater proficiency than they possess. This progression can be guided and encouraged, ensuring a method of self-evaluation for the students as most would not continue with texts hopelessly beyond their competence. They would, however, be prepared to work through texts which interest them and are largely within their competence. It should be noted that learner motivation is a key factor here. Self-paced study has been shown to be effective, though with the caveat that many students have difficulty managing their own instruction as was seen in the unexpected drop in completion rates among students using the TICCIT[16] system (Hofmeister and Maggs, 1984, pp. 3-10).

2.1.2 The benefits of computerised learning aids

As N. Garrett, p. 75 points out "the use of the computer does not constitute a method". Rather, it is a "medium in which a variety of methods, approaches, and pedagogical philosophies may be implemented" (1991, p. 75). Software, in other words, is there to assist in instruction, not to replace instruction. Because of the range of uses of computers they can easily provide a range of materials - replacing in many cases previously specialised materials. Candlin, et al. (quoted in Bell, 1981), broadly divide materials into two classes - content materials, which provide data and information, and process materials, which provide the learner with 'frameworks' for activities in which the data and information provided by content materials may be practised (Bell, 1981, p. 44). In his overview of CALL materials Warschauer (1996, Appendix A), however, divides CAI software into three categories; computer as tutor (this includes software that implements exercises in grammar, listening,

[16]TICCIT (Time-Shared Interactive Computer Controlled Information Television) was a major CAI system developed at the University of Texas and Brigham Young University (McNeil, 2004a).

pronunciation, reading, text reconstruction, vocabulary and writing), computer as stimulus (he mentions simulations as particularly useful here) and computer as tool (this includes word processors, concordancers, collaborative writing, reference works, Internet and authoring tools).

As with Bell, Warschauer notes that these classes do not need to exist independently. Increasingly these overlap in CALL programmes as the computer makes it possible to provide a learning framework and in this framework have access to a host of resources providing data and information that can be used in that framework. Hypertext, the 'universal' document format that makes the world-wide web possible, is a good example of this combination as the hypertext medium provides a framework for activities and interaction in a real-world setting as well as providing access to the single largest information resource in existence today.

A major advantage of computers is that the applications can be adaptable. In other words, applications can be designed (and have been designed) that provide for different levels of proficiency. Also one should guard against the perception that one single application should provide for all levels of proficiency. While this may be ideal, there is considerable place for programmes that focus on one particular aspect of language. Partly because of the limited scope of authoring packages which do not include tools for complex textual analysis, applications with a restricted linguistic scope have become the norm for CALL.

Two developments reflect this tendency. The first attempts to link a number of smaller applications as part of a larger package. These larger packages often allow sophisticated evaluations of learner performance to enable learners to progress from one applet[17] to another. These applets function, though, as stand-alone programmes each with its own goal and, at least conceptually, capable of existing independently of the governing package. Examples of this are the PLATO[18] system (Dyer, 1983, pp. 65-85), now in its fourth incarnation, and the TICCIT system, but these are far from the only ones (Levy, 1997, pp. 15-21).

The second method of circumventing the limiting scope of individual applications is to use these as a resource for encouraging communication among learners and between learn-

[17]An applet is a small programme or sub-programme. Applets often refer to small programmes run as part of a larger application or in a virtual operating environment. For example, Java programmes, which run on a virtual Java machine, are often called applets.

[18]PLATO (Programmed Logic for Automatic Teaching Operations) was originally designed to use a mainframe-based system rather than a smaller minicomputer because of greater program and storage capability (McNeil, 2004b).

ers and teachers or native speakers. Indeed Woolley notes that the addition of a conferencing facility very rapidly became an essential part of the package, helping create a remarkable sense of community among PLATO users (Woolley, 1994, pp. 5-7). As Schwienhorst says

> *Interaction is also of central concern in the concept of learner autonomy* (Schwienhorst, 1998, p. 122).

Consequently, much of recent CALL development focuses on this aspect of CAI. Further evidence of this can be seen from the number of aspects in recently proposed models of ideal CALL programmes that promote this kind of interaction. Consider, for example, the model described by Martha Pennington

> *[An ideal teacher or application would be one which]:*
>
> - *Helps learners develop and elaborate their increasingly specified cognitive representation for* (sic) *the second language;*
> - *Allows learners to experiment and take risks in a psychologically favourable and motivating environment;*
> - *Offers input to both conscious and unconscious learning processes;*
> - *Offers learners opportunities to practise and to receive feedback on performance;*
> - *Allows learners to learn according to their own purposes and goals;*
> - *Puts learners in touch with other learners;*
> - *Promotes cultural and social learning;*
> - *Promotes interactivity in learning and communication;*
> - *Exposes the learner to appropriate contexts for learning;*
> - *Expands the learner's "zone of proximal development";* [and]
> - *Builds to learner independence* (Pennington, 1990, p. 7).

As she notes: in all these ways the computer stands, along with the teacher, as a uniquely effective medium. A large part of that effectiveness arises from the aspects which promote interaction - offering opportunities to receive feedback on performance, putting learners in touch with other learners, promoting social learning and promoting interactivity in learning and communication are all factors that are difficult to encourage in a classroom environment but a necessary by-product of any tools that encourage learners to participate in discussions with people not connected to their learning environment.

2.1.3 The need for comprehensible output

Without involving human agents, the requirements for comprehensible output are harder to provide. Improvements in NLP, AI and discourse analysis increasingly make it possible to construct small artificial intelligences (called chatterbots, chatbots or just bots) which can, for a while, fool people into believing they are talking to a real person. These bots have enormous potential to enable learners to engage in communicative activity, including comprehensible output, in a completely risk-free environment.

Construction of these bots is, however, a time-consuming activity. The conversational range of chatbots is also limited to the context of their intended function. This is invariably a scenario that can be reasonably completely defined. As these chatbots and NLP systems operate essentially on the principle of responding to keywords using a set of pre-programmed responses, it should be fairly obvious that complex scenarios are, at present, beyond them. This is especially true in a language learning scenario as chatbots and NLP systems are essentially constructed to enhance game-play (in games) or provide information (in intelligent systems). A good example of this, developed specifically for language learning, is the speech interactive microworld developed by the Army Research Institute's Military Language Tutor (MiLT). This is effective in its role, but had to be restricted to fewer than 100 utterances, and use keyword recognition to achieve any robustness (Holland et al., 1999, pp. 339-359).

Using keywords as the basis of NLP means that applications often do not require grammatical or even comprehensible input. Input that matches keywords will trigger usable responses regardless of presentation of the input. Outside their intended fixed scenario these systems are unlikely to provide the same illusion of meaningful interaction and would thus lose much of their value as exercises furthering communicative competence.

A slightly more complex scenario is that of systems that use learned responses. This includes systems that generate neural networks as in the *meme* application (discussed in more detail in section 2.3.2) and programmes that construct relationship trees, such as *NICHOLE*. This is an open-source attempt at simulating a conversation by learning how words are related to other words (Howlett, 2000). Unlike other chatbot-like implementations, *NICHOLE's* design enables it to extend not only its vocabulary but also its knowledge base.

However, communicative competence is a highly complex ability. It includes grammatical accuracy, intelligibility and acceptability, contextual appropriateness and fluency (Nyyssöen, 1995, p. 160). This in itself makes chatbot design a difficult and complex process. A large part of the complexity arises because considerable care must be taken in constructing their responses to prevent initiating lines of conversation which go beyond the scope of the preprogrammed response set. This is especially difficult if chatbot answers are 'reasoned' responses derived from a form of knowledge database. Inevitably care must also be taken in defining parsing parameters that are flexible enough to cover the range of possible responses and initiations contained in input text. As a tool for language teaching, this is also a major limitation in that it means responses are limited with the result that chatbots cannot truly replicate conversation. In particular, they cannot replicate the creativity of conversation.

In addition to being a limiting factor in chatbot design, the nature of conversational input is also a key aspect that makes the existence of chatbots possible. Conversational text, and the interface to chatbots closely resembles this, is constructed of short responses seldom longer than a sentence. Because the scenarios in which chatbots 'converse' are known, most of the directions that conversation can take can be predicted in advance. This includes an outline of most of the meaningful user responses. The potential for novel responses, which would not be within the chatbot's range of conversation, increases as the length of input text increases. Consequently, chatbots are less able to respond to longer texts appropriately.

Currently, the only 'programmes' that do fulfil the requirements for comprehensible output and permit more substantial texts as output are applications which facilitate communication between people. These include chat lines (such as Internet relay chat or IRC), some games (such as role-playing games or RPGs) and email. At a larger textual level, the availability of the worldwide web allows students to publish web-pages which they can anticipate will be seen by a real-world audience. This has led to a number of student webrings being constructed that link sites designed by students - making it easy for students to read and comment on the work by their peers in other facilities, towns and countries. The opportunities these media provide for producing comprehensible output in a non-classroom environment means that they make effective communication tools and have been used as effective learning tools. A common feature of these communicative applications is that they are a beneficial by-product of an Internet connection. Communication over Internet puts language learners in contact with learning resources, increases opportunities for intercultural collaboration and dramatically improves global communication (Wells, 1993, pp. 79-88). It also makes them very close to the ideal expressed by David Little, in his concept of learner autonomy. He has repeatedly emphasised the importance of learners devising their own learning materials as the learners

> *experience the learning they are engaged on as their own, and this enables them to achieve to a remarkable degree the autonomy that characterizes the fluent language user* (Little, 1991, p. 31).

External factors can also affect the availability of network connectivity. In a South African context, for example, many institutions which could materially benefit from this sort of connectivity find it to be prohibitively expensive to provide for their students. More institutions are unable to provide even basic computer facilities. For those institutions that do have computing facilities, though, there should be means of encouraging their effective use even when network connectivity is not available. Part of this is finding a method of ensuring that comprehensible output will be generated without the necessity of having other people involved in the process - this being the key factor that makes Internet applications like IRC, RPGs and e-mail so effective.

This brings us back to chatbots as a possible AI solution for small texts in which keyword recognition can be the governing device for analysing communication. These are increasingly becoming more effective emulators of human conversation as can be seen in the developments of the entries of the Loebner prize for the application that best passes the Turing Test for computer intelligence[19] which have achieved a remarkable degree of sophistication. A fine example of this is the 2000 and 2001 Loebner prize winner *ALICE*. *ALICE* is a robot based on Artificial Intelligence Markup Language (AIML) and springs entirely from the work of Dr. Richard Wallace and the A.L.I.C.E. and AIML free software community based at `http://www.alicebot.org`. Robots such as these still cannot replicate the creativity of language but by careful design of responses can guide conversation in a direction that can appear to replicate this creativity. ALICE boasts a 30% success rate - defined as being able to fool people 30% of the time, over a five minute period, into thinking they are communicating with a person.

[19]In brief this can be summarised as: given an environment which restricts judgement to typed conversation (removing all externals that might prejudice judgement) then if a judge cannot tell the difference between human and computer responses in a conversation the computer can be said to be able to think (Turing, 1950)

In principle it should be possible to construct AIs capable of responding to longer texts provided these texts can be assessed in terms of the content of the text and that this assessment can be compared with the preprogrammed knowledge base of the AI to determine differences and similarities in content. Responses can then be constructed on the basis of this assessment, confirming or contradicting statements and providing information, if appropriate.

As the application described in this dissertation provides a method of contextual assessment in the form of comparing relationship networks (discussed in section 5.4), combining the application with a chatbot interface would be one potentially beneficial use of the system in increasing learner opportunities for generating comprehensible output.

2.1.4 Computer mediated instruction

Increasingly, interest in Computer Assisted Instruction (CAI) has moved toward using the computer as a resource or as a stimulus for group activities (Kenning, 1990, p. 73). Possibly the best application for use in a communicative approach to language teaching is a form of multi-user environment - a group activity on a global scale. Games like MUDs (Multi-user dungeons), MOOs (MUD Object Oriented) and MUSHes (Multi-User Shared Hallucination) provide a high level of real-time interaction in a shared environment. The ones mentioned here operate as worlds in which the various users perceive a shared environment through the medium of a text-based display. This shared environment can be that of a goal-orientated game, as in most of the MUD varieties, or a role-playing game in which the players co-operate in creating a world. Role-playing variants work particularly well as environments that encourage communication. This is taken to a logical extreme in many MUSH and MOO implementations. These become, in effect, an online co-operatively written book with no plot, hundreds of unscripted or partially scripted stories and the only requirement that players interact in character and cooperate to better build the shared experience of the particular world the game is designed to portray. In the words of Hamit

> ... *a cyberspace is defined more by the interactions among the [users] within it than by the technology with which it is implemented* (Hamit, 1993, p. 74).

These worlds can and do range from Medieval and Fantasy worlds (cf. the *Nightmare MUD*), often using ideas drawn from favourite books and TV shows (cf. *Diskworld MUD*), through period pieces (such as the *London by Gaslight MUD* which tries to recapture, as far as possible, the period and culture of Victorian London), to modern and futuristic worlds, such as virtual universities (cf. schMOOze University, accessible at `telnet://arthur.rutgers.edu8888` (Levy, 1997, p. 100 and Peterson, 2000, p. 1) and Diversity University, accessible at `http://www.du.org` (Schwienhorst, 1998, p. 5)) and Star Trek style space wars.

Increasingly these virtual worlds are being recognised as valuable resources for cultural and linguistic learning. As Elizabeth Reid puts it:

> *[MUDs are] a set of tools that can be used to create a sociocultural environment. Muds allow the depiction of a physical environment that can be laden with cultural and communicative meaning* (Reid, 1995).

Many of these environments are completely accessible via Internet, though some are limited to local networks. As a result, the participants very often represent a fair cross-section of the global Internet community. Participants include players from many countries covering both EFL and ESL speakers. In learning environments these can be used to provide a target language environment for real-time discussion between learners of that language (Warschauer, 1995; Pohio and Turner, 1995).

Given the range of English competences, the medium can be both forgiving and facilitatory as it brings ESL users into contact with EFL users in a non-threatening environment. Participants have no need to worry about taking the time needed to frame a response accurately. There is an inevitable lag in a communication medium where participants-at-talk may be sitting on different continents. Furthermore, load on the server and having to read responses before typing your own add to the delay in response times. This, combined with the fact that all perception of the virtual world takes place in the form of text, means that accuracy is preferred over speed. In the environments where role-playing is the defining characteristic, these delays are even more acceptable as here the creativity of the response is also cause of delay. A detailed creative response that enhances the environment perceived by the other participants is preferred over a bland response, even if this is accurate and quick. As a result these environments become valuable learning environments. Participants have more planning time for producing language while still being engaged in an activity where the pragmatics of communication are closer to spoken than to written communication (Ashwood, 1996, p. 93; Hoffman, 1996, p. 64).

Short of providing a complete natural language interface (allowing spoken conversation), a MUD comes close to being the ideal environment for effective communicative practise. As much of the benefit of these environments arises from the lack of pressure that results from having time to formulate and edit responses before they become visible to the rest of the participants (Phinney, 1996, pp. 137-152), allowing a completely natural language interface may not be an improvement either.

For all this, they do have some apparent drawbacks for use in an ESL classroom. For example, the interactions are completely unstructured. There is no way that the environment can be structured to fit a learning programme without some form of control over the environment. While it is possible to add these controls to the environment and have them enforced by the environment, these controls dilute the beneficial aspect (the spontaneity and creativity) of the environment to the point where it is doubtful as to its value.

It has been found that the students, especially the quieter students who ordinarily would not speak in class environments, are more inclined to contribute in the "safer" environment of online talk (Phinney, 1992). This minimises the effects of dominant personalities in face-to-face conversation (Hoffman, 1996, p. 64). A second apparent drawback, given the unpredictable origins of the participants, is the unpredictability of the quality of the responses (in terms of the fluency in English of the respondents) that participants will be exposed to. Whether this is really a liability though will depend on the results that are desired from the medium. Participants will not necessarily be exposed to a strictly grammatical use of English, a factor that has as corollary the probability that the English they will be exposed to is a better match for the sort of interaction they will encounter in real-life[20] than that contained in any prepared material. This includes exposure to the regional usages associated with different English-speaking countries, as well as cultural differences associated with the different backgrounds of the participants (But, 1999, http://imaginaryrealities.imaginary.com/volume2/issue8/learning.html). It should be noted that these online games become very real communities with all the trappings of real-life communities.[21] Experiences in these communities can be just as meaningful to the participants as what they experience outside the communities and often cover the same range of experience as real-life interaction (Pennington and Esling, 1996, p. 183). Similarly, the interaction that takes place in these virtual environments is just as real, creative and unpredictable as spoken conversation and requires a similar range of communicative skills as more common interactional environments (Pennington and Esling, 1996, p. 183) - with the added benefit that the environment encourages cooperation and collaboration. In the words of Amy Bruckman:

[20]"Real-life" is used in the Internet community to refer to any activity outside of the virtual world of a game or online community. In this context it refers to any non-computer mediated communication.

[21]Hence the term Virtual Communities to describe communities in online games and chat-rooms.

MUDs are Constructionist environments in which people build personally meaningful artifacts. But unlike many Constructionist environments, MUDs place special emphasis on collaboration, encouraging construction within a social setting (Bruckman, 1994, p. 1).

Essentially, apart from the shared virtual world and the text-based interface, what takes place is real-life communication. Communication with the same complexity, diversity and creativity as would be found in any non-computer mediated communication.

2.2 Grammatical models

Existing grammatical models cover a range of current research into systems of representing text and the process of generating and interpreting text. There are far too many theories to be able to examine them all in any detail; so, a few representative theories are discussed in this section. Inevitably there is a considerable overlap between the various theories, though the premises on which the theories are based may differ considerably. For example, as Kahane argues, Head-driven Phrase Structure Grammar (HPSG) can conceivably be viewed as a dependency grammar (Kahane, 2002, www.linguist.jussieu.fr/ skahane/HPSG.extraction.pdf).

The different approaches have different strengths and liabilities as models for use in developing CAI programmes. These strengths and limitations are examined in the light of the separate roles the grammatical systems would need to play in a textual analysis application: decoding texts, determining contextual linkages in the text, representing meaning in the knowledge base and, ultimately, generating output.

A general criticism of grammatical theories, as they can be applied in this application, is that they generally provide descriptions of language production. In general, grammars with greater lexicalisation perform better at language interpretation.

2.2.1 Generative grammars

[A] ***generative grammar*** *is a set of* formal rules *which projects a finite set of sentences upon the potentially infinite set of sentences that constitute the language as a whole, and it does this in an* explicit manner, assigning to each a set of structural descriptions ... *In recent years, the term has come to be applied to theories of several different kinds, apart from those developed by Chom-*

sky, such as Arc-Pair Grammar, Lexical Functional Grammar and Generalized Phrase -Structure Grammar... (Crystal, 1997, pp. 166)

The depth and range of research into generative grammars and the detail in their formulation make them an attractive starting point for any NLP application. As such, Transformational Grammar formed the basis for initial exploration into the possibility of constructing an analysis engine.[22]

This is a potentially very flexible and extensively documented grammar well suited to generating sentences given a basic starting point. It does, however, have the problem that its transformational nature makes it difficult to analyse sentences automatically.

Transformational grammars have as a basic assumption that "the kernel of the language consists of simple, declarative, active sentences, and that all others can be described more simply as transformations" (Bornstein, 1977, p. 39). The rules for these grammars "begin with directions for generating or producing structural descriptions of sentences, which are set forth in phrase structure rules" (Bornstein, 1977, p. 39).[23] Unfortunately this also means that the domain of these grammars is highly restricted (Ellis, 1995, p. 85). This reflects a tendency to model grammars with the aim of being able to make well-formedness judgements where each sentence has one and only one successful structural analysis (cf. Keller and Asudeh, 2002, p. 4). This provides an accurate analysis of a way in which phrases can be linked syntactically but does not provide a good way of determining this linkage without prior knowledge of the meaning of the sentence. Knowledge of the meaning of the sentence is rather a prerequisite for analysis.

Consider the surface similarity of the two sentences:

John is easy to please.

John is eager to please.

In the first sentence, "John" is the object of "please," while in the second sentence, "John" is the subject. They have the same surface structure but their deep structures reveal the different grammatical relations that account for their different meanings (Bornstein, 1977, p. 37).[24]

[22]While this section focuses on Transformational Grammar, the comments and criticism given in this section, however, apply to the other theories categorised as generative grammars.

[23]A rule is defined as a statement about the nature of the relationships that exist among various English sentences and the components of those sentences (Bornstein, 1977, p. 37).

[24]The distinction between deep structure and surface structure may have been more pronounced in earlier

Deep structures and surface structures are produced by two types of rules. Phrase structure rules generate the sentences that are formed by the deep structure. Transformational rules change the order of phrases in these sentences while transforming them into surface structures. Both types of rules are assumed to be part of the individual's linguistic competence so that the ability to perceive the deep structure of sentences includes the unconscious perception of grammatical relations which enables a native speaker of English to recognise the different grammatical relations illustrated in these sentences (Bornstein, 1977, p. 37).

This implies that decoding input involves a number of stages. First, the input is processed to determine which set of grammatical relations is in use. Then, the transformational rules that created the surface structure are removed, which reveals the underlying deep structure (the meaning) of the sentence. With the deep structure revealed, the meaning of the sentence is presented in its simplest form and can be understood.

This involves an inherent redundancy in that it becomes necessary to know the meaning of a sentence to determine which grammatical relations have been used. For example, unless one knows the meanings of the sentences in the two examples above, there is no way of distinguishing between their structures. The redundancy exists because, after determining the meaning of the message, there is no need to perform any transformations to determine the deep structure (deep structure is a representation of the meaning (Bornstein, 1977, p. 37)). In the first of the examples one has to first know that "John" is the object of "please" before one can realise that a transformational rule has been applied. While the deep structure may reveal different grammatical relations between the words it can only do this once those different relations have already been determined.

The concept of deep structure as a representation of meaning is, however, very attractive. The fundamental problem of programming inherent in the application developed here is one of representing meaning and doing so in a manner that allows comparison. For example, the sentence

Mary fixed the car.

is a simple declarative sentence of the structure "$S \rightarrow NP + VP$" which, when analysed, reveals the deep structure illustrated in Figure 2. The attractiveness of this representation

versions of Chomskyan approaches. The distinction still applies to recent Minimalist developments: even though there is no distinction between Surface-Structure and Deep- Structure, there is still a point (Spell-out) in which the derivation splits into two levels, phonetic form (PF) and logical form (LF). Furthermore, structures are still derived from other structures by operations such as move-a, even before Spell-out (Marantz, 1995, pp. 349-382).

Mary fixed the car

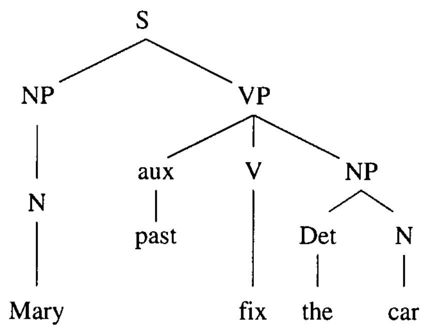


Figure 2: (taken from Bornstein, 1977, p. 50)

lies in the fact that differing surface structures with the same underlying deep structure are represented in the same way with only minor abstract markers to indicate the difference, making comparison of deep structures very easy, as can be illustrated with the following three sentences showing two transformations:

Kernel sentence: *Jim will pass the test.*

Question transformation: *Will Jim pass the test?*

Negative transformation: *Jim will not pass the test.*

The kernel sentence has a structure illustrated in Figure 3. This same structure is repeated in the deep-structures of the question and negative transformation forms of the sentence as illustrated in figures 4 and 5.

In addition, complex sentences are also reduced to simple declarative sentences with the same structure as used in the previous examples. A sentence such as

What she said was true?

(analysed in Figure 6) provides a simple representation reduced to two declarative statements. Summarising the relationships we have

"*she*" $-say_{past} \dashrightarrow$ "*what*" and

"*she said what*" $-is_{past} \dashrightarrow$ "*true*".

Jim will pass the test

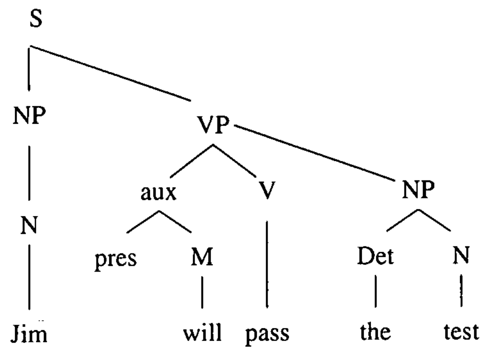


Figure 3: taken from Bornstein (1977, p. 51)

Will Jim pass the test

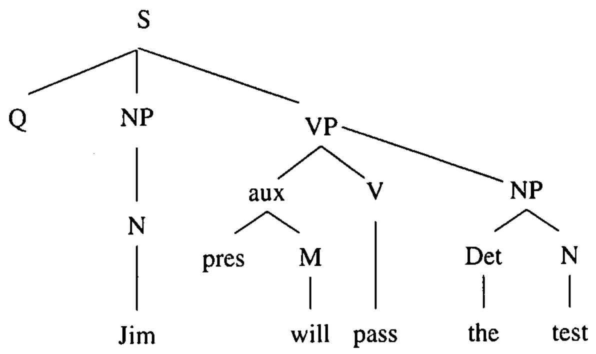


Figure 4: (taken from Bornstein, 1977, p. 51)

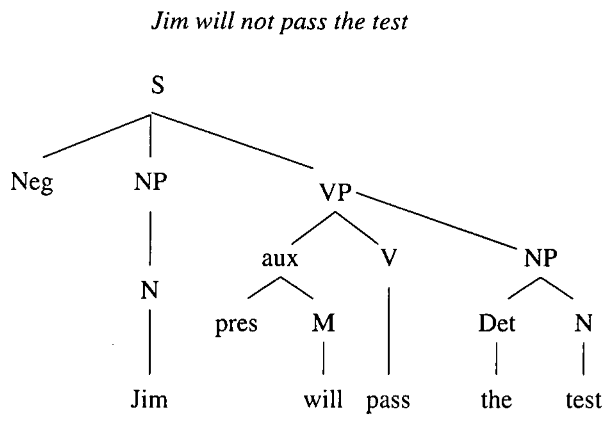


Figure 5: (taken from Bornstein, 1977, p. 51)

Should the knowledge base include another structure in the form

"she" −said → x

then one can evaluate the statement for validity, or supply an answer to a question like

"what did she say?"

This is precisely the type of comparison that the application is required to make. Every effort was thus made to retain this economy of representation in the final structure as it does make comparison operations between texts quite simple.

However, this is not a viable method for performing a computerised analysis. To determine the deep structure one must examine the sentence to determine what transformations have been applied. As was illustrated in the comparison of the two sentences "John is easy to please" and "John is eager to please", determining the transformations involved in generating a surface structure implies prior understanding of the meaning of the sentence. To complicate matters further, surface-structure transformations are frequently impossible to apply adequately in reverse as the transformations often imply missing data. As Covington, Nute, and Vellino put it "In order to undo the transformation, we must know the structure that it produced - but we must do the parsing in order to discover the structure" (Covington et al., 1988, p. 418). This is true, for example, of active to passive transformations in which the original subject becomes optional after the transformation.

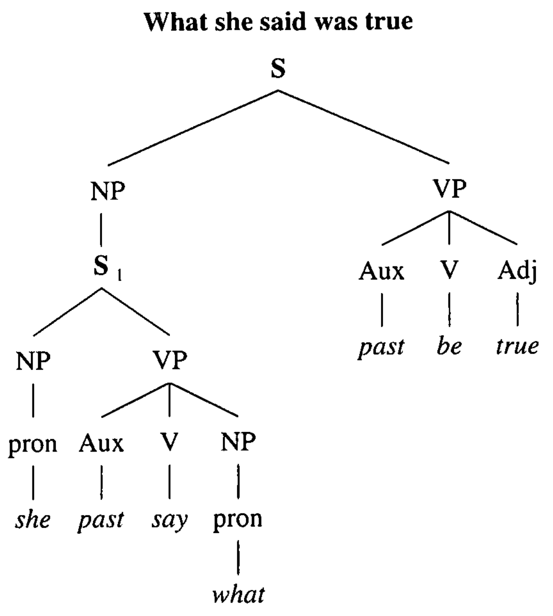


Figure 6: (taken from Bornstein, 1977, p. 185)

The only realistic way around this problem is to use some other grammatical formulation as the basis for decoding the text. From this, any transformations can be identified and possibly used as an indicator of the meaning of the text - or at least as an indicator of the network of relationships contained inside the text. Identifying these transformations, however, has no further benefit in determining the structure of the text.

In addition, most generative theories (such as those of Bresnan, 2001; Chomsky, 1981, 1995; Pollard and Sag, 1994) are not designed to handle gradient well-formedness (Keller and Asudeh, 2002, p. 4). This is a critical issue as the student texts used for input in this application are guaranteed to contain errors. This highlights the essential problem with generative grammar models. They have a very descriptive set of rules that, used correctly, can go a long way to explaining how responses are generated and provide an elegant analysis of relationships between phrases in a text. These may provide an outline for generating responses in an application but the set of rules is prescriptively large for use in this application.

Furthermore, the focus of most generative grammars, and indeed most grammars, is in the realm of grammatical competence (providing well-formedness judgements), and, while this explains the myriad of connections that can be made in language use, it does not provide a model for language performance (Winston, 1993, pp. 594-597). They are not able to determine sentence structures automatically when faced with the common problems of free variation, ambiguity and robustness (Keller and Asudeh, 2002, p. 4). This set of rules also requires extensive programming to cater for a reasonable range of sentence surface-structures.

In short, analysing text using a transformational model would involve an inefficient parsing engine. This is largely because of the number of times that the same sentence would have to be processed. It was also quite clear that this would need a sophisticated pre-parse analysis of input text. An analysis can only be made accurately given knowledge of the sentence content. As the purpose of grammar in this application is to provide a means of determining the sentence content, the transformational analysis of the input text largely became redundant.

2.2.2 Phrase structure grammars

Phrase structure grammars, as the name suggests, are defined in terms of the relationships between a phrase and its parts. Sentences are viewed as being composed of phrases, even if the phrase contains only one word. A number of theories have been developed from this core view of language. The main variants are Transformational Grammar (discussed in section 2.2.1) and General Phrase Structure Grammar (GPSG), though each of these has a number of offshoots which emphasise or de-emphasise aspects of the grammatical theory they developed from.

Of these, GPSGs have found more appeal among people committed to studying the implementation of grammar in a model of language processing (be that by humans or by computer) (Newmeyer, 1986, p. 210). GPSG and Context Free Phrase Structure Grammar (CF-PSG) have benefits computationally in a simple representation of relationships between phrases. This allows a convenient method of representing contextual linkages but it suffers from some of the same problems as a model for decoding sentences as does Transformational Grammar.

The main liability of any Phrase Structure Grammar is its dependence on the Phrase as the key grammatical unit. Computationally, this causes problems in that the types of the words in a sentence have to be known before phrase boundaries can accurately be determined. This is even more evident in the Head-driven varieties of Phrase structure grammar. These variously define a verb or object as the head of the sentence and construct a structural tree representing the relationships of the phrases in the sentence to this head. In so doing they define a representation of semantic content in terms of the structural tree. Unfortunately this presupposes that one can identify a head before beginning the analysis - an inherent contradiction in a computer-based analysis.

As with Transformational Grammar, this difficulty can in part be de-emphasised by using a large lexicon which includes type information for each word. This would mean that parsing becomes a matter of looking up the appropriate word and trying to determine phrase boundaries based on this information. In principle this works - until the programme encounters words which have multiple meanings (Miller, 2001, `http://www.kurzweilai.net/meme/`). This is particularly apparent when those meanings can result in the same spelling applying to words of different types. This brings us back to the problem of ambiguity. If we look at the line

"But I have promises to keep, and miles to go before I sleep."

from Robert Frost's poem, *Stopping by Woods on a Snowy Evening*, most of the words have a number of possible meanings. "But" for example has 11 different meanings, and "I" has a further three. This means that the two initial words, "But I" would have $3 \times 11 = 33$ possible compound meanings. Following this pattern we can build the following table of possible interpretations.

Word	Meanings	Interpretations	Types	Phrase structures
But	11	11	6	6
I	3	33	2	12
have	16	528	7	84
promises	7	3 696	3	252
to	21	77 616	2	504
keep	17	1 319 472	5	2 520
And	5	6 597 360	1	2 520
miles	5	32 986 800	1	2 520
to	21	692 722 800	2	5 040
go	29	20 088 961 200	6	30 240
before	10	200 889 612 000	3	90 720
I	3	602 668 836 000	2	181 440
sleep	6	3 616 013 016 000	4	725 760

(based on Miller, 2001, `http://www.kurzweilai.net/meme/`). The figures in the "meanings" and "interpretations" columns are drawn from Miller's article, while the figures in the last two columns are from definitions in the seventh edition of the Concise Oxford Dictionary (Sykes, 1983). These figures will vary in a functional application, partly because they are dependent on the dictionary used as a source, and partly because choices will have been made as to the degree to which seldom used or specialised variants of words are included (with the corresponding risk of incorrectly analysing a sentence if one of these missing variations is used). For example, the Concise Oxford lists five possibilities for "I":

1. as a letter of the alphabet;
2. as a pronoun;
3. as a noun representing the ego;
4. as an abbreviation for "Island" or "Isle"; and
5. as the symbol for Iodine.

Assuming this is the only difference between Miller's figures and the Concise Oxford, then there are approximately 2.4 trillion extra possible interpretations. The last two columns are particularly significant in that this demonstrates that it is impractical to try and determine the types of words in a sentence - and thus the phrase structure of the sentence - solely by means of looking up lexical information in a database. At best this can only hint at the appropriate type. There are so many possible interpretations available that another method of determining phrase structure would be needed before any phrasal analysis of the sentences is possible.

The strength of GPSG and its derivatives lies in representing relationships between phrases. Once the types of the words in a sentence have been determined, phrase structure rules can be used to partition phrases and subphrases into segments of meaning. At this level there are also pitfalls for the unwary.

Consider the two sentences:

The man in the coat and the hat went to town.

The man in the coat and the boy went to town.

The parts-of-speech used in these two sentences are identical, but they have very different interpretation structures based on the scope of the conjunction "*and*". It is easy to draw a phrase-structure tree describing the difference in scope. It is quite another to process these sentences on computer.[25] Without extensive lexical information to differentiate between the different characteristics of a "*boy*" and a "*hat*"; there is no way of determining which of the potential structure trees is the correct one.

2.2.3 Unification-based grammars

Another promising direction in the world of grammatical theory is that of unification-based grammars. These typically are easy to represent and understand as the grammars state relationships rather than procedures of computation. This is achieved by using "feature structures" to represent linguistic objects, which are essentially sets of constraints that characterise (partial) phonological, syntactic, semantic and contextual information concerning a given linguistic object. The fundamental operation on feature structures (and reason for the name "unification based grammar") is the "unification" operation, which combines the information from a set of compatible feature structures in a feature structure that contains all the information and only the information present in that set of feature structures (Giüngördü, 1997, p. 11). These theories are strictly declarative in that the grammars only specify what constraints are brought to bear during language processing but do not declare an order for those constraints to be satisfied (unlike more declarative grammars such as Government and Binding Theory (Chomsky, 1981) in which specific linguistic structures are successively transformed into different but nonetheless completely specified linguistic structures (Pollard and Sag, 1987, chapter 1)).

Grammars such as HPSG, have a few very attractive traits in constructing applications such as this one. Firstly, it is non-derivational and has a surface-oriented grammatical architecture (Kim, 2000, p. 7). This avoids many of the problems associated with generative grammars (described in section 2.2.1 on page 28). It is said to be surface oriented because it provides a direct characterisation of the surface order of elements in a sentence (Shieber, 1986, pp. 6-7). According to Shieber, this is assumed to be a characteristic of unification-based grammars as it uses the "unification" operation mentioned earlier.

From this it follows that these grammars have relatively few rules, and these are schematic and fixed (NLL, http://www.cs.sfu.ca/research/groups/NLL/2.html). On the other hand,

[25]This problem is considered in section 4.2.4 on page 124.

they will have rich sets of feature structures, often represented as complex lexical structures (Kim, 2000, p. 8).

The precision of these grammars (and HPSG in particular) has meant that they are used as a basis for a number of linguistic computer implementations (Borsley, 1991, p. 210). Computer science (along with a number of other fields) has also influenced HPSG in making this more precise (Pollard, 1997, p. 2). As Borsley (1996, p. 8) points out, however, HPSG has as its main goal to provide illuminating syntactic analysis, rather than providing computational implementations. He also notes that HPSG has to be adapted before it can be used in computer modelling, as is also the case with other frameworks.

The complexity of lexical information required, for example, for an HPSG based parser is an argument against basing a system on a unification-based grammar. Inevitably the grammar depends on a detailed and specifically tailored lexicon containing a large set of data for each word. At the moment this set has to be entered manually.[26] This consideration is true to a varying degree for all modern grammar theories in computational linguistics as these are all lexicalised to a varying degree (König, 1994, p. 5).

A second argument against this approach is the problem of determining phrase-heads. Determining these implies an analysis prior to determining the phrase-heads which are in turn needed to conduct the analysis. This could be a simple lexical lookup but this runs into the problem of ambiguity discussed earlier. The alternative of trying all possible phrase heads (in a worst-case scenario, all the words in the sentence) can result in extended processing times.

The result of this is that these grammars provide a very attractive framework for application development, but at the cost of time-consuming knowledge-representation overhead. Furthermore, this process requires human supervision to ensure accuracy. The biggest discouragement against using this sort of approach is that accuracy can only be ensured if the supervisor has a reasonable knowledge of the structure of the grammar as represented in the programme. This would limit the potential users to the programme developer and a few enthusiastic hackers - far from the aim of a generally accessible teaching tool.

Despite these reasons for not using unification-based grammars as the theoretical structure upon which this application is based, the simplicity of the underlying rule set of these grammars is a great attraction for this sort of application. As a result, as will be shown in section 3.2.3, the initial trials, while developing the dependency structures used in this

[26]As will be shown later, a large lexicon is unavoidable. The necessary complexity of an individual entry required by a grammar such as HPSG is the primary factor that argues against this as a model for computer-based interpretation.

application, drew extensively on the principle of head-phrase structure.

2.2.4 Dependency-based grammar

Recent developments in NLP have led to renewed interest in dependency-based grammars (Kettunen, 1999, www.nodali.sics.se/bibliotek/nodalida/1985_hki/NODA85-10/NODA85-10.txt). These grammars offer a freedom from the constraints of grammatical rules that is much closer to the requirements of this system. For example, Word Grammar, in its current format, defines syntactic structures in terms of dependencies to the extent that the syntax no longer uses phrase structure at all in describing sentence structure (Hudson, 1998, online). Instead everything that needs to be said can be described in terms of dependencies between single words (Hudson, 1994, p. 4990). Sentences are built directly by the combination of the words of the sentence, without resorting to groupings of words such as phrases. The main conjecture of dependency grammars is that the combination of the words of a sentence can be reduced to pairwise combinations of words (Kahane, 2002, p. 1).

Dependency-grammars such as the Meaning Text Theory (MTT) of Igor Mel'čuk and Richard Hudson's Word Grammar (a recent dependency-based grammar which uses some principles developed in MTT), set out to describe all aspects of language use and acquisition.[27] In practice this results in too complex a structure for the lexicon used in this application.[28] The structure of grammatical dependencies and ways in which dependencies can be inherited, though, make analysing sentences and constructing a representation of the *sense*[29] of the sentences a relatively simple task. MTT, for example, regards a natural language as a correspondence between meanings and texts (Kahane, 2001, p. 1).

In many ways grammars such as HPSG are remarkably similar to dependency-based grammars with the only major distinction being perception-based.

> *In many aspects, HPSG can be considered as a dependency grammar. Given that, in most cases, the description of a phrase is reduced to its head description, the combination of the head of a phrase with a subphrase can be seen*

[27]Keller and Asudeh's comment that "a generative grammar is empirically inadequate (and some would say theoretically uninteresting) unless it is provably learnable" (Keller and Asudeh, 2002, p. 2) could well be applied to dependency-based grammars as well.

[28]Word Grammar, for example, is defined in terms of the assumption that every linguistic concept inherits properties which are also available for non-linguistic concepts. As such an analysis of a grammar is incomplete unless it is part of a more general analysis of these parts of general knowledge (Hudson, 1994, p. 4990), making Word Grammar a theory, not only of grammar, but also of learning and knowledge representation.

[29]De Beaugrande defines '*sense*' as "the actual knowledge conveyed by a text element within its continuity of coherence."(de Beaugrande and Dressler, 1994, p. 82)

as the combination of two words: the head of the phrase and the head of the subphrase (Kahane, 2002, p. 1).

What is called a phrase in HPSG is, in dependency terms, the description of a word resulting of the combination with some of its dependents. In dependency terms, phrases are not linguistic entities such as signs, but only computational objects used to calculate the dependency tree (Kahane, 2002, p. 5).

As a result, as with Lexical Functional Grammar (LFG) and HPSG, this means that many of the rules that determine word order are contained in the lexicon, rather than explicitly stated as formal rules in the grammar. Inevitably, as with other unification-based grammars, this will lead to a large lexicon, with the problems associated with this. However, since most of the structures can be inherited from more general types, the complexity of the lexicon can be minimised, making it considerably smaller than that of a grammar that requires a detailed lexical description for every entry.

The benefit in parsing over Unification-based Grammars lies in the fact that dependencies are defined for all words. No processing is necessary to determine phrase-heads. Instead each and every word is processed in terms of its particular set of dependencies. This suggests a less complicated parsing structure than that required by phrase-structure grammars such as HPSG in that no rules are necessary to describe relationships between phrases. In short, any dependency-based system has the attraction of requiring less analysis than a constituency-based one, because the only units that need to be processed are individual words (Hudson, 1994, p. 4992).

According to Widdowson (1990, p. 97), "language learning is essentially learning how grammar functions in the achievement of meaning and it is a mistake to suppose otherwise". This view highlights one of the particular attractions of dependency-based grammars, namely that the dependencies do not only determine grammatical structure. More importantly, they are also strong indicators of the *meaning* structure of the sentence.

WG allows relatively rich dependency structures which have the benefit that almost all analysis can be conducted on what would be regarded as the surface structure of a text in other theories. In the process, the analysis yields much structural information of the kind shown in other theories by empty categories or underlying structures. This rich dependency structure is relatively easy to identify, because the additional structure is generally predictable from a simple basic structure; and once it is built, it is very easy to map onto semantic structure (Hudson, 1994, p 4992). Just as importantly, the dependencies that determine sense can be used to process sentences that do not fit the grammatical norm, mirroring the results of human interpretation of sentences.

In application design, identifying dependencies can be problematical. When lexicons become large, words will be entered which have more than one set of dependencies (since many words in English have different usages and meanings). Consider, for example, "about" which can be either an adverb or a preposition, "above" which can be an adjective, adverb or preposition, and "absent" which can be either adjective or verb. These examples are from a very small cross-section of a dictionary. As the complexity of the lexicon grows, further possibilities may present themselves.

Here too, though, WG is very attractive as a basis for building parsing engines. As all grammatical facts are located in 'isa' hierarchies there is no distinction between 'the grammar proper' and 'the lexicon'. In addition, the 'isa' hierarchies combined with the process of inheriting 'isa' relationships from parent classes of word make for a unique and simple method of handling exceptions (Hudson, 1994, p. 4993). For example, the default fact for past-tense verbs is that they are constructed as *morpheme* + *ed*. The exceptions would have an additional 'isa' relationship, so the word 'sing' would be defined in the lexicon with

$$structure\ of\ past\ [SING] = 'sang'$$

As this is a closer definition in the inheritance tree it would take precedence over the default $M + ed$ fact (Hudson, 1994, pp. 4990-4993). This feature of WG has become a fundamental aspect of the application presented here.

2.3 Artificial Intelligence

AI is used as a generic term for any system which imitates the responses of living organisms, though particular emphasis is given to that quality of living beings which enables them to learn and make probabilistic inferences based on learned data (Webopedia, 2004, `http://www.webopedia.com/TERM/artificial_intelligence.html`). As such it does not necessarily refer to any definition of intelligence applicable to people. The applications of AI examined in this dissertation are those which are in some way related to the problem of responding "intelligently" to naturally produced textual input.[30]

This section provides an overview of software that has at least superficial similarities

[30]This refers specifically to written texts. Speech recognition and production, while potentially extremely useful in a language learning context, are beyond the scope of this thesis. In a programme that could perform these functions, the differences in stress and intonation of normal speech would provide vital data that could be used to improve the accuracy of the representation of the *sense* of a text.

to the application presented in this dissertation. This software covers a range of Artificial Intelligence and Natural Language Processing systems. At the moment, the use of these types of programme is very limited in CAI systems. This can be attributed to two related issues. Firstly, most CAI applications have limited scope as they are designed to exercise particular aspects of language use. Secondly, AI systems are sophisticated programmes and it is difficult to set up the knowledge representation they require.[31]

These two factors are not compatible - there is no sense in developing an AI system to handle a programme with a small scope. Nevertheless, these systems show the most potential for enabling CAI programmes to accept communicative output. Furthermore, because many of these systems have some form of natural language interface, they are good indicators of difficulties that may be encountered in designing a text comparison system and, in some cases, indicate solutions to problems encountered in designing such a system.

2.3.1 Artificial intelligence models

Comparing texts contextually places a few basic demands on the nature of the lexical information contained in the lexical database that must make up a considerable part of this application. The first demand is that it must include some method of representing knowledge. At the outset I have assumed that knowledge can be represented as a set of relationships between words. This is borne out by Artificial Intelligence theories which are involved essentially with the constructing of knowledge trees - structures representing the relationships between concepts (referred to as nodes). The way in which these relationships are defined and interact depends on the AI model used.

2.3.2 Neural networks

In neural networks, for example, each node combines the individual influences received on its input links to determine the relationship between this input and the existing nodes. The relative strengths of the links between the nodes (represented as a mathematical weight associated with each link) determines the representation of knowledge in the database and can be used to provide models of reasoning and logical prediction based on that knowledge (Winston, 1993, pp 445-446). Neural Networks, for all their tremendous power, are difficult to set up, with a number of hidden factors that can easily lead to poor performance

[31]On average, three months of development is required to set up the knowledge base for CHAT, a fairly simple AI system designed to run small expert systems.

and sometimes total failure (Winston, 1993, p 468). Because of their complexity, in most knowledge-based systems, relationships are more firmly defined.

An example of the use of neural networks can be found in applications designed for automated essay evaluation in providing Graduate Record Examinations (GRE®) Writing Assessment scores and for summarising texts. These programmes are, in many respects, the closest in scope to matching the goals of the application generated for this dissertation as they also provide an evaluation of a complete text. In this case, however, the programmes search for identified features in the texts (such as terms like *in summary* and *in conclusion* for summarising (Powers, Burstein, Chodorow, Fowles, and Kukich, 2000, p. 5)). These features include content (in the form of prompt-specific vocabulary), rhetorical structure (indicated by rhetorical features that occur throughout extended discourse) and structure (the syntax of sentences) (Powers et al., 2000, p. 5).

