2002

An investigation of methods for improving accuracy of the SpeechWeb interface.

Linan. Li
University of Windsor

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®

AN INVESTIGATION OF METHODS FOR IMPROVING ACCURACY OF THE SPEECHWEB INTERFACE

By

Linan Li

A Thesis

Submitted to the Faculty of Graduate Studies and Research
through the School of Computer Science
in Partial Fulfillment of the Requirements for
the Degree of the Master of Science at the
University of Windsor

Windsor, Ontario, Canada

ABSTRACT

As with many computer speech applications, speech-recognition accuracy is one of main problems of the SpeechWeb system and requires theoretical research. Many approaches for improving speech-recognition accuracy have been carried out and some results have been found, but there is still a far way to go. A new approach, which involves embedding semantic constraints in the syntax of the recognition grammars has been developed by Frost. This report describes an investigation of the new approach and other potential solutions.

Keyword: SpeechWeb, semantic constraint, recognition grammar, speech-recognition accuracy, attribute grammar, W/AGE, nature-language.

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my advisor, Dr. Richard A. Frost. Without his help, this thesis would not have been possible especially during the spring of 2002. Dr. Frost has always given me concrete, timely and thoughtful instruction even though he is very busy as a Head of Department. In addition, Dr. Frost spent extra time to express his idea to me for whom English is a second language and spent extra time to help me improve my English writing. Finally, I would like to express my appreciate to Dr. Frost for his financial support during these months.

My appreciation also goes to committee members, Dr. Scott Goodwin, Dr. Myron Hlynka, and Dr. Weidong Zhang for having the patience to read drafts and for their attentive, invaluable comments, suggestions and relevant feedback.

I appreciate my wife for her understanding and help. And I would like to thank my classmate, Mr. Zhonglei Huang and my daughter, Amanda W. Li for taking time to test the recognition grammars.

I also would like to say thanks to all the faculty and staff members in School of Computer Science for the instruction and help they have given to me during the last two years.

TABLE OF CONTENTS

1. INTRODUCTION

SpeechWeb is a web of natural-language speech accessible applications. Since 1999, SpeechWeb applications and theoretical research have made much progress. The evolution of SpeechWeb gradually exposed two problems. One is an accuracy problem and another is a user-friendliness problem. This thesis investigates potential solutions and the thesis statement "Speech-recognition accuracy can be improved by the coding of semantic constraints in the syntax of the recognition grammars of the SpeechWeb System."

2. SPEECH-RECOGNITION ISSUES IN GENERAL

2.1 The following is a summary of speech-recognition issues according to Moore [Moore 1999]:

1) High-accuracy speech recognition with moderate to large vocabularies (hundreds to tens of thousands of words) requires a language model.
2) Typically, the language models used in speech-recognition systems pay little attention to the linguistic structure (grammar and semantic constraints) of utterances.
3) Many cases discovered in application development or research have led researchers to try to incorporate language models that reflect more linguistic structure into recognition.
4) There are two major methods for incorporating linguistic structure into the recognition process:

4a) Probabilistic or statistical language models, such as the n-gram model which uses a sliding two-word (bi-gram) or three-word (tri-gram) window to judge the likelihood of a recognition hypothesis, are normally applied using the following instance of Bayes' rule:

$$p(\mathrm{W}|\mathrm{A}) = p(\mathrm{A}|\mathrm{W}) \cdot p(\mathrm{W}) / p(\mathrm{A})$$

This equation characterizes the maximum-likelihood approach to speech recognition. The goal is to find the word sequence W that is most likely given the acoustic

evidence *A*. Bayes' rule expresses this in terms of the probability of *A* given *W*, which is provided by the acoustic model, the prior probability of *W* denoted by *p(W)*, which is provided by the language model, and the prior probability of *A* denoted by *p(A)*. Since the probability of *A* is constant for a particular recognition task, the term *p(A)* can be ignored, and the problem comes down to finding the word sequence *W* that maximizes the product of the estimates for *p(A/W)* and *p(W)*.

4b) In Moore's words: "A purely qualitative language model simply provides a specification of the permitted word strings and the search attempts to find the permitted word string that best matches the acoustics. In practical recognizers, such models are often specified in terms of a finite-state grammar. Moreover, the ability to define grammatical categories makes context-free grammars, a qualitative language model, much more suitable for defining linguistically-based language models, since they can at least model the gross surface structure of natural-language expressions."

5) Moore discussed the disadvantages of probabilistic and qualitative language models as follows:

5a) Disadvantage of probabilistic language models (derived from Moore, 1999):
Probabilistic or statistical language models usually have thousands to millions of parameters that have to be estimated empirically from thousands to millions of words of training data. To be effective, these data have to be from a source that is similar to the language used in the application. A language model trained on newspaper articles would provide a poor fit to the language used in a speech interface to a military command and control system. For novel applications, there may be no reasonable source of language-model training data until the system is built and in use. Moreover, even when application-specific training data is available, there are never enough data to estimate all of the parameters of the language model directly. Because of the sparse-data problem, some parameters of the model always have to be estimated by a combination of lower-order parameters of the model. For

example, in a trigram language model, there are always some trigrams that are never seen in the training data. Their probabilities have to be estimated from bigram and unigram data, to avoid assigning them a probability of 0.

5b) Disadvantages of qualitative language models (derived from Moore, 1999):

5bI) Qualitative language models simply classify strings as possible or impossible. Since in reality no human expert can anticipate all of the meaningful utterances that could arise in a particular application, invariably some strings that are actually uttered will be "out of grammar", and thus be miss-recognized, no matter how strong the acoustic evidence for them might be.

5bII) Purely-qualitative language models are unable to discriminate between two utterance hypotheses that are both classified as possible, even if one is far more likely than the other.

5bIII) Use of purely-qualitative language models makes it impossible to take advantage of robust interpretation strategies that have been developed for natural-language processing. Strategies going by such names as "template matching" or "fragment combining" make it possible to successfully interpret strings, even if they cannot be completely analyzed by the grammar used by the system. If the recognizer is constrained only to produce fully-grammatical hypotheses, these robustness strategies will never have the opportunity to be applied, even in cases where they would be successful if applied to the out-of-grammar string actually uttered.

5bIV) Purely-qualitative language models based on context-free grammars (CFGs) suffer in their inability to model the fine detail of constraints on natural language. Moore states, that the fine detail of constraints can be accommodated by encoding them in the syntax of the context-free grammar. However, this leads to verbose (long) grammars. This verbosity of context-free grammars in describing the details of natural language has led researchers in natural-language processing to augment

context-free grammars in various ways as discussed in section 2.2 to allow grammars to be more concise. Moore said that adding the constraints will increase the size of the grammars too much: "additional information such as this can be expressed in a context-free grammar, but only at the cost of greatly expanding the number of categories and rules" [Moore 1999].

6) Hybrid combinations of the two approaches are possible.

2.2 Methods for accommodating more detailed linguistic constraints (according to Moore)

Moore has investigated many researchers' qualitative-model approaches that are effective for the recognition of within-grammar utterances. For example, (i) in the procedural formalism, Augmented Transition Networks (ATNs) [Woods 1970] were widely used in the 1970s. In this formalism, context-free grammars were represented in the form of recursive transition networks, and the augmentations were defined by procedures attached to the arcs of the networks. (ii) In declarative formalisms, various styles of "unification grammar" have appeared since the mid-1980s. An example of a grammar rule written by SRI International's research group in Cambridge, England and in Menlo Park, California looks like this:

$$\text{S: [tensed = yes]} \rightarrow \text{NP: [person = P, num = N]}$$
$$\text{VP: [tensed = yes, person = P, num = N]}$$

In the notation presented above, grammatical categories are specified in terms of a major category symbol (such as S, NP, or VP), plus a set of feature constraints expressed by equations of the form "feature = value".

What unification grammars add to context-free grammars is the notion of feature constraints. The power of the formalism comes from the ability to constrain a feature not to a specific value, but to a variable that also appears as the value of some other feature, requiring the two features to have the same value, without having to specify what value that is. The name "unification grammar" comes from the fact that when the grammar is applied, the feature constraints are unified.

Moreover, Moore has summarized that a unification grammar, which is a qualitative model, when handling within-grammar utterances, can specify a context-free language more concisely than a context-free grammar, and unification grammars with finitely-valued features are equivalent to context-free grammars in expressive power.

2.3 Moore's methods to combine probabilistic models and qualitative models

Moore states that qualitative-language models are incapable of correctly recognizing out-of-grammar utterances. To use natural-language-based language models in a more robust way, Moore, and others who have had same idea, have investigated many numerical and probabilistic models, and has also developed some new models [Moore 1999].

In 1995, Moore's research group developed a more integrated approach to combining linguistic (qualitative model) and statistical (probabilistic model) factors in a single language model [Moore et al. 1995]. The approach was based on the idea that, even when the grammar fails to provide a complete analysis of an utterance, it is usually possible to find a small number of semantically-meaningful phrases that span the utterance.

In this language model, Moore uses the Gemini system [Moore et al. 1995] to analyze a recognition hypothesis as a sequence of semantically-meaningful fragments, but then they use n-gram statistics to estimate the probability of the hypothesis under that analysis. The resulting language model is a kind of multi-level n-gram model.

Since 1995, the approach has resulted in some improvement in recognition accuracy. Moore's group hopes for additional improvements in recognition accuracy by modeling more of the linguistic structure statistically. Moore recommends the use of "fully-statistical natural-language grammars", which include "probabilistic context-free grammars" and "probabilistic unification grammars" [Moore 1999]. But these approaches are not yet proven.