These are specific to a particular type and topic of essay and so involve training based on a sample of the tests that the application is required to evaluate. This sample covers the range of scores the test is intended to generate and so also provides a basis for scoring further tests. In operation this is very similar to the approach used for expert systems discussed in section 2.3.4 on page 50 as it essentially processes a text looking for specific text that has been specified as significant. As a result the content evaluation and rhetorical structure evaluation do not provide a real indication of the relationships between the structures in the text - rather they indicate that particular vocabulary and particular words indicating rhetorical structures are present in the text. The fact that this requires training on a particular test means that this form of evaluation is only practical in environments where large numbers of learners will be producing very similar material (Powers et al., 2000, pp. 18-24).

Another application where the potential of neural networks is demonstrated is as a solution for designing systems that can respond to normal conversation. This process can be illustrated by the *Meme Machine:*

> *The program's objective is to become Turing-test capable by learning conversation patterns and being able to fit typed phrases into these learnt scripts, generating responses based on the best-fit script the system has* (2001, `http://www.wintermute.demon.co.uk/meme`).

In this programme a virtual neural network is built up from direct language experience using a simple connectionist model. The model does not include any word knowledge (i.e. that spaces generally separate characters to generate a word level) or any other preconceived high-level language notions. In this way it attempts to learn conversation patterns without any grammatical preconceptions.

Because this sort of programme demonstrates emergent behaviour, a large amount of low level input data (ideally hundreds of megabytes) is needed before a high-level shape can be formed that will generate output resembling normal conversation. In the *Meme machine* this is demonstrated by the included sample training file - the text of *Alice in Wonderland* - which it cycles through a number of times for effective training.

In programmes of this nature, however, preconceptions about learning are unavoidable. The programme learns by building patterns. Consequently the models of learning used are heavily influenced by the type of pattern the designers believe the programme should look for. Regardless of how this is constructed, the very fact of the assumption that there will be patterns makes for a preconception that will influence the type of learning that the system is capable of.

The generated network cannot, however, be used as a source for comparison with another text. The connections in the network are between adjacent structures in the text. In other words, the *Meme-machine* uses a morpheme-based bigram model to represent texts. These structures store a representation based on the sequence of words in a text, rather than the sense of the text. Conversational patterns are modelled by recycling structures that it has previously encountered. So, if a user types 'hello' the *Meme machine's* response will be drawn from sequences that include the text 'hello'. It cannot create new responses as it does not have any inherent knowledge of the relationships in the structures it has created. If, however, it has had a sufficiently large training set, it may include enough variation in its data-network to be able to have enough variation in its responses to simulate conversation (Smith, 2001, http://www.wintermute.demon.co.uk/meme).

Consequently, while the *Meme machine* provides a great deal of insight into methods for training linguistic computer models, the model of learning used in the *Meme machine* does not provide a useful model for evaluating texts. The *Meme-machine* starts processing without any linguistic knowledge built into it - not even the concept that spaces delineate words. As such it will not have any basis for generating feedback on any aspect of textuality, be this well-formedness, accuracy, cohesion or cohesiveness. All the Meme-machine supplies is a network based on the order of memes in the training set. It cannot and is not intended to generate a representation of the sense of a text. Without this representation the task of determining if two texts are similar or if a text has the necessary content is impossible.

A method, thus, has to be found of representing the underlying structures inherent in the texts. One possible solution to this problem is outlined in this dissertation (cf. section 4.3).

2.3.3 Natural language processing

AI is a broad term covering a number of computer applications with the common goal of trying to duplicate at some level the processes involved in human reasoning. Usually this involves attempting to draw inferences and conclusions from a limited set of data. In other words, AI is essentially concerned with problem solving. In the realm of NLP and Computational Linguistics (CL) this is applied to three distinct processes. Possibly the most popular at the moment is Speech Recognition, which attempts to convert normal speech into texts that can be processed by computer. Given the availability of commercial Speech Recognition systems, this is becoming a powerful tool for ESL teaching. A significant problem associated with these systems is the difference in accent and rhythm used by speakers of a language in different countries, cities and even within a city. As with many limiting factors, it is possible to turn this into a benefit. Holland et al. report that, while recognition failures caused some initial frustration in MiLT system, the recognition rate improved during use, possibly because the students' pronunciation improved while using the system (1999, p. 352). The necessity of using a particular accent and rhythm in order for the system to recognise input speech has the potential of making speech enabled software a powerful training tool for teaching pronunciation.

A second form of NLP is speech generation. At first glance this is a considerably simpler problem than speech recognition as a reasonable reproduction can be achieved from a phonetic representation of words. Even without phonetic representation fully understandable output can be achieved by associating letters with sounds - an approach that has been used successfully in computer readers for visually impaired computer users for a number of years. Much of conveying meaning in speech, however, relies on tonal differences and inflection which are much harder elements to represent. As a result, achieving natural sounding speech is still a challenge, though here too the programmes currently available can give ESL students an indication of how words in English would sound and can prove to be a valuable resource for the ESL classroom.

The third process associated with NLP is deriving meaningful information from naturally produced text. In itself this covers a wide range of processes as it includes such diverse applications as machine translation, database interfaces and systems that simulate human conversation - - often called "expert systems".[32]

The ongoing interest in expert systems has resulted in considerable interest in formal grammars for natural languages in recent times. These systems are programmes that, in important respects, behave like an expert on some specific subject. As such they need a fund of factual knowledge, interface rules for making logical deductions from facts and a linguistic component for input and output in natural language (Koster, 1991, p. 15). As such these systems have a number of similarities with the chatbot programmes discussed in section 2.3.4 on page 50. The most important differences being the extent of the knowledge base of the expert system and the focus on interface rules for making logical deductions.

As expert systems are required primarily to provide solutions to problems, there is less of an emphasis on providing output that closely resembles interpersonal communication. The necessity of making logical deductions has one implication of particular interest in terms of their use in developing systems capable of fulfilling the requirements of allowing comprehensible output in CAI (the only part of these programmes that is of particular interest in this case). This is that the lexical definitions must contain sufficient information to ensure that logical deductions can be made. This is not the case with most chatbot applications where the only requirement is that the application be able to respond in a manner that is believable as a possible human response. To see the effect of this, consider, for example, the lexical definitions used in the Brainhat project (Dowd, 2004, http://www.brainhat.com). The method of knowledge representation used in this system is fairly common in expert systems. As such, the observations made about this system apply to expert systems in general.

In this expert system a "concept" is defined as the basic unit of meaning. All words, be they nouns, verbs, prepositions, adjectives or any other type, are defined in terms of the idea of a "concept" as the basic unit of meaning. This meaning, in turn, is represented by a set of ten[33] types of relationships that can exist between concepts. These relationships are defined as:

1. LABEL: a synonym of the word;

[32]Systems that simulate human conversation have a long history, since Turing proposed the Turing Test for conversation systems in the 1950's. A yearly contest is held for the Loebner prize (cf. `http://www.loebner.net/Prizef/loebner-prize.html`) to determine the best Turing system. To date the systems winning this prize still require a restricting scenario in order to function believably. No system has been developed capable of encapsulating sufficient real-world data to be able to respond adequately to the potential range of human conversation. The best to date is ALICE with 30% believability over a five minute period.

[33]Some AI applications define far more relationships than the ten used in Brainhat, with the result that the complexity of their initial lexicon is proportionately greater.

2. **CHILD**: a word that is a more specific case of another word. For example "rose", as a more specific case of the word "flower", can be defined as a child of "flower." Likewise a "ball" could be called a child of the word "toy";

3. **PARENT**: the opposite of child - links to the categories to which a concept belongs. In the above example, "flower" is the parent of "rose". Other unrelated parents of "rose" could be "colour" and "person" (for the name "Rose");

4. **WANTS**: common attributes associated with an object. For example, a "ball" would have a colour and a size - it would be defined with the relationships "wants colour" and "wants size." This type of relationship is intended to help AI systems distinguish between distinct meanings of a word. For example, the word "red" can mean both a colour and "communist". The "wants colour" relationship defined for a ball means that the AI system would have a bias in favour of the word "red" being understood as a colour in the context of a phrase like "the red ball". Similarly, if the concept of a country was defined with "wants politics", then the word "red " in the phrase "Red China" would be understood as meaning "communist";

5. **TYPICALLY**: similar in concept to the "wants" relationship. This is used to define characteristics typical of a concept. For example, a ball could be defined with "typically round";

6. **ORTHOGONAL**: Two concepts are "orthogonal" when they cannot exist together. So, for example, the colour "red" cannot also be the colour "blue". These would then be defined as "orthogonal" with respect to colour, as in:

```
blue

    child      colour
    orthogonal colour

red

    child      colour
    orthogonal colour
```

7. **RELATED**: concepts that are related to a word but not necessarily synonymous. For example, "happy" and "fun" are related though they have very different meanings;

8. **TENSE**: used to define time information for verbs and auxiliary verbs;

9. **NUMBER**: usually used with nouns and verbs. In Brainhat the only possible values are "singular" and "plural" though some systems allow a more detailed definition (Brainhat's developers intend implementing a more detailed definition of number in Brainhat in later versions of the programme);

10. **PERSON**: used with nouns to define "first", "second" or "third" person. (Dowd, 2004, http://www.brainhat.com/technote.php)

This results in a complex network of relationships for any given word - a network that is absolutely necessary if the system is going to function as a reasoning system.

While systems like this are often able to train themselves, they need an initial lexicon to serve as starting point. This lexicon defines an initial set of relationships which can be used to determine relationships with new concepts. This initial knowledge base would need to be fairly detailed as it would not be able to determine relationships for words in sentences totally unrelated to concepts already contained in the knowledge base. Constructing these initial knowledge bases is time-consuming as every word added results in a substantial increase in the number of links defined in the knowledge base. (Care must be taken to ensure that the words added do not result in circular definitions or jumps that would detrimentally effect the system's reasoning capabilities - a danger that increases as the size of the knowledge base increases. This danger is one reason why the use of these systems is generally limited to relatively simple scenarios which can be described fairly completely in the initial knowledge base (Dowd, 2004, http://www.brainhat.com/technote.php)).

This approach makes AI programming well suited to simple deductive reasoning. Inside its scenario, it is possible to have a programme answer sometimes fairly sophisticated questions and respond intelligently to those questions. The Brainhat system, for example, is capable of handling digression within a conversation, answer questions outside its predefined scripts and even answer questions with a question if it lacks sufficient information.

The limiting factors involved remain, however, that a system of this nature is not capable of much outside of the defined scenario and that these scenarios have to be constructed as a carefully planned set of relationships covering all the world information of the scenario. Obviously this network of relationships will have omissions. For example, none of the relationships listed here can describe the relationship that exists between the words "first" and "second" nor can these relationships define words like "hello" and "goodbye" sufficiently for this definition to ensure an expected response to a learner initiating or closing a conversation. (These particular sequences of responses, in particular, are usually programmed separately in any system that attempts to model conversational behaviour, be this chatbot or expert system.) It is important to remember that each extra relationship that is defined for words in the lexicon does not only apply to that set of words for which it is intended. A relationship that allowed the definition of a numerical would also be present in every word that does not have a numerical relationship. Similarly, the logic that enables the system to process entries containing these relationships would be applied to all the words in responses and in the knowledge base, not just those words that contain or define numerical relationships. Each relationship that is added would thus result in a considerable increase in the complexity of the lexicon and the time needed to process entries.

The last and most important limitation for use in a language teaching role is the lack of creative ability. Even with a deductive facility which enables it to "understand" texts inside its programmed scenario, the range of responses available to the system remains limited. Within its programmed scenario, however, Brainhat is capable of understanding language, evaluating ideas, and asking and answering questions. Because Brainhat is knowledge representation-based, it can digress within a conversation, answer questions outside its script, and keep rich context. It can even handle questions that are answered with questions (`http://www.brainhat.com/goals.html`). The limitation of this type of system stems from the fact that it still relies on a script to provide the structure for any interaction, which means that the scenarios in which it can be used are restricted to limited scope, prepared sessions such as personal assistants, kiosks, switchboards, and help desk applications.

An argument against the use of this sort of system as a basis for contextual evaluation is that the system requires a crafted knowledge-base. The programme cannot derive knowledge from a document - a feature that is necessary in an application which pretends to allow contextual comparison (the chosen method of enabling applications to support a communicative output).

2.3.4 Chatbots and artificial intelligences

Interactivity implies two conscious agencies in conversation, playfully and spontaneously developing a mutual discourse, taking cues and suggestions from each other as they proceed (Stone, 1995, p. 11).

To some extent the possible inadequacy of the medium can be compensated for by careful employment of chatbots - carefully constructed chatbots that can respond in a credible way as characters in the shared environment. These have become quite sophisticated. The virtual world makes an environment that can reasonably completely be encapsulated in a chatbot database, ensuring that interaction with the chatbot closely resembles the interaction between people. Advanced examples of chatbots are capable of responding to questions, keeping state information for conversations with different participants, enabling a believable continuity in conversation and can even learn from these interactions (an example of this sort of use is the Toei Rei chatbot - `http://imaginaryrealities.imaginary.com/volume2/issue5/toei_rei.html`). However, for all this sophistication chatbots have two major limiting problems. The first is that the design of the chatbot requires a reasonable understanding of programming in order to successfully set up the chatbot knowledge base. The second and more important limitation is that the chatbot's responses are, like its earliest predecessors, still limited to needing keywords in order for them to be able to respond at all to statements and questions (cf. Offereins, 1994, p. 260). There is no inherent ability to interpret novel text, though it is possible to programme a set of responses that obscure this inability (albeit that the limitation is a major one).

For all this, these chatbots, many of them with the resources of fairly sophisticated computing hardware to draw on, are considerably more sophisticated than their equivalents implemented in many commercial role-playing games (RPGs). The RPG implementations almost without exception operate on the principle of keyword recognition - to the extent that it is usually not necessary to enter phrases, let alone sentences, in order to trigger a desired response.

Outside of game environments chatbots are used to provide natural language interfaces to knowledge bases and provide information about selected subjects. The principle of operation, though, remains the same, with the responses tied to the particular area of expertise of the chatbot. In a game this is usually the game environment, while in these information systems it can be any set of knowledge in which most of the questions can be predicted in advance and predefined responses generated.

These are not genuine AI systems which can be defined as systems which attempt to model human reasoning. Because of the necessity for believability in the shared environment, MUD and MUSH implementations are often the most sophisticated implementations of these. In these there is often some overlap in terms of their having a limited problem-solving capability - even if this only means an ability to find their way from one part of the virtual world to another.

1181 2334X

The overwhelming liability of all chatbot implementations is the difficulty in properly determining the range of responses. Like the earliest implementation - Wiezenbaum's ELIZA programme (Weizenbaum, 1976) that has since been duplicated in many programming languages and distributed all over the Internet - and its descendant - ALICE, the 2000 and 2001 Loebner prize winner - these require a set of carefully scripted responses, often in the form of questions. The principle is simple. If a chatbot responds by asking questions, it is harder to pin it down and cause inconsistencies in its responses. At the same time this runs counter to the notion of lingual creativity. As a result, even when the chatbot responses are credible, they cannot truly replicate fluency in inter-personal communication.[34]

Genuine AI applications are designed to have the ability to make probabilistic inferences based on the data in a database. As such they have a greater ability to analyse input for keywords that can provide an entry point into the database. In a communicative teaching context, however, this is still a keyword-driven approach - a liability that is compounded by the fact that the parsing routines are limited to a fairly limiting set of question forms of sentences in order to better determine the search criteria. As these systems are intended primarily as knowledge-base interfaces, the generative capability of AI systems is also often weak if not non-existent. As with the generic question-form responses common to chatbots, this suppresses lingual creativity, making existing examples largely unsuitable for use in language teaching.

2.4 The evolution of linguistic ability

Theories of SLA give insights into how human linguistic ability may have evolved. At the same time, hypotheses of how humans evolved can shed light on the origins and potential of the human linguistic system, and in so doing provide a usable model for representing language and meaning computationally. Language has evolved to convey meaning, so a study of the evolution of language is, at the same time, a study of the development of ways of conveying and representing increasingly complex meanings.

As the development of computerised systems of text interpretation can be seen as a parallel to human linguistic evolution, an examination of the evolution of linguistic ability can provide useful guidelines for the development of NLP systems. In addition, this can

[34]Stubbs defines fluency as the ability to:

improvise, maintain continuity in speech and comprehension, respond immediately to unexpected utterances, make rapid changes of topic and speaker, and so on (Stubbs, 1983, p. 36).

provide insights into how language is represented internally which, in turn can be used to provide more effective environments for SLA.

2.4.1 Evolutionary origins

It is desirable to avoid complexity when describing the processes involved in language generation and interpretation. A simple description of these processes has numerous benefits, not least of which is the amount of processing time it saves. This time can be utilised better in determining the meaning of texts (as Ritchie points out, if someone is prepared to wait a second for a response most of that second will have to be used in generating the response and not in parsing the text (1987, p 233)). It is quite possible that this is true for people as well as artificial systems. Communication is a complex act yet it is accomplished with seeming ease. This argues strongly for a simple mechanism that does not demand much processing.[35] An integrated, comprehensive approach may actually lead to a simpler account of language overall than a fragmented, restricted one (de Beaugrande and Dressler, 1994, p xiv).

The selective forces acting on language to be usable by humans are significantly stronger than the selective pressure on humans to be able to use language. In the case of the former, a language can only survive if it is learnable and processable by humans. Many, such as Briscoe (2000), Kirby and Hurford (1997a,b) and Kirby (1998), argue that the language acquisition device must have evolved along with learnable languages, leading to a strong inductive bias in language acquisition.

Others argue that adaptation toward language use is one out of many selective pressures working on humans. Language is more likely to have adapted itself to its human hosts than the other way round (Christiansen, 1994, p 125).

As Hurford states:

> *The Bickertonian picture of over a million and a half years during which Homo erectus used protolanguages is easier to envisage as a continuum, with perhaps gradually expanding vocabularies, gradually faster speech and comprehension, and steady compression of (proto)language acquisition into the critical*

[35]This is supported by the recent history of the development and abandonment of linguistic theories. Theories are adopted when they seem to offer a simple explanation for grammatical phenomena and, in due course, are abandoned when they either fail to represent these phenomena fully or acquire a collection of peripheral rules to handle exceptions that ultimately leads to their failing to provide a conceptionally simple description of grammar (cf. Newmeyer, 1986).

> *period before puberty. Such gradual changes can be (intuitively) reconciled with the increase in brain size over the period* (Hurford, 1999, p 187).

Computationally this is useful as it provides a good guideline for developing a system governed by simplicity and ease of acquisition. Also in an evolutionary sense a complex system does not make much sense.

> *A theory of language should be consistent with the neo-Darwinian theory of evolution* (Worden, 1998, http://citeseer.nj.nec.com/context/201865/0).

One problem in the study of language evolution has been the tendency to identify contemporary features of human language and suggest scenarios in which these would be advantageous. This approach ignores the fact that if language has evolved, it must have done so from a relatively simple precursor (Nowak and Krakauer, 1999, p 8028). As humans evolved, communication evolved - and of necessity the languages used for communication also evolved.

Any grammatical theory that has complex structures and relationships as a basis will have to include a method of developing those structures. This means a process of developing gradually more complex structures as the communicative ability of the species that form our ancestry improved.[36] Either that or we have to assume that one of our predecessors evolved with a complete grammar imprinted in its organic circuitry before there was any need for the complexity of communication that this represents - a view implicit in the "principles and parameters" picture introduced by Chomsky (Worden, 1998, http://citeseer.nj.nec.com/context/201865/0).

While there is evidence that humans perceive and produce speech sounds as tokens of discrete categorical types, and that this ability is partly innate, it is also at least partly learned (Eimas et al., 1971). Even if innate, the ability to perceive such categories must have developed along with other linguistic abilities, as opposed to being present in fully developed form before the emergence of grammar (Batali, 1998, p. 5).

Worden (1998, http://citeseer.nj.nec.com/context/201865/0) developed a computational model of learning which demonstrates that language could be viewed as script functions learnt individually for hundreds, then thousands, of words. The syntax of language would be embodied in those script functions. Moreover, a script learning mechanism

[36]Using evolutionary game theory, Nowak and Krakauer (1999) mathematically model ways in which protolanguages can evolve in a nonlinguistic society. They argue that grammar originated as a simplified rule system that evolved by natural selection to reduce mistakes in communication.

> *is robust, efficient, and has plenty of evidence to learn from over years of childhood (any word can be learnt from about ten clear examples of its use, which a child does not have to wait long to hear). This model of learning works well computationally, seems to agree well with a lot of evidence about language acquisition, and suffers none of the conceptual difficulties which the 'principles and parameters' theory has with language change and bilingualism (which would seem to require parameters with intermediate or multiple values, thus losing much of the attractiveness of the parameter idea)* (Worden, 1998, http://citeseer.nj.nec.com/context/201865/0).

It seems more reasonable to assume that the processes of generation and interpretation developed as it became necessary and useful for these processes to exist. These processes would have been the simplest processes that would achieve the desired communicative outcome. They would have been based on traits that had already been established as beneficial (in an evolutionary sense). According to Worden (1998), this can best be explained by the hypothesis that language evolved out of primate social intelligence.

> *So, as a theory of how language arose - largely by reuse of pre-existing structures and operations in the brain for social intelligence and the theory of mind - the theory agrees with a wide range of data and constraints* (Worden, 1998, http://citeseer.nj.nec.com/context/201865/0).

The view adopted in this dissertation is that language and language learning are closely related to human pattern recognition. People have an innate tendency to see patterns. We not only see them, we try to create them - even when no patterns actually exist (as can be seen in the well-known 'gambler's fallacy' which has millions looking for patterns in the random numbers on a roulette table, roll of a dice or in lottery numbers). A similar process could explain how grammatical rules are learned - not from an inbuilt grammatical competence but rather from a highly developed system of pattern recognition. A pattern, in this sense, is a synonym for a generalisation. Seeing as the rules that are defined in grammatical systems can be better classified as generalisations, it follows that the patterns form the grammatical rules with which we are familiar. This makes the identification of patterns language learners receive during their formative years a very good candidate for an explanation of grammar acquisition - especially given the large body of training data available to them and the frequent reinforcement of evidence of correct acquisition.

2.4.2 New text forms

In the words of de Beaugrande and Dressler

> *The fact that humans can and do communicate successfully in a staggering range of settings indicates that there must be a limited set of powerful, regular strategies at work [. . .]. In its attempts to isolate single systems (phonology, morphology, syntax, etc) and to keep language distinct from everything else, linguistic research may have remained on a superficial plane that* increased *complexity of study rather than reducing it. On a more powerful plane, a simpler and more unified account of human language may yet be forthcoming* (de Beaugrande and Dressler, 1994, p. 220).

It is reasonable to assume that, as new textual forms are developed, the strategies already developed will be carried over to the new textual environment, rather than that new strategies will be invented for that environment. In this way, for example, road-signs could be developed which use an extremely abbreviated style of text presentation with the expectation that they would be consistently and correctly interpreted by the people who see them. The context of the sign provides the information to fill the missing dependencies in the abbreviated text. This ensures that a sign that, for example, reads

SLOW
CHILDREN
CROSSING

would be interpreted as meaning "in the context of driving here you should slow down because children might be in the road", rather than "the children here cross slowly" even though the latter interpretation is closer to that provided by grammatical well-formedness conditions. (It should be noted that the context is powerful enough that either interpretation would still be an effective warning to be careful because of the danger of children being in the road.)

It is logical to suppose that relations are largely developed when it becomes necessary and desirable to have means of expressing new concepts. Likewise it is logical to suppose that the process of generating communicative events and the process of interpreting these events must be related - the one could not develop independently of the other. It follows then that grammatical relations would be largely defined in the lexicon, instead of as a set of rules existing independently, simply because it would be simpler to define new relations in the lexicon - specifically associated with the new concepts - rather than define new rules that describe the concepts.

2.5 Text linguistics

Theories in text linguistics attempt to determine patterns in the structure of communication itself. In particular, studies in text linguistics focus on how texts function in human interaction. As the application developed here is intended to provide a tool that enables learners to experiment with communicating in English, it is necessary to have an awareness of the learners' expectations of the outcome of their communicative experiences.

A goal of the programme is to be able to evaluate texts contextually. This is a general statement and it is necessary to define what is meant by "contextually" in the context of this application.

2.5.1 Standards of textuality

Text linguistics defines a text as any communicative occurrence which meets seven standards of textuality. If any of these standards are not considered or have not been satisfied, the text will not be communicative. This means that non-communicative texts are regarded as non-texts. The seven criteria for textuality are cohesion, coherence, intentionality, acceptability, informativity, situationality and intertextuality.

COHESION entails the manner in which the words of the surface text are mutually connected within a sequence (de Beaugrande and Dressler, 1994, p 3). In other words, cohesion refers to the connectedness of the words in a text. The most obvious illustration of this is the language system of syntax which imposes organisational patterns on the surface text (de Beaugrande and Dressler, 1994, p 48). The surface components depend upon each other according to grammatical forms and conventions, such that cohesion rests upon grammatical dependencies (de Beaugrande and Dressler, 1994, p 3).

The second level of textuality is COHERENCE - the logical flow of ideas in a text. This comprises the ways in which the components of the textual world (the configuration of concepts and relations which underlie the surface text) are mutually accessible and relevant where concepts refer to a configuration of knowledge which can be recovered or activated in the mind and relations are the links between concepts which appear together in the textual world. De Beaugrande and Dressler (1994, p 4-7) show a number of ways in which concepts can be related.

In the application developed here, a simplified method of representation is used in which verbs and prepositions are regarded as defining relationships between objects. Because verbs generally define an agent and an affected entity (cf. de Beaugrande and Dressler, 1994, p 6) these links are viewed as vectors linking two concepts in the application (Li et al., 2003, pp. 85-86).[37] Effectively, a linked set of concepts thus represents the sense contained in a sentence (cf. section 2.5.3).

INTENTIONALITY is one facet of textuality that the application cannot evaluate. This refers to the text producer's attitude that the set of occurrences should constitute a cohesive and coherent text instrumental in fulfilling the producers' intentions. It involves, for example, the producer's intention to distribute knowledge according to some goal or plan. At present there is no way of determining or evaluating the producer's intentions. It should, however, be safe to assume that texts evaluated in this application do fulfil this criterion - the more so as text users normally exercise tolerance toward products whose conditions of occurrence make it hard to uphold cohesion and coherence altogether (de Beaugrande and Dressler, 1994, p 7).

ACCEPTABILITY represents the text receiver's attitude that the set of occurrences should constitute a cohesive and coherent text having some use or relevance for the receiver (de Beaugrande and Dressler, 1994, p 8). While it will not be possible to evaluate this properly, some indication of this can be gained, for example, by evaluating the degree to which learner texts contain relevant relationships as compared to those contained in a model answer (assuming the same topic). A text which does not have any corresponding relationships when compared to a model answer can be assumed to be off-topic and hence unacceptable.

INFORMATIVITY refers to the extent to which the occurrences of the presented text are expected as opposed to unexpected or known as opposed to unknown (de Beaugrande and Dressler, 1994, p 8). This is one of the hardest criteria to evaluate. As a teacher the content of a learner essay would probably not contain novel information - any text containing novel information would not be possible to evaluate in terms of acceptability as this can only be evaluated in comparison to a model text. A measure of divergence between a learner text and a model text could be used as an indicator of informativity, but this will be at the

[37]Parts of speech such as verbs and prepositions define links between two concepts and also indicate a direction of linkage (such as *cause* $\rightarrow$ *effect*, *agent* $\rightarrow$ *object*, *actor* $\rightarrow$ *target*, etc). The term 'vector' is used in this dissertation to indicate all parts of speech with these properties (cf. Li et al., 2003, pp. 85-86).

expense of using this as a measure of acceptability.

SITUATIONALITY includes those factors that make a text relative to a situation (de Beaugrande and Dressler, 1994, p 9). In different situations, the nature of what is an acceptable text changes. Again this is impossible to evaluate as it requires an extensive knowledge of different contexts as well as the types of text appropriate to those contexts (following from the discussion in section 2.4.2, this implies that appropriate dependencies need to be defined for each textual context). This application is limited to paragraphs and short essays at the moment, a factor which means that we can assume acceptability of any text which can be represented as a paragraph or short essay.

INTERTEXTUALITY, which refers to the factors which make utilisation of knowledge of one text dependent upon knowledge of one or more previously encountered texts (de Beaugrande and Dressler, 1994, p 11), also requires an extensive world knowledge. Once again it will not be possible to evaluate this aspect of textuality, though it may be possible to simulate the references to knowledge outside the text in some cases. An example of this would be initialising the application with a base text containing background information when evaluating texts. Any reference in the text being evaluated to knowledge contained in the background text would be evaluated consistently in each learner text.

In general, the computer has no predefined idea of an ideal text or even of a model of a communicative event. There is, thus, no standard against which any criterion which implies a judgement of the producer's efforts can be measured. Informativity can be tested as a measure of the degree to which a text contains novel relationships, but this will not provide an indication of the degree to which these relationships are acceptable. Acceptability can be tested as a measure of the degree to which a text contains expected relationships, though this is then at the expense of any indication of informativity. Situationality and intertextuality again require both extensive world-knowledge and knowledge of texts and that the programme make a judgement of the learner output in terms of its resemblance to data from that world knowledge. This, again, will not be possible.

It follows then that the application developed in this dissertation will focus, primarily, on the levels of cohesion and coherence. These will be discussed in more detail in the following sections.

2.5.2 Cohesion

There has been considerable overlap between theories in text linguistics and AI. This is especially evident at the level of cohesion. A number of simulations have been constructed which test the construction of networks representing relationships within a sentence - attempts in general to determine a single deep-structure underlying a number of possible surface structures. One such network system is the augmented transition network. This network is configured of nodes (comprising grammatical states) connected by links (defined by grammatical dependencies). The links in this system are synonymous with rules linking nodes of different types (de Beaugrande and Dressler, 1994, p 50).

This has been used as a starting point in determining a structure for constructing networks of relationships that represent knowledge structures of complete texts. Most abstract grammars have not had the real-time processes involved in the task of creating links as a prominent design criterion (de Beaugrande and Dressler, 1994, p 50). This is, however, an important consideration in any implementation of an engine aimed at textual analysis. Augmented transition networks consist of nodes (in this case grammar states) connected by links (in this case grammatical dependencies). Moving from one node to another involves a transition across a link - a process that demands the identification of the link as one of a number of dependency types (for example "subject-to-verb" or "modifier-to-head").

The primary difference between the augmented transition network and the network representation used in this dissertation is that defining links as grammatical dependencies limited the extent to which complex concepts could be represented. This representation works within a sentence but fails to account adequately for links outside a sentence as these links cannot always be defined in terms of grammatical dependencies. While a set of semantic dependencies has been included that allow links to be defined that refer outside an individual sentence this requires a greater level of detail in the lexicon than has currently been implemented. In principle, by allowing a cumulative network of nodes to be built for all the sentences in a multi-sentence text, the relationships defined in the network can reflect the interconnectedness of relationships inside that text (both inside a sentence and across sentences).

2.5.3 Coherence

> *If* MEANING *is used to designate the* potential *of a language expression (or other sign) for representing and conveying knowledge, then we can use* SENSE *to designate the knowledge that* actually *is conveyed by expressions occurring in a text* (de Beaugrande and Dressler, 1994, p 84).

This definition of sense plays an important part in this dissertation. Any network of relationships which is supposed to be a representation of a text is, by definition, a representation of the *sense* of that text.

A text "makes sense" because there is a continuity of senses in the knowledge activated by the expressions of the text (the sentences). It follows logically that sense is a cumulative event. The greater the degree to which the expressions in the text build on the sense that has been constructed by previous expressions, the greater the degree to which the text can be said to be coherent.

It is, however, impossible to put a quantifiable figure on the degree to which expressions have to influence each other in order for a text to be judged coherent. One can, however, assume that learner texts will be intended to be coherent and use this intention as a basis for evaluation. For example, this makes it possible to compare these texts with other texts known to be coherent and known to be meaningful.

This presupposes a means of representing texts in a relational network. In the system of processing proposed by de Beaugrande and Dressler (1994, p 95-97), surface text is parsed into a configuration of grammatical dependencies. Surface expressions are taken as cues to activate concepts. These are treated as steps in the construction of a continuity of sense and the extent of processing expended will vary according to whatever is useful for that task. In this attention would be directed primarily toward the discovery of control centres (i.e. points from which accessing and processing can strategically be done). De Beaugrande and Dressler call these most likely candidates "primary concepts" and these include:

OBJECTS conceptional entities with a stable identity and constitution;

SITUATIONS configurations of mutually present objects in their current states;

EVENTS occurrences which change a situation or a state within a situation;

ACTIONS events intentionally brought about by an agent.

Other concepts would be assigned to a typology of secondary concepts. The following summary is taken from de Beaugrande and Dressler (1994, p 95-97) and is in turn based on de Beaugrande (1980):

STATE: the temporary, rather than characteristic, condition of an entity;

AGENT: the force-possessing entity that performs an action and thus changes a situation;

AFFECTED ENTITY: the entity whose situation is changed by an event or action in which it figures as neither agent nor instrument;

RELATION: a residual category for incidental, detailed relationships like "father-child", "boss-employee", etc.;

ATTRIBUTE: the characteristic condition of an entity (cf. "state")

LOCATION: spacial position of an entity;

TIME: temporal position of a situation (state) or event;

MOTION: change in location;

INSTRUMENT: a non-intentional object providing the means for an event;

FORM: shape, contour, and the like;

PART: a component or segment of an entity;

SUBSTANCE: materials from which an entity is composed;

CONTAINMENT: the location of one entity inside another but not as part of substance;

CAUSE: the way in which a situation or event affects the conditions of another one;

ENABLEMENT: the way in which a situation or event creates sufficient, but not necessary, conditions to affect the conditions of another one;

REASON: a relation where an action follows as a response to some previous event;

PURPOSE: an event or situation that is planned to become possible via a previous event or situation;

APPERCEPTION: operations of sensorially endowed entities during which knowledge is integrated via sensory organs;

COGNITION: storing, organising, and using knowledge by a sensorially endowed entity;

VOLITION: activity of will or desire by a sensorially endowed entity;

RECOGNITION: successful match between apperception and prior cognition;

COMMUNICATION: activity of expressing and transmitting cognitions by a sensorially endowed entity;

POSSESSION: relationship in which a sensorially endowed entity is believed (or believes itself) to own and control an entity;

INSTANCE: a member of a class inheriting all non-cancelled traits of the class;

SPECIFICATION: relationship between a superclass and a subclass, with a statement of the narrower traits of the latter;

QUANTITY: a concept of number, extent, scale, or measurement;

MODALITY: concept of necessity, probability, possibility, permissibility, obligation, or of their opposites;

SIGNIFICANCE: a symbolic meaning assigned to an entity;

VALUE: assignment of the worth of an entity in terms of other entities;

EQUIVALENCE: equality, sameness, correspondence, and the like;

OPPOSITION: the converse of equivalence;

CO-REFERENCE: relationship where different expressions activate the same text-world entity (or configuration of entities); and

RECURRENCE: the relation where the same expression activates a concept, but not necessarily with the same reference to an entity, or with the same sense.

In addition, de Beaugrande defines a set of operators which further specify the status of linkage. This model allows detailed deconstruction of a text and generation of complex relationship networks representing the sense contained in the text.

As a model for computerised processing, though, it becomes unwieldy as each and every entry in the lexicon of the programme would require a definition in terms of a number of these concepts. In addition, while the taxonomy is organised from the perspective of a text receiver (a text producer's taxonomy could differ), there is an inherent problem in the taxonomy from a computational perspective. The list indicates value judgements of the importance of links. This presupposes an evaluation of the content prior to constructing the network of relationships. Computationally this is self-contradictory if this taxonomy is intended to be the primary means of evaluation and representation of content networks. In spite of this there is a clear superficial correspondence between this taxonomy and that used in the classification of objects in AI systems such as Brainhat (cf. section 2.3.3).

It is the contention of this dissertation that the simplest and most effective means of representing the relationships in texts is to use the relationships defined explicitly in the sentences of texts. In other words relationships are defined by the words used in the text, rather than by a symbolic representation of features. This factor enhances the attractiveness of WG as the theoretical basis of the application.[38] These relationships can grow and expand as the programme is used by allowing the structures inherent in sentences to define new relationships in the network. A new relationship in a taxonomy, such as that of de Beaugrande, would be extremely hard to extend should a new relationship need to be defined, and de Beaugrande does state that his list is not exhaustive. An open-ended list composed of a network of nodes linked by verbal vectors is, by definition, an open-ended system. If the logic of a system allows these nodes to be traversed at runtime then any number of nodes and types of relationship between these nodes can be defined. Common relationships would be represented in a similar format simply because of the similar terms of reference in their everyday use in a text.

The only real argument against this would then be the size of the training set necessary to build a sufficiently representative set of relationships. This is not a trivial argument. A complete system using this sort of linkage is not feasible simply because of the quantity of training data necessary (for comparison consider the amount of data children are exposed to as they build their knowledge networks).[39]

[38]Hudson claims that WG grammar is not in itself a model of either production or perception, but simply provides a network of knowledge which the processor can exploit (1998, p. 15)

[39]However, as is demonstrated in the application developed here, it may not be necessary to have a complete, predefined system of relationships to make this type of application function effectively

3 Application requirements

Application development involves an intersection between disciplines with very different sets of demands. In this instance we have language learning, grammatical theory and computer programming. When developing language learning software, none can exist independently. Programmes have to be designed with language learning goals as a primary requirement. These goals in turn are affected by the capabilities of computer hardware and the potential for representing those goals programmatically. This section examines the demands of these two fields and how the intersection of these disciplines affects the practical viability of this kind of application.

3.1 CALL design issues

Because the use of CAI overlaps with the actual content of the class, it is very easy to focus on the detail of the computer rather than CAI as being just another classroom tool. The most important considerations in this type of design are those that affect the application's use as an aid in language learning. In the words of Shannon Jacobs:

> *The underlying goals are to show what this kind of tool is capable of without getting lost in the "glamor" of the computer ... We need to remember that the students come first, then the course goals and lesson plans, and finally the specific teaching tools that help the students do the actual learning* (1999, p. 1).

From the outset, this application is aimed at analysing texts. This means that the focus of its use becomes a question of what kind of analysis is possible and how this analysis can be used to facilitate learning (for example, by providing information to a teacher or learner about areas where learners either have or lack certain communicative skills).

3.1.1 Feedback and flexibility

The nature of the feedback itself is very important as this plays an extremely important role in cementing knowledge. The feedback desired defines the goals of a CAI application and is, in turn, defined by the goals of the application. In this application the goal is to provide a textual analysis tool. Consequently, the feedback provided has to reflect this analysis and use this in a manner that supports the learner.

An analysis is an examination of the content of a text. It should provide an indication of the degree to which a learner has accurately conveyed a message. Ideally it should also provide an indication of the extent to which this message is coherent and of the cohesiveness of the text as a whole. Much of this depends on understanding the learner's message and comparing the learner's delivery of that message with that of an ideal speaker.[40]

Feedback must then reflect this analysis in a manner that supports the language learning efforts of a student and the teaching efforts of an educator. For example, if the teacher's role should, as Rope states, be control - "control, that is, of all pertinent aspects of what is presented to the student and what the student is invited to do with it"(Rope, 1985, p. 67), then the feedback generated by the application should support this role.

Feedback should highlight areas where a student has been effective in communicating and areas where the student has not been effective in communicating. Ideally, feedback would suggest methods of improving in accuracy, coherence and cohesion. It is not possible to do this explicitly at this early stage of development. It is, however, possible to determine (to an extent) areas where a learner has been successful or unsuccessful in conveying a message.

Given this fundamental data, good feedback then becomes a matter of presentation. The data gained from an analysis provides a summary of the content of the text. This then reflects the coherence and cohesion of the text. Shortcomings in either of these facets of communication will also be reflected in the summary generated from the text. Providing good and effective feedback, in turn, involves understanding the data on which the feedback is based and on knowing which kind of feedback will be most effective in a particular learning environment with a particular set of learners. This also means that this application must be designed with the aim of making the generated data accessible and meaningful. To assist in understanding the data that are generated, tools are provided which highlight the shortcomings in the data. Once these shortcomings have been identified appropriate feedback can be generated.

As students and teachers differ, the particular type of feedback that will be effective in one class will be different to the type of feedback that will be effective in a different

[40]This is impossible to achieve computationally at present simply because we do not have a model of language detailed enough to allow an application to 'understand' a learner's message (never mind understand a learner's message which contains errors in accuracy). It is possible, however, to simulate this to a degree. The extent to which this implementation achieves this is described in more detail in section 5).

class (cf Yamamoto, 2002, p. 5). Understanding the data obtained from analysis will help teachers and course developers to adapt the feedback their particular class needs.[41]

The data generated by the application does not need to change to suit a different environment. All that has to be adapted is how the feedback system is used and the feedback presented. This is a primary reason for using a modular approach to application design. To allow flexibility in terms of the type and nature of feedback and enable the application to serve as a plug-in component in other CAI applications the application supplies feedback in the form of raw data. The particular CAI applications which make use of this application would then interpret the data in a form accessible to learners. In this way purpose-specific feedback can be developed that will suit whatever environment this tool is used in.

The model adopted then involves a core application which provides output that describes the relationship network of an input text. Additional tools provide data from various comparison, merge and difference operations on pairs of these networks that highlight particular types of difference in these networks (see section 5.4 on page 168 for a description and examples of these operations). These differences highlight particular facets of a learner's text (accuracy, coherence or cohesion) and can thus be used to generate appropriate learner feedback.

3.1.2 Competence vs accuracy

In an L2 classroom context this application must provide data on two aspects of learner output. The first is syntactic - the degree to which learner output is grammatically accurate. The second is semantic - the degree to which learner output succeeds in conveying the learner's intended message.

There is also an inherent conflict of interest in these two aspects. Grammatical inaccuracy does not always result in lack of communication. An important consideration in this application, then, has to be the degree to which the application is capable of building semantic networks from grammatically inaccurate input. As has been observed in NLP application designs

Human language understanding involves interaction of several channels of information: syntactic, semantic, pragmatic and psychological. This renders syntactic structures extremely flexible and noise tolerant. Therefore, a rigorous

[41]As the primary intent of this application is to provide a tool that can be used to develop software and as a type of feedback that is appropriate will vary from application to application, we have only provided examples of the sort of feedback that can be generated from this application.

> *model of language competence are* (sic) *often powerless when facing performance data (Sun, Togneri, and Deng, 2003, p 8).*

Educationally, this is a particularly significant problem with beginning L2 students as these will have a higher proportion of inaccurate utterances than more advanced students. Computationally it is significant that allowing for grammatical inaccuracy means developing a system that is not bound by grammatical cues. Allowing grammatical inaccuracies increases the application's dependence on its lexical database. Meanings increasingly have to be defined in the lexicon and cannot as easily be resolved based on context.

In practical terms this means that a balance must be found between allowing some grammatical inaccuracies and using context to define meaning. This becomes a problem in identifying which errors to allow and which not. The grammatical formulation adopted here allows a measure of scalability in this regard. The grammar defines basic relationships between words and uses these to develop structures which describe those relationships. Much of grammar is used to refine the relationships defined by these basic structures. For example, grammatical constructs such as concord and tense provide extra clues to meaning and add precision to meaning. The basic relationships between the words, however, remain the same. Consider these simple sentences:

1. *The boy likes the dog.*
2. *The boy liked the dog.*
3. *The boy will like the dog.*
4. *The boys like the dog.*

The same basic relationship exists between the concepts "boy" and "dog" (in this case connected by "like"). Grammar supplies extra cues that influence meaning to a greater or lesser extent. For example, in (4) the concord helps indicate that more than one boy is involved and in (1), (2) and (3) the tense indicates a different time for the relationship, with potential further refinements of meaning depending on the context and stress (the meanings of these, for example, can be changed by varying the pitch and/or accent in the sentence).

The model adopted here is fairly primitive in this respect. In the grammar formulation used here these four sentences would be represented by very similar structures in the relationship network, reflecting the similarity in the fundamental meaning of the sentences.

This allows comparison between these sentences at that level showing that all four sentences have a basic similarity in meaning. A benefit of this is that minor errors in concord or tense will be ignored in generating the relationship network - reflecting the probability that these errors will not influence the understandability of the underlying message.

This is, in turn, a shortcoming if one wants to test the grammatical accuracy of each sentence in the text as the grammar does not, at the moment, make provision for the differences of meaning conveyed by these constructs.

In a CALL context this means that the results generated by the application will highlight gross errors in communication rather than minor inaccuracies in grammatical usage (minor in the sense that they do not interfere with the message). This is sufficient to make learners aware of ineffective communication and can be used to encourage self-correction. This is particularly valuable in that, as Lyster and Ranta conclude in their empirical study, corrective feedback can lead to learner uptake when there is "negotiation of form, the provision of corrective feedback that encourages self-repair involving accuracy and precision" and when cues are given to make students aware of the necessity of repair of ill-formed utterances (Lyster and Ranta, 1997, p. 42).

3.1.3 Implications for grammatical formulation

Accuracy in grammatical relations is subjective. There is a growing body of evidence showing that grammaticality is a gradient notion, rather than a categorical one (Keller and Asudeh, 2002, p. 4). As Levelt concludes "... it is an illusion to think that an objective absolute judgement of grammaticality is possible" (Levelt, 1974, p. 17). Most of the studies and theories of grammatical relations examine text at a sentence level. At this level grammatical relations are relatively easy to distinguish and are fairly fixed. It is possible to develop guidelines that consistently predict the acceptability of individual sentence constructions. Texts longer than an individual sentence are harder to deal with. The considerations in developing a grammatical formulation that describes multi-sentence texts is necessarily different to a formulation that focuses on sentence level grammar simply because there is an interaction between the sentences in multi-sentence text.

Sentences which do not appear grammatically accurate at face value can be grammatically accurate when taken in the context of a particular communicative event. This is especially evident in conversation. Here parties often use a 'verbal shorthand' in which parties do not need to state contexts if this is understood in terms of the conversation as a whole (either because the context has been supplied by an earlier turn-at-talk or because it is implied by the environment/situation of the conversation.)

This application is intended to analyse a text as a whole, rather than as a set of discrete sentences. In doing this we attempt to model a system that is similar in behaviour to that of a human. There are some fundamental hypotheses about the way in which people interpret texts which become cornerstones of the approach to analysis adopted in this engine.

The first and most important of these hypotheses is that people are predisposed to find meanings. People actively look for meaning in texts even when they recognise those texts as being ungrammatical. This is a fundamental assumption in that it decides the approach adopted for handling ungrammatical texts. If people look for meaning in texts that are ungrammatical then it follows that the interpreted meaning (the meaning as understood by a listener/reader) of a text is more important for effective communication than the grammatical accuracy of the text. If an application attempts to assist in students learning communicative ability then the focus of the application has to be on the students' interpreted meaning and how this relates to the intended meaning.

This means that there is an implied assumption inherent in any text used with this application. This is simply the assumption that the texts analysed here are intended to convey meaning. This is the same assumption any person makes when listening or reading. In attempting to give an evaluation of a learner's communicative ability, it follows that the software should also try to find meaning in the text, even when the text is not grammatically accurate. Grammatical errors thus only become significant when they interfere with the transfer of information taking place in a text.

These assumptions put boundaries on the level to which the application can test for grammatical accuracy. If it is weighted toward searching for grammatical accuracy at a sentence level then utterances which are acceptable in a multi-sentence context will be regarded as invalid. If, on the other hand, the grammar is made flexible enough (which can in this case mean vague enough) then the application will search for meaning in unacceptable responses - which is good - but will also allow many grammatically inaccurate utterances which are not acceptable even in an extended context. Finding a balance between these two poles will be affected by the CAI aims of the programme, which will, in turn, be influenced by the limitations the resulting compromise puts on the use of the programme.

3.1.4 Implications for lexicon construction

Contextual analysis operates at two levels. There is an evaluation of meaning and of expression, where meaning is quantifiable and expression a subjective evaluation of the effectiveness of delivery (style, conciseness, efficiency, etc). Seeing as it will be difficult to develop a measurable definition of expression, this dissertation is devoted to a study of the analysis of meaning in texts.

This requires an understanding of what makes up meaning. Meaning can be broadly split into two aspects - concrete and abstract. 'Concrete' reflects aspects of meaning that relate to the physical world around us. 'Abstract' covers basically everything else - aspects that are often defined by relating them to aspects of the physical world that in some way resemble the concept.

This categorisation is not absolute - different perspectives mean that this split will be unique for each individual (consider the extreme of a blind person who can only regard as abstract all the visual aspects that most people would categorise as concrete). A logical extension of this perspective would be to suppose a perspective in which all meaning was abstract and hence not related to anything concrete - as in a being who had no experience of the physical world. Our computer is such a being.

One practical example of this approach is provided by dictionaries. Here all meaning is defined in terms of the words contained in the dictionary itself. Even where reference is made to aspects of the physical world, these aspects are still defined in the dictionary in terms of words contained in the dictionary. Inevitably the definitions of meanings have to be recursive in nature.

In this view meaning cannot be an absolute because it has no concrete base. Instead it consists of relationships between definitions in the dictionary.

Sentence construction is a similar exercise in that this too defines relationships between the words in the sentences.

In this lies a manner of comparing texts contextually. This is achieved not by comparing the individual words - the choices of words will differ between individuals and the meaning intended by those words will sometimes also differ. Instead this is achieved by comparing the relationships between nodes as defined in the sentences.[42] These relationships contain the meaning of the text, and by determining similarities in the sets of relationships defined in the two texts we can construct a measure of the similarity of texts.

[42]Each node is a word or concept defined in by the relationships that encapsulate the meaning of the concept. This method of representation is described in more detail in section 4.2.2.

3.1.5 Design limitations

There is one unavoidable problem in any system that attempts to examine the semantic content of a text. The patterns that make up grammar can only identify relationships made explicitly between concepts in a text. This is a limiting factor in computerised interpretation as the application does not have the frame of reference to draw on that people do.

People, by contrast, use the given text as a starting point and activate external references, knowledge and experience to give particular meaning to the relationships defined in the text. For example in a sentence like

Tom smiled.

there is an explicit relationship defined between the object 'Tom' and the action 'smiled'. This is the relationship defined by grammar in the sentence. A human reader activates other knowledge and experience which means this is not interpreted solely as Tom performing the action of smiling. In one context, smiling implies that Tom is happy, or that he has heard or seen something funny. In another context he may simply be being friendly, consoling. In this second context the act of smiling also becomes an act of non-verbal communication - an action that is not explicit in the structure of the sentence. The first context could, potentially, by allowed in the application by defining 'smile' with links to 'happy' and 'funny', but then we have exhausted the range of adaptability of the application. The exact meaning of that smile is dependant on the context of the act - information that will only be available to the computer if it is explicitly provided. The second requires an evaluation of the social context of the event - this is a complex interpretation as it requires activating a broad range of social knowledge.

In any communication, there is thus an implicit environmental context which provides a target for references and makes the text simply a starting point for assigning meaning to otherwise unquantifiable concepts. This means that semantic content inherently depends on relationships defined between the symbols (words) that make up the language and concepts, actions or experiences outside the text. Many relationships are implied in texts - relying on shared experience or knowledge for the accurate transfer of information. Without this essential reference, the application will have no inherent capability to distinguish between two symbols.

To consider one extreme of this problem, the computer makes no inherent distinction between words like '*the*' and a symbol like '*cat*'. The only distinction it has is the definition in its lexicon of one as a determiner and the other as a common noun. With no means of defining external relationships, the only other data available to the computer is frequency of occurrence, and if this is used as a measure of the importance of a symbol, the application will have to conclude that the symbol '*the*' is one of the most important symbols in the English lexicon.

Only by adding artificial constraints to the grammar will the application be able to distinguish '*cat*' as the primary object in a sentence and not the determiner '*the*'. Similarly, unless there is an artificial and explicit relationship defined for the words '*cat*' and '*cats*' the application will regard these as two distinct symbols. If sufficient lexical information is given to determine that these both represent objects (nouns), the application will still need to have additional explicit relationships defined to 'know' that these both refer to a '*cat*' object in the physical world (and even if it has these defined, along with pictures, sounds, video footage, 3D models or any other related data, it will still not have an internal linkage that does relate these words to an object in the physical world).

3.2 Grammar design issues

In the broad overview of current directions in grammatical theory provided in section 2.2 it was shown that no single theory of grammar completely satisfied the theoretical requirements for an implementable grammar fulfilling all the requirements of this application. The most viable grammars for the purposes of this application appeared to be the dependency-based grammars, with Word-grammar coming closest to fulfilling the requirements of this application (cf. *Dependency-based grammar*, section 2.2.4 on page 40 and *Coherence*, section 2.5.3 on page 60).

It became necessary to design a grammar capable of generating a semantic network that gives a reasonably accurate representation of the content of a paragraph or essay. Many ideas were borrowed from Word-grammar in constructing this grammar (cf. section 2.2.4 on page 40). One early departure from the goals of most grammars (cf. Ellis, 1995, p 85 and Keller and Asudeh, 2002, p. 4) is that what this grammar does is only partially concerned with the grammaticality of input (cf. section 2.5.1 on page 57). No real judgement is made of the degree to which a text can be considered accurate or grammatically correct except to the extent that any errors result in an inability to determine semantic and syntactic relationships (cf. section 4.2.6 on page 131). The only place that these have in the application is when the errors result in unfilled dependencies in the sentence analysis or when words cannot be included in any structure describing the sense generated from these relationships (that is, when the words cannot be accommodated as part of any relationship defined by the analysed sentence structure).

The focus of this application is providing an automated means of comparing texts contextually (cf. section 2.5.1 on page 57). Consequently, the representation of language in the application is primarily designed with a semantic breakdown of the texts in mind, rather than a syntactic breakdown. In this context, a syntactic analysis is only useful in as far as it helps determine semantic content in the text. This also necessitates some modification of WG structure (cf. sections 2.2.4 on page 40 and section 4 on page 107) as this grammar is not in itself a model of either production or perception. Instead it primarily provides a network of knowledge which the processor can exploit (Hudson, 1998, p. 15).

In practice, this resulted in a dependency system that operates at two levels. The first is syntactic (cf. sections 4.1 on page 107 and 4.2.2 on page 109). Each word is assumed to contain a set of word types that can occur in its proximity. For the most part, this results in a pattern of dependency relationships between a word and its neighbours. As most words fit into a set of common characteristics, most of these dependencies can be defined in terms of classes that contain those characteristics (largely synonymous with the concept of parts-of-speech). Some words, however, may have additional dependencies that are unique to those words. These will need to be represented in the unique lexical structure of those words.

Dependencies are not only used to define the order evident in syntax, but also semantic relationships among words in a sentence (cf. sections 4.2.3 on page 116 and 4.2.4 on page 124). In other words, each dependency has a dual purpose. First, the dependency provides predictive information about the classes of the words in close proximity to the dependency. This aspect is implicit in the nature of any dependency formulation. Simply put, the fact that a word requires another word with particular characteristics to satisfy a dependency is a strong indication that a word with those characteristics exists in the text. So, for example, a determiner such as *the* is a good predictor that the sentence will contain a word with the characteristics associated with common nouns.

When considered together with the dependencies of the other words in a sentence this becomes a description of the syntactic structure of a sentence. Secondly, the dependency provides an indication of how words in a sentence are related to each other in terms of the meaning of that sentence. If a dependency is unfilled this second use can provide an indication of how the words/relationships in a sentence relate to other sentences in a text. In this way the semantic dependencies become indicators of the meaning derived in the second system. This is similar to the Semantic Dependency Net developed by Li et al. (2003, p. 83) to try and describe deeper semantic dependency between individual words, representing the meaning and structure of a sentence by these dependencies.

> *The second system is semantic - a representation of the meaning of sentences. This is a limiting structure representing the relationships between the objects in a text. Verbs and prepositions are used as the vectors forming these links and are, because of this, the determining elements of meaning in a text* (Li et al., 2003, pp. 85-86).

These two systems are analogous to describing the surface structure and deep structure of text, though deep structure in this sense in not the one used in generative grammars of "single formal object serving as input to the transformational rules" (Newmeyer, 1986, p. 74, cf. also Radford, 1988, pp. 401-420). The generative use of deep structures refers to an underlying *grammatical* structure of a sentence into which lexical items are inserted (Radford, 1988, p. 419). This is an artificial construct not suitable for determining meaning (cf. section 2.2.1 on page 28). As Fillmore states:

> *[deep structure] is an artificial intermediate level between the empirically discoverable "semantic deep structure" and the observationally accessible surface structure, a level the properties of which have more to do with the methodological commitments of grammarians than with the nature of human languages* (Fillmore, 1968, pp. 88).

The definition of deep structure is based on the assumption that there is an underlying grammatical structure to sentences. A corollary of this is that the entire meaning of a sentence is fully known to a speaker before he or she begins a sentence. This has to be the case in any system that uses movement rules to describe the patterns observable in text. If we examine our own behaviour when speaking, it is clear that this is not always the case. Very often we start sentences with no clear perception of the ideas we want to present. Instead these are developed as we speak. In this sense, the content of texts (especially texts generated in conversation) are chaotic. The relationships arrived at in the text are generated in an *ad hoc* manner with little prior planning. It is impossible for a deep or base structure of a sentence to have been constructed if the concepts contained in the sentence have not been fully determined, let alone for movement rules to be applied.

The view of deep-structure in this application is rather that deep-structure is synonymous to the set of relationships which contain the inherent meaning of a text. These relationships do not have a grammatical component as they may only include one-to-one links between objects.

To prevent confusion, I will hereafter refer to this as the link or relationship structure of text - meaning the set of relationships described in text. The link structures developed in the sample texts (described in more detail in section 5.3 on page 157) are not absolute - sentences involving multiple linkages could have a number of possible link structures depending on the perceived importance of the various links in the text. This non-absolute definition of link structures is also a viable explanation for the ambiguity of some texts and also why some seemingly ambiguous texts are not interpreted ambiguously by people engaged in conversation.

As a final aside, this is also a workable explanation of why many ungrammatical texts are still easily understandable - there is no confusion about the relationships that exist in the texts even if the method used to describe the relationships does not match our grammatical norm. Misinterpretation will only occur when it is no longer possible to correctly identify the relationships contained in text, an occurrence that will often be perceived as errors of grammatical usage.[43] This is one of the attractive features of WG as:

> *... WG accommodates deviant input because the link between tokens and types is the rather liberal 'Best Fit Principle' (Hudson, 1990, p. 45): assume that the current token* isa *the type that provides the best fit with everything that is known. ... There is no need for the analysis to crash because of an error* (Hudson, 1998, p. 15).

From this, it should be clear that, in this definition, grammatical relations are a convention used to convey these relationships to a second party. The process by which these relationships are represented is what is described by grammar and is a fairly close match for surface-structure. It is worth noting that this describes gross linguistic acts at the illocutionary level. Secondary speech acts (such as implication or connotation) are not described by this process at present.

Thus, the role of grammar in this model is facilitatory. The concepts we use in conversation are arrived at without any planning even though the relationships that these concepts have with each other are clear to the speaker. Grammar, at least as used in this application,

[43]Ungrammatical texts are very common yet people show a remarkable facility in understanding these texts and communicating accurately regardless of the lack of grammaticality. It seems logical then that any theory of grammar that does not include a process accounting for the ability of people to comprehend ungrammatical texts has serious flaws.

has two closely related purposes. It enables the speaker to convey these relationships in an unambiguous manner regardless of the order in which he thinks of them. Secondly, it provides a convention by which listeners can deconstruct sentences into the relationships that these represent. Grammar in this application is thus a set of meaning-seeking conventions rather than a set of well-formedness rules.

Interpreting sentences is the reverse of this. People cannot wait for an entire sentence to be spoken. Rather, they evaluate the units of text as they become aware of them.[44] Relationships are usually close together. The success of this model of interpretation and generation is to a degree determined by the simplicity of the structures. Simpler and fewer structures mean that a parser has fewer options to evaluate in order to develop a link-structure describing the relationships contained in a text.

3.2.1 Rules as generalisations

Unification-based grammars, which have most of their grammatical rules defined in the lexicon, offer a very reasonable explanation of the way human language learning and use may take place. However, they do not explain all human grammatical abilities. Consider the following sentence

"A beptly ampting snabe gam to fey."

If we know the sentence is English, even though (with two exceptions) the words in this sentence do not occur in any lexicon, it is possible to guess the role each of these words plays in the sentence. Our internal logic picks up small clues contained in sentences and in the words that make up the sentence and assigns roles to the words in the sentence based on grammatical expectations. The two words that are recognisable, "A" (a determiner) and "to" (a preposition) provide some indication of what might follow. Other clues lie in the suffixes "ly" and "ing". Using these clues we arrive at the probable interpretation shown in Figure 7.

As a more general example, consider the difference between the next two examples based on the above sentence:

[44]Experimental evidence (e.g. Marslen-Wilson (1973, pp. 522-523)) has shown that human language processing is highly incremental, meaning that humans construct a word-by-word partial representation of an utterance as they hear each word.

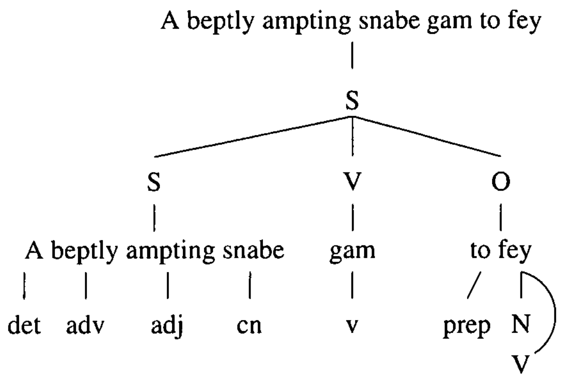


Figure 7:

"a [word]ly [word]ing [word] [word] to [word]"

and

"a [word] [word] [word] [word] to [word]."

The first has the same grammatical information as is contained in Figure 7, and so the same analysis is possible, as is shown in Figure 8.

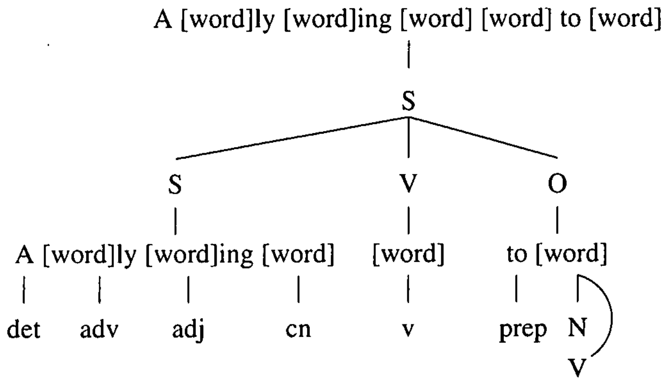

A [word]ly [word]ing [word] [word] to [word]
S
S
V
O
A [word]ly [word]ing [word]
[word]
to [word]
det
adv
adj
cn
v
prep
N
V

Figure 8:

A small change - removing the *"ly"* and *"ing"* suffixes - and this is no longer possible. Instead a number of distinct structures are possible, a few of which are listed in Figure 9.

```
A [word] [word] [word] [word] to [word]
|
├──► det [cn] [adv] [adv] [v] prep [N]
├──► det [adj] [cn] [v] [adv] prep [N]
├──► det [cn] [adv] [v] [adv] prep [N]
├──► det [cn] [adv] [v] [adv] prep [N]
└──► det [adj] [adj] [cn] [v] prep [N]
```

Figure 9:

Without additional information there is no way to distinguish among these.

The danger exists of too rigid a reliance on rules. Rules of any type are good indicators for structural analysis but that is all they are. Consider the examples above again. These rely on using the suffixes *"ly"* and "*ing*" as indicators for adverbs and adjectives, and in most cases this would be accurate. However, *"ly"* is not the exclusive property of adverbs. Some adverbs do not end in *"ly"* , a number of words do end in *"ly"* that are not adverbs (for example "*comely")* and the "*ly*" ending is not always a suffix (as in *"rely", "reply"* and *"fly").*[45] As with all the rules used here, it must be borne in mind that these are only indicators or guidelines, and any number of grammatical scenarios might develop that seem at first sight to flaunt one or other of these rules.

Rather than determine the part-of-speech of a word, indicators help confirm a previous approximation. Take, for example, the *"ing*" suffix. A rule that specifies that *"ing*" suffixes define continuous tense verbs is an over-generalisation that will lead to errors in analysis as it ignores the gerundive use of verbs. However, it is also true that no continuous tense verbs exist that do not have an *"ing*" suffix. This means that, while the programme cannot use the suffix as its primary method of analysis, it can use this suffix to confirm a reasonable approximation. Similarly it can eliminate the option of being a continuous tense verb from any word that does not have this suffix.

Even without a detailed lexicon this information is enough to make a good first approximation of the structure of many sentences.

[45]This is significant in that there is no way for an application to determine when this combination forms a suffix and when it does not unless it has a root form for a particular word in its lexicon.

3.2.2 Adjacent dependencies

A simple parser was constructed to demonstrate this principle.[46] This included a very small lexicon of approximately thirty words. These included determiners, conjunctions, some prepositions and variants of the verb "*to be*". Part of the aim behind this trial parser was to determine the minimum level of detail required in the lexicon to enable successful parsing. Manually constructing a lexicon is a laborious process so the aim was to avoid unnecessary effort in constructing the lexicon.

The simplest case of dependency (using only dependents of words immediately adjacent to each other) closely resembles a form of a bigram word model. In these models the probability of each next word in a sequence is determined from the current one - in other words from the word immediately adjacent to any particular word. As Christer Samuelsson (1998, p. 84) states, this results in "simplistic models of interword dependencies".

There are two primary differences between the model implemented here and a classic bigram model. The first is that the model implemented here included a number of pre-determined patterns in an attempt to improve the accuracy of the analysis. The second is that, in statistical language learning, the intent of a bigram model is to try and determine the probabilities of one word following another while the model here attempts to determine which class of word follows which in a sequence of words. By definition a bigram model requires substantial training data before any reasonable dependencies can be determined.

In a bigram model, these would be represented as dependencies between individual words - resulting in a large lexicon. A small corpus of sentences was constructed based on sentences generated by L2 students at the University of the Free State. In an attempt to avoid a large lexicon, patterns were identified in the order of words in this corpus that suggested assumptions about the type of words that could occur before and after an arbitrary word in a sentence.

These assumptions were organised into a list that only included a possible following type - dependencies worked from left to right exclusively. This does not include any phrasal rules, though phrase structures are implicit in a sequence of filled dependencies.[47]

- determiner $det \mapsto \{cn, adj, adv\}$

[46]The source for this version can be found in the directory `/trials/v1` on the companion CD.

[47]

A dependency analysis defines 'phrases' as by-products (rather than as the basic categories of the analysis) Hudson (1994, p. 4991).

- preposition $prep \mapsto \{det, adj, cn, pron, pn, conj\}$
- conjunction $conj \mapsto \{det, adv, adj, cn, pron, pn, conj, v\}$
- proper noun $pn \mapsto \{adv, v, conj\}$
- common noun $cn \mapsto \{adv, v, conj\}$
- pronoun $pron \mapsto \{adv, v, conj\, pn, cn\}$
- adjective $adj \mapsto \{adj, cn, pron, pn, conj\}$
- adverb $adv \mapsto \{adv, adj, v, conj\}$
- verb $v \mapsto \{det, adv, prep, pron, pn, conj\}$

These were the only parts-of-speech and the only dependencies defined in this version.

The reasoning behind using only immediate adjacency in the definition worked on the principle that the adjacency rules for neighbouring phrases would reinforce each other. Because of this, it would not be necessary to define rules which explicitly describe phrase content for each possible continuation. For example, determiners could have been defined as

$$det \mapsto \left\{ \begin{array}{c} \{cn\} \\ \{adj, cn\} \\ \{adv, adj, cn\} \end{array} \right\}$$

These definitions, however, are implicit in the adjacency structures. Defining these continuations for every case would require additional recursive processing and additional rules to allow parsing of sentences with (in this case) extended sequences of adverbs, adjectives or nouns. Instead the simpler basic structure used doesn't only allow sequences containing one of each part-of-speech but also allows extended sequences in phrases without any need for extra rules to govern this sort of extended structure, such as that in

The	really	incredibly	small	brown	dog	...
`det`	`adv`	`adv`	`adj`	`adj`	`cn`	

This list served two purposes. Its one use was to confirm sub-phrase sequencing determined from the first list. The second was to determine if the phrase sequences themselves were legitimate.

The third and last component of this model was a list of common suffixes. These were used to provide additional hints as to the type of a particular word.

This is sufficient for a surprising degree of accuracy in textual analysis, as can be seen in the examples shown below:[48]

The grey dog sat quietly on the mat today

$$det \rightarrow adj \rightarrow \left[\begin{array}{l} cn \rightarrow v \rightarrow adv \\ adj \rightarrow \left[\begin{array}{l} cn \rightarrow v \\ adj \rightarrow cn \end{array} \right. \end{array} \right. \rightarrow prep \rightarrow det \left[\begin{array}{l} cn \\ adj \end{array} \right. \rightarrow adv$$

Sit

$$\left[\begin{array}{l} adv \\ v \\ pron \end{array} \right.$$

The boy is named John

$$det \rightarrow adj \rightarrow v \rightarrow \left[\begin{array}{l} cn \rightarrow adv \\ v \rightarrow \left[\begin{array}{l} adv \\ cn \end{array} \right. \\ adj \rightarrow cn \end{array} \right.$$

The dog ran very slowly

$$det \rightarrow adj \rightarrow \left[\begin{array}{l} cn \rightarrow v \rightarrow cn \\ v \rightarrow \left[\begin{array}{l} adj \rightarrow cn \\ cn \rightarrow adv \end{array} \right. \\ adj \rightarrow cn \rightarrow cn \end{array} \right.$$

[48]The Perl source-code for this version is available on request.

Modifications to this version were developed that increased the number of parts-of-speech (for example, adding tenses and a special case for the verb 'to be'). This enabled the parser to more accurately capture the aspects of sentence structure to which these changes apply, but has the penalty that, without additional rules, it also increases ambiguity in parsing. The dependency structures for the different tenses, for example, are almost identical, resulting in an inability to distinguish between these without some additional cue. Irregular forms of tenses make it impossible to apply a blanket generalisation (such as an expected 'ed' suffix in past tense forms) to resolve this ambiguity.

An important consideration in interpreting these examples is that the aim of this analysis is to identify link relationships in a sentence. The sparsity of dependency data that this model uses in arriving at this analysis inevitably leads to a number of potential structure trees. These trees are then reduced by merging similarities in the parse trees. This gives a structure from which is possible to derive nodal relationships.

A particular weakness in this version is that it relies on there being recognisable cues in the input. Many sentences are constructed without any prepositions, determiners or words with recognisable suffixes. In these cases the only clues become the start and end points of the sentence. Consequently the results of the analysis of these sentences have a very low level of accuracy which manifests itself in an analysis tree with too many branches. This in turn results in an analysis with multiple verb possibilities which creates ambiguities in determining nodal relationships. As a result, objects are not defined clearly enough to enable reasonable comparison of the resultant relationship networks.

This problem of multiple verb possibilities also becomes particularly apparent when sentences are analysed which do not follow the pattern of the simple sentences examined above. Questions, for example, are often characterised by a change in the normal sequence of words. For example a simple sentence such as:

"You will go to town."

can be converted into a question by moving 'will' to give

"Will you go to town?"

Handling this change means that the process of analysis will have to allow for both possibilities. With no lexical definitions for the words involved in the movement this means that the analysis will not be able to distinguish between these two sentence structures.

In this case, since the programme structure did try to allow for this particular variation, the programme returns the analysis tree

$$\left[\begin{array}{c} adv \dashrightarrow \left[\begin{array}{l} v \dashrightarrow adv \\ adv \dashrightarrow v \end{array}\right. \\ v \dashrightarrow adv \dashrightarrow adv \\ pron \dashrightarrow \left[\begin{array}{l} v \dashrightarrow adv \\ adv \dashrightarrow v \end{array}\right. \end{array}\right. \dashrightarrow prep \dashrightarrow \left[\begin{array}{c} adj \\ cn \\ pron \end{array}\right.$$

for both sentences.

The dependencies used here are bound to the basic structure of the sentences they are based on. Other structures can only be supported by adding dependencies to the existing definitions, which inevitably increases the number of potential sentence structures for a given text. Each additional dependency for a particular type increases the potential for ambiguities in the analysis.

3.2.3 Sub-phrase structure dependencies

As this project moved closer to developing a lexicalised version, a second trial parser was developed that modelled something resembling the structure of lexical entries.[49] This entailed a considerable expansion of parts of speech similar in scope to the types covered in the lexicalised version (see section 3.4 for a discussion of the final grammatical formulation and lexicon structure). Additions to the first trial version included a better formulation of phrase heads. It also included a set of dependencies to better describe sequencing within a phrase. This information was contained in three separate dependency lists which were later merged to form the sequencing dependencies of the lexicalised versions.

The first of these is a list of possible phrase and sub-phrase initialisation types:

- determiner $det \succ \{det, advadj, cn, pron, pn\}$
- conjunction $conj \succ \{cn, vcpron, pn, det, prep, adv\}$
- proper noun $pn \succ \{pn\}$
- common noun $cn \succ \{cn\}$
- pronoun $pron \succ \{pron\}$

[49]This version can be found in the directory `/trials/v3` on the companion CD.

- adjective $adj \succ \{adj, adv\,cn, pron, pn\}$
- adverb $adv \succ \{adv, v, vp, vc, i, ip, pp, h, hp\}$
- simple tense verb $v \succ \{adv\}$
- continuous tense verb $vc \succ \{adv\}$
- past tense verb $vp \succ \{adv\}$
- future tense auxiliary $f \succ \{adv, v, i, h, pp\}$
- present perfect auxiliary $h \succ \{adv, vc, pp, ipp\}$
- past perfect auxiliary $hp \succ \{adv, vc, pp, ipp\}$
- verb "to be" $i \succ \{adv, pp, vc\}$
- past tense verb "to be" $ip \succ \{adv, pp, vc\}$

This is also similar to head structure in HPSG, though with less importance attached to the semantic importance of the phrase head. Depending on the implementation, HPSG parsers define the sentence head as the verb or subject of the sentence. All possible sentence trees are then computed based on this head. Instead the head for phrase structure here referred to any legitimate starting word for a traditional noun, verb, adjectival or adverbial phrase. This better matches the WG approach where, for example, determiners are used as the head for noun phrases (Hudson, 1998, p. 15).

The second rule is a list of possible phrase end types for a particular initialisation:

- determiner $det \prec \{cn\}$
- preposition $prep \prec \{cn, pron, pn\}$
- proper noun $pn \prec \{pn\}$
- common noun $cn \prec \{cn\}$
- pronoun $pron \prec \{pron\}$
- adjective $adj \prec \{cn, pron, pn\}$
- adverb $adv \prec \{v, vc, pp, vp, adv, cn, pron, pn\}$

- simple tense verb $v \prec \{v, adv\}$
- continuous tense verb $vc \prec \{vc, adv\}$
- past tense verb $vp \prec \{vp, adv\}$
- past participle $pp \prec \{pp, adv\}$
- future tense auxiliary $f \prec \{v, vc, pp, adv\}$
- present perfect auxiliary $h \prec \{v, vc, pp, adv\}$
- past perfect auxiliary $hp \prec \{v, vc, pp, adv\}$
- verb "to be" $i \prec \{v, vc, pp, adv\}$
- past tense verb "to be" $ip \prec \{v, vc, pp, adv\}$

A noun phrase, for example, cannot end with an adjective. An analysis that included this possibility would be an incorrect analysis. Instead any phrase beginning with an adjective has to end with a noun, be it proper or common.

Lastly, the third list was of required part-of-speech types for a particular phrase initialisation (phrase dependencies that have to be filled but do not need to be adjacent):

- determiner $det \prec \{cn\}$
- preposition $prep \prec \{cn, pron, pn\}$
- proper noun $pn \prec \{pn\}$
- common noun $cn \prec \{cn\}$
- pronoun $pron \prec \{pron\}$
- adjective $adj \prec \{cn, pron, pn\}$
- adverb $adv \prec \{adv\}$
- simple tense verb $v \prec \{v\}$
- continuous tense verb $vc \prec \{vc\}$
- past tense verb $vp \prec \{vp\}$

- past participle $pp \prec \{pp\}$
- future tense auxiliary $f \prec \{v, vc, pp\}$
- present perfect auxiliary $h \prec \{v, vc, pp\}$
- past perfect auxiliary $hp \prec \{v, vc, pp\}$
- verb "to be" $i \prec \{vc, pp\}$
- past tense verb "to be" $ip \prec \{vc, pp\}$

This often coincides with a valid phrase terminator, but not always. For example, a verb phrase has to contain a word of type verb, even though it may begin with an adverb and end with an adverb.

While this might appear a more rigorous set of rules than the set defined for section 3.2.2, in reality this is not the case. The phrase end list, for example, does not add any material benefit in terms of accuracy. (Verbs and adverbs, for example, are not defined as legitimate continuations following an adjective in the dependency structures outlined in section 3.2.2. This means that the phrase end rule made explicit by requiring a noun ending for an adjective initiation is implicitly defined section 3.2.2.) The reason for its inclusion in this version, however, is that it is a means of reducing the number of passes across a sentence required to generate an analysis. With no lexicon on which to base analysis this became a very important consideration as the number of possible combinations easily enters the thousands.

The two significant additions in the second version are the more precise definitions of phrase types and extension of the dependency definitions to allow a greater range of input sentence structures. As is shown in the example that follows, this results in increased ambiguity as there are now more potential trees (each potential sentence structure identified by the programme is shown complete as the number of possibilities becomes too many to represent conveniently in a tree structure):

The grey dog sat quietly on the mat today

1. $det \dashrightarrow adv \dashrightarrow adv \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow cn \dashrightarrow vp$
2. $det \dashrightarrow adj \dashrightarrow adj \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow cn \dashrightarrow v$
3. $det \dashrightarrow adj \dashrightarrow adj \dashrightarrow cn \dashrightarrow vp \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$

4. $det \dashrightarrow adj \dashrightarrow adj \dashrightarrow cn \dashrightarrow v \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
5. $det \dashrightarrow adj \dashrightarrow cn \dashrightarrow vp \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
6. $det \dashrightarrow adj \dashrightarrow cn \dashrightarrow v \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
7. $det \dashrightarrow adj \dashrightarrow cn \dashrightarrow f \dashrightarrow v \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
8. $det \dashrightarrow adj \dashrightarrow cn \dashrightarrow f \dashrightarrow vp \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
9. $det \dashrightarrow adj \dashrightarrow cn \dashrightarrow adv \dashrightarrow v \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
10. $det \dashrightarrow adj \dashrightarrow adv \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow cn \dashrightarrow vp$
11. $det \dashrightarrow adj \dashrightarrow adv \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow cn \dashrightarrow v$
12. $det \dashrightarrow cn \dashrightarrow vp \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
13. $det \dashrightarrow cn \dashrightarrow v \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
14. $det \dashrightarrow cn \dashrightarrow v \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
15. $det \dashrightarrow cn \dashrightarrow f \dashrightarrow adv \dashrightarrow v \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
16. $det \dashrightarrow cn \dashrightarrow adv \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow vp$
17. $det \dashrightarrow cn \dashrightarrow adv \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow vp$
18. $det \dashrightarrow cn \dashrightarrow adv \dashrightarrow vp \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
19. $det \dashrightarrow cn \dashrightarrow adv \dashrightarrow v \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
20. $det \dashrightarrow cn \dashrightarrow adv \dashrightarrow adv \dashrightarrow vp \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
21. $det \dashrightarrow cn \dashrightarrow adv \dashrightarrow adv \dashrightarrow v \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
22. $det \dashrightarrow adv \dashrightarrow adj \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow vp$
23. $det \dashrightarrow adv \dashrightarrow adj \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow v$
24. $det \dashrightarrow adv \dashrightarrow adj \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow v$
25. $det \dashrightarrow adv \dashrightarrow adj \dashrightarrow cn \dashrightarrow v \dashrightarrow prep \dashrightarrow det \dashrightarrow adj \dashrightarrow cn$
26. $det \dashrightarrow adv \dashrightarrow adv \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow cn \dashrightarrow vp$
27. $det \dashrightarrow adv \dashrightarrow adv \dashrightarrow adj \dashrightarrow cn \dashrightarrow prep \dashrightarrow det \dashrightarrow cn \dashrightarrow v$

Many of these possibilities exist as different trees solely because of differences in tense. For example, the only difference in the trees in (1) and (2) is that there is no way of deciding whether the verb is past or present tense. Removing the tense differentiation reduces the number of possibilities by approximately two-thirds. This version generates 27 variations for a grammatical sentence using the clues and dependencies in its rule set. However there are only four positions for verbs in these results. This indicates that, where we to add a feature enabling this version to construct semantic linkages, it would have four possible relationship structures representing potential meaning in the text.

3.2.4 Discussion

The level of ambiguity that results from a detailed implementation of a grammar is a strong argument in favour of simplicity in the grammar formulation. Because there can be any number of phrase segments in a sentence, the only viable way to test for valid phrases is to use a recursive routine - repeatedly testing all possible combinations until combinations are found that agree with the dependencies laid down in each of the rule sets. As sentence complexity increases the computational delay caused by processing these possibilities becomes increasingly noticeable.

For example, consider a simple sentence:

The dog sat.

Using the adjacency rules (based on the only clue word in the sentence, the determiner "*the*") and no other processing (such as testing for the presence of verb-phrases) gives the following possible combinations:

1. $det \rightarrow cn \rightarrow cn$
2. $det \rightarrow cn \rightarrow adv$
3. $det \rightarrow cn \rightarrow pp$
4. $det \rightarrow cn \rightarrow vp$
5. $det \rightarrow cn \rightarrow v$
6. $det \rightarrow adv \rightarrow cn$
7. $det \rightarrow adv \rightarrow adv$
8. $det \rightarrow adv \rightarrow pp$
9. $det \rightarrow adv \rightarrow vp$
10. $det \rightarrow adv \rightarrow v$
11. $det \rightarrow adj \rightarrow cn$
12. $det \rightarrow adj \rightarrow adv$
13. $det \rightarrow adj \rightarrow pp$
14. $det \rightarrow adj \rightarrow vp$
15. $det \rightarrow adj \rightarrow v$

This makes for a total of 15 sentence combinations that match a basic three word sentence with a leading determiner. Even though the routine ultimately returns only two possible matches -

1. $det \rightarrow cn \rightarrow vp$
2. $det \rightarrow cn \rightarrow v$

the routine still has to process each of the 13 other possibilities before it can determine their invalidity. This increases enormously as soon as we process more complex sentences. In a simple sentence such as:

The grey dog sat on the mat.

which with three recognised word cues (two determiners and a preposition) has more cues than we can expect from most input, the adjacency rules generate this pattern of possibilities:

$$det \rightarrow \left\{\begin{matrix} cn \\ adv \\ adj \end{matrix}\right\} \rightarrow \left\{\begin{matrix} pp \\ cn \\ v \\ vp \\ f \\ adv \\ adj \\ vc \end{matrix}\right\} \rightarrow \left\{\begin{matrix} pp \\ cn \\ v \\ vp \\ f \\ adv \\ adj \\ vc \end{matrix}\right\} \rightarrow prep \rightarrow det \rightarrow \left\{\begin{matrix} cn \\ adv \end{matrix}\right\}$$

This gives a grand total of 384 possible sentence combinations. An increase in the number of words in the sentence will result in a logarithmic increase in the number of potential sentence combinations. This means that it is inevitable that there will be a noticeable delay in long sentences.

Increasing the complexity of the rule sets in an attempt to refine this output further can affect the processing time considerably. Adding even one type, say a type to account for abstract nouns, would cause four additional positional entries (one for every noun position) in the previous examples analysis. This would have the net result that there would be 972 possible sentence combinations for the routine to process. This reinforces the argument for simplicity even though it comes with the penalty of generalised analysis and the corresponding potential for incorrectly identified contextual links. There is thus an inevitable trade-off in the design between allowing greater flexibility in interpretation and between achieving acceptable processing speed.[50]

At this stage two strategies were used to try to reduce the number of times this recursive routine needed to run. The first was by preprocessing sentences to remove the more definite impossibilities from the tree. These include removing continuous verb options if the words lack an *"ing"* suffix and eliminating options that have definite positional dependencies if those dependencies are not met. Past participles, for example, have the dependency of needing a verbal auxiliary (*"i"*, *"ip"*, *"h"* or *"hp"*) preceding them in enough sentence forms to be able to use this to eliminate this option if it occurs without these dependencies. Removing just one "*pp*" possibility from the sample analysis results in needing only 336 passes, meaning it should arrive at the same analysis in only 88% of the previous time. Removing the "*vc*" options then reduces this to 252 iterations - 66% of the original processing time.

The second strategy was skipping the development of trees wherever this seemed possible. The tree starting

$$det \dashrightarrow adv \dashrightarrow v \dashrightarrow \ldots$$

contains an immediate impossibility - a determiner has a dependency requiring a noun continuation. This enables all possible combinations that might be developed from this structure to be eliminated (16 passes for "*v*"; add to this the iterations for "*vp*", "*vc*", and "*pp*" and this saves 64 passes).

Ultimately this model of the application relies on having a dependency structure detailed enough to determine contextual links while at the same time simple enough not to result in redundant detail in object recognition.

3.2.5 Conclusions

Inevitably, as with all generalisations, any rule-based system will fail when the complexity of language is considered. The systems described here work for a fairly simplistic sentence structure. Question forms and complex forms will result in a high error rate. Each allowance for a special case in the set of rule lists increases the uncertainty of the parse process - increasing the chance that a number of grammatical possibility trees will be built from the same input sentence and resulting in longer parses for all structures. Each extra

[50]As will be shown later, we have opted for greater flexibility as this version is essentially a 'proof of concept' rather than a viable tool.

tree will create uncertainties that these rules cannot resolve. Consequently, a rule based system will not, on its own, be able to determine conceptual units accurately over a sufficiently large set of sentences for this to be a viable type of parsing engine for developing relationship networks.

It is also inevitable that sentences will be input in which the computer cannot identify any cues. Take for example the simple sentence

"Very few people work without pay."

Apart from the start and end of sentence markers there are no cues in the sentence that give a rule based engine a place to start. There are no common suffixes and predefined words that it can recognise. As a result it has the following base analysis:

$$\left\{\begin{array}{c} cn \\ adv \\ adj \\ v \\ vp \\ f \end{array}\right\} \rightarrow \left\{\begin{array}{c} pp \\ cn \\ v \\ vp \\ f \\ adv \\ adj \end{array}\right\} \rightarrow \left\{\begin{array}{c} pp \\ cn \\ v \\ vp \\ f \\ adv \\ adj \end{array}\right\} \rightarrow \left\{\begin{array}{c} pp \\ cn \\ v \\ vp \\ f \\ adv \\ adj \end{array}\right\} \rightarrow \left\{\begin{array}{c} pp \\ cn \\ v \\ vp \\ adv \\ adj \end{array}\right\} \rightarrow \left\{\begin{array}{c} pp \\ cn \\ adv \\ v \\ vp \end{array}\right\}$$

which yields 61 740 possible sentence combinations. Even after the three phrase structure rules have been applied this leaves us with 522 possible sentence structures built using these rules.

Another way of looking at the problem is highlighted by a refinement that allowed the application to eliminate possibilities containing excess verbs from the list. This leaves us with 124 sentence possibilities containing one verb phrase each. While this may seem a considerable improvement, it still leaves too many possible objects that can be identified. As verbs are used to define relationships between objects this would lead to a proliferation of links from this one sentence - links for every possible "*cn*" to every other possible "*cn*" using any and all of the words in the list as potential verb vectors. The knowledge base would include a correct representation of the sentence but this would be buried under the weight of the inaccurate relationship networks. Not only would processing these links result in extremely slow performance but the application would be unable to compare texts as there would be too many unrelated relationships defined for the text for the comparison to have any meaning.

The only viable method of reducing this problem is to provide a larger database of cues from which to work. These cues would include many common words (entered with as complete a grammatical definition as possible) in the hope that this will provide sufficient information to analyse sentences. In other words, given a word cue it should be able to use the more common rules and cues already included in the programme to generate a detailed enough grammatical analysis of most sentences to allow for contextual links to be identified.

In an attempt to provide a fairly general set of words for this expanded lexicon I have used the set of words identified by the Longman Dictionary of Contemporary English for the definitions in the dictionary. This was chosen largely because this set of words is supposed to have non-cyclic definitions which are then used to provide the meaning definitions for the remaining 60 000 words in the dictionary.[51] Ideally the lexicon should be expanded and all additions should be defined as relationships between this base set of definitions, as has been attempted in the Longman Dictionary, though this has not been implemented in this proof-of-concept version. Words regularly encountered in student texts have been added to this list to facilitate processing student documents.

The combination of this larger lexicon and the rules defined earlier which made it possible to analyse sentences containing unrecognised words. This enables increasingly accurate grammatical analysis, though as the lexical entries ideally need extra relationship entries to represent knowledge type relationships these lexical entries will seldom be complete.

Given the original aim of identifying conceptual objects in a sentence this version of the analysis engine could be called a qualified success. The enlarged lexicon provides a high degree of accuracy, though inevitably there will be ambiguities that the programme cannot reconcile. This will happen especially when a word that it already has in its lexicon used in an entirely new manner.

Consider the word "*will.*" Originally this was entered into the lexicon as a future tense auxiliary. It also has three other distinct usages - as a personal name, a verb and the common noun name for a testament.

In a sentence like

Will the book fit on the shelf

The programme will not be able to distinguish between "*will*" as a verb as a future tense auxiliary unless it is able to recognise the other verbal cues in the sentence. If a lexicon

[51]This facility can be useful if a meaning definition is included in the lexical entries for words, though as Bran Boguraev points out (Ritchie, 1987, pp 225-256) the definitions do contain circularities.

containing these words[52] is added to the version discussed in section 3.2.3, the lexicon combined with the dependency structures is able to accurately arrive at the one valid structure for the sentence

> *"Will Will will the will to Will?"*

This sentence contains sufficient cues in the determiner and preposition to eliminate the ambiguity in this instance.[53] Similarly, the error rate in analysing the sample sentences dropped considerably - generally yielding accurate grammatical structures and with very few ambiguities at a grammatical level.

In short, this version of the programme was a qualified success in that it provided a reasonably accurate analysis of grammatical dependencies in a text when it had enough clues to work from. It cannot work without a large lexicon but it does provide a potential solution to a very important problem given that it is not feasible to include every word in the English language in the lexicon. Simple dependency systems like this one can derive type information using known words as clues. This is vital given that texts that are encountered will almost certainly contain unrecognised words. These unrecognised words may be spelling errors or simply words not contained in the lexicon but it is imperative that a functional version of this application be able to incorporate these correctly in the sentence structure and, hence, in the final relationship network derived from that structure.

3.3 The importance of context

This dissertation is largely practical in that it attempts to construct a model for comparing textual content. This is far from as simple a statement as it might at first appear. This statement includes a number of basic assumptions about the nature of texts - assumptions which will define the type of text the programme is capable of handling. The first assumption has to be that the texts have meaning. If we follow Grice's assumptions, summarised by Steven Bayne as:

> *for utterance occasion meaning:*
>
> *"U meant something by uttering x" is true iff* [if and only if]

[52]This is done in the version available in the directory `/trials/v4` on the companion CD.

[53]In this one instance the analysis is more accurate than that generated by version 4.1 of the link grammar parsing engine - a well-developed parser that relies fairly heavily on a large database of words with part-of-speech tagging.

1. *U intended, by uttering x, to induce a certain response in A.*
2. *U intended A to recognize. at least in part from the utterance of x, that U intended to produce that response*
3. *U intended the fulfilment of the intention in (2) to be at least in part A's reason for fulfilling the intention mentioned in (1) (Bayne, 2000, p. 2).*

When applied to this application this implies that a learner writes a text in the knowledge that it will be read, that its message is comprehended and that the reader will respond appropriately to text.

A second important assumption is that printed text contains the complete meaning of the sentence. This assumption patently cannot be true for all texts. Consider the request

Can you open the door?

This has two possible interpretations. The base content of the sentence can be interpreted as a request for information

"*are you physically capable of opening the door*";

however, this is very often used with the intended meaning of

"*open the door*"(Morgan, 1977).

Without considerable knowledge not only of the content of the text but also of its context, this sort of implication cannot be inferred. Information needed to determine the correct interpretation of this sort of text includes the relative status of the speaker and listener, as well as physical details of the listener (in this example, the listener may be disabled or too young to reach the door handle). Even given these details, there is still no absolute guarantee of a correct interpretation as inflection and other extra-linguistic aspects of the communication, such as body-language, that cannot be represented in the printed text also play a part in modifying or clarifying a particular interpretation.

To have a reasonable chance of successfully comparing texts then, we have to be able to assume a context. It is as yet impossible to design a knowledge base capable of handling all possible contexts. Even without the limitation of memory and processing power, this is not possible in part simply because a large part of the contexts in which we communicate involve manipulating our environment. Except for a few specialised cases, this is simply not possible for a programme.

As a result, the context assumed for this programme consists of written monologues. In particular, given the intended application of the results in the development of teaching aids for the English Second Language (ESL) classroom, this study focuses mostly on essay- and paragraph-length texts produced by ESL learners. The programme does not inherently support external references in it knowledge-base, however this knowledge-base can be 'primed' by using a training text containing the potential targets of these references. Without this training we also have to assume that in these texts the majority of references will be to elements in the texts themselves rather than to external items, events or texts.

This makes for a fairly easily determined context for the text and limits the variation in topic that the application will be exposed to. As a result, comparison is made that much simpler. Despite these artificial limitations, there is no reason why the principles used for generating these comparisons cannot be applied to larger texts as well, providing that the context of the text is clearly defined.

3.3.1 Beyond the sentence

A common problem of analysis is that most theories of grammar develop to describe the sentence level structure of text. Even with grammars that attempt to define "deep-structures" for the meaning underlying the apparent structure of produced sentences invariably talk of the "deep-structures of sentences" (Newmeyer, 1986, p. 74). This poses serious problems when trying to develop a system of analysis for texts which can include a number of sentences. Much of sentence structure is determined by the surrounding text. In many ways this principle is similar to what Kraus referred to as the cumulative nature of conversation (Greyling, 1987). Volosinov also expounds this view of communication when he describes language as "a continuous generative process implemented in the social-verbal interaction of speakers" (1973, p. 98).

In a simple statement like

"This is good."

"*This*" can take the place of a wide range of structures with very different underlying principles. Consider these pairings

Is this chocolate? This is good. and
People believe in the rule of law. This is good.

In the first, "*this*" refers to a single tangible object (the chocolate). In the second, the reference is to a considerably more complex structure - "*belief in the rule of law*". In interpreting these statements context and non-verbal communicative acts become very important. The first sentence, for example, would probably be accompanied by a gesture clarifying the context of the word "*this*". A grammar, such as generative grammar and its derivatives, that attempts to define rules governing "*this*" usage will have two vastly different deep-structures to represent. This hardly begins to touch on the problems caused when this variation of reference is compounded by different sentence structures, as in:

This chocolate is good. This is good.

Here there are two distinct usages of the word and a number of possible interpretations, partly depending on the inflection used when making the statement. For example, depending on context, the first "*this*" can select between a range of chocolates or can serve as emphasis for the word "*chocolate*". This second "*this*", for example, can refer to the chocolate, or the statement that "*this chocolate is goo*d", or in a different context it can be a negation of the first statement reading "*the chocolate isn't as good as this other stuff*".

Many words and phrases exist solely to show the relationships between one sentence and the next. Consider words and phrases such as *"firstly", secondly", "for example"* and *"on the other hand"*. These phrases have no proper place in a grammar that does not consider texts in the light of multiples of sentences.

In practice though, these words have a considerable influence on the correct interpretation of sentences and can cause otherwise perfectly grammatical sentences to appear ungrammatical. Consider the statement:

X distrusts Y.

This illustrates a simplified grammatical sentence of the form $NP \rightarrow V \rightarrow NP$. This is quite valid grammatically and can be used to generate a number of sample sentences, for example,

Peter distrusts Paul.
Americans distrust Russians.
We distrust our neighbours.

If we add the phrase *"for example"* to these then the acceptability of the phrases can be brought into question, even though the addition is grammatically valid on its own.

For example, Peter distrusts Paul.
For example, Americans distrust Russians.
For example, we distrust our neighbours.

If we link two of these sentences using the phrase "*for example*", then the acceptability of the sentences becomes conditional on the order of the linkage. A valid arrangement would be:

We distrust our neighbours. For example, Peter distrusts Paul.

An invalid arrangement would be

**Peter distrusts Paul. For example, we distrust our neighbours.*

In both cases there is no error in the basic grammatical structure of the sentences. They are generated from the same grammatical rule. The problem arises because the sentences are tied. A phrase exists that necessitates that the legitimacy of the sentences be evaluated in terms of the context of the sentences. The phrase "*for example*" presupposes that what follows will be a more specific case related to a previously stated generalisation. As "*Peter distrusts Paul*" is more specific than the example, the sentence "*For example, we distrust our neighbours*" appears wrong. (Considering a still larger context for this segment could cause us to have to re-evaluate this segment - Peter and Paul are very common names often used figuratively to represent everybody. In this sense, "*Peter distrusts Paul*" could plausibly be the more general statement.)

It should be noted that there is nothing ungrammatical in the order itself. A different linking phrase makes this order perfectly acceptable, as in the paragraph

Peter distrusts Paul. This shows we distrust our neighbours.

If we focus on the level of the grammar of individual sentences, it is possible to derive rules that show that these structures use different grammatical structures to make these connections, however, even given the different structures that may or may not be at the heart of these two paragraphs, the fact remains that the validity of the examples cannot be evaluated except in the context of the rest of the paragraph, as can be seen if we define an appropriate context for the invalid example

**Peter distrusts Paul. For example, we distrust our neighbours.*

This paragraph would be perfectly valid given a more general leading sentence, for example:

We distrust outsiders. For example, we distrust our neighbours.

In other words, many errors treated as grammatical errors are errors because a context for the sentence is not defined. As grammatical analysis focuses on the sentence, grammatical sentences then become those that include sufficient contextual information in the sentence for the context of all the elements in the sentence to be reasonably clear. Consider this example from Marantz (1981, p. 192) which he uses to illustrate the way some transitive verbs, like *eat*, appear freely without an object while others, like *lock*, do not:

Elmer locked the porcupine cage late last night.
**Elmer locked late last night.*

The second sentence is grammatically invalid in that "*lock*" requires an object. None the less the latter sentence (utterance) is acceptable in English conversation. If this object is supplied elsewhere in the context, then this not only becomes acceptable but also contributes to the overall cohesion of the cumulative text. It is in effect one of the devices (ellipsis) that contribute to the stability and economy of long stretches of text (cf. de Beaugrande and Dressler, 1994, p 49).

The error in this sentence then becomes an indicator of reference. The problem of definition is not so much one of some verbs in English being able to appear without an object and others not. Instead, verb use could be better stated as being acceptable when there is sufficient contextual information to fill the verb's contextual dependencies. A problem in processing these is that this acceptability depends not only on there being a context sufficiently well known for the text producer but also for the receiver.

So, for example, a verb, like "*eat*", does not require an explicit context. The context in which this word is used is already well defined in the culture of English usage. "*Eat*" includes the contextual information of common meal times, breakfast, lunch, supper, midnight snacks, popcorn at the movies, and so on. This makes a statement like "*let's eat*" perfectly acceptable and understandable. '*Lock*', on the other hand generally, does not have as much predefined context so "*lets lock*" does not, at face value, seem as acceptable. This would change should the external environment provide an appropriate context (such as when leaving home or getting out of a car).

3.3.2 Parsing requirements

Any non-trivial natural language understanding system needs a component for assigning syntactic structures to its input, that is, a parser. ... *Subsequent semantic and pragmatic components try to figure out the meaning of an utterance with the help of the structure assigned to it by the parser* (Plaehn, 1999, p. 5).

One of the problems associated with a parsing system in an educational context is the definition of the goal of the system. There are essentially two possible approaches (discussed from a CALL perspective in section 3.1.2).

The first aims to determine grammatical accuracy. In a system based on this parsing structure, any sentence that does not meet the requirements for grammatical accuracy would be rejected as unparsable. For example, in her paper on grammar design (in computational linguistics) König defines the essential task of a grammar as being "to specify pieces of a syntax tree" and of a parser or interpreter as "to construct a complete syntax tree for a given string or input semantics from the partial trees specified in the grammar (König, 1994, p. 17).

In a language learning implementation, however, this would have the effect of forcing learners to generate grammatically accurate sentences when doing an exercise which uses this type of parsing system as a basis. In any exercise which is intended to encourage learners to express meaning using English, this absolute view of grammar can become very frustrating. In these types of exercise, flexibility in analysing sentences would be more useful.

This leads us to the second approach to parsing. This approach assumes that, regardless of the accuracy of the grammar used, texts contain messages. This message is all important and the engine needs primarily to decode the message. Therefore, even ungrammatical sentences must be analysed as containing sense and the programme must respond to these as though they were grammatically accurate.

Consider these ungrammatical examples:

**The dog the man die*

**The dog the man bite*

If we start from the assumption that these sentences contain messages then we can make an attempt at interpreting them by applying the generalisations and conventions that make up grammatical rules in a flexible manner. For example, we could use these two generalisations about sentences in assigning sense to these sentences.

1. English has a basic structure of Subject $\dashrightarrow$ Verb $\dashrightarrow$ Object. This can be represented in phrase structure as NP $\dashrightarrow$ VP.

2. Intransitive verbs do not require an object while transitive verbs do require an object. Transitive verbs would then be represented in phrase structure as VP $\Rightarrow$ VP $\dashrightarrow$ NP.

"*Die*" is an intransitive verb. As such the sentence does not require an object. This leaves a text in which an ambiguity exists as there are two potential subjects with no means of distinguishing which one is the intended subject. This would result in two possible analyses for the structure NP $\dashrightarrow$ VP, namely

NP $\dashrightarrow$ VP	*The dog* $\dashrightarrow$ *die*
NP $\dashrightarrow$ VP	*The man* $\dashrightarrow$ *die*

This is the same structure that would result from analysis of the grammatically accurate sentence

The dog and the man die.

This interpretation is reinforced in this instance by the fact that the verb "*die*" expects a plural subject. However, this is not at issue as, without an explicit object, the same analysis would result if the singular form of the verb ("*dies*") had been used.

In the second sentence "bite" can be both intransitive and transitive.

This results in further ambiguity. The first interpretation is for the intransitive use of "bite" and uses the same process as was used for the first sentence. This would result in the analysis

NP $\dashrightarrow$ VP	*The dog* $\dashrightarrow$*bite*
NP $\dashrightarrow$ VP	*The man* $\dashrightarrow$ *bite*

which is the same structure as would result from the sentence

The dog and the man bite.

The second interpretation is for the transitive use of "bite". This requires an object, though from the sequence of words in the sentence this is missing. The simplest way of resolving this conflict is to use the nearest NP in the sentence to fill this dependency. The nearest NP in this structure is the NP "the man" resulting in the following analysis.

NP → VP	*The dog →bite*
⇒VP → NP	*bite → the man*

or

NP → VP	*The man→bite*
⇒VP → NP	*bite → the dog*

Different stresses in the delivery of this sentence can suggest either one of these interpretations. Given the world-knowledge that dogs sometimes bite people and the closer proximity between noun and verb, the more common interpretation would be the sentence

The dog bites the man.

It is also perfectly possible that the originator of these malformed messages did not intend any of these meanings. This does not mean that the attempt at understanding is wasted. An important consideration is that the malformed text is seen as a valid attempt at communication, which means that applications can respond in terms of the interpreted message. In a conversation simulation, for example, this would show up as an obvious misunderstanding requiring clarification or remedial moves by the originator of the message. In a written exercise (a paragraph or essay for example), this would show up as both a lack of cohesiveness in the text as a whole and as off-topic text. The way in which applications respond to this naturally depends on the context of the application. Regardless of this context, the fact that an inaccurate meaning has been assigned allows these applications to generate feedback based on this misinterpretation - again allowing the potential for clarifying and remedial moves.

This is very similar to the process of *negotiation of meaning* defined by Long as

> *... the process in which, in an effort to communicate, learners and competent speakers provide and interpret signals of their own and their interlocutor's perceived comprehension, thus provoking adjustments to linguistic form, conversational structure, message content, or all three, until an acceptable level of understanding is achieved* (Long, 1996, p 418).

In other words, the error is not explicit. Instead the error is implicit in the response. As a result the learner must self-correct. This, in turn, requires evaluating his or her own communication, identifying shortcomings and modifying this communication to be more understandable. As Lapkin and Swain put it

> *noticing a problem "pushes" the learner to modify his/her output. In doing so, the learner may sometimes be forced into a more syntactic processing mode than might occur in comprehension* (Lapkin and Swain, 1995, p. 373).

This mode helps learners internalise new forms (Holliday et al., 1989, pp 63-90) and in so doing improve the accuracy of their existing grammatical knowledge (Ellis and Nobuyoshi, 1993, pp 203-210).

The same principle would be applied regardless of what dependency has not been met. It should be noted that this second parsing system allows a simpler parsing structure for grammatically accurate sentences in which dependencies are not met by adjacent phrases. This is a fairly common problem for adverbs associated with verb phrases - illustrated in these two sentences:

> *Slowly the man walked home.*
> *The man walked home slowly.*

In both these sentences the adverb "slowly" (with a verbal dependency) is associated with the verb "walked". It is separated from this verb by an intervening phrase. This missing dependency can easily be assigned using the sense-finding parsing structure described above. In a parsing structure requiring absolute accuracy, each of these sentences becomes a special case that requires explicit testing as they do not follow the general pattern of phrase structure that can be represented as rules.

These two approaches to parsing require fundamentally different approaches to sentence structure. In the first approach, sentence structure is used to determine sense. In the second, phrasal dependencies determine sense. These two views cannot be seamlessly combined.

Any approach that does attempt to combine them will result in a system that contains two independent parsing systems in which the results of these parsing systems are combined, or in which the second parsing system is used only when the first fails. A programme that attempts to apply both systems will require independent representation of the patterns associated with sentence structure.

If this system needs to allow for interactive texts (as in a conversation), then it becomes increasingly likely that the input texts will involve errors in grammatical accuracy. These texts would not necessarily be incorrect in terms of the context of the interaction. Consider the following short interaction:

> *Q: Who went to town?*
> *A: The man.*

In this context the answer (A's contribution) violates the "rule" that sentences should contain a verb. In the context of the question, though, it is still a legitimate contribution. Any programme designed to handle this sort of text would have to ignore the apparent error in accuracy in order to correctly analyse the text. A possible approach to doing this would be by allowing a flexible parse which can pick up the verbal dependency from the preceding sentence.

3.4 Grammatical structure

3.4.1 A lexicalised dependency system

The lexicon-free dependency structures trial programmes developed in sections 3.2.2 on page 80 and 3.2.3 on page 84 had two key shortcomings. The adjacency structure provides a reasonable grammatical analysis of a simple sentence but it does not handle sentences which contain few part-of-speech clues. It also does not fulfil the all important criterion of providing an analysis of the *sense* of the sentence. Sense would have to be derived by post-processing the results of the analysis. This is a vital omission given the aim of the application and required a reworking of the grammar to better describe this key aspect of language.

There are two primary reasons for redesigning the grammar with a semantic focus rather than adding a post-process semantic analysis to the existing grammar. Firstly, as will be demonstrated in the following sections, the majority of ambiguities occur at a semantic level rather than a syntactic level. Secondly, many sentences generated in conversation could be classed as ungrammatical in the sense that they do not match the norms identified for common sentence structure. These sentences are, however, often appropriate in the context of the conversation as a whole and so need to be accounted for in the grammatical system.

As far as processing is concerned, dependency-based systems offer a significant advantage in being able to successfully accommodate deviant input. For example, WG accommodates deviant input because the link between tokens and types is the rather liberal 'Best Fit Principle' (Hudson, 1990, p. 45): assume that the current token 'isa' the type that provides the best fit with everything that is known. The default inheritance process which this triggers allows known characteristics of the token to override those of the type; for example, a misspelled word such as misspelled can 'isa' its type, just like any other exception, though it will also be shown as a deviant example. There is no need for the analysis to crash because of an error (Hudson, 1998, p. 15).

Dependencies, thus, form the basis of all structural analysis. Conceptionally a text is regarded as a whole that depends on its parts. This leads to a tree structure with the whole text as conceptual head. Dependencies in individual sentences should all relate to this overall text object.

3.4.2 Satisfying dependencies

The easiest level to illustrate using dependency structures to construct structural analyses is at the level of the sentence. A sentence is a conceptual unit in its own right. As such it has inherent dependencies, syntactic and semantic. The single most important dependency for well-formedness is the requirement 'has verb'. Seeing as this application is not intended to process conversation this dependency is treated as a prerequisite. For conversational purposes, though, it would be necessary to allow the dependency to be filled in other sentences, as in the text

A) *Did John go to class?*

B) *No, home.*

where the verbal dependency of the second utterance is filled in the first. It may also be necessary to allow the broader *vector* class (which includes both verbs and prepositions) to satisfy this dependency as this would allow texts such as

A) *Where is the milk?*

B) *On the table.*

I say 'may' as this can also be processed using the verb in the first utterance to fill the verbal dependency of the second. It does become more of an issue when non-verbal communication is added into the equation. Assuming that the context is clear to both parties (for example *(B)* knows that *(A)* will need milk) we could have this as a meaningful interaction:

[A enters]

B) *On the table.*

Until a grammar is developed that can assume missing content (in this case *(B)* could be responding to an, as yet, unasked question) we will have to settle for allowing the second utterance at face value in this formulation. The target environment of the current version of the application does place greater constraints on what we can class as well-formed texts. With formal written input, we can assume that all valid texts will contain sentences with verbs.

4 The grammar in brief

4.1 Overview

It is impossible to avoid the inherent structure of phrases when trying to determine grammaticality. This structure can be represented in terms of the dependencies of the individual words. The fact that words share properties of behaviour means that it is possible to generalise - classifying words by their behaviour and associating dependencies with that behaviour. One method of classifying these is expressed the classic definitions of parts-of-speech.

The rules governing this behaviour in the dependency-based structure adopted in this application can be summarised as:

- Each word must be the grammatical dependent of a word in the sentence.
- Each word can have one and only one grammatical dependent.
- A word can be a dependent of itself.

These three rules effectively determine existence of phrasal units inside a sentence. An additional minor set of dependencies is necessary to determine sequencing within a phrase. Together these allow for fairly complex constructions, including nested phrases and conjunctions. While these are called grammatical dependencies, they do not guarantee that a sentence which meets the requirements can be regarded as well-formed (a sentence containing only a noun-phrase, for example, would fill the above criteria).

Sentences are constructs that determine the relationships between the concepts within the sentence. In this application these concepts are primarily determined by the nouns in the sentence. Following this line of reasoning, the relationships between concepts are defined by the verbs in the sentence. This, then, determines the requirements of the second set of dependencies in the grammar of the application, called, for want of a better distinction, the semantic dependencies.

In this grammar, concepts are classified as nouns and relationships as verbs because of the similarity between these classes and the classifications of traditional grammar. It should be noted though that in this grammar the classes may contain members that do not traditionally fit the definition of verb or noun. For example, because of an overwhelming similarity in dependency structure, prepositions are also classified as verbs in this system, though with a different priority level.

These dependencies can be summarised as:

- Each member of a vector class (V) has both a left and a right dependency, though neither has to be filled.
- Each member of an object class (N) has one verbal dependency that must be filled (this can be filled by a V phrase either before or after the N).
- Dependencies have between zero and one dependent but not more than one.[54]
- A sentence is a superset of the N class and, as such, has a verbal dependency.

To allow for multi-sentence texts:

- Words can have a dependency that refers to concepts defined outside the current sentence.

This is primarily aimed at codifying the behaviour of external references (such as "*this*", "*he*", "*these*", "*them*", etc).

While the dependency structures presented here and the logic that governs the functionality of the structures has been constructed specifically to determine relationships in English language texts, the principles governing the structures are not bound to any language. A benefit of using dependency structures defined in the lexicon is that an application such as this one could be ported to a non-English language and structure with relatively few changes to the programme itself. Essentially translation would involve defining different dependency relationships and building a lexicon in the target language.

4.2 Lexicon

As a dependency-based system, almost all grammatical rules are implicit in the structures contained in the lexicon. As this lexicon then governs all processing it warrants more detailed consideration.

[54]Conjunctions cause branching interpretations but each interpretation should only allow one dependent.

4.2.1 Overall structure

There are five classes of word allowed in this implementation. These are:

sentence a category for sentences (symbol: 'S');

noun a broad category containing all objects and concepts (symbol: 'N');

vector a broad category containing all verbs and prepositions (symbol: 'V');

reference a category for all types which refer to other categories (symbol: 'Ref'). Adjectives and adverbs fall into this category.

conjunction a category for types that make explicit links between words/phrases of the same category (symbol: 'CONJ')

There are also entries defined as having no category, which includes most punctuation and words which may have significance in terms of well-formedness but do not for determining sense (such as determiners).[55]

Conceptually the lexicon is arranged as a tree structure in which words inherit the characteristics of their parents in the tree (as illustrated in Figure 10). This allows a flexible structure as dependency requirements specific to a word or a class of words can be added without affecting the other words in the lexicon.

Figure 10 shows only the top level of the lexicon tree. Each end node in this diagram is itself further divided (verbs, for example, are further divided into past, present and perfect forms). There is no formal constraint on tree depth. Even this top structure can change (should a new type be identified which does not fit as a derivative of one of the pre-defined base classes) without affecting the functionality of the programme or the grammar.

4.2.2 Node structure

Each word in the lexicon can be viewed as a node in a knowledge network. As words can be more than one type this information is duplicated for each type in the node. To illustrate

[55]This basic class structure is implicit in many dependency-based grammars. For example, Tesnière (1959) postulates that there are only four main parts-of-speech verb, noun, adjective and adverb. With the following relations and (Kahane, 2002, pp. 9,10) demonstrates that this economy can be carried over to grammatical formulations (such as HPSG) with differing theoretical viewpoints.

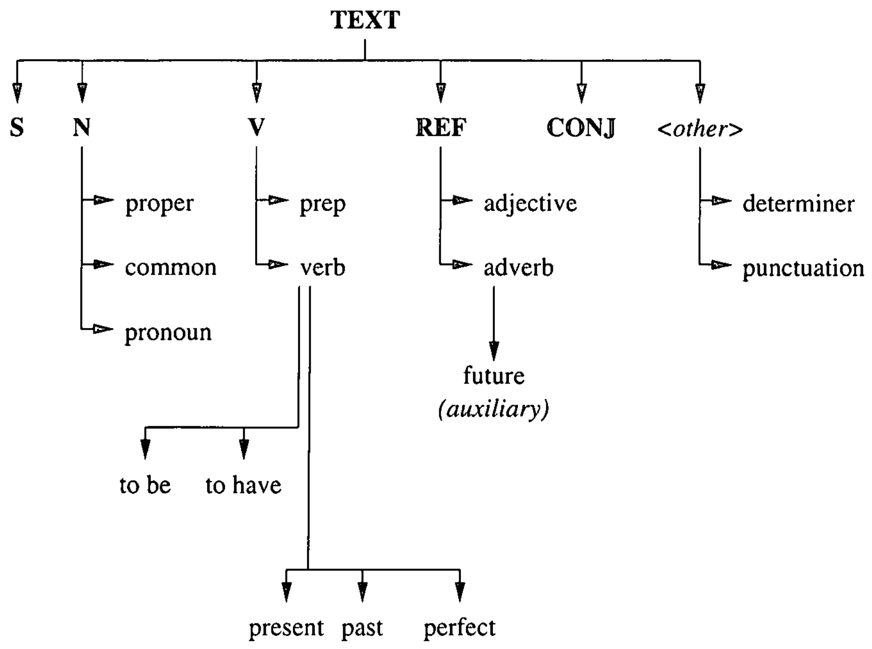


Figure 10:

this concept, let us consider the word '*race*' which has both a verb and noun usage. At its simplest[56] this word would be represented as:

$$race \mapsto \left\{ \begin{array}{ccc} cn & \mapsto & \{stem \mapsto race\} \\ v & \mapsto & \{stem \mapsto race\} \end{array} \right\}$$

A separate definition in the form $cn \mapsto \{stem \mapsto race\}$is necessary to allow for cases where words with identical spelling have very different meanings and potentially different dependency structures. Each type of the word would then inherit the properties defined in its parent class ('*cn*' or '*v*'). Including this inherited data makes this simple definition conceptionally equivalent to:

[56]The representation of relationships or knowledge structures in the lexicon is covered separately in later sections 4.2.3 on page 116 and 4.3 on page 133.

$$race \mapsto \left\{ \begin{array}{ll} cn \mapsto & \left\{ \begin{array}{lcc} stem & \mapsto & \{race\} \\ dep & \mapsto & \{cn\} \\ seq & \mapsto & \{cn\} \\ ldep & \mapsto & \{v, vp, i, ip, prep\} \\ rdep & \mapsto & \{v, vp, i, ip, prep\} \end{array} \right\} \\ v \mapsto & \left\{ \begin{array}{lcc} stem & \mapsto & \{race\} \\ dep & \mapsto & \{v\} \\ seq & \mapsto & \{adv, adj, vc\} \\ ldep & \mapsto & \{S, cn, pn, pron\} \\ rdep & \mapsto & \{adj, cn, pn, pron\} \end{array} \right\} \end{array} \right\}$$

The '*dep*'[57] and '*seq*'[58] structures are analogous to phrasal grammatical dependencies. These group words form conceptual entities. Phrases, as with WG, are implicit in these dependencies but do not play a part in the grammar (cf. Hudson (1998, p. 2)). The '*dep*' dependency indicates required characteristics of a phrase head and the '*seq*' dependency indicates word order dependencies.[59] This is similar to the concept of syntactic dependencies and topological constituents outlined by Kahane and Gerdes where a syntactic dependency expresses the relationship between the lexical head of a constituent and the lexical head of a subconstituent without reference to the linear order of these two elements and topological constituents are "lineally ordered groups of words appearing naturally in the layout of the word order of the sentence" (Kahane and Gerdes, 2002, p. 3). As in this implementation the syntactic dependency are independent though related notions. Also, as with Kahane and Gerdes, the syntactic dependency is an intermediate description closer to the meaning than the topological phrase structure, which is closer to the form of the language.

In this implementation a word may satisfy its own '*dep*' requirements - meaning that the smallest permitted unit is an individual word (if that word matches all required dependencies). Determiners, for example, are defined with

$$det \mapsto \left\{ \begin{array}{lcc} dep & \mapsto & \{cn, pn\} \\ seq & \mapsto & \{cn, pn, adv, adj\} \end{array} \right\}$$

[57]An abbreviation for '*required dependency*'

[58]An abbreviation for '*sequential dependency*'

[59]It is not strictly necessary to represent this as two distinct structures; however ,this provides a short-cut in processing that eliminates unnecessary recursion.

which means that they can be followed by a common noun, proper noun, adverb or adjective and require a common noun or proper noun to fill its dependencies. Dependencies are bi-directional in that a word that satisfies a dependency may occur before or after a word. Adverbs are defined as

$$adv \mapsto \left\{ \begin{array}{lll} dep & \mapsto & \{v, vp, vc, i, ,ip, pp, adj\} \\ seq & \mapsto & \{v, vp, vc, i, ip, h, hp, f\} \end{array} \right\}$$

which means that the adverb's dependencies can be filled by any one of a verb (past, present or continuous tense), verb 'to be' (past or present tense) or an adjective. This definition covers structures such as

The dog barked loudly.

The dog loudly barked.

The dog barked at the postman loudly.

Simplified diagrammatic representations of the dependencies for these structures are given in figures 11, 12 and 13.[60] Figure 13 also illustrates another property of these dependen-

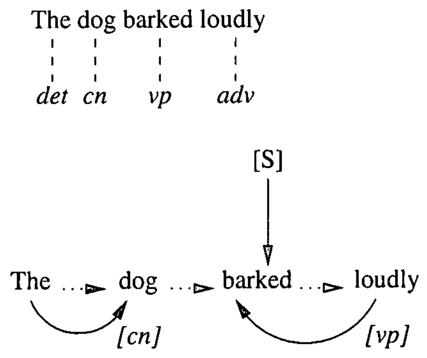


Figure 11: Syntactic dependencies for "The dog barked loudly"

cies. Dependencies can be filled even when other phrases intervene - the only requirement is that the dependencies required by the intervening phrases are completely satisfied first.[61]

[60]Only the dependencies which determine conceptual groupings (phrases) are shown. Sentences are treated as larger concept groupings with a verbal dependency (indicated by the arrow labelled [S]).

[61]This differs from the WG definition where the adjacency principle requires all phrases to be continuous (with some exceptions) (Hudson, 1994, p. 4992). As a consequence some structures will succeed here which

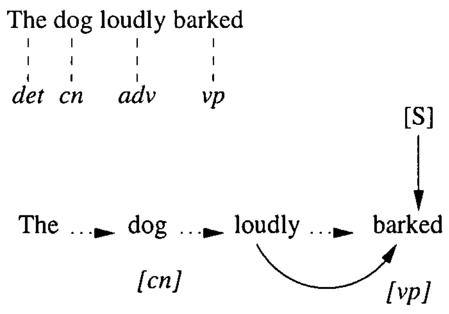


Figure 12: Syntactic dependencies for "The dog loudly barked"

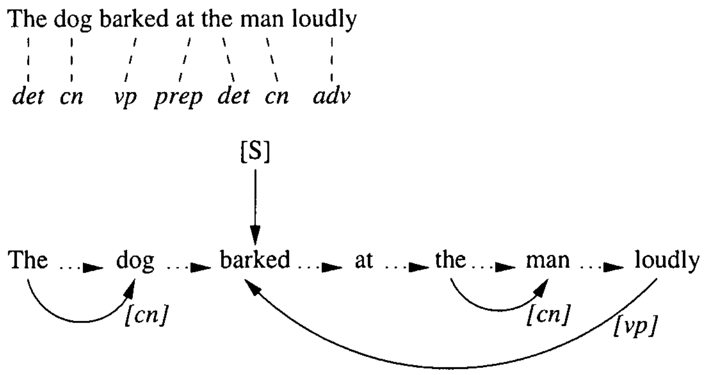


Figure 13: Syntactic dependencies for "The dog barked at the man loudly"

To return to the definition of the node for 'race', the left and right dependencies[62] defined for its noun variant indicate requirements for its use in a subject or object relation to a vector (verb or preposition). For words belonging to the V (vector) class both left and right dependencies have to be satisfied. For N (noun) classes only one of these must be satisfied. The same structures are used to resolve references inside and outside a sentence in words belonging to the Ref (reference) class. Resolving the left and right dependencies forms the basis for determining semantic content in a sentence.

From Figure 13 it can be seen that a word or concept can be the target of more than one left and right dependency. This provides a means of indicating how relationships combine in building sense in a sentence. Each unique relationship structure that can be built from the possible combinations of left and right dependencies in a sentence is referred to as a thread (as it defines a "thread of sense" in the sentence). The network of relationships for a sentence thus consists of the combination of all the threads of sense in a sentence. In Figure 11, for example, the combination of ldep and rdep determine the thread of

$$\begin{tikzcd} dog \arrow[r] & bark \arrow[d] \\ & loud \end{tikzcd}$$

In Figure 14 the dashed arrow indicates this relationship. The bi-directional nature of this relationship (the right dependency of '*dog*' and the left dependency of '*barked*' define the same relationship but from different point of view) becomes necessary in representing the relationships affected by branch conditions (such as conjunctions).[63]

Figure 15 illustrates the effects of left and right dependencies in a slightly more complex sentence (using the example from Figure 13). The different priority levels for verbs and prepositions ensure that the verb 'basked' is correctly identified as defining the primary relationship in this sentence. The resulting threads (semantic relationships) can be

may not ordinarily be regarded as well-formed. For example a sentence such as:

*? *The man in the hat wearing the coat looked old.*

Will be treated as containing the concept '*in the hat*' nested in the concept '*the man wearing the coat*'. The goal here is to attempt to determine a viable set of relationships which represent the meaning of the sentence regardless of the grammaticality of its structure.

[62]These are abbreviated 'ldep' and 'rdep' in the node structure of the lexicon used in this application.

[63]The alternative is allowing a word to have a number of left or right dependencies. This is, however, computationally unfeasible as it means recursive logic structures become necessary to process these. This is both time-consuming and memory intensive.

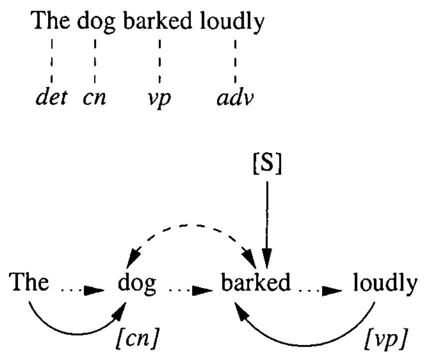


Figure 14: Left and right dependencies for "The dog barked loudly"

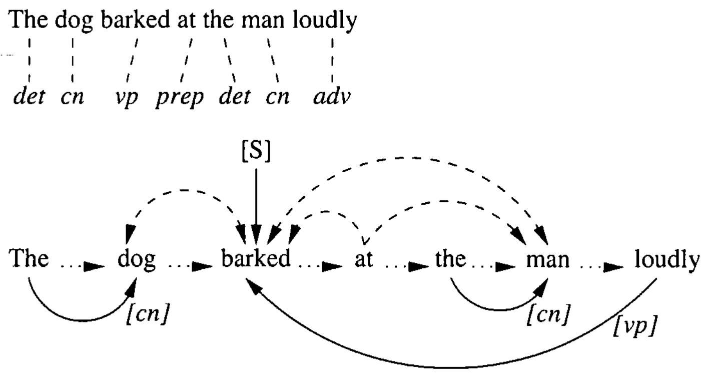

The dog barked at the man loudly
det cn vp prep det cn adv
[S]
The dog barked at the man loudly
[cn]
[cn]
[vp]

Figure 15: Left and right dependencies for "The dog barked at the man loudly"

represented as:

$$\begin{tikzcd}
& \textit{bark at} \arrow[r] & \textit{man} \\
\textit{dog} \arrow[r] \arrow[ur] & \textit{bark} \arrow[d] & \\
& \textit{loud} &
\end{tikzcd}$$

4.2.3 Sentence structure

Texts are processed from left to right. As a text is processed, trees that describe the potential structures of the sentence are generated. As a result, these relationships are very fluid - changing as each successive word in the sentence is encountered. This should model human processing to a degree. Experimental evidence (cf. Marslen-Wilson (1973, pp. 522-523)) has shown that human language processing is highly incremental, meaning that humans construct a word-by-word partial representation of an utterance as they hear each word. In practical terms this provides a simple and convenient starting point for analysis as no effort has to be made to identify a potential phrase-head or other starting point inside the sentence.[64] Instead, the structure is determined solely based on the dependencies of the individual words that make up the sentence (cf. Hudson (1994, p. 4990)).

To illustrate this process graphically, consider the sentences first introduced in section 2.2.1 on page 28:

John is easy to please.

John is eager to please.

As can be seen in figures 16 and 17 these develop very similar dependency structures.[65] The two sets of structures in each figure illustrate two processes in the analysis. The first

[64]This approach is also drawn from current Word-grammar theory which, for example, uses determiners as heads for noun phrases (Hudson, 1998, p. 18). This is a logical choice given that this type is a good predictor of later content (a determiner 'expects' a noun dependent). However, this choice is disputed in published WG analyses and may turn out to be wrong (van Langendonck, 1994, pp. 243-259).

[65]This is typical of dependency-based grammars. For example Tesnière (1959) proposed the following relations for the four main parts of speech (verb, noun, adjective and adverb):

- verb actants are nouns,
- verb modifiers are adverbs,
- noun dependents (actants or modifiers) are adjectives and
- adjective and adverb dependents are adverbs.

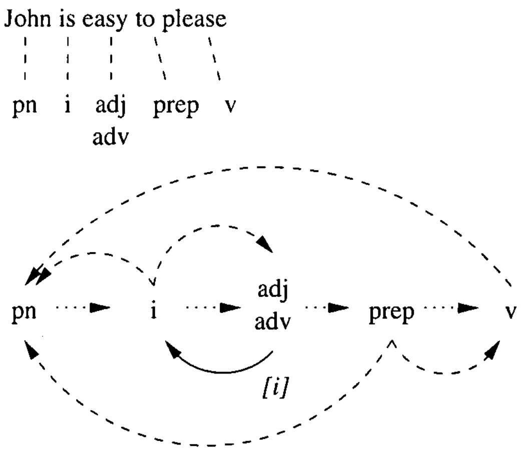


Figure 16:

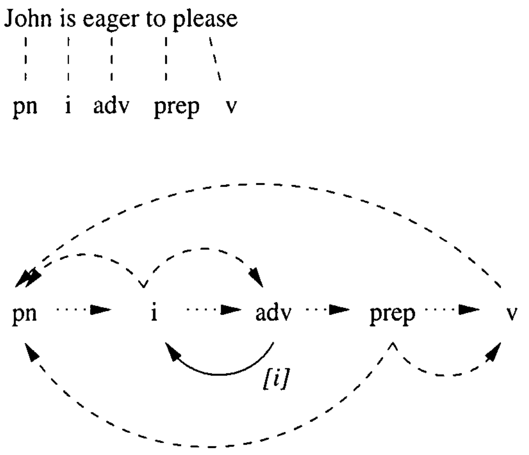


Figure 17:

resolves immediate word-based dependencies. These dependencies govern the grouping of words into conceptional units inside a sentence (generally equivalent to phrases). The second dependency structure governs relationships between these conceptual units and between sentences in longer texts.

There should only be one valid set of word-based dependencies for a given set of word-types in a sentence but there can be a number of valid threads. A best-fit approach is used which retains the thread sequences which have the fewest dangling threads (thread dependencies that remain unsatisfied). Even then it is possible to have a number of potentially equally valid thread structures - particularly when conjunctions affect these structures.[66]

The implementation of the 'best-fit' used in this application uses verbs to determine thread dependencies. A numeric value is assigned to the concepts based on a concept's position in relation to a vector (a verb or preposition). This value changes depending on the distance between the concept and a vector - which provides a priority level for each of the concepts inside a sentence and for the relationships between these concepts.

To facilitate this process each class of word has been given a default priority level. For most types this value is 0. For vectors that can satisfy a sentence dependency the value is 1 and for other members of the vector class (prepositions and continuous-tense verbs) the value is 2. The process of assigning values is a simple additive operation. As each new phrase-head is discovered one of three operations occurs:

- The priority-level is added to that of the previous phrase-head if the two phrase-heads are of different classes;
- The priority-level is incremented by 2 if the two phrase-heads are of the same class; or
- The priority-level of the previous phrase-head is used if the two potential heads from part of a compound (for example, when the first is verb type and the second is a preposition).

Only vectors have initial priority levels larger than 0. All other types have an initial priority of 0. As a result, the target of a vector will have the same priority level as the vector. Any

When a word of a part-of-speech other than adjective wants to modify a noun it must be "transferred" into an adjective, that is, it must be covered by a special word which masks it and makes it appear as an adjective. The transfer of nouns (into adjectives and other parts-of-speech) is effected by prepositions. (Kahane, 2002, p. 8)

[66]This problem is discussed in more detail in section 4.2.4.

object with a lower priority level becomes a potential origin of the vector (with one proviso: if a vector (*v1*) with a lower priority lies between the potential origin and the vector looking for origins (*v2*) then the priority level of the potential origin must be lower than the priority-level of *v1*) . The last piece of the puzzle required to make this work is the primary vector. This vector is defined as the vector that satisfies the vector dependency of the sentence and is assigned a priority level of one.

This process can be demonstrated in the following sentences.

1. *The girl smiled at me.*

2. *The puppy the boy likes looks fat.*

3. *The man in the coat walked home with his wife.*

The sentence (1) (illustrated in Figure 18) is the simplest case. An initial priority level is

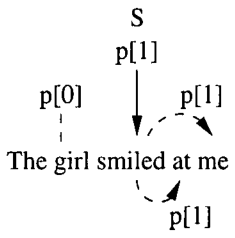


Figure 18:

assigned to the first object (*'girl'*) and a priority-level of 1 is assigned to the primary vector (*'smiled'*). The word *'at'* is classified as a vector with a higher priority-level (a preposition) and takes its value from the adjacent phrase's head. In this case the words themselves are adjacent. Specifying that it uses the phrase-head allows other words and, in some cases, phrases to physically separate the two concepts while still preserving this sense. Lastly, the object *'me'* is assigned a priority-level based on its default priority-level plus that of the adjacent phrase - a calculation resulting in a value of 1. Later, when determining threads, this will indicate that the *'me'* is the target of the vectors *'smiled'* and *'at'*.

The second sentence (illustrated in Figure 19) is more complex in that we have two adjacent objects. A value of 2 is added to the priority-level of the first object to give the priority level of the second object. As it is not the vector that satisfies the sentence's vector dependency, an additive operation gives the value of the *'likes'* vector. Finally, the *'looks'*

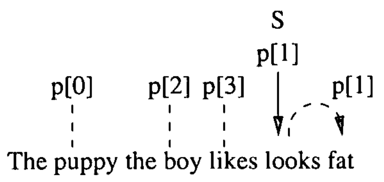


Figure 19:

vector does satisfy the sentence's vector dependency so its value is set to 1. This sentence includes two candidates for primary vector ('*likes*' and '*looks*'). To determine which of these is the correct option the programme attempts to build sentence structures for both. However, if '*likes*' is used as the primary vector then the object '*boy*' is left without any associated vector as is illustrated in Figure 20. (A visual metaphor for this rule illustrated

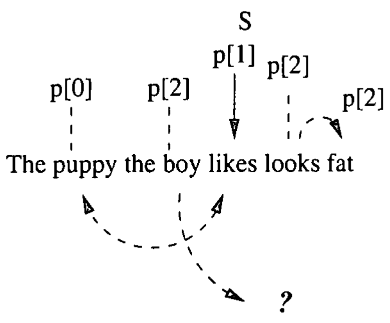


Figure 20:

by the drawing is 'thread lines may not cross'). Consequently, only the first variant is used.

The last sentence (illustrated in figures 21 and 22) gives the best illustration of using

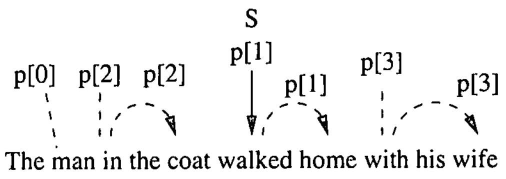


Figure 21:

these priority-levels to determine the origins of vectors and of processing sentences with more than one potential thread of sense. The value of the vector '*in*' is based on its default

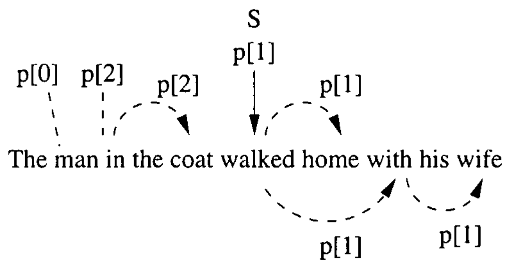


Figure 22:

priority level added to that of the previous object. The value of the object '*coat*' is then determined by this vector. In Figure 21 the same process is used to determine the values of the priority levels of the vector '*with*' and '*wife*'. Each of the objects with a priority lower than that of one of the vectors is a potential origin of the vector - with one exception. The object '*coat*' may not serve as an origin for '*with*' as the '*walked*' vector lies between the two, with the result that the priority level for '*coat*' is too high. The object '*man*' can, however, serve as an origin for '*with*'.

As Figure 22 illustrates, the programme builds representations for each of the possible senses contained in the sentence. The priority-levels in this illustration link the vectors '*walked*' and '*with*' to indicate that these should be related in the sense structure of this sentence.

The links outlined by the relationships between priority levels form the primary means of determining thread structure inside a sentence. To further illustrate this process consider the ungrammatical sentences below:

**The boy the dog bit.*

**The boy the dog bit the man*

When a primary vector (main verb) is determined it is assigned a priority level of one. Priority level calculation then proceeds using the priority level of the primary vector as a starting point. This allows two levels of thread dependency, one linking to the left and one to the right.[67] The basic formula for satisfying these two dependencies requires the thread on the left to be of a lower priority than the thread on the right. As is shown in figures 23 and 24 the thread dependency for '*the dog*' remains unfilled while in the sentence

[67]Certain conditions, such as conjunctions, can create multiples of objects which share a particular priority

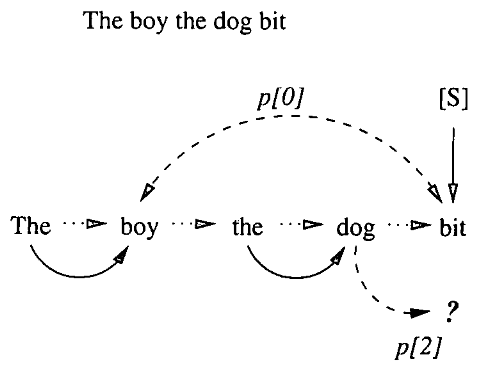


Figure 23: Unfilled thread dependency example: "The boy the dog bit"

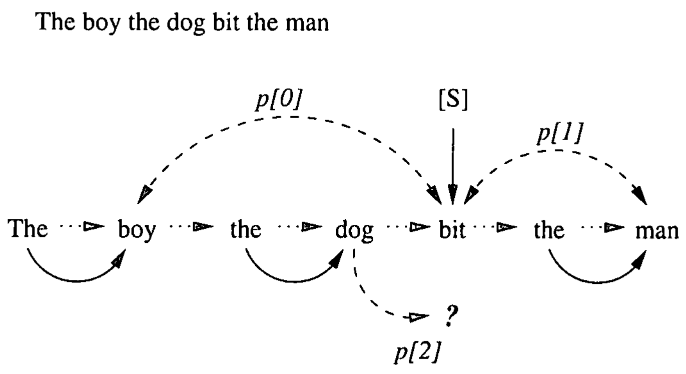


Figure 24: Unfilled thread dependency: "The boy the dog bit the man"

The boy the dog bit is blind.

(illustrated in Figure 26) the thread dependencies of both '*the boy*' and '*the dog*' are satis-

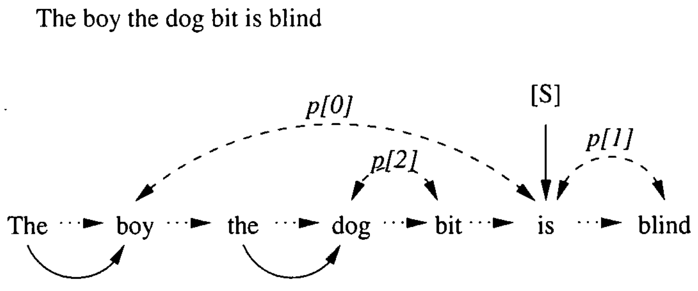


Figure 25: Filled dependency example: "The boy the dog bit is blind"

fied. The process of assigning priority levels also assists in determining the primary vector in the sentence. In the sentence examined in Figure 26, for example, an attempted analysis with the verb '*bit*' as primary vector would result in unfilled dependencies, as illustrated in Figure

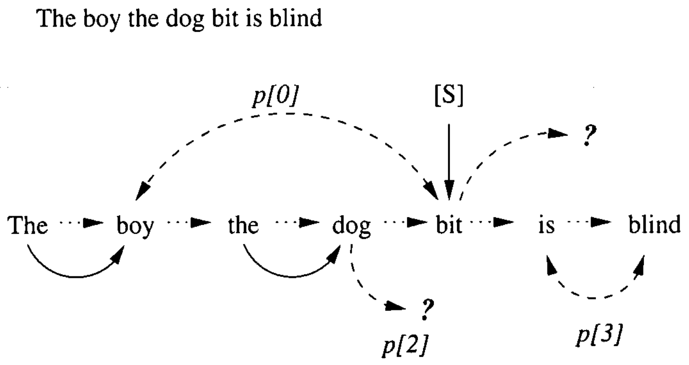


Figure 26: Priority levels determine primary vector: "The boy the dog bit is blind"

This priority level also forms the basis for parsing sentences containing conjunctions - these are treated as conditions that cause concepts to share a priority level, thus allowing a vector to be satisfied by more than one object or concept of that level. This is illustrated more fully in the section 4.2.4.

level (referred to here as branch conditions). In these cases the thread dependency would define relationships linking to more than one object or concept at that level.

It is also worth noting that this process actively tries to assign meaning. In the two ungrammatical sentences above the process of assigning meaning will fail in the sense that it will not be able to find a cohesive thread for each of the concepts identified in the sentence. The thread patterns illustrated in figures 23 and 24 do, however, also indicate the sense that the application will use in further processing - the best fit for finding a cohesive thread in the sentences (essentially ignoring the role of '*the dog*' in these sentences).

4.2.4 Semantic dependencies at work

Broadly speaking, there are two types of dependency used in this parser. The first dependency type is syntactic.

The second type of dependency is semantic and is not necessarily bound by sentence boundaries. This dependency type has two functions, partly decided by the structures that are allowed to fill the dependency. The first function tracks threads of sense inside a sentence. A 'thread' in this instance refers to a potential interpretation based on the syntactic structure of the sentence. Threads can differ between two readings of the same sentence - a single, grammatically well-formed sentence, can contain more than one thread of sense. This mechanism provides a means of managing ambiguities as well as determining reference inside a sentence.

Consider this generic example:

$N_a\, prep\, N_b\, CONJ\, N_c\, verb\, N_d$

The conjunction in the sentence creates a branch condition - two thread sequences are possible here with equal validity. These can be represented schematically as shown in Figure 27 and Figure 28 (dashed arrows show how the conjunction links concepts, solid arrows show the vector assigned to an object):

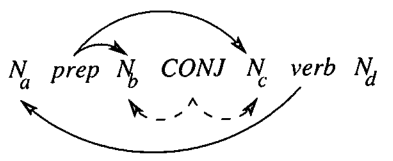


Figure 27: Conjunction linkage summarising first possibility

A simple definition for conjunctions, as it would apply in this application, could be stated as:

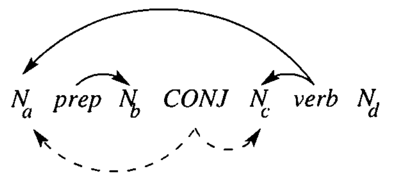


Figure 28: Conjunction linkage summarising second possibility

A conjunction causes words of the same type and containing similar characteristics to be evaluated together.

or, as a set of dependencies,

A conjunction links the two concepts with the most similar set of syntactic and *semantic dependencies as being in the same scope.*

So a conjunction is defined as linking the two most similar objects that satisfy the thread structure. If N_c is more similar to N_a then the thread pattern diagrammed in Figure 28 will be chosen. If, however, N_c is more like N_b the thread pattern diagrammed in Figure 27 will be used.

To further illustrate this practically, consider the following two sentences which have exactly the same grammatical structure (at least as far as the parts of speech used and their sequence are concerned):

1.	The	man	in	the	coat	and	the	hat	went	to	town
	det	*cn*	*prep*	*det*	*cn*	*conj*	*det*	*cn*	*vp*	*prep*	*cn*

2.	The	man	in	the	coat	and	the	boy	went	to	town
	det	*cn*	*prep*	*det*	*cn*	*conj*	*det*	*cn*	*vp*	*prep*	*cn*

There is no indication, at this level, of what the scope of the conjunction "*and*" should be. Given a person's superior lexical information we can see that, in the first example, the conjunction has "*coat*" and "*hat*" as items in the same scope (illustrated roughly in Figure 29). In the second the scope links "*man*" and "*boy*." In terms of processing *sense* this distinction is vital as it determines which of the nouns should fill the subject dependency of the verb "*went*" (illustrated roughly in Figure 30).

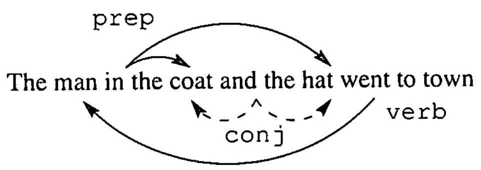


Figure 29: Conjunction linkage for "The man in the coat and the hat went to town"

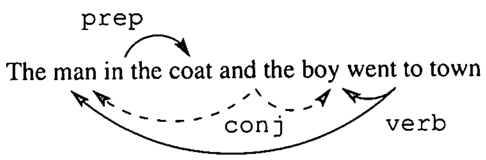


Figure 30: Conjunction linkage for "The man in the coat and the boy went to town"

Ultimately the meaning of the sentence is based on a weighted comparison of threads.[68] Those threads with the highest weight are used as being the most likely ones to represent the meaning of the sentence. The second function supplements the syntactic dependencies. Semantic dependencies can have syntactic implications - in other words syntactic analyses which provide more satisfied thread dependencies will be regarded as better formed and more likely interpretations than sentences with fewer satisfied thread dependencies. Thread dependencies can be specified as 'left', 'right' or 'both left and right', meaning that they have to be satisfied either on the left, or on the right or on both left and right of a particular words position in the sentence. Verbs and prepositions (vectors) for example often have thread dependencies that need to be filled on both left and right. Nouns (concepts) on the other hand have an either/or declaration. For nouns-as-subject the thread dependency has to be satisfied on the right. For nouns-as-object the thread dependency has to be satisfied on the left. A corollary of this is that the noun cannot have both a left and a right thread dependency for a given thread of sense. Only one applies.[69]

[68]Each thread interpretation is assigned the same weight. As words often have a number of interpretations this regularly results in hundreds of possible interpretations for a given sentence, often with very minor differences between the interpretations. The weights of identical interpretations are then added together. The highest weighed interpretations from this merged set are regarded as the most likely interpretations of the sentence.

[69]An interesting side effect of this process is that the generated sense threads very closely match the "simple, declarative, active sentences" (Bornstein, 1977, p. 39) that form deep structure in generative grammars.

4.2.5 Long-distance dependencies

Conjunctions are, obviously, only one indication of needing a structure that allows a number of potential candidates to fill the left dependency of a vector. Another occurs often in multi-sentence texts. Pronouns and some adverbs often serve to link the content of a sentence to external events or previous sentences in a text. These can also be viewed as dependencies - though these dependencies will have no impact on the grammatical analysis of a sentence.

In this application concepts that can have external dependencies have an extra dependency defined (abbreviated '*exdep*' in the lexicon). This dependency does not affect the analysis of sentence structure, but doe influence the construction of node structures. In brief, to determine the target of this dependency the application first tries to satisfy the dependency from the sentence containing the reference (searching both left and right of the word containing the extra dependency). If this fails, the application tests earlier sentences in the text for one which satisfies the dependency condition or which contains a concept which satisfies that condition. If this remains unfilled then it tries to satisfy the dependency from later sentences in the text as these are processed.

The full process of determining the best target for an extra dependency is a more complex procedure, very similar to the process of resolving the sense threads generated by the conjunctions in section 4.2.4. Similarity in objects is used to filter the potential targets of a references in much the same manner as similarity is used to determine the potential targets of conjunctions. The relationships and inheritance of an object are compared and the one with the greatest number of shared relationships wins.[70] A useful feature of this approach is that exactly the same logic can be used to determine the targets of references outside the sentence and, given a lexicon with sufficient built in world knowledge, outside the text.

The lexicon used in the programme is far simpler in construction than that suggested in section 4.2. In the implementation presented here, semantic references (with the exception of a stem word) have been omitted in the lexicon definition. Provision is, however, made

[70]This is also the single greatest lack in the implementation provided here. While this has been implemented in the programme itself it depends on a very detailed lexical definition for its functionality. This detail includes numerous relationships defined for each word in the lexicon. This work is far from complete. In order not to prejudice the results by potentially crafting a lexicon based on the target texts this detail was removed entirely from the lexicon used in this implementation.

This means that the application cannot use this method to distinguish between threads of sense in a text and uses its fail-safe routine. This allows all threads to co-exist. Probable meaning is assigned by an additive function. As any sentence will potentially have many threads and many potentially valid grammatical analyses the programme merges these to determine which structure is most common and which thread the most likely interpretation of a sentence.

for these in the lexicon structure and in the logic of the programme. This omission is partly a factor of time - adding semantic relationships to the approximately 7000 entries in the lexicon is an extremely laborious process. Partly, though, this has been by intent. Any semantic relationships defined in the lexicon will, inevitably, reflect the priorities and world view of the person adding those relationships - much as the choice of words in the lexicon reflects the priorities of the person who built the lexicon in the first place.

The logic of the application is intended to build semantic networks with the idea that these will be an accurate reflection of the information in a text. Any extra semantic information that is predefined will influence the networks generated from a text. This would be desirable in scenarios where a text makes explicit or implicit reference to information not contained in the text. In this scenario the lack of predefined semantic information is a liability. In scenarios where one does not want to 'pollute' the relationships in a text with external references this is a definite bonus. Furthermore, while the examples here do not illustrate this use, it is always possible to use a training text to build a semantic network providing contextual background prior to doing an analysis of learner text.

This inevitably impacts on the performance of some aspects of the programme. For example, the programme uses a little logic to find a best (optimal) fit for dependencies that cross sentence boundaries (long-distance dependencies or references) such as *'this'* and *'he'*.[71] This logic examines the semantic links defined for each potential target of the reference to find a target with the best fit.

Implicit references link to information that forms part of the shared world-view of the communicants. These are handled as a by-product of the weights assigned to objects in the text. As the relationships between these references and the objects they refer to are dependent on semantic information, these will suffer from the omission of a semantic component in the lexicon as well. However, this does not impact on the functionality of the programme to the same extent as explicit references because implicit references cause the creation of unique object definitions.

Explicit references, on the other hand, do not necessarily result in the creation of unique object definitions. The same referent (for example, *'he'*) is more likely to reference different objects at different points in the text - a factor that can cause ambiguity in the programme output. A corollary of this which governs the logic of this application is that, in most cases, this means that the referent will be part of the text, which means that it is possible to find a reasonable target for the reference. This may not be the intended target - 'reasonable' in

[71] A brief discussion of this type of reference was made in section 3.3.1.

this sense means 'acceptable' - which in turn implies that the text producer can accept the error in reference as resulting from an unintended ambiguity.

Consider the following rather artificial paragraph which contains a number of implicit and explicit references or long-distance dependencies (the explicit references are the only ones I want to examine here).

> *(1)Sally was worried. (2)Peter is flying home. (3)Tom is driving home. (4)It was raining. (5)He crashed. (6)She heard the accident. (7)They loved each other.*

Consider the reference *'he'* in sentence (5). *'He'* would be defined in the lexicon as shown in abbreviated form in Figure 31. The logic tests the link relations of the previous words in

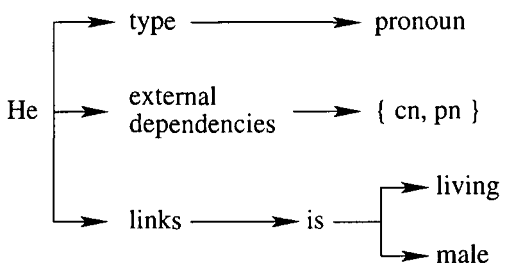


Figure 31:

the sentence word in the text, first to find words that match the basic $\{cn, pn\}$ dependency (the grammatical dependency), then to find words which have the greatest similarity in link structure (the semantic dependency). Both *'Tom'* and *'Peter'* are potential candidates here as the words would be defined with the same links '$is \mapsto living$' and '$is \mapsto male$' . Because *'Tom'* has closer proximity to the reference it has a higher priority and is seen as the best candidate.

The lookup for *'he'* is the simplest case. The lookup for *'She'* in sentence (6) is more interesting. The lexical definition for this word is almost identical to the definition for *'he'*. The only difference is that *'she'* would include the link '$is \mapsto female$' in the place of the '$is \mapsto male$' link for *'he'*. In the logic to find the best semantic fit for the dependencies two candidates are found. The first is *'Tom'* (with one link similarity) and the second *'Sally'* (with two link similarities). Because of the two link similarities *'Sally'* will have a higher priority in this case and will be the target of the reference.

There are two references in this paragraph that the current logic cannot handle. The first is *'they'* in sentence (7). This requires logic that allows the reference to match more than once and across sentence boundaries. Resolving the problem becomes a problem of resolving the scope of the reference. If one allows the word to match across sentence boundaries, where does it stop matching? Should it match *'Peter'* and *'Tom'*, *'Tom'* and *'Sally'* or all three people?

In human processing, this results in ambiguity - a factor which seems to indicate that the answer to the problem is that all three interpretations are valid. This ambiguity is reflected in the programme in that references for all three scenarios are be created. This would not impact significantly on the functioning of the programme, when it is used to compare texts. The network that the programme develops contains all three interpretations, though at lower probabilities of certainty of interpretation than it would have with only one possible interpretation. If the underlying assumption of this application is correct - namely that the key points of a text will gain a higher priority than less important statements in the text because of repetition and frequency of reference - then this will still result in an increase in the priority level of the nodes in the network which this links to.

It would, however, affect its use in an interactive mode, should it be used to provide the back-end to a chatbot or natural language interface, because the programme does not have the extensive external reference that the context provides people. This would not be harmful in a language learning setting. It would result in a lack of coherence in output based on a discontinuity of sense between the input and the generated output (cf. de Beaugrande and Dressler, 1994, p 84). The errors in communication result from a lack of clarity of the internal referents in the input text itself - a factor that can be used to initiate reformulation. In effect this is one aspect of the process of negotiation of meaning which is so beneficial to second language acquisition (Long, 1996, pp. 413-468).[72]

The second problem case is the reference *'It'* in sentence (4). This word does not refer to anything explicit in the text. The programme does not include any external information and the logic in its current state does not provide for searching external information should this exist. As such, the programme has no way of correctly resolving the reference. Instead, we examine the way the programme does handle this scenario as an indicator of the sort of

[72]The implementation of this particular process is not complete. The current lexical definitions do not make any distinction between singular and plural. As a result there is no means for distinguishing between the behaviour of words such as *'they'* which require plural targets and *'he'* which requires a singular target. As this omission does not significantly impact on the functionality of the programme, it has been left for a future version (the more so as the semantic component of the lexicon that is required for this logic to function fully has not been added).

error that can result from its processing.

The text defines the semantic relationship

$$\text{'}it - is \mapsto raining\text{'}.$$

The semantic dependency for 'it' is

$$it \mapsto \{cn, pn, S\}.$$

The logic tests for any word in the text which fills this dependency (i.e. a common noun or a proper noun) and has a link defined that references *'rain'*. With no semantic information it will, at the moment, fall to its optimal fit match of the noun class word in the previous sentence with the lowest weight. This turns out to be *'Tom'*. The consequence of this is that the final knowledge structure will erroneously link *'Tom'* and *'rain'*. This is not a very serious error in the context of the overall network structure in that it is a relatively low value semantic link. Similar references are grouped together additively. As such they will have significance when they occur in sufficient number but will not have a significant value when they occur in isolation.

The definitions used in the dependency relations are still evolving. This last reference error, for example, is resolved with the addition of a self-reference dependency linking to the sentence in which the reference occurs. This allows structures such as

'It is raining' and

'This is chocolate'

to be parsed correctly.

4.2.6 Grammatical errors

The use of a large lexicon, though clearly unavoidable, comes hand in hand with all the problems associated with having a large lexicon discussed earlier. In particular the analysis engine will have problems with sentences containing words already defined in the lexicon but used in a form for which it does not have an entry (as when the word "*will*" was used in section 3.2.5 while the lexicon only had an entry for its use as a future tense auxiliary). When a word is incorrectly identified this will, in most cases, cause the recursive phrase rule test algorithm to fail.

A failure, in this case, means that the algorithm is unable to find a valid sentence structure matching the rules that have been defined. This means that an error has occurred in understanding the grammatical structure of the sentence, and has one of two possible causes. Firstly, the parser may have a lexicon which fails to adequately describe the words used in the text, and secondly, the sentence itself may contain grammatical errors.

Given a reasonably large and descriptive lexicon, the dependency structures can be used to make guesses as to the types of unrecognised words and new usages of words already contained in the lexicon. This involves repeated parsing of the sentence - first using the routines as they are currently contained in the programme and, if that fails because of one of the reasons mentioned, then by analysing the sentence using only the minimal set of cues already identified. The word possibilities listed in the tree that results from this second test are then tested against the entries contained in the lexicon to find existing words with alternative entries. Re-processing sentences in this way has the unfortunate side-effect of slowing analysis considerably for these sentences (the default configuration of the current version of the application disables this facility and an alternative routine that bypasses some of the strictures governing dependency rules is used instead). In most cases the solution to the problem posed by a faulty analysis would be the sentence structure containing the most possible correct identifications. Those words in this solution sentence that are unrecognised can then be treated as new usages of existing words and entered into the lexicon as such - either on a permanent or temporary basis.

In general, if no solution structure is found or if the solution structure contains unrecognised entries this can be taken as an indication of a grammatical error or of a valid grammatical structure not covered by the generalisations used to analyse sentences. In most cases this will mean a grammatical error in the input text. For example, the ungrammatical sentence

**the dog bit cat the*

will result in missing grammatical dependencies as is illustrated in Figure 32. It is also worth noting that the unfilled dependencies in this example are at a syntactic level (unlike the errors illustrated in figures 23 on page 122 and 24 on page 122 which were at a semantic level). As a result it is possible to determine a probable thread of sense in this sentence. While this may not be the meaning the learner intended, the application will continue processing using the sense '*dog* $\rightarrow$ *bit* $\rightarrow$ *cat*'. Any response or feedback generated to this statement would thus use this sense. If the understood sense is not the learner's

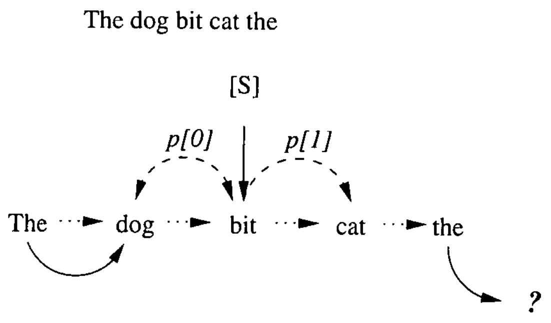


Figure 32:

intended sense, this will be seen in the misunderstanding that results. This thus becomes implicit feedback of the correctness of the statement which, as Robinson (1991, pp. 155-167) demonstrates, can be more effective than explicit feedback.

4.3 Representing relationships

The key to the viability of the application lies in the extent to which it can encapsulate the meaningful content of a text. This has to be done before any evaluation can be made of the content - either implicitly or explicitly. This process has been touched on during the discussion of the structure of the lexicon[73] as this goal heavily influenced the development of the structure used for representing words in the lexicon.

The governing factor in lexicon development was the consideration that the nodes in the lexicon would be used to construct relationship structures that would define new concepts that needed to be represented in the lexicon. Furthermore the relationships defined between nodes would need to be represented in a way that enabled them to be used, in turn, as nodes in constructing larger knowledge structures. The node definition, thus, had to be both flexible and easily extensible, capable incorporating new information very easily and quickly.

This section describes in greater detail the requirements for representing meaning and the structure of the nodal representation of some of the different types of relationship that may be encountered in a text.

[73]Discussed in section 4.2.

4.3.1 Object-driven relations

The approach to representing meaning in this application is an object-driven approach in which the subject of a sentence is viewed as the governing object. For this study, this is assumed to apply to all types of sentence, be they active, passive, compound, complex, statements or questions.

For sentences containing verbs spanning two or more words (as in the passive voice), the first verb is classed as governing and the latter verbs as object information in their own right. This allows analysis as shown below

1. *The boy threw the ball*
   - $boy \rightarrow throw \rightarrow ball$
2. *The ball was thrown by the boy*
   - $ball \rightarrow is \rightarrow throw$
   - $ball \rightarrow by \rightarrow boy$
   - $ball \rightarrow throw\ by \rightarrow boy$

It is important to note that in testing for similarity, we have no inherent interest in the meanings of the phrases. For all this, the strong correlation in meaning between the active and passive versions of the sentence means that it may be preferable to represent finite verbal relationships in terms of two links - one for each object in the sentence. In this case this would result in the following relationships

1. *The boy threw the ball*
   - $boy - throw \rightarrow ball$
   - $ball - is \rightarrow throw$

As easily, we could convert the relationships defined in the passive voice sentence to an active link representation; however, finite verbs are represented as a set of at least two relationships as this gives a more complete representation of the relationships involved and provides an easier comparison of contextual information. In a comparison between texts for contextual similarity, the similarities are easier to find in this latter representation.

Consider the two sentences below. These are similar but do not have identical content. The extent of the similarity is relatively easily determined using the passive representation of relationships.

1. *The boy threw the ball*
   - $\boxed{ball - is \dashrightarrow throw}$
   - $boy - throw \dashrightarrow ball$

2. *The ball was thrown*
   - $\boxed{ball - is \dashrightarrow throw}$

When the objects are reduced to their most basic form, however this is represented, this will provide an exact match for the object ball in "$ball - is \dashrightarrow throw$". The data referring to boy will not match as this is absent in the second set. If the passive sentence had contained the continuation "*by the boy*" then there would have been a linking relationship between boy and ball, even though the form of the link is not identical which would indicate a strong similarity, though again not an identical similarity. If we assign a very rough scoring system - a two column table for matches and misses and assigning one for every match or miss and half for a near miss, then the first example would have one match and one miss, a score of 50%. The second would have one match and a near miss, a score of 75%.

It should be noted that at no point in this comparison has it been necessary to refer at all to the meaning of the text or objects. Granted, the objects are the same in both texts. Nonetheless, it should be possible to use link relationships to provide a basis for comparing cases involving different objects that are not identical without ever having to explicitly compare the objects or the texts in terms of the meaning of these texts. Implicitly comparing nodal relationships can be said to be a comparison of meaning, but only to the extent that the nodal representation of the relationships in the texts can be said to represent the deep structure of the sentences in the texts.

It should be noted that it is not the intent of this application to detect usage errors of the sort included in artificial and intentionally erroneous text as would be formed by a sentence like

*"*the boy was thrown by the ball*".

Instead the application attempts to construct a sense network regardless of the acceptability of the text, with the following result:

1. * *The boy was thrown by the ball.*
   - $boy - is \dashrightarrow throw$
   - $boy - by \dashrightarrow ball$
   - $boy_{thrown} - by \dashrightarrow ball$

The errors that ungrammatical text would cause in the sense network would show up as misunderstandings which would, in turn, require identification and repair by the student - a potentially beneficial factor for use in the L2 classroom.

4.3.2 Passive relations

If this process of simplification is extended to its logical conclusion, then it has some interesting practical developments, especially when this is combined with the representation of meaning as proposed in section 4.3.1.

In English, many of the various movement rules have been proposed to handle common grammatical features such as passive voice and wh-movement. These transformations are probably not necessary for interpreting and generating sentences if one uses a model governed by relationships between objects. In this model the surface structure of a text is used as a strong indicator of the relationships between objects in a sentence. Passive voice and wh-questions simply define alternative relationships between objects. Sentence analysis is thus not a matter of application of grammatical rules, but rather of determination of objects and identification of their relationships. The governing concept is how these objects are defined. A proper definition, then, would be one that ensures the best encapsulation of meaning. For example, in terms of the link-structure described in section 4.3.1 there is no difference in grammatical representation between the sentences

The ball was thrown.

and

The ball was hard.

Both of these define links to the object *'ball'*. Even though the first is a passive sentence and the second a copula, the two sentences both add information about the properties of the ball. The first adds information about its state (it is moving) and the second about its structure (it is hard). In other words, even though we identify two distinct grammatical structures (passive sentence and copula) the function of these sentences is the same when we consider the relationships that they define. The difference in perceived grammatical structure does not exist because of a difference in the use of the link ('is') but because of a difference in the target node (a verb target such as the past participle 'thrown' includes implicit relationships, such as "who threw the ball").

Any additional phrases that extend these sentences can be treated in the same way, as in these sentences.

The ball was thrown by Sam.
The ball was hard as nails.
The light was green by default.

In each of these, the prepositional phrases are properties of the object nearest it as well as the head object of the sentence yielding structures such as

1. *The ball was thrown by Sam.*

    - $ball - is \dashrightarrow thrown$
    - $thrown - by \dashrightarrow Sam$
    - $ball_{thrown} - by \dashrightarrow Sam$

2. *The ball was hard as nails.*

    - $ball - is \dashrightarrow hard$
    - $hard - as \dashrightarrow nails$
    - $ball_{hard} - as \dashrightarrow nails$

3. *The light was green by default.*

    - $light - is \dashrightarrow green$
    - $green - by \dashrightarrow default$
    - $light_{green} - by \dashrightarrow default$

It follows then that it is not necessary to have a distinct structure to handle passive sentence forms. They use the same set of dependency structures to determine the inter-relationships as do copulative verbs. The differences in meaning are defined largely by any extra dependencies of the target nodes.

4.3.3 "Wh"-question forms

The same principles observed in section 4.3.2 also apply to other structural variations in grammar. "Wh" forms have been used as an indicator of the necessity of movement rules in grammar formulations. These are not necessary, though, in a determination of the relationships between the objects in a sentence. Consider the question

- *"Who threw the ball?"*

This is analysed using the same structures as were used in sections 4.3.1 and 4.3.2 to give the following breakdown:

1. *"Who threw the ball?"*
    - $who - throw \dashrightarrow ball$
    - $ball - is \dashrightarrow throw$

The "wh" words such as "*who*" are references which means that, like other references, they are defined with an extra set of dependencies indicating they refer to objects potentially outside the sentence. These would then be used to provide a target for the reference implied by "*who*".[74]

4.4 Defining similarity

One of the key questions that determine the viability of this project is the question of what we are prepared to define as equivalent texts.[75] Language provides a number of ways of expressing the same set of information.

This is more than just a problem of identifying synonyms. Context, for example, plays an enormous role, not least because context often provides a large body of assumed information that can indicate similarity between otherwise different statements. Similarly many concepts can be similar from one perspective but totally different from another.

This section describes some of the problems associated with determining similarity and the solutions used in this application.

[74] Further discussion of references and methods of handling them are provided in sections 4.4.5 and 4.2.5.

[75] Some problems affecting comparison are examined in section 4.3.1.

4.4.1 Degrees of similarity

When examining texts to determine the criteria for similarity, it soon became clear that there were a number of types and degrees of similarity. Much of similarity is also dependent on context. This problem can be illustrated in a rather simplistic though very real manner by these two sentences:

The boy stood in the corner.

The boy sat in the corner.

These statements can be said to be quite similar. They use the same concepts ("*boy*" and "*corner*") and their verb links both indicate a positional relationship. If the similarity tests the answer to a question such as

Where was the boy?

then the two statements can be said to be similar in that they have the same linkage, linking "*boy*" to "*corner*". If, on the other hand, the similarity tests the answer to a question

What was the boy doing?

the sentences cannot be regarded as similar as that they contain verbs or vectors indicating different activities. This can be complicated further by comparing the sentences to other possible responses that convey similar information, such as:

The boy was in the corner.

This sentence also has the contextual linkages linking "*boy*" and "*corner*"; however, the vector does not indicate an activity, which means that it will have the same similarity relationship to the first question as the first two sentences. On the other hand, it does not in any way answer the second question. In other words, while it is not similar to the first two sentences in any way (it does not indicate an action), it is also not dissimilar to either of the first two sentences in that the statement in this example can allow either action from the first two sentences.

Another type of problem indicating a contextual requirement of the base lexicon is captured in the sentences:

The boy was sitting in the corner.

and

The boy was seated in the corner.

These sentences are to all intents and purposes perfect matches for the second sentence. This will not be recognised, however, unless the application has a record in its database indicating that the vectors "*sat*", "*was sitting*" and "*was seated*" are equivalent.

Consequently a hierarchical arrangement of information is necessary[76] in the lexicon to allow at least minimal recognition of equivalence between concepts when the similarity is not obvious (that is, when they are not explicitly given the same name in two different texts.) For example, the sentence

The child sat in the corner

could answer either question accurately, but the similarity to the second sentence would not be fully apparent unless the application has some record indicating that "*boy*" and "*child*" are equivalent. In longer texts this sort of similarity is more likely to be described in the text itself, even while the number of concepts that are unique also increases, but this is not something that can be relied on to determine similarity.

4.4.2 Determining similarity

The examples in 4.4.1 suggest that there are three parts to determining similarity in a sentence - similarity in subject, vector and target. If all of these are similar in two statements then we can assume that the two statements are synonymous.

Determining the similarity of two sentences then becomes an issue of determining the similarity of the objects in that sentence. This requires defining a set of relationships between words that can be used computationally to determine the extent of similarity between objects. For example, to process the examples used in section 4.4.1, we could define an explicit relationship between the concepts "*child*" and "*boy*" indicating that these two words are similar or synonymous. This simple definition of similarity has a problem, though, when adding a definition of similarity for "*child*" and "*girl*". This would not be a problem because these definitions are incorrect, but because they allow the logic of $boy \mapsto child \cup girl \mapsto child \therefore boy = girl$. Obviously "*child*" and "*boy*" are only similar in

[76]The approach adopted in this application is described in section 4.4.2.

certain contexts. In this case girls and boys are similar in the context of being children but different in the context of sex.

The solution adopted by traditional AI systems (discussed in section 2.3.3) is to use a number of types of relationship. These define conditions of similarity in different contexts. For example, a parent-child relationship can be defined for objects enabling structures to be defined. ("*Leaves*", "*branches*" and "*roots*" are children of "*trees*", "*children*", "*parents*", "*mom*", "*dad*", "*boys*" and "*girls*" are children of "*families*" and "*people*", and so on.) Another relationship could indicate antonyms, or related concepts (items like "*bat*", "*ball*" and "*pitch*" are all related to the concept "*cricket*"). These additional relationships allow a degree of reasoning, hence their use in AI and NLP systems.

There are, however, inherent problems with these predefined relationships. Each relationship type, for example, effectively identifies a context for similarity or difference. Inevitably this restricts the logic of the AI system to functioning within the contexts defined by these relationships as it will not have a basis for comparing objects that fall outside of these relationships. Even though AI systems are designed with fairly broad and descriptive sets of relationships, a finite set of relationships will not be able to cope with the extreme flexibility and variance possible in language. Lexicon construction also becomes a concern as these relationships have to be defined for each word in the lexicon and a chain of relationships will have to exist linking each word to other words in the lexicon if this is to provide any basis for comparison. By extrapolation, therefore, we can deduce that no taxonomy will be complete enough to represent all relationships possible in language without becoming similar in complexity to the language it tries to represent.

Given the scope of this application, it did not seem practical to attempt a system of predefined relationships. The approach adopted here uses a dynamic system. "*Girl*" and "*boy*" share many of the same link structures (*is* $\mapsto$ *person*, *is* $\mapsto$ *child*, etc) so the application tests for similarities in the references while placing extra emphasis on the contexts defined in a text to determine similarity. "*Girl*" and "*boy*", for example, would have different link structures for sex, so these would have greater weight in a context which includes reference to sex. Using a deductive method of determining similarity such as this one means it should not be necessary to have a taxonomy specifying explicit relationships between objects. What is more, the actual reasoning involved is quite elementary - a simple comparison yielding a calculated representation of similarity - so it should not affect processing speed detrimentally.[77]

[77]This is almost the exact reverse of the approach adopted in the WordNet project (`http://www.cogsci.`

By not defining these links, the lexical structure is allowed optimal flexibility in representing new relationships and in determining relationships in new data - it is not necessary to have separate definitions for synonym, antonym, or any other relationship between words as this can be determined comparing the link structures of the words. This flexibility does, however, carry a cost. The cost is twofold - it involves more processing than would a system with a finite set of predefined relationships and, secondly, this cannot work unless the lexicon contains an extensive set of predefined links from which it can determine relationships such as synonymity.[78]

In this way the majority of similarities in meanings are defined by similarities in the relationships linking similar target nodes. These in turn can be divided into two categories. These are relationships defined in the lexicon (a pre-determined, static and unchanging world-view) and relationships defined at run-time (derived from an input text). In a very simplistic sense these can be viewed as analogous to long- and short-term memory respectively.

4.4.3 Lexical requirements for similarity

Unlike an information retrieval system (such as Brainhat, described in section 2.3.3), the reasoning ability needed in this application is minimal. It only aims to compare texts. This allows us considerably to simplify the link structure maintained in the database. For now, the links will be defined as:

1. **TYPE**: a phrase type indicator. This provides a means of identifying the targets of syntactic and phrasal dependencies and provides a method of specifying these.

2. **STEM**: stem of words defining the concept. At least one of these will be needed for every type that is defined for the word, though more will be allowed. (When represented in the database this means that this and the previous type can be combined.) Unless explicitly defined, either as rules for formation or as an explicit relationship, the application would not 'know' the similarity between these different forms (as in

princeton.edu/~wn/), an online lexical reference system. In WordNet the most important relation is similarity of meaning (Miller et al., 1990, p. 241). Cross-references between the contexts defined for each word in the lexicon are used to correctly determine the sense of the word, and from this return its meaning. WordNet now boasts a large and detailed database, but it has been a slow development as the WordNet source files are all written by lexicographers and are the product of a detailed rational analysis of lexical semantics (Beckwith and Miller, 1990, p. 304).

[78]The implementation of this is covered in more detail in section 5.4.4.1 on page 176.

"*sit*", "*sat*", "*sitting*", "*seated*", "*seating*" and "*seat*"). This list would have to be maintained independently for each distinct usage (part-of-speech) of a word. "*Seat*", for example, is both verb and noun. As a noun it would be synonymous with words such as "*chair*" and "*bench*" rather than with verbs. This became a necessity because of the shortage of training data for the application. As more texts are used to provide training data for lexicon building, it is hoped that the similarities of different forms of a word will be determined using the same logic structures that determine similarities between other objects.

3. **LINK**: every concept linked to this word will be linked by a dynamically maintained list of verbal vectors and their associated weights in the form of an array containing three elements, the vector, the weight and the target object.

As a minor addition to the lexicon intended to make recognition of inter-concept links more viable and to simplify the comparison as far as possible, this version of the application stores all short-term verb vectors as links to and from the stem (root) concepts of words where these have been identified in the lexicon. This increases the chance of a match but is not a substitute for better representation of sense in the lexicon.

Another feature of this lexicon structure is that link relationships have two types of entry in the relationship network. The first of these uses the LINK field in a node entry to record the relationship between subject, vector and target. This does not, however, only specify a relationship between two concepts. It also describes a process of building a relationship. This relationship is in turn added to the relationship network as an individual node in its own right. So, for example, if we have a simple sentence such as

The man is old.

The sentence defines a relationship $man \rightarrow is \rightarrow old$. This is added to the LINK field of the node '*man*'and is also used to create a new node in the lexicon named $'man\ is\ old'$. In this way entire relationships can become the origins or targets of further relationships. In other words, a sentence like

The old man walked home.

could be represented visually as

$$\left.\begin{array}{c} \textit{man} \\ \downarrow \\ \textit{is} \\ \downarrow \\ \textit{old} \end{array}\right\} \longrightarrow \textit{walk} \rightarrow \textit{home}$$

The link structure also allows easy representation of questions. This is particularly important as questions are processed using exactly the same rules as statements. Consequently, a question such as

Is the man old?

will define two potential senses for the sentence as there are two potential targets for the vector '*is*' (*is* $\rightarrow$ *man* and *is* $\rightarrow$ *old*). What might at first seem a complicating factor - the missing origin for the vector - becomes our saviour in representing this question in the network. In this implementation a missing origin is represented using unnamed node (represented in the network as $'?Q'$). This leads to the following node structure for this sentence:

$$\begin{tikzcd} & is \arrow[r] & man \\ ?Q \arrow[ur] \arrow[dr] & & \\ & is \arrow[r] & old \end{tikzcd}$$

Lastly, representing relationships as nodes allows for complex concepts to be defined at lexical level. An idiom such as 'to kick the bucket' can be represented as the node *'kick bucket'* with the concept '*die*' as its stem. A simple test during the final phase of sense generation tests the lexicon for structures of this type and incorporates the sense from these in the developing network.[79]

A similar principle allows compound concepts to be represented as single nodes. An example of this would be the concept "dog show". In determining grammatical structure two adjacent nouns can define a compound or indicate the start of a new phrase (as is the

[79]see the file '*genSense.pm*' for the implementation of this.

case in the sentence "is Toni a Taurus?"). As a result processing forks at this point to allow for both possibilities and generates one syntax tree for the concept as a compound and one for the concept as a new phrase (only those syntax trees that satisfy all the dependency requirements of a sentence and the concepts in the sentence are retained for use in building *sense*). When building the relationship network compounds are combined to form a single node and the application lexicon is checked to see if a compound of that form already exists. If it does exist information from the lexicon is incorporated in the new node definition in the relationship network.[80]

4.4.4 Multi-sentence texts

Texts containing more than one sentence are harder to compare in some senses and easier in others. Seeing as we are trying to determine similarity, the problem of determining similarity of vectors is increasingly unnecessary. In longer texts the information is built up in the relationships between sentences and the concepts in those sentences.

This view of texts borrows heavily from Rhetorical Structure Theory (RST) which describes the structure of written monologues. One of the most basic assumptions of RST is that coherence can be modelled by means of named relations which hold between adjacent text units. Such relations can be used to structure texts by iteratively applying relations thereby composing complex text units out of smaller ones*(Fischer et al., 1994, p. 210).*

In this implementation, a network is formed in which concepts are related to each other. The weights that are generated while determining sense threads now serve a different purpose.[81] The weights of all the relationships that make up a node are added together. When compared to the other potential nodes from that sentence this provides an indication of the importance of the node in the sentence. If the node is a reference we can now add this node weight to the weight of the target of the reference (a weight that will have been determined earlier when processing the sentence containing the target of the reference).

If the network of relationships generated in the sentence containing the reference is also incorporated in that of the target of the reference we not only have an indication of the importance of the concept in the text as a whole (from its accumulated weight) but also an indication of the sense of this concept in the text (from the network of relationships that

[80]In the case of "dog show" this leads to a node labelled *'dog show'*. A planned improvement in the lexicon, however, will require these to be labelled as *'dog + show'*. This will facilitate comparisons involving vector relationships between these two concepts.

[81]This weighting system and its implementation are described in section 4.2.3 on page 116.

have been constructed around this concept). A fair degree of similarity can be determined by comparing two concept lists to see if they contain a similar list of concepts, followed by an analysis of the significant differences in the texts.[82]

4.4.5 Links between sentences

As with RST, the textual model employed in this application assumes that

> *in general, a relation imposes an asymmetric structure on the connected text units. For a given pair of related text units, the so-called nucleus corresponds to the unit which contains highly relevant information, while the satellite carries less significant information; the satellite can be either substituted or left out without significantly changing the overall meaning of the discourse.(Fischer et al., 1994, p. 210)*

This assumes that, generally, paragraphs are constructed around a primary topic. The secondary topics add information in some way relevant to the primary topic. Consequently monologues employ a number of techniques for relating different parts of the text to the primary topic.

Some of these links are formed explicitly, for example, by restating concepts (the sentence subjects) or by referring to these concepts using one of a fairly small set of reference words (words like "*these*", "*they*", "*he*", "*she*" and "*it*" which contain object semantic dependencies). Most of these references occur in adjacent sentences, though this is not always the case. Usually, when references are made to concepts in non-adjacent sentences, this will be indicated by a word such as "*similarly*", "*firstly*", "*secondly*", "*but*", "*however*" and "*therefore*". These words have semantic dependencies which refer to entire sentences.

These inter-sentence links makes it possible to represent logical structures in sentences and paragraphs in a single data-structure. The node structure described in section 4.2.2 allows more than just a description of relationships inside a sentence. As the relationships that make up a node are defined as relationships between nodes it is possible to incorporate each node in a larger relationship network that describes the relationships between nodes in different sentences. Ultimately an entire text can be represented as a network of relationships between nodes.[83]

[82]This is described in more detail in section 5.4 on page 168.

[83]Section 5.3 provides a practical illustration of this process.

These larger relationship networks provide a means of testing the similarity of larger text segments (such as paragraphs). Given two such structures covering two separate paragraphs, it is possible to compare these and generate a list of similarities and differences in the two structures. If these relationship networks reflect the content of the paragraphs reasonably accurately then the list of similarities and differences in the relationship structures can provide a strong indication of the extent to which the two paragraphs as similar or different.

The process of comparison works on a simple principle. If the nodes in a network are regarded as a set (in a mathematical sense) then the intersection of the two sets of nodes will contain the nodes shared in both networks. Likewise, the difference of the two sets will contain those nodes that are not in one or other of the two networks. A measure of similarity can then be made by calculating the proportion of nodes in the intersection set in relation to the number of nodes in one or other of the original sets.

The practical application of this is not this simple. Similarity can be perceived in a number of ways, as was described in section 4.4.2. Nodes can be similar in some ways and different in others, making it hard to make a definitive judgement of their similarity. Because of this alternative algorithms have been implemented to allow different levels of similarity in the intersection set that describes the shared content of the texts.[84]

The intersection and difference node sets can be used to provide a range of feedback:

- Questions can be generated to expand user input to cover areas not included in the intersection.
- Confirmatory responses can be generated to positively reinforce the validity of responses within the intersection.
- Relationships defined in the network of the model text but not in the learner's structure can be used as a source for generating answers to questions or as new information which can enhance the learners interaction with the text.

The precise method of responding to comparisons depends very much on the nature of the application that uses this type of system. This is intended to provide a means of assessing multi-sentence texts but can easily be adapted to handle shorter texts such as those that would be found in conversation.

[84]The process of comparison is described, with practical examples, in section 5.4.

A network representing the relationships in a dialogue is built up in the same way as a network representing the relationships in a paragraph. The only difference in building the network is that it is built one turn of talk at a time. The most recent turns would define relationships with a high priority - so responses based on high-priority relationships in the network would tend to be generated using references to the most recent statements (as the priorities are built additively, occasions will occur when previously defined relationships will have a higher priority).

Greeting-greeting and closing-closing sequences can be defined in terms of semantic dependencies, leading to an automatic use of these conversational patterns based on the high priority defined for these dependency relationships. However, with the exception of sequences with clear verbal cues like these, the application will tend to follow an initiation-response pattern. The degree to which the system can make initiations is dependent on the particular implementation and the degree to which the implementation is intended to actively acquire information. Even then initiations will probably have the artificial quality apparent in chatbot implementations.

4.5 Limitations

The implementation of dependency based grammar in this application imposes its own limitations on the nature of the texts that it can process. There are many features of written language which have not been considered in the design process. In some cases written language employs conventions which do not affect the underlying grammatical structure. One such feature is direct speech which allows a speaker's statement to be split in a number of ways as shown below:

Tom said: "Happy birthday. I've brought you a gift."

"Happy birthday. I've brought you a gift," said Tom.

"Happy birthday," said Tom, "I've brought you a gift."

In these three sentences the direct speech forms a text within the context of a larger text. As a result this text form is one of those not supported in this version of the programme.

Many of the symbols used in writing do not play a significant role in determining the content of the message. Consequently, punctuation marks are generally ignored in processing texts. Apart from those symbols that indicate the ends of sentences, the only punctuation mark that has any functionality associated with it in the application is the comma. As commas are treated as a form of conjunction this functionality is very basic and not intended to represent the proper use of commas in any real way. To allow some processing of texts containing commas these have been classified as a variation of conjunction. This allows correct processing of sentences such as:

Tom, Dick and Harry play soccer.

The officer in charge, his partner and the suspect drove away.

In each of the sentences above the comma can be interpreted as taking the place of the conjunction 'and'. This incomplete implementation does not take into consideration the other uses of commas (such as to indicate extra information in the sentence). Sentences such as

Fred, an elderly gentleman, fell asleep during the movie.

and

Firstly, let me introduce myself.

Are not processed correctly at present. The application tries to process these as conjunctions, and if that fails, ignores the comma and tries to process the sentences as if it did not exist.

This has meant that the texts used in the examples in section D have been edited slightly when punctuation marks are encountered which have not been allowed for in the programming. Editing has, however, been kept to a minimum in these texts - even to the point of including a elements that are not supported in the application (such as the use of parenthesis). Many of the spelling and grammatical errors in the texts have also been retained.[85] The texts which are not discussed in section D have not been edited. These generally take longer to process as editing inevitably simplified sentence structures slightly. Relationship networks for these texts (generated by the application) can be found in the directory `eg-output` on the companion CD.

A design goal in the implementation of the system has been to use the same dependency criteria to evaluate all sentence types. This means that the resulting networks will be less accurate in sentence types that fall outside the range of sentences the application is designed

[85]The original versions of texts that have been edited can be found in the `eg-texts\originals` directory.

to handle. Nevertheless, as will be demonstrated in the next section, the range of sentences and texts that can be processed is large. Even at this early stage of testing the range is large enough to make the programme usable as an aid in evaluating texts.

5 Implementation

The rule sets presented here are all derived from the dependency-based view of sentence structure developed in section 3. This view was chosen for two primary reasons. The single most important criterion, given the intent of the programme, is that the sentence analysis should also represent an analysis of the *sense* of the sentence. Dependency structures can be defined in such a way that the sentence analysis which fulfils the dependencies also provides an outline of the *sense* of a sentence (cf. sections 2.2.4 on page 40, 4.2.3 on page 116 and 4.2.4 on page 124). The second reason is that dependency criteria are easier to represent computationally than are the criteria for phrase construction as dependency operations function solely at the surface level of text. Dependency relations do not require mapping the surface structure analysis onto a lower level of representation.

5.1 The process of analysis

The application uses a bottom-up, procedural model of processing texts. In this approach each sentence in a text is processed in sequence. In processing a sentence each word in the sentence is also processed in sequence. This operation of processing a text can be regarded as being composed of three phases:

1. The first phase is a lexical lookup. This provides all possible types of a word identified in the lexicon. Should the word not be contained in the lexicon a subset of primary word-types is returned after being processed for suffix clues.

2. Each type returned by the lexical lookup is tested recursively using grammatical dependency information to eliminate all obviously invalid grammatical combinations of the particular word sequence of the sentence.

3. The grammatical structures are processed in turn, testing for syntactic dependencies. The syntactic dependencies are used to generate *sense threads* - each sense thread being a possible, unique, interpretation of the *sense* of the sentence.[86]

These processes will be examined in more detail during the course of this section.

[86]Because the lexicon only includes parts-of-speech information at present, a number of threads are often possible from the combinations of parts-of-speech.

5.1.1 Overview of primary processing events

1. The most basic level of processing is that which separates a text into its constituent words and sentences.[87]

   In this implementation most punctuation is processed solely as sentence end markers. This is true for exclamation marks, question marks, full stops, semi-colons, colons and commas. This means that text structures such as direct speech and sentences containing information in parenthesis are not supported.

   Each sentence will be processed separately to determine the potential grammatical sentence structures that can be formed of the particular words and sequence contained in the sentence.

2. The first phase of processing a sentence finding those words defined in the lexicon. If a word is contained in the lexicon then information for each part-of-speech of that word in the lexicon is returned. If the word is not contained in the lexicon a primitive guess is made at the possible types open to that word.[88] This guess is based on the process outlined in the lexicon-free trial described in section 3.2.3.

3. All possible combinations of the types identified are now tested for a limited degree of grammaticality in a recursive routine. This test returns true for structures in which all the grammatical dependencies of the words in the sentence are met.[89]

   In a conceptual sense, the manner in which conjunctions are handled is not ideal in that conjunction dependencies are not defined in the lexicon itself. Instead these are determined from an examination of the adjacent preceding word and any preceding words which have filled dependencies.

   Conditions are regularly encountered that allow for a number of possible structures, for example, compound noun and verb phrases or conjunctions for which more than one potential left dependency exists. In this, new processes are started, one for each possible structure.

[87]The process described here assumes that input is from a plain text file containing one or more paragraphs. An interactive mode also exists which allows the user to input individual sentences and view the resulting network.

[88]The routines governing this process are defined in the file *sentence.pm*

[89]The default dependencies are defined in the file *rules.pm* and the implementation of the logic that resolves the dependencies in the file *genTree.pm*. A few words (such as '*do*') have additional dependency information specific to that word defined in the lexicon in the file *lexicon.pm*.

4. Each valid grammatical structure is, in turn, tested to determine if it contains a valid thread of sense. A valid thread is defined as a sentence in which at least one semantic dependency is met for all the objects (noun class phrases) in the sentence. As with determining grammatical dependencies, this is a recursive process which tests all possible threads of sense for the existence of valid threads.

   If no valid grammatical structure exists, or no valid thread of sense can be found, then the failed structures are processed to construct a basic sense structure.

   Each sense thread represents a possible semantic interpretation of the words in the sentence. In other words, each thread defines one (and only one) set of relationships based on the words in the sentence.[90]

5. Each thread is now converted into a representation of the relationships in the sentence. Each thread is assigned a weight (based on the number of threads that exist for a sentence).

   All possible representations are furthermore merged additively to create a network representing all possible relationships in the sentence (additively in this sense means that the new node contains the relationships defined in each node being merged and that the weights of each node are added together as well).

   In the final structure the nodes with the highest weight should represent the primary interpretation of the sentence. This also, however, allows for ambiguity in interpretation.[91]

6. Once a sense network has been defined for each sentence in the text an attempt can be made to resolve anaphoric and cataphoric references in the text. A best fit routine searches through the text assigning weights to potential targets of the reference based on their proximity to the reference and the degree to which they share defined relationships (as no relationships are defined in this version of the lexicon, this implementation will only correctly identify references where these relationships are defined in the text).[92]

7. The last part of this process is, once again, a merge of sense networks. This time the sense networks of each sentence are merged additively to generate a network

[90]The functions governing this process are contained in the file *genThread.pm*

[91]This is implemented in the file *genSense.pm*

[92]This is defined in the file *refs.pm*

representing all the relationships defined in the text.

A consequence of this model is that frequently referenced objects have high weights while objects with very few instances in the text or few references will have low weights. The weight of the node thus becomes an indicator of the importance of the node in the context of the text.

8. This network is sent to the standard output where it can be redirected to either a file or another programme.

In this implementation, the resulting network contains nodes representing both the primary objects in the text and nodes representing the compounds formed by the links between these objects. These compounds will regularly be unique to a particular potential sense thread in a sentence and will thus have a very low weight. Despite this, they are included in the network in order to provide a means of distinguishing between different interpretations in a sentence. A preferred interpretation will form a particular construction more often than a less likely interpretation, resulting in more instances of the preferred construction before a merge operation takes place. As the merge operation combines the weights of these additively, this will cause a final construct for the preferred construction with a higher weight than that of less likely constructions.

5.2 Considerations of well-formedness

As is described in sections 4.2.3 on page 116 and 4.2.6 on page 131, the application attempts to assign meaning to ungrammatical sentences in the same way that it assigns meaning to grammatically accurate texts. Unfilled dependencies, and the type of dependency that is unfilled, do, however, provide an indication of the type of error a learner has made and can be used as a basis for a determination of the seriousness of the error. For example, in the sentence

**Tom slept like baby*

the missing determiner (causing a sequencing dependency for '*like*' to be unsatisfied, illustrated in Figure 33) does not affect the comprehensibility of the statement. In this case the application is able to determine a valid sense thread as the dependencies used to determine sense are unaffected by the error. In this way the current format of the application concentrates on building *sense* structures from text rather than grammatical accuracy. A report

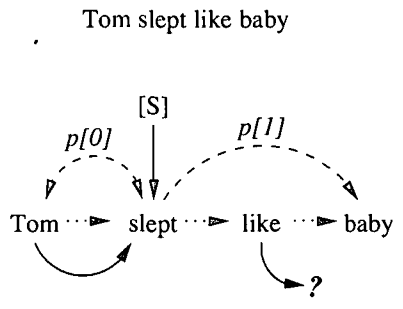


Figure 33:

of unfilled dependencies can be used to provide some indication of grammatical accuracy and of the degree to which a particular error is likely to affect comprehension. This report does not include all errors. To improve flexibility in determining sense relationships, the grammar used allows a range of input structures. As a result some errors will go unnoticed as they require a more restrictive set of dependencies and do not affect the process of generating a relationship network from the text.

Unlike the minor error[93] illustrated in Figure 33 the errors illustrated in figures 23 on page 122 and 24 on page 122 do affect comprehension as these cause an inability to create a relationship network that captures all the information contained in the sentence. For the purposes of this dissertation, the application is required to attempt to build *sense* structures even when no grammatical analysis of a sentence exists in order to have a basis for comparing the content of two texts. Here the grammatical structure is used primarily to eliminate obvious non-interpretations from its *sense* processing routines. (It is possible to generate output describing the reasons for the application's inability to process a particular sentence[94] though this will require an understanding of the workings of the application's internal grammar to be useful.)

[93]Minor errors in this sense are errors that still allow the programme to build an appropriate sense structure for the sentence. These include things like missing determiners and incorrect tenses. More serious errors are errors that prevent the application from building a representation of sense. These include things like incorrect word order and incorrect usage of a word (such as when a word that is identified in the application lexicon as a verb is used as a noun, discussed in section 5.3.2).

[94]In *rules.pm* change the $::VERBOSEPARSEFAIL line to one of:

```
$::VERBOSEPARSEFAIL = 1;
$::VERBOSEPARSEFAIL = 2;
```

5.3 Examples

In each of these examples very small extracts from the start essays written by first-year students at the University of the Free-State have been used. This is to enable demonstrating the data encapsulated in the relationship network. As the relationship network for a complete essay can be quite large, it is not practical to show this operation in detail here. To illustrate the process, however, summaries from the results of processing four complete texts are provided in appendix D.

A number of potentially unnecessary relationships are defined in the sense structure. The reason for this is that the network is twofold but based on a simple concept - the network is intended to be a structure used in comparing texts. The first reason, then, is that we wanted a method of representation which would allow both nodes and the relationships to be accessed in exactly the same way. In other words, the relationships defined for a node should create a structure that can be used in turn as part of later relationship structures. This means that the structure composed of the relationships between nodes must be able to be used as a node in its own right. The second reason was simply that allowing a number of structures to be defined for a sentence reduces the possibility of missing similarities between texts which will inevitably have structural differences.

The significance of a relationship in a sentence (a calculation based on its proximity to the primary subject and verb and the complexity of linkage necessary to achieve the relationship) is used to filter unlikely relationships from the generated network. This means that if a more restrictive set of relationships is required, for example illustrating only those relationships directly defined by a verb or preposition, this can be achieved simply by increasing the value used to filter unlikely structures from the network.

5.3.1 Processing a sentence

To start with let us examine the results from processing a single sentence (from the learner text shown in appendix D.1) which reads:

Xenophobia is the hatred or fear of foreigners.

If we process this sentence the programme returns a network containing 11 nodes which can be diagrammed approximately as illustrated in Figure 34. Part of the reason for the number of nodes (not all of which are obvious in the diagram) and also the reason that this illustration can only be an approximation is that each of the relationships defined in the network

Xenophobia is the hatred or fear of foreigners

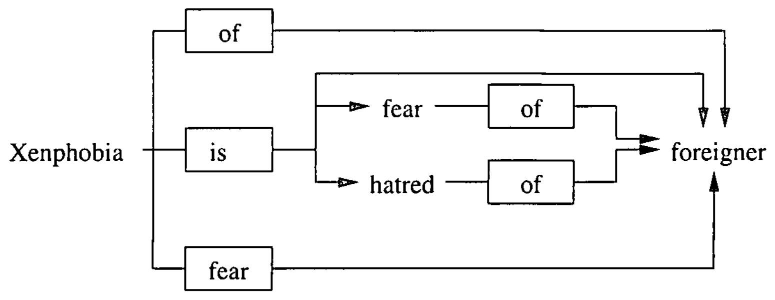


Figure 34:

is also represented as a node in its own right. This is accomplished by creating a node definition for each element in the relationship. For a relationship defined by a simple sentence (consisting of an origin, vector and target[95]) this results in a node *'origin vector target'*. In the above example this means that when the network defines an *'is'* relationship between *'xenophobia'* and *'hatred'* it also creates a node *'xenophobia is hatred'*. Each node is also associated with a weight indicating the likelihood that this structure is intended in the sentence. This is calculated using very simple principles. The programme generates a set of all potential interpretations of a sentence (taking into consideration the particular dependency definitions of the words in the sentence). Each possible combination of the concepts and vectors in the sentence is represented in a unique structure (this can be a very large structure if the sentence includes a number of vectors, conjunctions or words which have a number of possible types). Each of these interpretations is given the same weight - such that the sum of all the weights of all the sentences equals 100. So, for example, in a sentence with 10 interpretations each interpretation scores 10, and in a sentence with 5 interpretations each interpretation scores 20. Each interpretation represents a different arrangement of the relationships inside the sentence - however the same relationship structure will often be present in a number of interpretations. The weights for each of these are added together to give the accumulated weight for a particular relationship of node in the sentence. This makes a score of 100 the logical maximum that a given relationship can score in a sentence. A

[95]The terms origin, vector and target are used rather than more traditional grammatical terms (such as subject, linking verb, subject complement in this example) to emphasise that the application and grammar use the same processes regardless of the particular nature of the vector. This is particularly important as the vector class is a broader category than the traditional verb as it includes both verbs and preposition.

score of 100 would also indicate that this was the only legitimate sense for that concept in a particular sentence.

The nodes generated for the sentence illustrated in Figure 34 are shown in Table 1 along

Node	Weight
Xenophobia	100
fear	54
hatred	54
foreigner	100
fear of foreigner	23
hatred of foreigner	23
xenophobia fear foreigner	31
xenophobia of foreigner	54
xenophobia is hatred	38
xenophobia is fear	46
xenophobia is foreigner	15

Table 1:

with the accumulated weight for each node. Because the relationships between nodes are also represented as nodes in the relationship it is difficult to represent these weights in the diagrams depicting these relationships. As a result, the fact that the relationship shown in Figure 34 for '*xenophobia is foreigner*' has a very low weight cannot be shown. In spite of this, these weights form an integral part of the relationship network and fundamental to the performance of the application as a whole.[96]

Using exactly the same format for representing both nodes and the relationships between the nodes allows complex concepts to be represented as simple nodes. These relationships can then be used as building blocks in constructing further relationships. In addition this provides a means of representing idiomatic expressions such as '*kicked the bucket*' which would be represented by a node '*kicked bucket*' and objects consisting of compounds (such as '*town hall*' and '*Alexander the Great*').

Each of these can then be located in the lexicon without the need for preprocessing to determine if any of the words in the sentence is part of one of these structures. In the case of '*Alexander*' this means both the text '*Alexander the Great*' and '*Alexander called*

[96]As nodes also represent relationships this also means that these diagrams can only represent the first level of relationship practically. Arrows cannot conveniently capture level of relationships which occur when previously defined relationships are in turn related to other concepts and relationships (although an attempt is made in Figure 46 on page 212, appendix D.2).

"Great"' have a node *'alexander great'* defined. For the first of these the compound is created automatically (a potential sense thread is defined for any adjacent noun phrase and discarded later if it does not yield a legitimate grammatical tree). In the second, the two concepts '*Alexander*' and '*Great*' are connected by a sense thread formed by the vector "*called*."[97]

5.3.2 Samples with errors

Errors are an inevitable part of the input and this application attempts to function in spite of these errors. To illustrate this, a few errors will be introduced to the sentence analysed in section 5.3.1.

The original sentence reads:

Xenophobia is the hatred or fear of foreigners.

The errors examined could be classed as typographical errors but illustrate a few key problems in processing sentences with errors and demonstrate the potential of a dependency based approach for processing sentences with errors. The examples described here will cover:

1. **Xenophobia is the hatred or of foreigners*
2. **Xenophobia is the hated or fear of foreigners.*
3. *Xenophobia aaa the hatred or fear of foreigners.*
4. *aaa is the hatred or fear of foreigners.*
5. *Xenophobia is the aaa or fear of foreigners*
6. *Xenophobia is the hatred or aaa of foreigners*

(1) In this sentence one of the words of the sentence has been omitted. The grammar used does not allow words to be left out of a sentence arbitrarily.[98] This makes this sort of error difficult to allow for. In operation the programme will be left with words with unfilled dependencies. The word '*or*', in this case, requires a word that either allows a continuation using one of the forms illustrated in figures 35 and 36.

Xenophobia is the hatred or [NOUN] of foreigners

$$\begin{tikzcd}
\text{cn} \arrow[r] & \text{v} \arrow[r] & \text{det} \arrow[r] & \text{cn} \arrow[r] & \text{conj} \arrow[r] \arrow[l, dashed, bend right] \arrow[r, dashed, bend left] & \text{[NOUN]} \arrow[r] & \text{prep} \arrow[r] & \text{cn}
\end{tikzcd}$$

Figure 35:

Xenophobia is the hatred or [VECT ...] of foreigners

$$\begin{tikzcd}
\text{cn} \arrow[r] & \text{v} \arrow[r] & \text{det} \arrow[r] & \text{cn} \arrow[r] & \text{conj} \arrow[r] \arrow[llll, dashed, bend right] \arrow[r, dashed, bend left] & \text{[VECT ...]} \arrow[r] & \text{prep} \arrow[r] & \text{cn}
\end{tikzcd}$$

Figure 36:

Xenophobia is the hatred or of foreigners

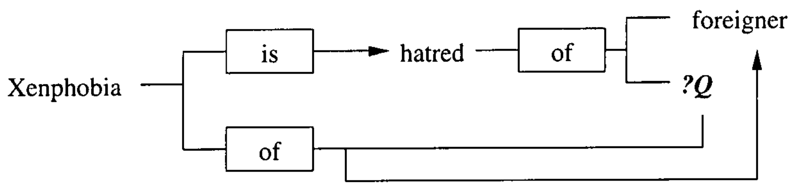


Figure 37:

The resulting network is shown in Figure 37. Two things are of particular interest here. The first of these is that the relationships that are not directly affected by the omission are correctly identified (these are *'xenophobia is hatred'* and *'xenophobia of foreigner'*). Secondly, the programme tries to build relationship structures even when parsing is affected by the omission. In this case this results in node definitions that include an indication that there is missing content - indicated by the node *'?Q'*.[99] This enables relationships to be defined for the missing content in the form *'xenophobia of ?Q'* and *'hatred of ?Q'*.

(2) This is the most interesting error as, in this case, the programme has to cater for the fact that the word '*hated*' is identified as a verb based on its lexical information. As a result, processing this sentence will be based on its use as a verb. Also, if '*hated*' is treated as a verb, then the rules governing conjunctions (described in section 4.2.4) mean that the verb type of '*fear*' will be used. The resulting structure (outlined in Figure 38) shows the effects of this treatment. Initial processing cannot generate a

Xenophobia is the hated or fear of foreigners

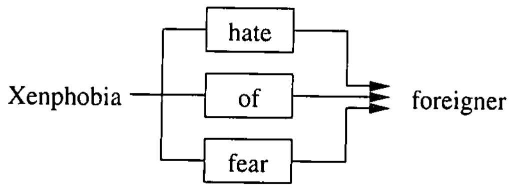


Figure 38:

legitimate grammatical tree for this sentence as the analysis has two potential verbs and a determiner with unfilled dependencies (illustrated roughly in Figure 39).

The first phase of determining a best-fit for the sentence relaxes one of the rules of phrase order (namely the rule that threads may not cross described in section 4.2.3). Bypassing this rule allows the dependency for the determiner to be satisfied by the

[97]This principle is also discussed in section 4.4.3 on page 143.

[98]It may be feasible to do this by allowing words with unfilled dependencies to hint at possible ways of completing the sentence but this has not been verified experimentally.

[99]The *'?Q'* node has special significance in the relationship network as it is also used in building relationships involving questions.

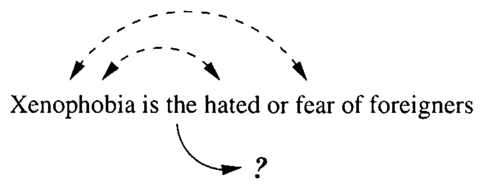


Figure 39:

concept '*foreigner*'. It also enables us to identify a probable source of the error in the sentence. This does not detect that the word '*hated*' has been used incorrectly but does show its consequences. Consequently, the programme provides this feedback on the error

```
Some dependencies violate boundaries
In tree:
[cn (d1)], [i (d4)], [det (d8)], [pp (d4)], [conj (d6)], [v
(d6)], [prep (d7)], [cn (d8)],
[i at 2] - dependency at 4
```

which indicates that the dependency for '*is*' can only be filled from inside another phrase.[100]

This approach is limited, however, given the range of errors that can be made. If an error occurs that the programme cannot accommodate in this way then it resorts to using the partially successful parses to define relationships.[101]

An alternative approach which has the potential of avoiding the problem inherent in having a word incorrectly identified in the lexicon is to fall back to using cues and simple adjacency dependencies to try and determine the type of a word (as was done in the trial versions described in 3.2.2). The single greatest problem with this approach is that the programme would have to do this for all words in the sentence.

[100]In this output the figure indicated by the letter 'd' refers to the index of the word satisfying a syntactic dependency.

[101]These can be seen, along with a description for the reason a particular grammar tree failed by setting the value of $::VERBOSEPARSEFAIL to 1 or 2 (depending on the level of detail desired) as in one of:

```
$::VERBOSEPARSEFAIL  = 1;
$::VERBOSEPARSEFAIL  = 2;
```

This results in large amounts of recursion, much memory use and very slow processing.[102] The best-fit approach used instead of this cannot determine which word was used incorrectly; however, it does generate a relationship network that can be used.

(3) above is the least interesting of the examples as it is passed with almost seamless results. The verb '*aaa*' is not recognised ('aaa' is used as this is not part of the lexicon and does not contain any cues in suffix or prefix that might aid in processing. As is shown in Figure 40 the network very closely resembles the network in Figure 34. The de-

Xenophobia aaa the hatred or fear of foreigners

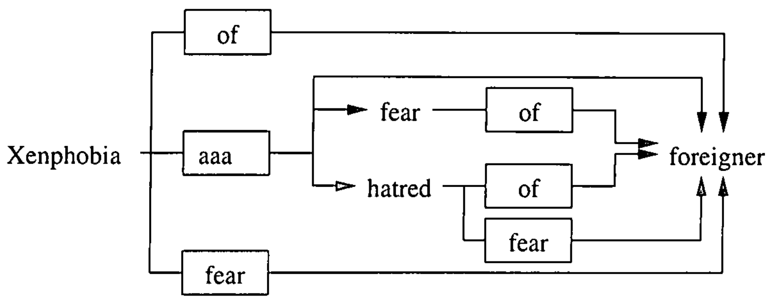


Figure 40:

pendencies of the identified words in the sentence enable the application to correctly determine that '*aaa*' is being used as a verb and generate the appropriate relationship network. One additional relationship is defined (as compared to the network from Figure 34) for $'hatred\ fear\ foreigner'$ but this shares the spot for lowest weight with $'xenophobia\ aaa\ foreigner'$.The weights for these two nodes are low enough that they are normally discarded from the output as being below the threshold for probability of interpretation. This threshold is set to 10 in the current implementation.[103]

[102]The code necessary for this approach is in the file 'sentence.pm'. To verify this experimentally change line 11 of 'sentence.pm' to

```
$::UseNoLexOnFail = 1;
```

[103]This can be changed by changing the value for $::WEIGHTHORISON in *rules.pm*. The value of 'auto' is allowed for single sentence use. This value allows the application to adapt the threshold in accordance with the number of nodes generated from the input. As 'auto' shows significant values only its use is not recommended for texts longer than a sentence. The format for this line would be one of:

```
$::WEIGHTHORISON = 10;
```

defined for this sentence but were discarded by the application as falling below the threshold level it has for showing weights.

(4), (5) and (6) In these sentences (illustrated in figures 41, 42 and) the nouns '*xenopho-*

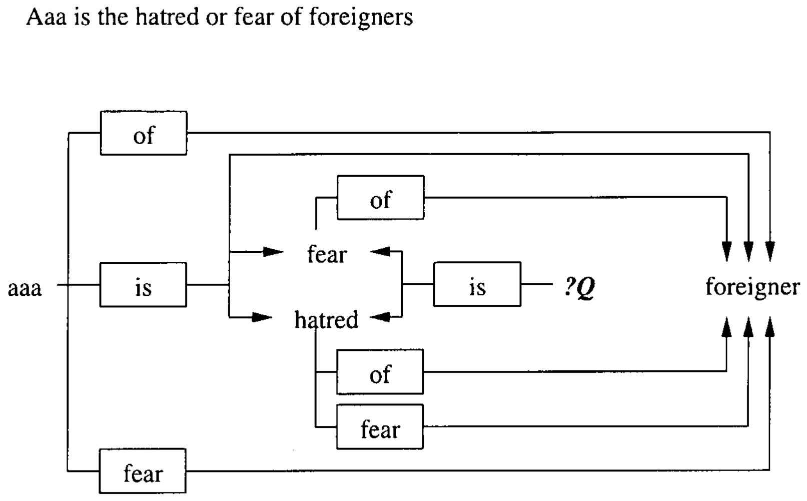


Figure 41:

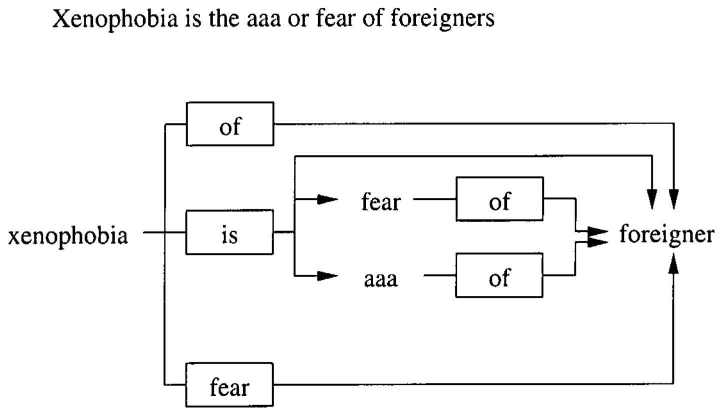


Figure 42:

bia', '*hatred*' and '*fear*', respectively, are replaced with an unrecognised word. As

$::WEIGHTHORISON = 'auto';

Xenophobia is the hatred or aaa of foreigners

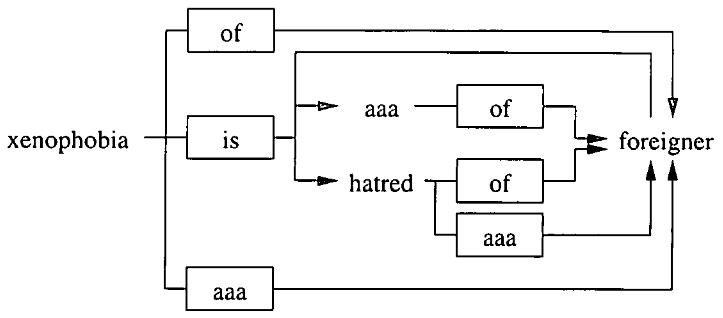


Figure 43:

with sentence (3), this is still sufficient to build a very similar network to that of Figure 34. The only exception is in (4) where there is slightly less dependency information available to the parser than in the other sentences. Consequently, extra nodes are defined representing relationships with the 'unknown node' ($'?Q'$), though as these nodes have very low weights they do not affect processing in general.

The resulting networks can be used as an indicator of the similarity between the sentences. For example, a simple comparison of the nodes from Figure 34 and the nodes from Figure 41 (illustrated in Table 2) shows that six of the eleven nodes (or 55%) are identical. Furthermore, four of the nodes that differ define relationships and these four nodes are identical at the relationship level (where only vector and target object are considered).[104] Depending on whether the relationships are taken into account in comparison this indicates a moderate to strong likelihood that the sentences cover the same topic. The chances are also very good that the concepts '*xenophobia*' and '*aaa*' in these two sentences are synonymous (the links recorded for these concepts match perfectly).

[104]These figures are from comparing sentence (1) to sentence (4). If we reverse the order the primary concept comparison is slightly worse given that there are more nodes defined for (4). The relationship comparison though will show six nodes which correspond to nodes from (1).

Nodes from Figure 34		Nodes from Figure 41
Xenophobia	$\neq$	*aaa*
	$\neq$	*?Q*
fear	$=$	*fear*
hatred	$=$	*hatred*
foreigner	$=$	*foreigner*
fear of foreigner	$=$	*fear of foreigner*
hatred of foreigner	$=$	*hatred of foreigner*
hatred fear foreigner	$=$	*hatred fear foreigner*
xenophobia fear foreigner	$\neq$	*aaa fear foreigner*
xenophobia of foreigner	$\neq$	*aaa of foreigner*
xenophobia is fear	$\neq$	*aaa is fear*
	$\neq$	*?Q is fear*
xenophobia is hatred	$\neq$	*aaa is hatred*
	$\neq$	*?Q is hatred*

Table 2: Comparing nodes from two similar sentences

5.3.3 Multiple sentences

The process of building sense networks from multiple sentences is simply an extension of the process demonstrated in sections 5.3.2 on page 160 and 5.3.2 on page 160. The networks for each sentence are combined additively. In other words, the information of each node in each of the networks representing a sentence is merged to form a new network containing all the information from all of the sub-networks. This is a simple operation when the nodes are unique in the sub-nets. When a node occurs in more than one subnet then each of the relationships is defined in the new node as well. In addition, the weights of the two nodes in the sub-nets are added together which means that the weight value becomes an indicator of the importance of the concept represented by that node in the text as a whole.

This example will use a slightly larger fragment from the same text as used in 5.3.2 on page 160:

Xenophobia is the hatred or fear of foreigners. A foreigner is a person from another country. They were born in a different country and they look different.

This sample (illustrated in Figure 44) also illustrates one of the limitations of the current implementation and the extent to which this affects processing. The generated network for this text contains 53 nodes representing both primary concepts and the relationships

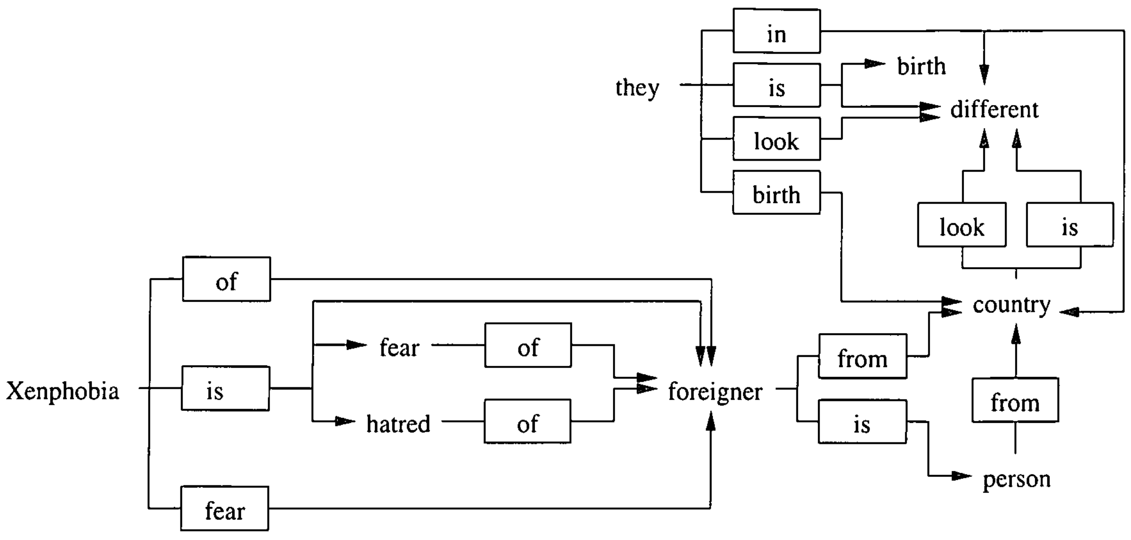


Figure 44:

between those concepts. The weights calculated for each node are based on the importance of the nodes in a sentence. As these are added together when the networks of the various sentences are combined, this means that important concepts in a text will have values larger than 100% (100% being the maximum value a concept can have in a single sentence).

For example, the concept '*xenophobia*' has a weight of 100. This weight comes from its having only one possible sense in the first sentence. '*Country*' and '*foreigner*' are present in two sentences and in both sentences have weights close to 100. When the nodes from these two instances of the words are merged the weights of each instance are added together. This forms part of the process of combining the relationship networks from individual sentences to form a network representing the text as a whole. This gives weights for '*country*' and '*foreigner*' in the relationship network for the paragraph of 179 and 200 respectively. .

5.4 Using the application

This application is not intended to exist as an application on its own. Instead, in keeping with the idea of building generalised modules that can be used to add functionality to educational programmes (as stated in section 1.5.3 on page 13), this application is intended to provide functionality for other educational programmes. These should provide the facility for translating the resulting networks into usable information.

The discussion in this section, thus, focuses on how these applications would be able to derive usable information from the relationship network. Usable information, in this context, means information that can be used to provide feedback as to a learner's success in conveying a particular message.

As the focus of this dissertation is not on the design of these applications, the examples provided here are fairly elementary. They should, nonetheless, provide a good illustration of the potential of this software.

5.4.1 Using the relationships in the network

In some senses the resulting network is much like a sophisticated concordancer. Concordancers provide examples of word usage by listing the words along with lines from texts which contain the word. In the process they provide an indication of the context in which one can use a particular word and the ways in which that word can be used.

As the relationship network developed from a text can also be viewed as a description of the context of the words, the network can be used in much the same way as a concordancer. So, for example, if asked for a description of the concept '*xenophobia*' one can use the relationship networks to generate these responses (the output listed here is not meant to form sentence structures - this is simply a listing of the relationships defined for these concepts intended to provide a description of context):

From text 1 (section D.1 on page 203)

```
Xenophobia is phenomenon
Xenophobia is hatred
Xenophobia with desperate
Xenophobia have consequence
Xenophobia can result
Xenophobia of foreigner
```

From text 2 (section D.2 on page 208)

```
xenophobia like prejudice
xenophobia like race
xenophobia like result
xenophobia to prejudge
xenophobia to one
xenophobia to race
xenophobia to result
xenophobia of free
xenophobia of equal
xenophobia has consequence
xenophobia has result
xenophobia oppose one
xenophobia emphasise feel
xenophobia emphasise equal
xenophobia with desperate
xenophobia is prejudge
xenophobia is race
xenophobia is complex
xenophobia is phenomenon
```

A fairly simple routine can expand on these definitions by testing to see which are defined by prepositional relationships (such as *'xenophobia can result'* or *'xenophobia with desperate'* in the set from the first text) and expand these relationships to provide a better indication of word meaning, as in:

From text 1 (section D.1 on page 203)

```
Xenophobia is phenomenon
Xenophobia is hatred
Xenophobia with desperate (have consequence)
Xenophobia have consequence
Xenophobia can result (to person)
Xenophobia can result (to job)
Xenophobia can result (in person)
Xenophobia can result (religious hatred)
Xenophobia can result (lose job)
Xenophobia can result (is this)
Xenophobia can result (is country)
Xenophobia can result (is native)
```

```
        Xenophobia of foreigner
```

Another scenario where this could be employed is providing hypertext markup in online materials. The relationships of any word in the text can be determined and displayed with very little effort, as well as a summary of the sense in any given portion of the text.

5.4.2 Using the weights

The relationship network does not only contain a record of the relationships in a text. It also records an indication of the importance of a concept in the text as a whole. This is a very rough indication based on the number of times that concept or relationship is used or referred to in a text. Nonetheless this provides a piece of information which can be used in a number of ways.

5.4.2.1 A measure of cohesiveness One practical use of the weighting system is in providing an overview of the cohesiveness of the text as a whole. The maximum weight a concept can have in a given sentence is 100. A score of 100 indicates that it had the same representation in every potential thread of sense generated for a given sentence.

In a multi-sentence text this weight is increased every time this concept is accessed, be this directly or by reference. Increase by reference is only partially successful at the moment as it depends to a large extent on having a fairly detailed set of relationships for each word in the lexicon. Without this, references have only those relationships defined in the text to draw on in determining targets.

Nevertheless this still provides a means of evaluating the interconnectedness of the concepts in the text. A concept can only have a weight larger than 100 if it is referred to in more than one place in the text. Setting the weight filter to 'auto'[105] uses the number of nodes in the network as the filter weight. This provides a rough means of comparing the interconnectedness in texts of different lengths.

The more nodes there are in a text with high weights, the greater the interconnectedness of concepts inside the text. This is a corollary of the definition of weights in this application as high weights can only occur when concepts are accessed in a number of sentences in a text. So, for example, if a concept is used twice in the text the weight for that concept has

[105]To see this in action use the supplied perl 'compare.pl' script with '-Wauto' as in

```
perl compare.pl [input_file] -Wauto
```

a potential maximum of 200 (the figure may be smaller if the concept is seen as potentially having other senses). Similarly, when a concept is identified as the target of a reference the weight of the target concept is also affected.

If we use the texts in sections D.1 (t1) and D.2 (t2) and use a weight filter of 'auto' we find that the filtered network for t1 contains ten nodes. The filtered network for t2 only contains five nodes. Considerably more experimentation will be necessary before this score can be used in any way to provide a measure of coherence. At present this is only useful when comparing texts based on the same topic. When comparing t1 and t2, for example, the score indicates that t1 is more cohesive than t2.[106]

This is not the only means of performing a contextual evaluation using the generated network. A more detailed method of evaluating cohesion is described in section 5.4.3.

5.4.2.2 Simple summary Node weights also provide a tool for making simple summaries of the content of a text. In this example, we set a cut-off point for node representation of a cumulative weight of 400.[107]

If we just list the nodes (arranged in descending order by weight in table 3) we can

NODE	WEIGHT
person	1031
they	867
country	752
who	599
xenophobia	550
foreigner	522
hatred	500
who is there	500
different	494
fear	436

Table 3: Key nodes identified by weight

generate a summary of the text. This summary is composed of the relationships defined for the primary nodes in the text (as identified in table 3). The summary provided here is

[106]A critical comparison of the two texts will tend to support this conclusion as the paragraphs in t2, while coherent in themselves, have very little in common. The latter paragraphs particularly show signs of extreme haste in writing and contain a number of incomplete sentences.

[107]400 is chosen as a high weight which should result in a fairly restrictive relationship structure and a very brief summary.

simply a list of the relationships taken directly from the relationship network for the text. It is not intended to be a coherent text in itself. Considerable refinement of this output is possible to make it more readable, such as reconstituting these relationships as sentences. For economy, only verbal links with a target identified as belonging to the noun class in the lexicon are used in this summary.

```
person fear weakness
person channel fear
person is country
person is native
person see weakness
person like weakness
they is different
they hurt you
country to work
country tend less
country tend work
country lose job
xenophobia have consequence
xenophobia is hatred
xenophobia is phenomenon
foreigner exploit country
foreigner tend work
foreigner tend less
foreigner work less
foreigner is person
foreigner develop country
foreigner develop less
foreigner develop work
hatred cause fear
hatred channel fear
hatred is fear
fear is change
fear is ignorant
```

The listing above involves a very simplistic filtering of the relationships that were defined. Seeing as these relationships are also represented as nodes with weights, a more complex set of filters can also be defined which only allows the highest weighted relationships to be included in the output. This representation of the relationships has potential benefits in providing feedback. As each relationship involving one of the primary concepts is defined in the simplest terms possible, feedback can easily identify which of these relationships have been included or omitted in a learner's text.

5.4.3 Coherence and cohesion

In processing a text, this application provides two indicators of coherence and cohesion in a text. The first of these is obtained from its ability to parse sentences. If a sentence parses successfully, it means the application was able to determine relationships for all the structures in the sentence. The resulting network should be a reasonable reflection of the sense of the sentence (cf. de Beaugrande and Dressler, 1994, p 84).

There are two cases where the resulting network may not be an accurate reflection of the text. The first of these will occur when the programme cannot resolve all the syntactic dependency requirements of the words in the sentence. This will occur if a text is input which contains a grammatical error, if the sentence contains a unexpected usage of a word defined in the application lexicon (as illustrated in Figure 38) or if the sentence structure is such that reveals an error in the computational representation of the grammar. As can be seen from the output from the sample texts analysed in section D this does not occur very often.[108] The programme does have the option of giving a report of the attempted grammatical structures which can be used to identify the cause of the error. As the report is based on the application's internal representation of grammar this is will probably not be generally useful without an understanding of the internal grammar. As such it is not considered as part of the feedback process and is disabled by default.[109]

[108]Failure to successfully resolve all syntactic dependencies is indicated by the message

[109]By default just the message *"unable to determine basic structure of this sentence"* is displayed. It is possible to have more detailed output in this case by setting one of the flags in the file 'rules.pm'. Set the $::VERBOSEPARSEFAIL parameter to one of the following values:

```
$::VERBOSEPARSEFAIL = 1;
$::VERBOSEPARSEFAIL = 2;
```

The first will give a listing of the phrase-types used (which can be used, for example, to determine when

The second reason for failing to build an accurate relationship network occurs when the semantic dependency requirements of the words and the sentence are not satisfied. This condition is very rare as it is almost always possible to generate some relationships from the sentence. If it is not possible to determine any relationships, a message to that effect is displayed.[110]

This, however, is not where the strengths of this application lie. The usefulness of the programme does not stem from the process of building the relationship network but from the things one can do with that network once it is built, which brings us to the second indicator. An overview of the resulting networks (such as those provided in section D) can provide an indication of this from observing the extent to which relationships are defined between nodes. Nodes which do not have relationships defined or groups of nodes which have little contact with other groups will occur when there are no ties between these groups. If all the nodes in the text are connected in the relationship network (either directly or via other nodes) then there can be said to be a cohesive argument in the text.

This does not automatically guarantee coherence. It is perfectly possible to have a sentence which successfully defines relationships that is still not fully coherent. Coherent implies that the various nodes in the text may logically be used together. This in turn requires a representation of knowledge in the lexicon (as discussed in section 4.4.2). As any predefined knowledge creates a bias in the application in favour of those relationships defined in the knowledge base this has been omitted from this version of the application. Consequently, as there is no representation of meaning in the lexicon, there is no direct method of determining that a sentence such as

The dog talked to the fish as it flew by.

is not coherent (especially as this is largely dependent on context). There are, however, indirect methods that we can use in combination with this approach to provide an indication of coherence (described in the next section). An additional benefit of the indirect approach is that it also provides the context that is missing in the example above.

phrases have been incorrectly identified). The second adds the values for left, right and syntactic dependencies to the output.

[110]The message reads *"unable to build any sense threads in this sentence."*

5.4.4 Testing by comparison

One relationship network can only provide insight into the relationships defined in that text. Two or three such networks, on the other hand, allow for an interesting range of applications. Having more than one text provides an indirect method of doing the contextual evaluation (one of the initial aims of this project) without the need for a predefined knowledge-base.

In brief, the problem is that there is no direct way of determining the accuracy of a text, either in a grammatical sense or in terms of the accuracy of its content. Without a formal description of the text type there is also no direct way of determining if the text is even an acceptable member of a particular text type (cf. de Beaugrande and Dressler, 1994, p.182-186; Warschauer, 1996, p. 6).

This is where using two or more networks from texts that are intended to be similar becomes useful.[111] In this process the texts are treated as sets of data (in the mathematical sense of the word set). As described in section 4.4.5, treating the nodes in a relationship structure as elements in a set enables the use of very simple but powerful operations which can be used to compare two or more relationship networks. An intersection of two networks, for example gives an indication of shared content. A difference, on the other hand, indicates information which is exclusively contained in one or other of the networks. The particular operations discussed here are intersection, difference and union.

These operations are most useful when one of the sets is based on a model answer - in other words when one of the sets is guaranteed to contain all pertinent information. In this case the operations determine the degree to which the content (defined by the relationships in the network generated from the text) agrees with the content in the model answer.

5.4.4.1 Dealing with synonyms Before we can begin a discussion of operations which attempt to make comparisons of two texts one has to confront the problem created by synonyms. If two texts cover the same topic but different terminology is used in the two texts this will make the texts harder to compare automatically. This is particularly important given the fact that there is no record of similarity (such as synonyms) in the lexicon used in this application.

However, this is not seen as a particularly serious problem in this application for two reasons. The first is simply that it is unlikely that the terminology used will differ in all

[111]A very basic comparison programme is provided along with the analysis programme to illustrate a few of the ways in which a sense network can be used.

respects. There will be overlaps (the likelihood of this is increased slightly by representing nodes using just the stem of the word, rather than the form actually used in the text). Understandably enough this is not something one would want to rely on for generating results. The second method is far more useful and it is implicit in the relationships that make up the network.

Simply put - similar concepts will tend to have a similar set of relationships defined. Synonyms can be detected, without the need for an explicit relationship defining their similarity, because the respective nodes will have a very similar set of relationships defining meaning. If the test for similarity is not limited to the words themselves (the node roots) but also takes into consideration the relationships defined in the nodes, one can determine similarity even when synonyms have been used.

This is, in effect, exactly the same process that is used in determining the potential targets of references in the text - the only difference is that the references search for similarities in the lexical definitions while here we are only concerned with similarities of use in the relationship networks themselves.

Since the relationships are also defined as nodes, it is very easy to do a search through the node list for nodes that represent similar relationships. Using the networks from the texts in sections D.1 (t1) and D.2 (t2) again we can build the list of words used in similar contexts[112] shown in table 4. These words have been determined as being similar as they have similar links defined in the respective nodes. As can be seen this includes some words that have been used identically in both texts (such as *'consequence'*). Most of the entries, however, are words that are used in the same context in the two different texts (that is, they have similar meanings in the context of their use in the texts). This table is intended simply to demonstrate that this approach to determining similarity when faced with synonyms is viable. If used in an educational context, links would also have to be provided to the texts themselves to provide examples in their actual context.

5.4.4.2 Intersection We can use the fact that we can determine similar concepts to determine where two texts are similar. This is effectively a mathematical intersection of the two sets of data which we can define as:

Given two relationship networks, the intersection is a relationship network con-

[112]This is done using the 'compare.pl' script with the '-s' switch as in

```
perl compare.pl -s [filel] [file2]
```

WORDS FROM T1		WORDS FROM T2
consequence	$\Longleftrightarrow$	consequence
country	$\Longleftrightarrow$	one
culture	$\Longleftrightarrow$	accept
culture	$\Longleftrightarrow$	self-esteem
culture	$\Longleftrightarrow$	benefit
fear	$\Longleftrightarrow$	feel
fear	$\Longleftrightarrow$	desire
foreigner	$\Longleftrightarrow$	person
hatred	$\Longleftrightarrow$	hatred
hatred	$\Longleftrightarrow$	unrest
native	$\Longleftrightarrow$	someone
phenomenon	$\Longleftrightarrow$	one
that	$\Longleftrightarrow$	feel
they	$\Longleftrightarrow$	person
they	$\Longleftrightarrow$	it
they	$\Longleftrightarrow$	area
this	$\Longleftrightarrow$	it

Table 4: List of words with similar relationship structures in texts t1 and t2

taining only those concepts defined in both the original networks.

An intersection of relationship networks is an extremely useful concept as it forms the primary means of testing for coherence in a text. If one of the two networks is a known quantity (a model answer) then an intersection shows the degree to which the concepts in the model answer are also contained in the learner's text. The larger the intersection the better the correlation between the two texts.

The intersection also shows where the two texts are similar. It includes concepts that have been used in both texts and concepts that can be determined as being similar.[113] Potentially this could be used to highlight sections of a text that can be determined as similar and to provide cross-references between the two texts. This would also reveal a measure of the extent to which the texts are different (discussed in more detail in section 5.4.4.3).

Even with the determination of similar concepts it is unlikely that an intersection will be very large. The nodes in the networks include both concepts and the representations of the relationships between nodes. As such the majority of the nodes will be determined by

[113]See section 5.4.4.1 on page 176 for a discussion of this process.

the writer's word-choice and style of sentence construction. These nodes are effectively filtered by the intersection process.[114]

Following from this we can distinguish three possible scenarios resulting from the comparison. In the first scenario there may be no intersecting nodes or a few intersecting nodes with low weights. In this case the texts would be considered different and, hence, essentially covering different topics. Other operations such as the similarity search ('compare.pl' with the '-s' switch) or filtering the nodes in the network based on the learner's text ('compare.pl' with the '-Wauto' switch) may reveal more information in this case. The filter operation in particular will provide an indication of the particular focus of the text.

The second scenario occurs when there are relatively few intersecting nodes but these have high weights. An example of this occurs when comparing the texts in Appendix D.[115] This indicates that the texts have a common thematic subject (indicated by the correspondence of strongly weighted nodes).

A third scenario arises when there is a strong correlation between the relationship networks of two texts. In this event, the intersection contains both a large number of nodes and nodes with high weights. This indicates that the two texts are functionally equivalent - as is shown when comparing the texts in Appendix D.3 (t3) and Appendix D.3.3 (t4). These two texts are different drafts of one student's essay.[116] This third scenario can only occur when both the topic concepts and the style of writing are similar in the two texts as the majority of nodes reflect relationships partly determined from word-order. Because the structure of the nodes in a relationship structure are composed completely from the information contained in the text the nodes will reflect the writing style of the author of the text as much as they reflect the content. This means that the relationship networks of two texts should show significant differences even when they have very similar content. These differences should be particularly evident in the links defined inside the nodes of the network. Consequently

[114]Using the 'compare.pl' script we can gain a variety of views of this intersection. In addition to filtering the relationship nodes from the intersection we can use the '-W' weight switch to show only those nodes with a high weight - indicating relative importance in the text. At the moment the intersection network preserves the values in the first file of input and uses these for filtering results. Seeing the relative importance of concepts in the second file requires swapping the order of these on the command line.

[115]The intersections for these texts generally range from 30 to 50 nodes, with the majority of these being large weights. The majority of nodes in a network are formed by combining objects, links and their targets to form compound nodes. As these compound nodes are dependent on the particular word order within a text, they will frequently be unique in each text.

This makes the number of shared nodes appear artificially small in relation to the total number of nodes in a network. The shared nodes, however, are primarily the heavily weighted nodes from a network. The constructed nodes will have low weights and so are of less significance in the comparison.

[116]These two texts share 390 nodes - approximately $\frac{2}{3}$ of the nodes in each network.

a strong correlation between two relationship networks should only occur when comparing drafts of the same text (for example to determine changes). If this is seen when comparing texts supposedly by different authors a strong correlation is a very good indication that one of the texts is plagiarised.

5.4.4.3 Difference Difference is the converse of the intersection operation described in the previous section. As such we can define this as

> Given two relationship networks, "difference" generates a relationship network containing only those concepts defined in the first network which are not defined in the second network.

This definition also indicates the directional nature of the operation. If one wants to determine which concepts occur in the second text but not in the first, one merely has to swap the order of the two files on the command-line.[117]

As strongly weighted nodes represent nodes which were frequently referenced in the text, these will tend to correspond with key concepts in the text. It follows that if the difference network includes a number of strongly weighted nodes, then it is likely that the particular focus of the two texts is different.

Alternatively, if the intersection operation has shown that the texts do have common ground, then the difference operation will provide an indication of any extra information contained in the first text and not in the second.

As with the intersection operation there is a third scenario which will result when the two texts are very similar. In this case the difference operation will return a very small network. As with the intersection, this is a strong indication that the two texts have both the same subject matter and the same author.

5.4.5 Output format

Currently, the output from all the operations in the *compare.pl* script are represented as a new relationship network. Intersection, thus, gives a relationship network containing only those nodes that occur in both files and difference returns a network showing only those

[117]Use the 'compare.pl' script with the '-d' switch as in:

```
perl compare.pl -d [file1] [file2]
```

nodes that do not occur in one of the files. A benefit of this approach is that it allows the output of these operations to be used as input for further operations - such as when comparing the proportion of nodes with high weights to those with medium or low weights (this use is demonstrated in section 5.4.6).

A friendlier interface is certainly possible and could allow automation of the process of arriving at many of the conclusions that have been made in describing these operations. This has yet to be implemented, however, as the interface will depend on the particular necessities of the environments in which the application is used.

Conceptually this would be linked to a HTML-server as providing the most portable platform for feedback. In addition HTML is a well documented markup language which has a growing number of well-developed design tools available, making it relatively easy for people with moderate computer literacy to design exercises that make use of the facilities this tool provides.

5.4.6 Interpreting operations

Interpreting the results obtained from the operations described above depends on the particular environment in which this sort application is used. In an environment where errors in language usage need to be identified, for example, neither the network nor the comparison script will be much use. In other contexts these can be both useful and valuable. This section and section 5.5 outline some ways in which these features can be put to good use.

A very important operation is the filter. Networks can be filtered to show the importance of particular nodes in the network. As the importance is a measure of the degree to which concepts are related in the network, this also provides a strong indication of the cohesion of the argument in a text. This does not, obviously, provide a qualitative judgement of inter-sentence coherence and cohesion except in as far as the weights of the concepts in a network are influenced by the process of resolving references. Much of the qualitative evaluation has to be left for operations which involve comparing the network of a learner text with that of a model answer. The model then provides both a context for the texts and a basis for implicitly evaluating coherence and cohesiveness.

As was shown in section 5.4.2.1 on page 171 the text given in section D.1 on page 203, (t1) had a better score in this regard than the text in section D.2 (t2). Additional information about the cohesiveness of the texts can be gained from using specific weights as opposed to the automatic weight choice used in section 5.4.2.1 on page 171. Using a weight of 100, for example, would reveal all concepts identified as having a priority level equivalent to that of a primary concept in a sentence. If a text does not contain many supporting sentences (in other words it consists largely of unsupported statements), then there will be a number of nodes in the 100 to 200 weight range. Text t2 has a total of 16 nodes with weights above 100. Of these, 16 nodes (62.5%) fall within the 100-200 weight range and only 6 above the automatic threshold for determining significance in the text. By comparison tl has 25 nodes above the 100 weight range but only 8 (32%) between 100 and 200.[118]

This is a start, but not yet enough for a qualitative judgement of the cohesiveness of these texts. All it indicates is that there are concepts which have cohesive ties (though the links defined for the nodes do also indicate what those ties are). A second consideration has to be whether the text is accurate in the sense that it contains the content that it is supposed to contain. This is the level of coherence. While an indication of the degree to which a text is coherent can be gained from examining the relationships defined for key topics in a text, a better measure can be obtained by comparing a learner text to a model answer.

As the model is a known quantity the intersection and difference operations can be used to provide a measure of the degree to which a student text is both accurate in the sense that it contains the necessary information and is coherent. In this operation we use both the relationships between nodes and the weights assigned to the nodes. As concepts are referred to in the network, the weights associated with the concepts increase. In addition a relationship is defined representing that form of access. If the nodes with high weights show a strong correlation with those in the model, there is a good indication that the argument in the learner text is both accurate and coherent.

This process can be illustrated by letting tl play the part of the model answer. Using tl as the model answer, we can generate an intersection set that shows that the two texts have 34 identical nodes, 10 of which have values larger than 200. If we represent this as a proportion of the significant nodes in the texts above this weight level then tl shows a 66% similarity (15 nodes at this weight level).[119]

If a measure of the degree to which a text does not contain necessary information the difference operation can be used. Using the difference operator on tl and t2 with tl as the model we see that 5 nodes are defined above the 200 level in tl and not in t2 (33%

[118]A slightly more sophisticated approach would also takes into consideration the types of these nodes and their respective weights. References, for example, also create nodes when the target of the reference cannot clearly be determined. In t2 the node with the highest weight is that for 'it'. A more accurate consideration would thus use the sums of the weights of the nodes in the different weight as an indication.

[119]tl is used as the model in this comparison. Using t2 as the base reveals only 6 common nodes at this weight level but with a correspondence of 100% at this weight level - which is to be expected as it is a shorter and less informative text.

of the nodes determined as significant at this weight level). These include concepts such as '*foreigner*' and '*fear*', which may be considered vital concepts given the topic of these assignments (the students were asked to write short essays about xenophobia).

An additional method of comparison which provides further insight into the accuracy of the content of a learner's text uses a slightly more complex evaluation of similarity than simple intersection. Instead the notion of similarity described in section 5.4.4.1 has to be used. In this way nodes are tested not only for their presence in the relationship network but also for having a similar set of relationships as the nodes in the model text.[120] If there is a correlation of nodes but the weights in the learner text are low then the probability exists that the text is not accurate (it may still be coherent if other nodes have high weights and the references defined for these nodes are directed toward the nodes shared with the model text).

5.5 Use in the ESL classroom

Although the application is not intended as a stand-alone application it can be used as such. The interpretation of results is not automated to the degree that they would be in a complete package but this does not mean that they can not be used in an ESL classroom. The output from the *compare.pl* scripts operations, for example, can be used as a score of the similarity of two texts. This lends itself to any scenario where a response has to be tested for accuracy of content (as opposed to grammatical accuracy).

5.5.1 Scenario 1

A scenario that fits this description is one in which learners are given a passage to read and requested to summarise the content. Their comprehension of the passage is tested by the degree to which they successfully identify and reproduce the key points from the text. The application can then be used to test the answers, either by comparing the answers against its own summary of the text (using the strongly weighted nodes approach of identifying these as described in section 5.4.2.2 on page 172) or by comparing against a model answer (as described in section 5.4.4 on page 176).

[120]This has not yet been reduced to a single operation in the *compare.pl* script. As a result repeat processing is needed to achieve this effect. This can be done by making a union of two intersection sets (the relationships defined in the nodes of an intersection set depend on which text is used as the basis for comparison and can be changed by swapping the order of the input files on the scripts command-line) and testing this union for similarities.

5.5.2 Scenario 2

A similar scenario in which this can be used is that of comprehension exercises. These often involve questions with answers ranging from one or two sentences to small paragraphs. In addition the content of the paragraphs is very predictable as it is defined by the context of the passage on which the exercise is based. As in the summary scenario, learner output can be compared against model answers. As the passage provides a context for the texts the lexicon can be primed for that context by generating a network for the passage. References used in the answers will then have suitable targets defined which will allow accurate comparison. Furthermore, the passage also provides a context that the application can use for identifying concepts used synonymously in the different texts (as described in section 5.4.4.1 on page 176).

5.5.3 Scenario 3

An exercise for students with a lower level of proficiency would be to give them a set of words and require them to re-arrange these in a sentence. With this absolute control of vocabulary, the application can easily identify learner's who succeed in the exercise. Far less computationally expensive methods exist for doing this already. The benefit of using an analytical tool in this exercise lies in the ability to generate feedback for those who fail to order the words correctly. This is achieved by analysing the relationships defined in the learner's output, giving a report on those relationships in which the learner was successful as well as a report of those relationships in which the learner was not successful. If considered useful in the context of the exercise, a report can also be generated showing the incorrect relationships determined in the learner's output.

5.5.4 Scenario 4

A similar exercise for students with intermediate proficiency is to give them a set of words and request them to write a short passage that uses those words. This is the most complex task of those described here from an evaluative perspective as it allows the greatest opportunity for free variation in content and sentence structure. In this scenario the application can be used to test that the concepts have been used, that valid relationship structures are defined using these concepts and can provide a score which should give an indication of the cohesiveness of the text as a whole (done by testing the ratio of nodes with weights greater than the maximum for a sentence against the nodes with weights close to the maximum for a sentence - as described in section 5.4.6 on page 181).

6 Use and limitations for ESL

The results generated using the application serve as guidelines for its use. They also highlight limitations in the programme. This section discusses a selection of these features.

6.1 Target population and target texts

The texts used in testing the application were all generated by students in the English department at the University of the Free State. Their level of proficiency and qualification means that the texts they produce will be markedly more complex than texts produced by ESL students at, for example, school level. A complex text containing many references (both internal and external) will inevitably result in a larger reference network than a simpler text. This is not necessarily a problem, however, as it is more likely to allow sufficient information in the resulting sense structures for the logic which determines targets for references to function.

The complexity of the text does, however, put a practical size limit on the length of text the application can process. As can be seen in the summaries of processing provided in section D on page 202 a text of between 25 and 35 sentences takes between one and three minutes to process. The time taken to process texts is inevitably proportional to the complexity of the text, so a complex text of 10 sentences may take longer to process than a simpler text containing 20 sentences. Just as inevitably, the complex text will result in a richer relationship network.

Processing time becomes the limiting factor as this determines the speed at which feedback can be generated. In scenarios where instantaneous feedback is desired one would have to use shorter or less complex texts - probably in a five to 10 sentence range. In scenarios where a wait of several minutes is acceptable, the length and complexity of the texts can be increased.

Another limiting factor is that of the model answer used. Longer texts allow for a greater range in student responses and, consequently, will vary further from a model answer than would shorter responses.

The programme's ideal focus is environments that encourage short contextual responses. An example of this sort of text is the short paragraphs common in school comprehension passages in both language and literature testing. In texts of this length the context can be reasonably well determined and controlled, which limits the potential for variation and allows greater accuracy in evaluating content.

For all this the application still has a potential use in language learning. It can relieve much of the drudgery from evaluating short passages and provides information that can be used to identify achievements or shortages in the content of learner texts. This can in turn be used to provide feedback about the coherence of the passages and the degree to which necessary content has been included or omitted.

Another disadvantage for using this application in an ESL classroom is that the application is designed as a module for use in other CAI programmes. As such it does not include accessible computer-generated feedback. Automated generation of feedback is quite feasible. The processes discussed in the section describing the interpretation of results (section 5.4.6) are ones that can be automated. In short passages with little complexity the results would be available in a matter of seconds even from this unoptimised version of the application. Feedback from longer texts with greater complexity can take a few minutes - a delay that may or may not be acceptable depending on the environment in which the application is used.

Importantly, this feedback is based on the content of the learner's communication rather than the grammatical form of the communication.

6.2 Contextual feedback

The current feedback system is designed as an interface for a post-processing programme and does not produce feedback that is easily accessible for a user. The results need to be interpreted. The interpretation of these results is, however, a task that can be automated. None of the results generated in section 5.4.6 on page 181, for example, require value judgements to be made. Instead, these are made implicitly by the use of a model text as the basis for comparison.

It is possible to provide a mathematical measure of similarity, though a format for this which accurately expresses the nature of the similarity still has to be identified. For example, the comparisons between most of the texts in Appendix D have an intersection of between 30 and 50 nodes out of a total of between 300 and 400 nodes. Most of these nodes, however, define particular relationships and thus will have weights lower than the base weight for a sentence (unless, of course, these are used in turn to build larger relationships). This makes a comparison of similarity possible as primary concepts will have weights greater or equal to the base-weight of a sentence (as described in sections 3 on page 172 and 5.4.6). This also means that the majority of the nodes in the network structures representing the texts are not significant even in that text. A better measure of similarity is then the degree to which the intersection of 30 to 50 nodes represents the set of significant nodes from the two network structures. Even texts which are apparently identical, such as the texts in Appendix D.3 and Appendix D.3.3, show a number of differences in network structure. These texts show an intersection of 235 nodes (from 328 nodes in the second text which is used as the model in this run) where only 27 of these have weights larger than the base weight of a sentence. Once again the similarities are among the set of more significant nodes - though here there is also a greater coincidence in nodes with lower weights. As these weights are determined from the probability that a concept will have a particular sense in a sentence this is a very strong indicator that the intersection covers similar concepts - particularly as the intersection at lower weights represents an intersection in nodes that represent particular relationships defined in the two texts.

An intersection alone is not sufficient, obviously, to tell if there is a correspondence between the texts. It is here that the difference between texts becomes important. The difference structure can be examined to see if it contains significant nodes (nodes with a high weight). This operation (described in more detail in section 5.4.4.3 on page 180) reveals the extent to which the topic of one text is different to that of another. If the difference does not contain any significant nodes then the topics are similar. On the other hand, if a difference network does contain significant nodes there is an indication that the one text contains information that is not in the second. Depending on which file is being used to make the difference network, this can either be information that should have been included but was not or an indication (model text - learner text) of areas where the learner had strayed off topic (learner text - model text). This use is demonstrated in section 5.4.6 on page 181.

6.3 Processing speed

As this programme a trial version, no real effort has been made to optimise performance. The necessity of optimisation is apparent in the time taken to process sentences containing complex structures.

The programme determines *sense* by first constructing a structure containing all potential threads of sense and then reduces this by eliminating impossible interpretations. For complex sentence structures, this results in thousands of possibilities being generated even when only a few of these turn out to be valid. Inevitably this is a time-consuming process. Depending on the operating environment in which the Perl interpreter is running, extreme cases containing a number of conjunctions and nested sentences can cause the Perl environment to run out of memory.

Sentences for which no valid sense thread can be identified contribute significantly to the slowness of this operation. An additional test is performed on these sentences to try and construct a "best-fit" network of sense threads which approximates the intent of the sentence. In a worst case scenario, this process is applied to all sentences which failed the test for grammatical dependencies. As the number of ungrammatical possibilities invariably outweighs the grammatically possible sentences, this process can be slow. This violates the second basic requirement for a CALL programme identified in section 1.3, namely that the programme should be fast.

Considerable optimisation in the programme is possible. For example, processes which cannot yield viable sense threads can be aborted as soon as the lack of viability becomes apparent. At the moment, however, the programme continues to process these as though they were viable. Similarly, the process of building sense when no grammatically correct parse has been found could be improved considerable by keeping a record of the most viable of the ungrammatical parses. The recursive routines used in building sense from parsed responses can take a long time, especially when dependency requirements have not been met. Not processing the less likely possibilities can improve performance considerably.

This version made no attempt at efficient design or memory management. But while this would improve performance considerably, it would not be sufficient. At the moment the programme takes, on average, between 1 and 3 minutes on a fast computer with minimal workload to process a text of the length and complexity of those contained in Appendix D. This is still too slow to create immediate feedback for this length of text. There are two ways of bypassing this problem though. The first involves limiting the length of texts to four or five sentences. At this length the delays should be less noticeable as sentences would be processed in a few seconds. Alternatively a medium such as e-mail could be used for sending feedback. The delays inherent in this medium mean that a wait of a few minutes while the text is processed will not be noticed and the feedback can still be fast enough to be useful.

The performance of the application could also be improved (both in terms of accuracy and usefulness and in terms of speed) if one includes concordance information in the lexical database. Data which indicates the frequency with which individual words are used would provide a way of filtering likely interpretations and ranking these accordingly (Ritchie, 1987, p 248).

6.4 Further developments and research

The implementation discussed in this dissertation is potentially useful. Some additional development will be required before it meets the requirements of robustness, speed and ease of use that are expected of consumer software. This section describes some of the key improvements that will need to be added before this application can meet user expectations.

6.4.1 Changes in programme logic

Speed is a major consideration in an application of this nature. The most promising line of reasoning for reducing processing speed is that of enabling predictive parsing of sentences. This could be added, for example, by including concordance information as suggested in section 6.3 on page 188. This addition would involve very small changes to the current programme (the addition of a comparison operation in the 'genTree[121]' subroutine). The changes to the lexicon, however, would entail a substantial amount of work as this information would have to be entered for every sense of every word in the lexicon.

A second improvement which could, potentially, improve processing speed is to develop a more precise method of defining conjunctive dependencies. The method used at present uses weights within the sentence to determine phrase scope. This does not test semantic dependencies until late in the processing. This causes the generation of far more possible sense interpretations than is necessary.

6.4.2 Lexical sense

The lexicon, which currently only includes 7000 entries covering all recognised word forms, allows considerable scope for improving the performance of the programme - both in the sense of improving processing speed and in the sense of improving the accuracy of the generated relationship networks.

An obvious lack in the current system is that of the links that would enable the semantic dependencies defined for references to work. This requires adding elementary word

[121]found in the file '*genTree.pm*'

definitions to the lexicon in the form

$$\begin{tikzcd} & is \arrow[r] & male \\ man \arrow[ur] \arrow[dr] & & \\ & is \arrow[r] & person \end{tikzcd}$$

This also implies that the lexicon requires a better classification of sense. For example, the word "*man*" can refer to either an individual male homo-sapiens or to the collectivity which is all homo-sapiens, male and female.

In many ways this is similar to the subcategorical features used in generative grammars. These provide a categorical system that enables words to be categorised so that they are grouped by their similarities yet still able to provide a means of identifying differences. In this way the common features of, for example, *proper* and *common* nouns could be used to define a primary category of nouns, with sub-categories which include pointers to the different usages of the types. Instead of having two distinct types for proper and common nouns these would be classed in a lexicon under the same type - that of noun - but with a simple binary feature such as [$\pm$*COMMON*] to differentiate between them (Radford, 1997, pp. 59-63). A categorisation of this order, though, is not feasible given the range of features that can influence semantic processing.

In dependency-based grammars this can be accounted for by inheritance of properties. In Word-grammar, an example of a dependency based grammar, properties and meanings are inherited from more general words. The word "*dog*", for example, includes properties associated with the concept "*pet*" and the concept "*mammal*"(Hudson, 1998). Grammatical features, such as the adjacency rules that apply to a particular word, are also inherited. As a result problems associated with specific instances of word use can be resolved by defining the word as inheriting properties of base types that include the two divergent applications of the word.

The Cognitive Science Laboratory at Princeton University (under the direction of Professor George A. Miller) is developing an online lexical reference system named WordNet (`http://www.cogsci.princeton.edu/~wn/`). In this system English nouns, verbs, adjectives and adverbs are organised by semantic into synonym sets, each representing one underlying lexical concept - a design that is inspired by current psycholinguistic theories of human lexical memory (Miller et al., 1990, pp. 235-244). Different relations link the synonym sets.

Preliminary investigation suggests that integrating the WordNet system with the current lexical structure would be the most feasible method of providing the classification of sense necessary to allow more accurate representation of texts, though this does come at a price. The greater detail inherent in a database such as WordNet makes it more difficult to process entries where a word contained in the database is used in a sense that is not contained in the database.

6.4.3 Error handling

The texts used in testing were edited slightly to remove some of the spelling errors and to simplify complex sentences containing much punctuation. This was necessary because very little effort was made at handling punctuation properly. Punctuation was not considered significant in determining the viability of this sort of application.

In real-world application, however, punctuation is a feature of language use. Some additions will have to be made to the programme to allow for this.

A more significant problem of real-world use is the fact that texts by ESL students will seldom not contain errors. To date this has been a big problem for those parsing systems that have been attempted (Chen and Warden, 1999, http://www.cyit.edu.tw/ warden/). The programme must be able to recover gracefully regardless of whether errors result from incorrect word choice, bad word order, bad spelling, or any other of the myriad of possibilities that will occur in student texts. Recovering gracefully is also a term open to definition. This can mean giving feedback on these errors, attempting to derive meaning from these sentences anyway, eliciting a repair and rephrase, or any combination of these and any unnamed feedback strategies.

The approach adopted in this application has been to process the input as though the errors are simply words not included in its lexicon. An alternative way of handling these would be to couple this to a spell-checker which could weed out the more obvious spelling errors, though at a slight penalty in terms of speed.

6.5 Future research

In classroom experiments that illustrate ... complementarity, theoretically grounded learning materials and strategies to facilitate L2 learning are selected or developed by researchers. The researchers then work with parti-

cipating teachers toward classroom use of these materials and strategies, followed by classroom research on their impact on students' learning (Pica, 1997, p. 54).

In the context of Pica's comment we have essentially reached the end of stage one. The development of a functioning application capable of being used to evaluate contextual information in texts has been a substantial undertaking. With this application available a number of avenues of research become available.

The application developed here is intended to provide a tool to assist in the construction of further CALL applications. The usefulness of the application and its viability as a CALL application still need to be tested fully. In particular, the application tested in ESL classrooms to determine the degree to which it does or does not facilitate the learning process.

The feedback system also need further study. The current application provides very elementary feedback which requires interpretation by the user. This is partly because the judgement of which information in the relationship network represents significant contributions is extremely primitive. The sample of texts used in testing this application are extremely small[122] and do not include any from the target population of ESL students. This means that the programme must be tested with a larger sample of texts, preferably those generated by the target population, and the results must be analysed to determine if the relationships considered significant in the resulting relationship networks accurately reflect the significant relationships in the original texts.

This research would enable developers to revise the definition of significance used in this implementation and to construct more informative and more effective feedback.

[122]Appendix D does not contain all the texts used in testing.

7 Conclusion

It was seen that learning should take place in an environment that makes it possible for the learners to produce comprehensible output. This in turn requires that the learners' output is comprehended. The application described in this dissertation provides a method of simulating this comprehension and using the simulation to generate analyses that can be used to provide contextual feedback to the learner.

To facilitate this process an application was constructed that could be used as a component in CALL programmes to provide this facility. The application generates a network of relationships based on those contained in an input text. A number of operations are demonstrated using this relationship network that provide insight into the cohesiveness of a text. Furthermore, by comparison with a model text, operations are demonstrated which allow for forming evaluations of the accuracy and coherence of a learner text.

By enabling knowledge structures to be constructed from ordinary text input, it provides a tool that can facilitate the development of applications which require evaluating the content of texts. Furthermore, the programmes themselves are designed so that the output can be redirected and post-processed by other applications. This allows, for example, inclusion of the output in hypertext documents and allows other applications to interpret the results in a more user-friendly manner than that provided by the *compare*.pl script.

The programme lacks some features necessary for full usability and has yet to be tested in the ESL classroom - though suggestions as to how it can be used even in its current format are made in section 5.5 on page 183. The results generated from sample texts derived from student output do, however, show that the programme is viable. It will, in other words, be possible to add support for unrestricted learner text production in CALL programmes. The texts used in testing were between one and two pages in length - a factor which demonstrates that learner output in CALL programmes need not be limited to brief, sentence length responses.

In brief, given some small additions and further testing, it is possible to design CALL programmes which allow students to produce communicative output and to expect and receive relevant automated feedback on that output.

A Frequently used abbreviations

A.1 Acronyms

AI Artificial Intelligence

CAI Computer Assisted Instruction

CAL Computer Assisted Learning

CALL Computer Assisted Language Learning

CL Computational Linguistics

EFL English as a First Language

ESL English as a Second Language

ESP English for Specific Purposes

GPSG General Phrase Structure Grammar

HPSG Head-driven Phrase Structure Grammar

IRC Internet Relay Chat

LFG Lexical Functional Grammar

MUD Multi-user dungeons

MOO MUD Object Oriented

MUSH Multi-User Shared Hallucination

MTT Meaning Text Theory

NLP Natural Language Processing

OS Operating System

rl Real Life (life outside of a virtual community)

RPG Role Playing Game

RST Rhetorical Structure Theory

SLA Second Language Acquisition

TEFL Teaching English as a First Language

TESL Teaching English as a Second Language

A.2 Dependency abbreviations

N noun class

Ref reference class

S sentence class

V Vector class

adj adjective

adv adverb

conj conjunction

cn common noun

det determiner

eos end of sentence marker *(not used in the final version)*

f future tense auxiliary

h present perfect auxiliary

hp past perfect auxiliary

i verb "*to be*"

ip past tense of verb "*to be*"

prep preposition

pn proper noun

pp	past participle
pron	pronoun
sos	start of sentence (*used internally to indicate an instance of the S class*)
v	simple tense verb
vc	continuous tense verb
vp	past tense verb

B Programming Language Choice

The initial programming for this project was done in C++ with the idea that this relatively low level language would improve the speed of the final programme. This has since been abandoned in favour of Perl for a number of reasons. Firstly, while good C++ programmes are faster than their scripted equivalent, there is a considerable increase in development time associated with C++. Perl provides a number of facilities for handling strings that are not as functionally available in C++. Furthermore, scripting languages, like Perl, handle memory usage internally while low-level languages like C++ require explicit memory management. Typically, Perl scripts use twice as much memory as C or C++ programmes (Prechelt, 2000, p. 29). This can become significant as the recursive routines used to determine the sense of sentences can use much memory when processing complex sentences. As at least a quarter of programming in C applications relates to memory management (a factor that weighed heavily against C in this experimental version), and since this is essentially an exercise in determining the viability of this type of tool, this factor has been ignored as insignificant.

Secondly, even though C++ compilers are available for all platforms, the differences in the compilers and platforms make C or C++ code only marginally portable. Part of the aim of this project is that it be able to run on any Operating System (OS).

Thirdly, the experimental approach adopted meant a rapidly changing code-base. This lends itself more readily to a scripting environment where changes do not have the planning overhead associated with the more complex C++ environment.

Lastly when this project was in its infancy, Python was relatively unknown, Java unheard of and Perl widely distributed, as can be seen by the way Perl rapidly became the backbone upon which much of the scripting of the Internet was done. This also means that, while much of the development has been done on UNIX (Linux) systems, only minor changes would be needed to use the final product on Microsoft or Apple systems. When a reasonably final version of this project is achieved a C version will be developed, again in the interest of speed. Until then, Perl will suffice.

Languages used in AI experimentation, such as Prolog, LISP and Scheme, were not considered. Part of the aim of this dissertation is to develop an application which could be integrated into existing systems. These languages were seen as harder to port to more traditional programming environments, not as easy to integrate with Internet applications and web-pages and as providing too slow an operating environment to be effective as a classroom tool.

C Rules defined in the final version

Because the dependencies structures identified in the application are based on observed phenomena it is inevitable that they will be more restrictive than those that govern human language generation and interpretation.[123] At the moment they do, however, cover a sufficiently large range of language input to enable the application to be usable. The shortcomings of the application are ones that can be eliminated as part of the process of improvement and refinement and not inherent in the principles on which the programme is based.

C.1 Grammatical dependencies

prepositions ***(prep)*** `prep`

determiners ***(det)*** `cn, pn`

adjectives ***(adj)*** `cn, pron, pn`

adverb ***(adv)*** `v, vp, vc, pp, i, ip, adj`

pronouns ***(pron)*** `pron`

proper nouns ***(pn)*** `pn`

common nouns ***(cn)*** `cn`

simple tense verb ***(v)*** `v`

past tense verb ***(vp)*** `vp`

past participle ***(pp)*** `i, hp, h`

continuous tense verb ***(vc)*** `vc`

verb "to be" ***(i)*** `vc, pp, i`

past verb "to be" ***(ip)*** `vc, pp, ip`

present perfect auxiliary ***(h)*** `pp, vc, i, ip`

[123]These are discussed in section 4.

past perfect auxiliary ***(hp)*** `pp, vc, i, ip`

future tense auxiliary ***(hp)*** `v, pp, vc`

new_sentence_marker ***(_sos)*** `v, vp, i, ip, pp`

C.2 Minor sequencing dependencies

determiners ***(det)*** `adv, adj, cn, pn`

adjectives ***(adj)*** `adv, adj, cn, pron, pn`

adverb ***(adv)*** `adv, adj, v, vp, vc, i, ip, h, hp, f`

pronouns ***(pron)*** `pron`

proper nouns ***(pn)*** `pn`

common nouns ***(cn)*** `cn`

simple tense verb ***(v)*** `adv, adj, vc`

past tense verb ***(vp)*** `adv, adj, vc`

past participle ***(pp)*** `adv, adj`

continuous tense verb ***(vc)*** `adv, adj`

verb "to be" ***(i)*** `vc, pp, adj`

past verb "to be" ***(ip)*** `vc, pp, adj`

present perfect auxiliary ***(h)*** `adv, vc, pp, det, cn, pron, pn`

past perfect auxiliary ***(hp)*** `adv, pp, vc, det, cn, pron, pn`

future tense auxiliary ***(hp)*** `adv, v, i, h, pp, det, cn, pron, pn`

C.3 Semantic left dependencies

prepositions ***(prep)*** `v, vp, vc, cn, pron, pn`

adjectives ***(adj)*** `v, vp, i, ip, prep`

pronouns ***(pron)*** `v, vp, i, ip, prep`

proper nouns ***(pn)*** `v, vp, i, ip, prep`

common nouns ***(cn)*** `v, vp, i, ip, prep`

simple tense verb ***(v)*** `_sos, cn, pron, pn`

past tense verb ***(vp)*** `_sos, cn, pron, pn`

past participle ***(pp)*** `_sos, cn, pron, pn`

continuous tense verb ***(vc)*** `_sos, cn, pron, pn`

verb "to be" ***(i)*** `_sos, pn, cn, adj, pron, adv`

past verb "to be" ***(ip)*** `_sos, pn, cn, adj, pron, adv`

present perfect auxiliary ***(h)*** `pn, cn, pron, adj`

past perfect auxiliary ***(hp)*** `pn, cn, pron, adj`

future tense auxiliary ***(hp)*** `pn, cn, pron, adj`

C.4 Semantic right dependencies

prepositions ***(prep)*** `cn, pn, pron`

adjectives ***(adj)*** `v, vp, i, ip, prep`

pronouns ***(pron)*** `v, vp, i, ip, prep`

proper nouns ***(pn)*** `v, vp, i, ip, prep`

common nouns ***(cn)*** `v, vp, i, ip, prep`

simple tense verb ***(v)*** `cn, pron, pn, adj`

past tense verb *(vp)* `cn, pron, pn, adj`

past participle *(pp)* `cn, pron, pn, adj`

continuous tense verb *(vc)* `cn, pron, pn, adj`

verb "to be" *(i)* `vc adj, pp, pron, pn, cn`

past verb "to be" *(ip)* `vc adj, pp, pron, pn, cn`

D Sample texts

The programme's relationship network construction was tested using the texts included below. These texts were written by students in the Department of English at the University of the Free State. As a result the texts are generally longer and consist of longer and more complex sentence structures than would those of the target population of the application - namely ESL students.[124]

The texts have been edited slightly to correct some spelling mistakes.[125] As this programme only has rudimentary routines for handling some forms of punctuation, some of the sentences which contained, for example, many commas had to be split up.

It is not feasible to provide a hard-copy of the complete output from the programme. The resulting sense network is stored in an ASCII file in a format that is intended to be human-readable. The problem in presenting the data here is one of economy. The network for the first text presented here, for example, contains 311 nodes. As these include both concepts and relationships it is not feasible to represent this graphically.[126]

Instead, output from an analysis of the first three sentences of each text has been provided to give an indication of the resulting network.[127]

[124]Additional texts used in testing can be found in the directory `eg-texts` on the companion CD. Relationship networks from these texts are in the directory `eg-output`.

[125]Not all spelling errors were corrected. The intended operational environment of the programme means that spelling errors are an inevitability and the programme must be able to handle these. The primitive handler in this implementation (which generates every possible interpretation) means that spelling errors result in extreme memory usage and poor performance. As a result some of the errors were corrected to improve processing speed.

[126]The complete output from each text is available on the companion CD in the directory `eg-output`.

[127]The exact results will differ slightly from the complete textual analysis in that the complete analysis will inevitably add additional references to those contained in the sample networks.

Similarly not all the sample texts have been included here. In total 17 sample texts on xenophobia were used to test the programme. The texts presented here are a descriptive subset of these sufficient to illustrate the functioning of the programme.

Lastly, the time indicated is affected by the type of computer running the test and the load on the computer at that time. In this case the computer is an Intel P4 2.8Ghz with 512Mb RAM. At the time of writing, this is a reasonably fast system. Implementations running on the sort of hardware generally available to schools will not perform as well. Neither will implementations running on computers with heavier workloads (such as web or database-servers).

The actual results of the programme on the companion CD may differ slightly from the results shown here as development remains an ongoing process. Memory use in particular has been made more efficient and some additional feedback options are being made available (settings for these can be found in the beginning of the file *rules.pm*).

D.1 Text 1

D.1.1 Text

Xenophobia is the hatred or fear of foreigners. A foreigner is a person from another country. They were born in a different country and they look different. They have different customs. They cook differently. They eat different foods and they dress differently. The only way that these differences could affect you is in an aesthetic sense. The differences cannot affect you physically. They cannot hurt you.

Xenophobia is a complex phenomenon. People fear that which is different. The reason for this fear is ignorance, ignorance of difference and change. However, some people may see fear as a weakness. For some people, hatred is a way of channeling this fear. One does get circumstances where hatred is not caused by fear. Hatred is caused by something else.

People may feel that foreigners exploit our country's resorces. Foreigners from less developed countries tend to work for less; this can result in people who are native to their own country losing their jobs. When people lose their jobs they become desperate. Xenophobia combined with desperation can have awful consequences. It is not only regarding the work place that people tend to be xenophobic. They are xenophobic in religious circles too. Xenophobia can result in ethnic and religious hatred.

There are people who suffer from xenophobia. There are also those who do not. There are people who embrace foreigners for the currency that they bring into the country. There are also those who see foreigners as a way of learning about other countries and cultures. Lastly there are people who themselves are foreigners when they visit a foreign country. These would rather be welcomed than feared.

D.1.2 Output

This network contains 430 nodes and took 1 minute 13 seconds to process (95 potential nodes were discarded as having weights below 10[128] - the threshold level of significance for these trials).

The visual feedback from generating the network is merely intended to inform the user of the progression of the analysis. As such it has a variety of levels of information available which can be enabled by setting variables in the file *rules.pm*. The default is for a minimum of progress information with an indication of grammatical errors identified by the application.[129]

D.1.2.1 Screen output The following text gives an indication of the progress of the application in building the relationship network for a text:

```
1> [Xenophobia is the hatred or fear of foreigners .]
Done with syntax - 3 possible trees
2> [A foreigner is a person from another country .]
Done with syntax - 1 possible trees
3> [They were born in a different country and they look different .]
Done with syntax - 11 possible trees
4> [They have different customs .]
Done with syntax - 1 possible trees
5> [They cook differently .]
```

[128]This translates as the concept was discarded because less than 10 percent of the possible contexts of a concept involved that relationship).

[129]To allow redirecting output the relationship network is output to STDOUT while status reports and progress information is sent to STDERR. Either or both of these output streams can be redirected. Both are output to the screen by default.

```
Done with syntax - 1 possible trees
6> [They eat different foods and they dress differently .]
Done with syntax - 2 possible trees
7> [The only way that these differences could affect you is in an aesthetic
sense .]
Done with syntax - 78 possible trees
8> [The differences cannot affect you physically .]
Done with syntax - 1 possible trees
9> [They cannot hurt you .]
Done with syntax - 6 possible trees
10> [.]
10> [Xenophobia is a complex phenomenon .]
Done with syntax - 2 possible trees
11> [People fear that which is different .]
Done with syntax - 3 possible trees
12> [The reason for this fear is ignorance , ignorance of difference and
change .]
Done with syntax - 75 possible trees
13> [However , some people may see fear as a weakness .]
Done with syntax - 23 possible trees
14> [For some people , hatred is a way of channeling this fear .]
Done with syntax - 18 possible trees
15> [One does get circumstances where hatred is not caused by fear .]
Done with syntax - 35 possible trees
16> [Hatred is caused by something else .]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
5 potential error levels
Trying parses with 2 dependency errors
Trying parses with 4 dependency errors
17> [.]
17> [People may feel that foreigners exploit our country has resorces .]
```

```
Done with syntax - 62 possible trees
18> [Foreigners from less developed countries tend to work for less ;]
Done with syntax - 10 possible trees
19> [this can result in people who are native to their own country losing
their jobs .]
Done with syntax - 463 possible trees
20> [When people lose their jobs they become desperate .]
Done with syntax - 6 possible trees
21> [Xenophobia combined with desperation can have awful consequences .]

Done with syntax - 16 possible trees
22> [It is not only regarding the work place that people tend to be xenophobic
.]
Done with syntax - 276 possible trees
23> [They are xenophobic in religious circles too .]
Done with syntax - 2 possible trees
24> [Xenophobia can result in ethnic and religious hatred .]
Done with syntax - 5 possible trees
25> [.]
25> [There are people who suffer from xenophobia .]
Done with syntax - 1 possible trees
26> [There are also those who do not .]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
6 potential error levels
Trying parses with 1 dependency errors
Trying parses with 5 dependency errors
27> [There are people who embrace foreigners for the currency that they bring
into the country .]
Done with syntax - 8 possible trees
28> [There are also those who see foreigners as a way of learning about other
countries and cultures .]
Done with syntax - 40 possible trees
```

```
29> [Lastly there are people who themselves are foreigners when they visit
a foreign country .]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
4 potential error levels
Trying parses with 1 dependency errors
Trying parses with 2 dependency errors
Trying parses with 3 dependency errors
30> [These would rather be welcomed than feared .]
Done with syntax - 1 possible trees
```

D.1.3 Commentary

A coherent thread of sense could not be found in sentences 3, 6, 28 and 30. These all contain words that are identified as references in the lexicon and so are subject to the lack of lexical information in resolving the target of the reference. As can be seen from sentences like 20, 27 and 29 this is not always a problem. The failing is most prominent when the reference has to be resolved outside the current sentence and, consequently, the grammatical dependencies on which building a thread of sense partly relies also lie outside the sentence. If no target for the reference can be found these dependencies are unsatisfied and no thread can be built that supports them.

D.1.4 Resulting network

This is based on the first three sentences from the text:

> Xenophobia is the hatred or fear of foreigners. A foreigner is a person from another country. They were born in a different country and they look different.

This network can be diagrammed as shown in Figure 45.[130]

[130]This is the same network discussed in more detail in section 5.3.3.

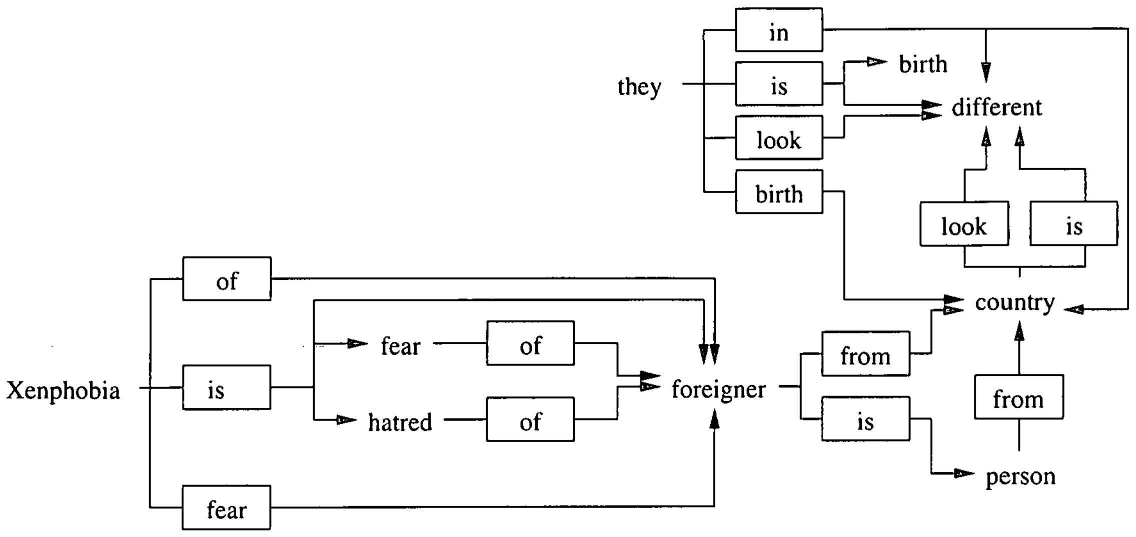


Figure 45:

D.2 Text 2

D.2.1 Text

Xenophobia is a complex phenomenon as opposed to a simple one. It is not a condition that one just born with. It is rather one with many aspects and causes. It is driven by, among other things, a desire for financial benefits. It is also one way of boosting one's self-esteem. This is at the expense of someone else. Xenophobia can be likened to prejudice and racism. Ultimately it develops from negative attitudes that people in one area feel against those from another area.

Xenophobia, combined with desperation, has awful consequences as opposed to positive results. Xenophobia does not emphasize any feelings of freedom or equality. It has no regard for human rights. Desperate people tend to act irrationally. Desperate people who are already negatively predisposed towards others are surely going to behave even worse.

Ethnic and religious hatred, as opposed to ethnic and religious acceptance. Ethnic and religious acceptance promotes feelings of harmony and satisfaction. Also important is the positive emphasis on one's feelings of self-worth. Ethnic and religious hatred promotes civil unrest, violence and chaos. Ethnic and religious unrest destroys societies. It splits them and fragments the lives of those who are hated and the lives of the haters.

D.2.2 Screen output

This network contains 324 nodes and took 1 minute 18 seconds to process (60 nodes where discarded as being below the threshold weight for inclusion).

```
1> [Xenophobia is a complex phenomenon as opposed to a simple one .]
Done with syntax - 4 possible trees
2> [It is not a condition that one just born with .]
Done with syntax - 11 possible trees
3> [It is rather one with many aspects and causes .]
Done with syntax - 6 possible trees
4> [It is driven by , among other things , a desire for financial benefits
.]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
6 potential error levels
Trying parses with 1 dependency errors
Trying parses with 2 dependency errors
Trying parses with 3 dependency errors
Trying parses with 4 dependency errors
Trying parses with 5 dependency errors
5> [It is also one way of boosting one has self-esteem .]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
3 potential error levels
Trying parses with 1 dependency errors
Trying parses with 2 dependency errors
6> [This is at the expense of someone else .]
Done with syntax - 2 possible trees
7> [Xenophobia can be likened to prejudice and racism .]
```

```
Done with syntax - 5 possible trees
8> [Ultimately it develops from negative attitudes that people in one area
feel against those from another area .]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
3 potential error levels
Trying parses with 1 dependency errors
Trying parses with 2 dependency errors
9> [.]
9> [Xenophobia , combined with desperation , has awful consequences as opposed
to positive results .]
Done with syntax - 25 possible trees
10> [Xenophobia does not emphasize any feelings of freedom or equality .]

Done with syntax - 6 possible trees
11> [It has no regard for human rights .]
Done with syntax - 3 possible trees
12> [Desperate people tend to act irrationally .]
Done with syntax - 1 possible trees
13> [Desperate people who are already negatively predisposed towards others
are surely going to behave even worse .]
Done with syntax - 1 possible trees
Could not determine sense threads in recognised sentence structure.
Trying with partial sentence structures
14> [.]
14> [Ethnic and religious hatred , as opposed to ethnic and religious acceptance
.]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
4 potential error levels
Trying parses with 1 dependency errors
Trying parses with 2 dependency errors
Trying parses with 3 dependency errors
15> [Ethnic and religious acceptance promotes feelings of harmony and satisfacti
.]
Done with syntax - 4 possible trees
16> [Also important is the positive emphasis on one has feelings of self-worth
.]
Done with syntax - 1 possible trees
17> [Ethnic and religious hatred promotes civil unrest , violence and chaos
.]
Done with syntax - 2 possible trees
18> [Ethnic and religious unrest destroys societies .]
Done with syntax - 2 possible trees
19> [It splits them and fragments the lives of those who are hated and the
lives of the haters .]
"haters" is not defined in LEXICON Going to try "haters" as possible [cn,
]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
7 potential error levels Trying parses with
1 dependency errors Trying parses with
2 dependency errors Trying parses with
3 dependency errors Trying parses with
4 dependency errors Trying parses with
5 dependency errors Trying parses with
6 dependency errors
```

D.2.3 Generated sense network

This is based on the first three sentences from the text:

> *Xenophobia is a complex phenomenon as opposed to a simple one. It is not a condition that one just born with. It is rather one with many aspects and causes.*

This network can be diagrammed as shown in Figure 46. This is also an example of one of

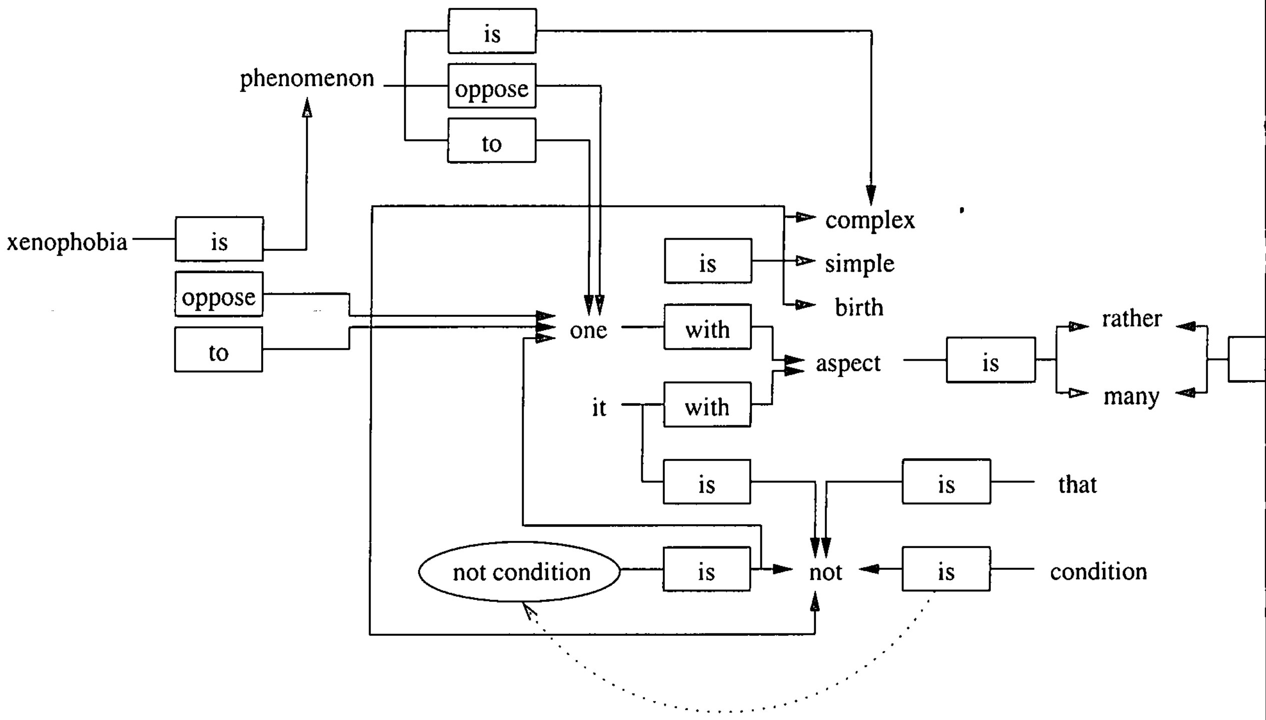

phenomenon
is
oppose
to
xenophobia
is
oppose
to
complex
simple
birth
is
one
with
aspect
is
rather
many
it
with
is
is
that
not condition
is
not
is
condition

Figure 46:

the reasons why these illustrations can only be approximations of the internal representation of the network. The network for the second sentence in this example defines a relationship for *'not condition'* and this relationship is in turn linked to the concept *'one'*. An attempt is made here by representing the relationship in a circle and its connectivity with dotted lines, but this loses the effect of the actual representation, namely that the actual relationship defined in the lexicon has its own relationships. Internally, this allows a very rich representation that unfortunately cannot be shown in this diagram.

Processing the second sentence of this segment also illustrates the application's attempt to process texts which contain errors. This implementation is not programmed to assume missing content in a sentence.[131] As a result the programme proceeds with the implicit assumption that there is no missing content in the sentence. In this case the missing 'is' (the sentence should end '... that one *is* just born with') would define an explicit relationship between '*condition*' and '*born*' as is indicated in the relationship structure generated for the corrected sentence (Figure 47).

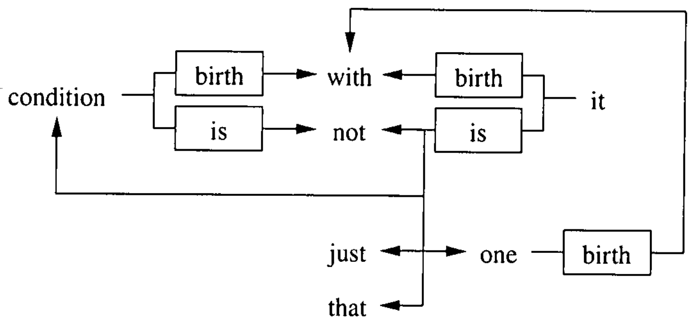


Figure 47: It is not a condition that one *is* just born with.

D.3 Text 3 and text 4

These two texts are drafts of the same essay. The two drafts are included to indicate the process of comparison when there is a high degree of similarity between two texts.

[131]In cases such as this, there are enough cues in the dependencies of the surrounding words to make this possible. It would, however, involve further recursive structures that would slow performance. As a result no attempt has been made to allow this.

D.3.1 Text 3

Xenophobia is defined as the irrational dislike or fear of people from other countries. I do not believe that one wakes up one morning deciding to be Xenophobic. I believe that Xenophobia is casual. Personally I have a fear of heights. This is because I almost fell out of a six story building once. Ever since then I detest high places.

Xenophobia cannot be seen as a neatly formed diagram that can be broken down and analysed. One can speculate about many sources where Xenophobia could stem from. These include negative media stereotyping or family or cultural intolerances towards "outsiders". Our perception of other people is often influenced by premature assumptions about certain cultures and their countries.

Racist parents can enforce racist beliefs on their children. This causes them to form hatred towards others. This could eventually lead to warring attitudes between cultures. On the other hand one could develop a fear for a certain racial group. If your family believes that all Africans are criminals you may develop a fear of all Africans.

The strange thing is that the world is one enourmous global village with various cultures, races, religions and languages. These are mixed across national and international borders. South Africa boasts a very diverse culture. There are the eleven official languages and their cultures. There are also many other diverse families living here from other countries also. They are all South African as well.

Xenophobia is a last desperate attempt to hide behind a shield when people feel insecure or threatened. Therefore Xenophobia should be rephrased. "It is the intense fear or dislike of anyone who threatens ones sense of security and normalcy."

D.3.2 Screen output for text 3

This network contains 411 nodes and took 1 minute and 39 seconds to process (42 nodes discarded as below the threshold weight).

```
1> [Xenophobia is defined as the irrational dislike or fear of
people from other countries .]
```

```
Done with syntax - 11 possible trees
2> [I do not believe that one wakes up one morning and decides
to be Xenophobic .]
Done with syntax - 64 possible trees
3> [I believe that Xenophobia is casual .]
Done with syntax - 5 possible trees
4> [Personally I have a fear of heights .]
Done with syntax - 1 possible trees
5> [This is because I almost fell out of a six story building once
.]
Done with syntax - 117 possible trees
6> [Ever since then I detest high places .]
Done with syntax - 6 possible trees
7> [.]
7> [Xenophobia cannot be seen as a neatly formed diagram that can
be broken down and analysed .]
Done with syntax - 25 possible trees
8> [One can speculate about many sources where Xenophobia could
stem from .]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
5 potential error levels Trying parses with 1 dependency errors

Trying parses with 2 dependency errors
Trying parses with 3 dependency errors
Trying parses with 4 dependency errors
9> [These include negative media stereotyping or family or cultural
intolerances towards " outsiders " .]
Relaxing boundary rules
Some dependencies violate boundaries
In tree:  [adj (d4)], [v (d2)], [adj (d2)], [cn (d4)], [cn (d4)],
[conj (d7)], [cn (d7)], [conj (d9)], [adj (d10)], [cn (d10)], [prep
(d11)], [punc (d11)], [cn (d13)], [punc (d13)], [punc (d15)],
In tree:  [adj (d4)], [v (d2)], [adj (d2)], [cn (d4)], [cn (d4)],
[conj (d9)], [cn (d7)], [conj (d10)], [adj (d10)], [cn (d10)],
[prep (d11)], [punc (d11)], [cn (d13)], [punc (d13)], [punc (d15)],

[conj at 6] - dependency at 9
Done with syntax - 2 possible trees
10> [Our perception of other people is often influenced by premature
assumptions about certain cultures and their countries .]
"assumptions" is not defined in LEXICON
Going to try "assumptions" as possible [cn, ]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
8 potential error levels
Trying parses with 1 dependency errors
Trying parses with 2 dependency errors
Trying parses with 3 dependency errors Trying parses with 4 dependency
errors
Trying parses with 5 dependency errors
Trying parses with 6 dependency errors
Trying parses with 7 dependency errors
11> [.]
11> [Racist parents can enforce racist beliefs on their children
.]
Done with syntax - 2 possible trees
12> [This causes them to form hatred towards others .]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
3 potential error levels
Trying parses with 1 dependency errors
Trying parses with 2 dependency errors
13> [This could eventually lead to warring attitudes between cultures
.]
```

```
"eventually" is not defined in LEXICON
Going to try "eventually" as possible [adj, adv, ]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
3 potential error levels
Trying parses with 1 dependency errors
Trying parses with 2 dependency errors
14> [On the other hand one could develop a fear for a certain racial
group .]
"racial" is not defined in LEXICON
Going to try "racial" as possible [adj, cn, ]
Done with syntax - 12 possible trees
15> [If your family believes that all Africans are criminals you
may develop a fear of all Africans .]
Done with syntax - 412 possible trees
16> [.]
16> [The strange thing is that the world is one enourmous global
village .]
"enourmous" is not defined in LEXICON
Going to try "enourmous" as possible [adj, cn, ]
Done with syntax - 23 possible trees
17> [One with various cultures , races , religions and languages
.]
Done with syntax - 2 possible trees
18> [These are mixed across national and international borders
.]
"mixed" is not defined in LEXICON
Going to try "mixed" as possible [pp, vp, ]
"national" is not defined in LEXICON
Going to try "national" as possible [adj, cn, ]
Done with syntax - 8 possible trees
19> [South Africa boasts a very diverse culture .]
Relaxing boundary rules
```

```
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
0 potential error levels
Unable to build any sense threads in this sentence.
Listing primary concepts in sense structure without defined relationships

20> [There are the eleven official languages and their cultures
.]
"eleven" is not defined in LEXICON
Going to try "eleven" as possible [adj, adv, cn, i, ip, pn, pp,
v, vp, ] Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
5 potential error levels
Trying parses with 1 dependency errors
Trying parses with 2 dependency errors
Trying parses with 3 dependency errors
Trying parses with 4 dependency errors
21> [There are also many other diverse families living here from
other countries also .]
Done with syntax - 16 possible trees
22> [They are all South African as well .]
Done with syntax -
15 possible trees
23> [.]
23> [Xenophobia is a last desperate attempt to hide behind a shield
when people feel insecure or threatened .]
Done with syntax - 108 possible trees
24> [Therefore Xenophobia should be rephrased .]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
5 potential error levels Trying parses with
1 dependency errors Trying parses with
```

```
3 dependency errors
Trying parses with 4 dependency errors
25> [" It is the intense fear or dislike of anyone who threatens
ones sense of security and normalcy .]
"normalcy" is not defined in LEXICON
Going to try "normalcy" as possible [cn, ]
Done with syntax - 15 possible trees
```

D.3.3 Text 4

Xenophobia is defined as the intense dislike or fear of people from other countries. I do not believe that one wakes up one morning deciding to be Xenophobic. I believe that Xenophobia is casual. Personally I have a fear of heights. This is because I almost fell out of a six story building once. Ever since then detest high places.

Xenophobia could stem from many sources. These can be negative media stereotyping or family and cultural intolerances towards outsiders. Our perception of other people is often influenced by premature assumptions about certain cultures and their countries.

Racist parents can enforce racist beliefs on their children. If a specific family believes that all Africans are criminals one may develop a fear of all Africans.

The strange thing is that the world is one enourmous global village with various cultures, races, religions and languages. These are mixed across national and international borders. South Africa boasts a very diverse culture. There are the eleven official languages and their cultures. There are many other diverse families from other countries living as well. They are all also South African as well.

Xenophobia is a shield for people to hide behind when they feel insecure or threatened. Therefore Xenophobia should be defined as the intense fear or dislike of anyone who threatens ones sense of security.

D.3.4 Screen output

This network contains 328 nodes and took 1 minute and 44 seconds to process (37 nodes discarded as below the threshold weight).

```
1> [Xenophobia is defined as the intense dislike or fear of people
from other countries .]
Done with syntax - 11 possible trees
2> [I do not believe that one wakes up one morning and decides
to be Xenophobic .]
Done with syntax - 64 possible trees
3> [I believe that Xenophobia is casual .]
Done with syntax - 5 possible trees
4> [Personally I have a fear of heights .]
Done with syntax - 1 possible trees
5> [This is because I almost fell out of a six story building once
.]
Done with syntax - 117 possible trees
6> [Ever since then detest high places .]
Done with syntax - 3 possible trees
7> [.]
7> [Xenophobia could stem from many sources .]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
3 potential error levels
Trying parses with 1 dependency errors
Trying parses with 2 dependency errors
8> [These can be negative media stereotyping or family and cultural
intolerances towards outsiders .]
Done with syntax - 1 possible trees
9> [Our perception of other people is often influenced by premature
assumptions about certain cultures and their countries .]
"assumptions" is not defined in LEXICON
Going to try "assumptions" as possible [cn, ]
```

```
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
8 potential error levels
Trying parses with 1 dependency errors
Trying parses with 2 dependency errors
Trying parses with 3 dependency errors
Trying parses with 4 dependency errors
Trying parses with 5 dependency errors
Trying parses with 6 dependency errors
Trying parses with 7 dependency errors
10> [.]
10> [Racist parents can enforce racist beliefs on their children
.]
Done with syntax - 2 possible trees
11> [If a specific family believes that all Africans are criminals
one may develop a fear of all Africans .]
"specific" is not defined in LEXICON
Going to try "specific" as possible [adj, ]
Done with syntax - 707 possible trees
12> [.]
12> [The strange thing is that the world is one enourmous global
village .]
"enourmous" is not defined in LEXICON
Going to try "enourmous" as possible [adj, cn, ]
Done with syntax - 23 possible trees
13> [One with various cultures , races , religions and languages
.]
Done with syntax - 2 possible trees
14> [These are mixed across national and international borders
.]
"mixed" is not defined in LEXICON
Going to try "mixed" as possible [pp, vp, ]
"national" is not defined in LEXICON
```

```
Going to try "national" as possible [adj, cn, ]
Done with syntax - 8 possible trees
15> [South Africa boasts a very diverse culture .]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
0 potential error levels
Unable to build any sense threads in this sentence.
Listing primary concepts in sense structure without defined relationships

16> [There are the eleven official languages and their cultures
.]
"eleven" is not defined in LEXICON
Going to try "eleven" as possible [adj, adv, cn, i, ip, pn, pp,
v, vp, ]
Relaxing boundary rules
Unable to determine basic structure of this sentence.
Using best fit of the possibilities
5 potential error levels
Trying parses with 1 dependency errors
Trying parses with 2 dependency errors
Trying parses with 3 dependency errors
Trying parses with 4 dependency errors
17> [There are many other diverse families from other countries
living as well .]
Done with syntax - 8 possible trees
18> [They are all also South African as well .]
Done with syntax - 9 possible trees
19> [.]
19> [Xenophobia is a shield for people to hide behind when they
feel insecure or threatened .]
Done with syntax - 27 possible trees
20> [Therefore Xenophobia should be defined as the intense fear
or dislike of anyone who threatens ones sense of security .]
```

```
Done with syntax - 60 possible trees
```

D.3.5 Generated sense network

This is based on the first three sentences from the text:

> *Xenophobia is defined as the intense dislike or fear of people from other countries. I do not believe that one wakes up one morning deciding to be Xenophobic. I believe that Xenophobia is casual.*

Only one diagram (Figure 48) is provided for texts 3 and 4 as this segment is the same in both texts.[132]

The word '*one*' is used in two senses in the second sentence. This particular usage is not handled well in the relationship network as there is little in the resulting network that distinguishes between the two usages. As a result there is a cyclic relationship defined for '*one wake one*'. This shortcoming applies to generated network as the programme has to recognise the distinction to be able to parse the sentence. If enough detail is added to the lexicon to enable the programme to determine the targets of external references then this shortcoming will be far less evident as referents would be replaced in the lexicon with the targets of the reference.

As with Figure 46, Figure 48 suffers at the hands of two-dimensional representation. Some effort has been made to indicate instances where references have in turn been used as nodes in the process of constructing richer representations of the sense of the text. However this cannot capture the potential for linkage. In this extract in particular (partly because of the failure to build an accurate representation of the second sentence) the process of using relationships in the process of building further relationships extends for several levels. This results in complex nodes, such as '*not belief that one decide*', which have significance for the internal representation of the sense of a text but are impractical to represent in this 2-dimensional representation.

This text also demonstrates the limitation of the current method of processing commas or sentences which contain many conjunctions. These cause numerous forks in the processing sequence. As a result a sentence with many commas will be extremely slow to process. One of the few editing changes in these two texts, for example, was to sentence (16) which originally read

[132]Section 5.4 on page 168 includes a discussion of this similarity.

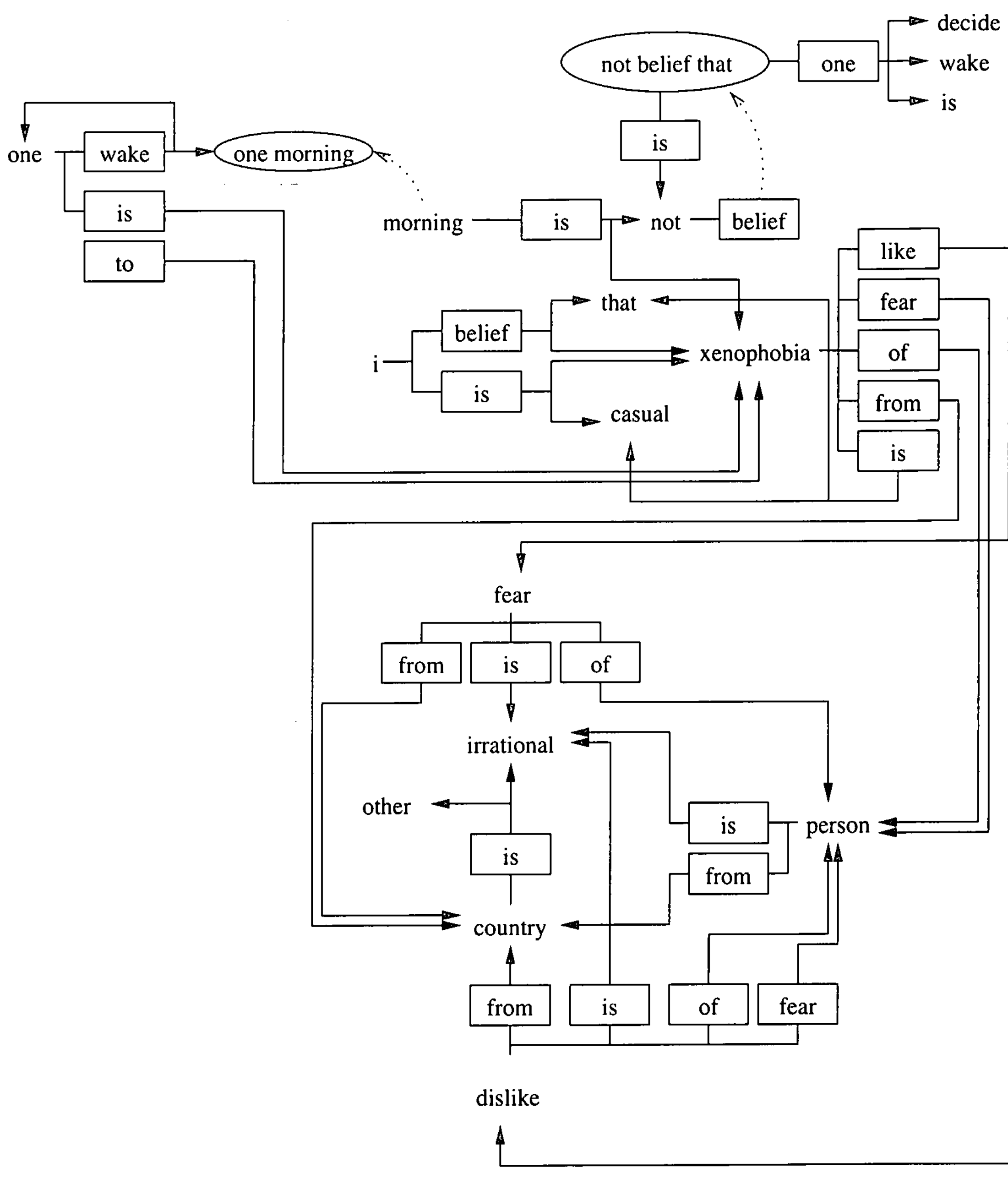

not belief that
one
decide
wake
is
one
wake
one morning
is
to
morning
is
not
belief
like
fear
of
from
is
that
belief
i
xenophobia
is
casual
fear
from
is
of
irrational
other
is
is
person
from
country
from
is
of
fear
dislike

Figure 48:

> *The strange thing is that the world is one enourmous global village with various cultures, races, religions and languages.*

Editing split this into two shorter sentences. The application can process this sentence. However, as there is only primitive support for commas at present, this sentence alone takes approximately 19 minutes to process.

E Programme manual

E.1 Introduction

The applications that form part of this dissertation were written in the Perl programming language. This means they will run on any system that includes the Perl scripting environment. The Perl environment is freely available and versions exist for a wide range of operating systems, including a number of flavours of UNIX, Apple Macintosh and Microsoft.

If you are using or have access to a UNIX-like operating environment then Perl is probably already installed on the system. Your system-administrator will be able to tell you if this is installed or will be able to install it.

A version of the perl operating environment (freely available from ActiveState at `http://www.activestate.com/Products/ActivePerl/`) is supplied on the CD that accompanies this dissertation in the directory `bin`. The perl binaries are packaged to install using Microsoft's Installer which can also be found in the `bin` directory.

The version of Windows Installer provided on the CD (INSTMSIA.EXE) is for Microsoft Windows 95, 98 or ME. A version for Windows NT SP4 and Windows 2000 (called INSTMSIW.EXE) is available for download from Microsoft's download site `http://www.microsoft.com/downloads/`. Windows Installer is already part of the Windows XP OS.

E.2 Installation

E.2.1 Windows 95, 98 and ME only

- Install the Windows Installer by running the programme INSTMSIA.EXE which can be found in `bin` directory of the CD.

  (The latest version of this programme is available from `http://www.microsoft.com/downloads/`).

- Restart your computer if requested to do so and continue with section E.2.2.

E.2.2 All Windows versions

- Install the Perl package by opening the file ActivePerl-5.6.1.638-MSWin32-x86.msi using Windows Installer.

- ActivePerl should correctly configure itself for your system. In the unlikely event that it does not, you may have to add the location of the Perl executable to your PATH in the file '*autoexec.bat*' as in

  ```
  PATH = %PATH;c:\Perl
  ```

  (assuming that 'c:\Perl' is the directory you chose for the Perl installation).

- Restart your computer if requested to do so or if you have modified your '*autoexec.bat*' file.

- At this point you can probably run the scripts in the `scripts` directory of the CD. If however you choose to copy these onto your computer make sure that all the modules (filenames ending in *.pm) are in the same directory as the scripts (files ending in *.pl).

E.2.3 Apple Macintosh

The application has not been tested in a Macintosh environment. However, Perl scripts are one of the most portable programming environments currently available (cf (Raymond, 2003, Ch 17)) so the scripts will probably run without any problems on this system provided there is a Perl interpreter for Macintosh installed. MacPerl, a Perl interpreter Macintosh, can be obtained from `http://www.macperl.com/`.

E.2.4 Unix

The scripts were written on a number of Linux systems and so should function on any compatible system. The first line of the files *analise.pl* and *compare.pl* should point to the location of the Perl binary (usually `/usr/bin/perl`). If Perl is installed in a different directory, you may have to change these lines to reflect the location of the perl binary on your system.

E.3 Running the software

E.3.1 Overview

The scripts use the command line as interface.

To execute one of the scripts:

- Change to the directory containing the scripts - on a Windows system, execute the instruction

  ```
  cd e:\scripts
  ```

  if you are running the scripts from the CD and your CD-drive is 'e:'.

- To run a script - use the format

  ```
  perl script [options] [file1] [[file2]]
  ```

  The analise.pl script would thus be started with the instruction

  ```
  perl analise.pl [file]
  ```

- To redirect output to a file add a redirection instruction in the form

  ```
  > output.txt
  ```

  An example of redirecting output for the *analise.pl* script would be

  ```
  perl analise.pl input.txt > output.txt
  ```

  Note: This may not work in earlier versions of Windows. Support for pipes of this nature is relatively new to the Windows environment. If redirecting output is not permitted, a similar effect can be achieved by setting the value for $::LOGGING to 1 in analyse.pl and/or compare.pl. This line should thus read

  ```
  $::LOGGING = 1;
  ```

E.3.2 *analise.pl*

This script has two modes - an interactive mode and a batch mode. The interactive mode provides a simple interface that allows one to type individual sentences or load files for processing. The batch mode processes a file from the command line and then exits immediately.

A set of configuration options is available in the file '*rules.pm*' which governs the degree to which analise.pl reports grammatical errors and the detail provided for these reports.

Some also enable portions of the programme which may allow greater accuracy in representing the relationships of words not defined in its lexicon but at the cost of extremely long processing runs and extreme memory use (the analysis script uses multi-dimensional hashed-array structures to hold much of its processing information and *Perl* preallocates fairly large amounts of memory for these). Enabling these options can cause memory problems on computers with limited resources.

E.3.3 *compare.pl*

This file provides basic utilities for inspecting the relationship networks generated by *analise.pl*. All this application's functions are accessed from the command line.

Available options are:

-Wx show only those nodes with weights above x (with no space between the W and the number)

-Mx show only those nodes with weights below x (with no space between the M and the number)

-i generate an intersection network from two files (the shared nodes are returned with the relationships of the first file in the sequence)

-d generate a difference network from two files (the shared nodes are returned with the relationships of the first file in the sequence)

-u generate a network containing all the nodes from both files (relationships in shared nodes are combined)

-s scan the nodes in a network for similarities

To use the application enter a command along the lines of:

```
perl compare.pl -W100 [input_file]
perl compare.pl -s -W200 [input_file]
perl compare.pl -W100 -i [file1] [file2]
perl compare.pl -W100 -d [file1] [file2]
```

To save the output use the redirection pipe as in:

```
perl compare.pl -W100 [input_file] > [new_file]
perl compare.pl -s -W200 [input_file] > [new_file]
perl compare.pl -W100 -i [file1] [file2] > [new_file]
perl compare.pl -W100 -d [file1] [file2] > [new_file]
```

References

Arisland K.Ø. 1994. The good, the bad, and the unusual in computer assisted learning. URL: http://www.ifi.uio.no/~ftp/publications/others/KOArisland-1.html.

Ashwood D. 1996. Hypermedia and CALL. In Pennington (1996), pages 79–95.

Batali J. 1998. Computational simulations of the emergence of grammar. In Hurford J., Studdert-Kennedy M., and Knight C., editors, *Approaches to the Evolution of Language - Social and Cognitive Bases*, pages 405–426. Cambridge University Press, Cambridge. URL: http://cogsci.ucsd.edu/~batali/papers/grammar.ps.

Bayne S.R. 2000. Grice's theory of meaning. URL: http://umd.edu/~traum/CA/Papers/bayne.ps.

Beckwith R. and Miller G.A. 1990. Implementing a lexical network. *International Journal of Lexicography*, 3(4):302–312. URL: ftp://ftp.cogsci.princeton.edu/pub/wordnet/5papers.ps. Revised: 1993.

Bell R.T. 1981. *An Introduction to Applied Linguistics: Approaches and Methods in Language Teaching*. St. Martin's Press, New York.

Bornstein D.D. 1977. *An Introduction to Transformational Grammar*. Winthrop Publishers, Cambridge, Mass.

Borsley R.D. 1991. *Syntactic Theory: A Unified Approach*. Edward Arnold, London.

Borsley R.D. 1996. *Modern Phrase Structure Grammar*. Blackwell Publishers Ltd., Cambridge, MA.

Bresnan J. 2001. *Lexical-Functional Syntax*. Blackwell, Oxford.

Briscoe E. 2000. Grammatical acquisition: Inductive bias and coevolution of language and the language acquisition device. URL: http://citeseer.nj.nec.com/briscoe00grammatical.html.

Bruckman A. 1994. Programming for fun: Muds as a context for collaborative learning. URL: ftp://ftp.cc.gatech.edu/pub/people/asb/papers/necc94.ps.

Brumfit C., Phillips M., and Skehan P., editors. 1985. *ELT Documents 122: Computers in English Language Teaching*. Pergamon, Oxford.

But D.P. 1999. August, Muds as social learning environments. URL: `http://imaginaryrealities.imaginary.com/volume2/issue8/learning.html`.

Chapelle C.A. 1997. Call in the year 2000: Still in search of research paradigms. *Language Learning & Technology*, 1(1):19–43. URL: `http://polyglot.cal.msu.edu/llt/vol1num1/chapelle/default.html`.

Chapelle C.A. 1998. Multimedia call: Lessons to be learned from research on instructed sla. *Language Learning & Technology*, July, 2(1):22–34.

Chen J.F. and Warden C.A. 1999. A review of computer error correction in the efl research. URL: `http://www.cyit.edu.tw/~warden/`.

Chomsky N. 1972. *Language and Mind*. Harcourt Brace Jovanovich, New York, enlarged edition.

Chomsky N. 1981. *Lectures on Government and Binding*. Foris Publications, Dordrecht.

Chomsky N. 1995. *The Minimalist Program*. MIT Press, Cambridge, MA.

Christer Samuelsson B.L. 1998. Linguistic theory instatistical learning. In Powers D., editor, *NeMLaP3/CoNLL98: New Methods in Language Processing and Computational Natural Language Learning*, pages 83–89. ACL. URL: `http://citeseer.nj.nec.com/585476.html`.

Christiansen M.H. 1994. *Infinite Languages, Finite Minds: Connectionism, Learning and Linguistic Structures*. PhD thesis, University of Edinburgh.

Christochevsky S. 1997. Teachers and multimedia. In Baggot L., editor, *Proceedings of CAL-97, University of Exeter*. URL: `http://www.media.uwe.ac.uk/masoud/cal-97/papers/christ.htm`. Paper No 158.

Colburn C.M. 1998. Online strategic interaction: Esl role-playing via internet relay chat. *The Internet TESL Journal*, June, 4(6).

Covington M.A., Nute D., and Vellino A. 1988. *Prolog Programming in Depth*. Scott, Foresman, Glenview, IL.

Cruttenden A. 1994. *Gimson's Pronunciation of English.* Edward Arnold, London, fifth edition.

Crystal D. 1997. *A Dictionary of Linguistics and Phonetics.* Blackwell, Oxford, fourth edition.

Darian S. 2001. Adapting authentic materials for language teaching. *Forum*, 39(2):2–9.

de Beaugrande R. 1980. *Text, Discourse and Process.* Longman, London.

de Beaugrande R. and Dressler W. 1994. *Introduction to Text Linguistics.* Longman, New York.

Dowd K. 2004. Brainhat: Natural language environment. URL: `http://www.brainhat.com`.

Dumitrescu V. 2000. Authentic materials: Selection and implementation in exercise language teaching. *Forum*, 38(2):20–22.

Dyer P. 1983. *Instructional procedures for implementing the Strategic Interaction Method in an intensive English as a Second Language program.* PhD thesis, University of Delaware.

Ehsani F. and Knodt E. 1998. Speech technology in computer-aided language learning: Strengths and limitations of a new call parafigm. *Language Learning & Technology*, July, 2(1):45–60.

Eimas P.D., Siqueland E.R., Juscyk P., and Vigorito J. 1971. Speech perception in infants. *Science*, 171:303–306.

Ellis R. and Nobuyoshi J. 1993. Focused communication tasks. *English Language Teaching Journal*, 47:203–210.

Ellis R. 1995. Appraising second language acquisition theory in relation to language pedagogy. In Cook G. and Seidlhofer B., editors, *Principle and Practice in Applied Linguistics*, pages 73–89. Oxford University Press, Oxford.

Fillmore C. 1968. The case for case. *Universals in Linguistc Theory*, pages 1–90.

Fischer M., Maier E., and Stein A. 1994. Generating cooperative system responses in information retrieval dialogues. In *7th International Generation Workshop - Kennebunkprt, Maine*, pages 207–216. URL: `http://citeseer.nj.nec.com/577092.html`.

Gancarz M. 1995. *The Unix Philosophy*. Digital Press, Boston.

Gass S. 1997. *Input, interaction, and the second language learner*. Lawrence Erlbaum Associates Publishers, Mahwah, NJ.

Güngördü Z. 1997. *Incremental Constraint-based Parsing: An Efficient Approach for Head-final Languages*. PhD thesis, University of Edinburgh.

Greyling W.J. 1987. The typicality of classroom talk and its implications for the training of teachers. Master's thesis, University of the Orange Free State.

Hamit F. 1993. *Virtual Reality and the Exploration of Cyberspace*. Sams Publishing, Carmel, Indiana.

Higgins J. 1985. Should teachers learn to programme? In Brumfit et al. (1985), pages 69–77.

Hoffman R. 1996. Computer networks. In Pennington (1996), pages 55–78.

Hofmeister A. and Maggs A. 1984. *Microcomputer Applications in Education and Training*. Holt, Rinehart, and Winston, Sydney, Australian edition.

Holland V.M., Kaplan J.D., and Sabol M.A. 1999. Preliminary tests of language learning in a speech-interactive graphics microworld. *CALICO Journal*, 16(3):339–359.

Holliday L., Lewis N., Morgenthaler L., and Pica T. 1989. Comprehensible output as an outcome of linguistic demands on the learner. *Studies in Second Language Acquisition*, 11:63–90.

Howlett P. 2000. About N.I.C.H.O.L.E. URL: `http://nicole.sourceforge.net`.

Hubbard P.L. 1996. Elements of call methodology: Development, evaluation and implementation. In Pennington (1996), pages 15–32.

Hudson R. 1990. *English Word Grammar*. Oxford, Blackwell.

Hudson R. 1994. Word grammar. In Asher R., editor, *Encyclopedia of Language and Linguistics*, pages 4990–4993. Pergamon, Oxford.

Hudson R. 1998. Word grammar. In Agel V., Eichinger L.M., Eroms H.W., Hellwig P., Heringer H.J., and Lobin H., editors, *Dependency and Valency. An International Handbook of Contemporary Research.* Walter de Gruyter, Berlin. URL: `http://ftp.phon.ucl.ac.uk/home/dick/wg-entry.htm`.

Hurford H.R. 1999. The evolution of language and languages. In Dunbar R., Knight C., and Power C., editors, *The Evolution of Culture*, pages 173–193. Edinburg University Press, Edinburgh. URL: `http://www.ling.ed.ac.uk/~jim/dunbar.knight.power.s.ps`.

Jacobs S. 1999. Authoring software and lesson planning. *CALL-EJ Online*, 1(1).

Kahane S. 2001. What is a natural language and how to describe it? meaning-text approaches in contrast with generative approaches. *Lecture Notes in Computer Science*, 2004:1–?? URL: `http://citeseer.nj.nec.com/478467.html`.

Kahane S. 2002. Hpsg as a true dependency grammar. on a more lexical treatment of extractions. URL: `http://www.linguist.jussieu.fr/~skahane/HPSG.extraction.pdf`. Unpublished draft.

Kahane S. and Gerdes K. 2002. Phrasing it differently. URL: `http://talana.linguist.jussieu.fr/~kim/papiers/PhrasingItDifferentlyFin%al.pdf`.

Keller F. and Asudeh A. 2002. Probabilistic learning algorythms and optimality theory. *CALL-EJ Online*, 33(2).

Kenning M.M. 1990. Computer-assisted language learning. *Language Teaching*, 23:67–76.

Kettunen K. 1999. On modelling dependency-oriented parsing. URL: `http://www.nodali.sics.se/bibliotek/nodalida/1985{\_}hki/NODA85-10/NODA%85-10.txt`.

Kim J.B. 2000. *The Grammar of Negation: A Constraint-Based Approach*, chapter 1. CSLI Publications, Stanford.

Kirby S. 1998. Fitness and the selective adaptation of language. In Hurford J., Studdert-Kennedy M., and Knight C., editors, *Approaches to the Evolution of Language*, pages 359–383. Cambridge University Press, Cambridge.

Kirby S. and Hurford J. 1997a. The evolution of incremental learning: Language, development and critical periods. *Edinburgh Occasional Papers in Linguistics*, 97(2).

Kirby S. and Hurford J. 1997b. Learning, culture and evolution in the origin of linguistic constraints. In Husbands P. and Harvey I., editors, *4th European Conference on Artificial Life*, pages 492–502. MIT Press, Cambridge.

Koedinger K.R. and Ritter S. 1996. An architectue for plug-in tutor agents. *Journal of Artificial Intelligence in Education*, 7:315–347.

König E. 1994. A study in grammar design. In *Proceedings of the International Conference on Computational Linguistics*, Kyoto, Japan. URL: http://citeseer.ist.psu.edu/508729.html.

Koster C.H.A. 1991. Affix grammars for natural languages. URL: http://citeseer.nj.nec.com/koster91affix.html.

Krashen S. 1982. *Principles and practice in second langauge aquisition*. Pergamon, Oxgord.

Lapkin S. and Swain M. 1995. Problems in output and the cognitive processes they generate: A step towards second language learning. *Applied Linguistics*, 16:371–391.

Larsen-Freeman D. and Long M. 1991. *An introduction to second language aquisition research*. Longman, London.

LeLoup J.W. and Ponterio R. 1998. Using www multimedia in the foreign language classroom: Is his for me? *Language Learning & Technology*, July, 2(1):4–10.

Levelt W. 1974. *Formal grammars in linguistics and psycholinguistics*, volume 3. The Hague, Mouton.

Levy M. 1997. *Computer-Assisted Language Learning: Context and Conceptualization*. Clarendon Press, Oxford.

Li M., Li J., Dong Z., Wang Z., and Lu D. 2003. Building a large Chinese corpus annotated with semantic dependency. In Ma Q. and Xia F., editors, *Proceedings of the Second SIGHAN Workshop on Chinese Language Processing*, pages 84–91. URL: http://www.aclweb.org/anthology/W03-1712.pdf.

Little D. 1991. *Learner Autonomy 1: Definitions, Issues, and Problems.* Authentik, Dublin.

Long M.H. 1996. *The role of linguistic environment in second language acquisition.*, pages 413–468. Academic Press, San Diego.

Lyster R. and Ranta L. 1997. Corrective feedback and learner uptake: Negotiation of form in communicative classrooms. *Studies in Second Language Acquisition*, (19):37–61.

Marantz A. 1981. *On the Nature of Gramatical Relations.* MIT Press, Cambridge, Mass.

Marantz A. 1995. *The Minimalist Program*, pages 349–382. Blackwell Publishers, Cambridge, MA.

Marslen-Wilson W.D. 1973. Linguistic structure and speech shadowing at very short latencies. *Nature*, 244:522–523.

McNeil S. 2004a. A hypertext history of instructional design. URL: `http://www.coe.uh.edu/courses/cuin6373/idhistory/ticcit.html`.

McNeil S. 2004b. A hypertext history of instructional design. URL: `http://www.coe.uh.edu/courses/cuin6373/idhistory/plato.html`.

Miller G.A. 2001. March, Ambiguous words. URL: `http://www.kurzweilai.net/meme/frame.html?main=/articles/art0186.html`.

Miller G.A., Beckwith R., Fellbaum C., Gross D., and Miller K.J. 1990. Introduction to wordnet: an on-line lexical database. *International Journal of Lexicography*, 3 (4):235–244. URL: `ftp://ftp.cogsci.princeton.edu/pub/wordnet/5papers.ps`. Revised: 1993.

Morgan J. 1977. Conversational postulates revisited. *Language*, 53:277–284.

N. Garrett N. 1991. Technology in the service of language learning: Trends and issues. *Modern Language Journal*, 75(1):74–101.

Netcraft. 2003. November, Market share for top servers across all domains august 1995 - november 2001. URL: `http://www.netcraft.com/survey/`.

Newmeyer F.J. 1986. *Linguistic Theory in America.* Academic Press, Inc., Orlando, Florida, second edition.

NLL. 2002. Theoretical work (post 1988). URL: `http://www.cs.sfu.ca/research/groups/NLL/2.html`.

Nowak M.A. and Krakauer D.C. 1999. Evolution. *Proceedings of the National Academy of Sciences of the United States of America*, July, 96:8028–8033.

Nyyssöen H. 1995. Grammar and lexis in communicative competence. In Cook G. and Seidlhofer B., editors, *Principle and Practice in Applied Linguistics*, pages 159–170. Oxford University Press, Oxford.

Offereins M. 1994. Interactive learning and natural language systems. In Brouwer-Janse M.D. and Harrington T.L., editors, *Human-Machine Communication for Educational Systems Design*, pages 255–262. Springer-Verlag, Berlin.

Pennington M.C. 1990. The power of the computer. *Language Teaching*, 23:1–14.

Pennington M.C., editor. 1996. *The Power of the CALL*. Athelstan, Houston, TX.

Pennington M.C. and Esling J.H. 1996. Computer-assisted development of spoken language skills. In Pennington (1996), pages 153–189.

Peterson M. 2000. Schmooze university: A virtual learning environment. *Tesl-EJ*, December, 4(4).

Phinney M. 1992. October, Using electronic mail: Exploring our virtual worlds. Paper presented at the Mexico-Caribbean-Latin America Joint TESOL conference. Acapulco, Mexico.

Phinney M. 1996. Exploring the virtual world: Computers in the second language writing classroom. In Pennington (1996), pages 137–152.

Pica T. 1997. Second language teaching and research relationships: A north american view. *Language Teaching Research*, 1(1):48–72.

Plaehn O. 1999. *Probabilistic parsing with discontinuous phrase structure grammar*. PhD thesis, University of Saarland.

Plass J.L. 1998. Design and evaluation of the user interface of foreign language multimedia software: A cognitive approach. *Language Learning & Technology*, July, 2(1):35–45.

Plass J., Chun D., Mayer R., and Leutner D. 1998. Supporting visual and verbal learning preferences in a second language multimedia learning environment. *Journal of Educational Psychology*, 90(1):25–36.

Pohio V. and Turner J. 1995. September, Using text-based virtual reality in the language classroom - a narrative. URL: `http://elicos.qut.edu.au/moo/mpaper.html`. Paper presentend at the 8th Annual ELICOS Association Conference, Esplanade Hotel, Freemantle, Australia.

Pollard C. 1997. Lectures on the foundations of hpsg. URL: `http://www-csli.stanford.edu/~sag/L221a/cp-lec-notes.pdf`. Unpublished manuscript: Ohio State University.

Pollard C. and Sag I.A. 1987. *Information-Based Syntax and Semantics: Fundamentals*. Center for the Study of Language and Information, Stanford, Ca.

Pollard C. and Sag I.A. 1994. *Head-Driven Phrase Structure Grammar*. University of Chicago Press, Chicago.

Powers D.E., Burstein J.C., Chodorow M., Fowles M.F., and Kukich K. 2000. Comparing the validity of automated and human essay scoring. *GRE Board Research Report*, (98-08aR). URL: `http://ftp.ets.org/pub/gre/gre_98-08ar.pdf`.

Prechelt L. 2000. An empirical comparison of c, c++, java, perl, python, rexx, and tcl for a search/string-processing program. URL: `http://wwwipd.ira.uka.de/EIR/`.

Radford A. 1988. *Transformational Grammar: A First Course*. Cambridge University Press, Cambridge.

Radford A. 1997. *Syntactic Theory and the Structure of English*. Cambridge University Press, Cambridge.

Raymond E.S. 2003. The art of unix programming. URL: `http://www.faqs.org/docs/artu/`. to appear.

Reid E. 1995. Virtual worlds: Culture and imagination. *Cybersociety*, pages 165–167.

Ritchie G. 1987. The lexicon. *Linguistic Theory and Computer Applications*, pages 225–256.

Ritter S., Brusilovsky P., and Medvedeva O. 1998. Creating more versatile intelligent learning environments with a computer-based architecture. In Goettl B.P., Halff H.M., Redfield C.L., and Schutte V.J., editors, *Intelligent Tutoring Systems*, pages 554–563, San Antonio, Texas. Proceedings of the 4th International Conference, ITS '98.

Robinson G. 1991. Effective fedback strategies in call: Learning theory and empirical research. In Dunkel P., editor, *Computer-assisted language learning and testing. Research issues and practice*, pages 155–167. Newbury House, New York.

Rope A. 1985. Calling the tune. In Brumfit et al. (1985), pages 65–68.

Schwienhorst K. 1998. May, The "third place" - virtual reality applications for second language learning. URL: `http://www.tcd.ie/CLCS/assistants/kschwien/Publications/eurocall97.htm`.

Shieber S.M. 1986. *An Introduction to Unification-Based Approaches to Grammar.* CSLI Publications, Stanford.

Smith B. 2001. 'readme' file for the Meme Machine v4.x. URL: `http://www.wintermute.demon.co.uk/meme`.

Sperberg-McQueen C.M. and Burnard L. 1994. Guidelines for electronic text encoding and interchange. URL: `http://etext.virginia.edu/TEI.html`.

Stevens V. 1989. A direction for call: From behavioristic to humanistic courseware. In Pennington M., editor, *Teaching languages with computers: The state of the art*, pages pp. 31–43. Athelstan, La Jolla, CA.

Stone A.R. 1995. *The War of Desire and Technology at the Close of the Mechanical Age.* The MIT Press, Cambridge, MA.

Stubbs M. 1983. *Discourse Analysis.* Blackwell, Oxford.

Sun J., Togneri R., and Deng L. 2003. A robust parsing technique for spoken language understanding based o n conceptual relational grammar. URL: `http://citeseer.nj.nec.com/315238.html`.

Swain M. 1985. Communicative competence: Some roles of comprehensible input and comprehensible output in its development. In Gass S.M. and Madden C.G., editors, *Input in second language aquisition*, pages 235–252. Rowley, MA, Newbury House.

Sykes J.B., editor. 1983. *The Concise Oxford Dictionary.* Oxford University Press, New York, seventh edition.

Tesnière L. 1959. *El'ements de syntaxe structurale.* Klienckieck, Paris.

Turing A.M. 1950. Computing machinery and intelligence. *MIND - A Quarterly Review of Phsycology and Philosophy*, LIX(236):433–460. October, URL: `http://www.abelard.org/turpap/turpap.htm`.

Underwood J. 1984. *Linguistics, computers, and the language teacher: A communicative approach.* Newbury House, Rowley, MA.

van Langendonck W. 1994. Determiners as heads? *Cognitive Linguistics*, (5):243–259.

Vincent M. 1985. Appendix: Comments by an efl methodologist. In Brumfit et al. (1985), pages 79–82.

Volosinov V. 1973. *Marxism and the philosophy of language.* Seminar Press, New York. Translated by L. Matejka and I.R. Titunik.

Warschauer M. 1995. *E-Mail for English Teaching.* TESOL inc., Alexandria, Va.

Warschauer M. 1996. Computer-assisted language learning: An introduction. In Fotos S., editor, *Multimedia language teaching*, pages 3–20. Logos International, Tokyo.

Webopedia. 2004. Artificial intelligence. URL: `http://www.webopedia.com/TERM/artificial_intelligence.html`.

Weideman A.J. 1988. *Linguistics: A Crash Course for Students.* PATMOS, Bloemfontein, revised edition.

Weizenbaum J. 1976. *Computer power and human reason: From judgement to calculation.* W. H. Freeman, San Francisco.

Wells R. 1993. The use of computer-mediated communication in distance education: progress, problems, and trends. In Davis G. and Samways B., editors, *Teleteaching: Proceedings of a conference held at the University of Trondheim*, pages 79–88. International Federation for Information Processing (IFPIP), the University of Trondheim and the Norwegian Computer Society.

Widdowson H. 1990. *Aspects of Language Teaching*. Oxford University Press, Oxford.

Winston P.H. 1993. *Artificial Intelligence*. Addison-Wesley, Reading, Mass., third edition.

Woolley D.R. 1994. PLATO: The emergence of on-line ommunity. *Computer-Mediated Communication Magazine*, July, 1(3):5–7.

Worden R. 1998. June, The evolution of language from social intelligence. URL: `http://citeseer.nj.nec.com/context/201865/0`.

Yamamoto S. 2002. Can corrective feedback bring about substantial changes in the learner interlanguage system? *TESOL WebJournal*. URL: `http://www.tc.columbia.edu/academic/tesol/Webjournal/Yamamoto.pdf`.