2.4 Summary of Moore's analysis

Moore makes two important observations:

1) Adding semantic constraints to CFGs increases their size considerably and complicates construction and analysis of the CFGs.
2) Qualitative language models, such as those based on CFGs, cannot recognize out-of-grammar utterances and are therefore not very robust.

Moore suggests various methods to overcome these problems, in particular:

1) Coding semantic constraints in "augmented" grammars and then compiling these augmented grammars into CFGs. This overcomes problem 1 to some extent.
2) Combining probabilistic techniques with qualitative techniques to help accommodate out-of-grammar utterances.

3. THESIS STATEMENT

Although Moore's observations and proposed solutions are appropriate for "large-language" interfaces, we claim that the coding of semantic constraints in CFGs (and the use of the resulting purely-qualitative language models) is appropriate for use in the SpeechWeb system. This is because SpeechWeb divides the application into small language components and the user is expected to know what can or cannot be asked at each node in the navigable web of application nodes. Also the grammars, even with semantic constraints will be small enough to be manageable.

> Speech-recognition accuracy can be improved by the coding of semantic constraints as syntax in the recognition grammars of the SpeechWeb system.

The reasons for our belief in this statement are (i) that the SpeechWeb system is intended to provide interfaces to well-defined language sources, and (ii) that the sources are sub-divided into small hyper-linked sihlos and therefore, users can "learn" the language to be used at each sihlo.

4. SPEECHWEB

4.1 Definition of SpeechWeb

Frost has developed a system called SpeechWeb, which is a web of natural-language speech applications [Frost 2002a]. Applications are located on standard web servers, and are based on natural-language. The "speech browsers" run on remote PCs and use IBM speech-recognition technology. So the most apparent difference from the conventional web is that SpeechWeb applications use natural-language speech access to the data instead of the keyboard, visual scanning and mouse input. This change extends the applications of the conventional web. Because somewhere, sometimes, people cannot use the keyboard, mouse screen or printer for access, and so on, especially for visually-challenged users. And our human speech ability is one of our fundamental behaviors. Humans used speech to communicate before writing, and it is faster than writing.

4.2 An Example Dialogue with SpeechWeb

Here we present an example of a SpeechWeb application, which involves three sihlos called "Solarman", "Monty" and "Judy". Solarman can answer questions about the solar system, Monty is a student at the University of Windsor, and Judy knows

some poems. They were made and are maintained by Frost's natural-language research group at the University of Windsor, Ontario, Canada. The following is a transcript of a dialogue, which took place in March 1999, as part of demonstrations given to high school students visiting the University of Windsor (this example is from [Frost and Chitte 1999]).

```
End user => Hello, is Monty here?
Computer => Hi, yes, I am Monty.
End user => Hello, Monty.
Computer => Hello, how are you?
End user => Fine thanks.
Computer => Good so am I.
End user => How old are you?
Computer => Older than you think but younger than my friend solar
man does.
End user => Who is solarman?
Computer => He knows about the solar system.
End user => Can I talk to solarman?
Computer => Yes I will get him for you.
HYPERLINK FOLLOWED TO THE SOLARMAN SIHLO,
  SOLARMAN GRAMMAR DOWNLOADED,
  AND CHANGE OF VOICE TO THAT OF AN OLDER MAN.
Computer => Hello I am solarman.
End user => What do you know, solarman?
Computer => I know about the planets, moons and people who
discovered them.
End user => Which moons were discovered by Hall?
Computer => Phobos and Deimos.
End user => Which planet is orbited by Deimos?
Computer => Mars.
End user => How many moons were discovered by Hall or Kuiper?
Computer => Four.
End user => Do you know any poems?
Computer => No, but my friend Judy does.
End user => Can I talk to Judy?
Computer => Yes, I shall get her for you.
HYPERLINK FOLLOWED TO THE JUDY SIHLO,
```

```
   JUDY GRAMMAR DOWNLOADED,
   AND CHANGE OF VOICE TO THAT OF A WOMAN.
Computer => Hello, I am Judy, would you like to hear a poem?
End user => Yes, I would like to hear a poem.
Computer => (JUDY READS A SHORT POEM.)
```

4.3 Components of SpeechWeb

The components of SpeechWeb include a speech (voice) browser, which can run on a standard PC, and a web of Sihlos.

- The speech browser

Each end-user has to use a speech browser running on their PC. The prototype of the speech browser is written in Java. It uses the IBM ViaVoice speech-recognition engine and IBM's implementation of the Java speech APIs (Application Programming Interfaces) [Frost 1999]. These browsers accept spoken natural-language input and respond with synthesized voice output. Speech browsers can access the remote application servers by hyperlinked Internet web as other conventional web browsers. In this kind of SpeechWeb system, the role of speech browser is same as other conventional web browsers in conventional web system: taking input data, finding the remote servers, navigating the web, and outputting the data. Actually SpeechWeb is a part of conventional web.

- A web of Sihlos

Sihlos are hyperlinked applications, which reside on servers on the Internet. "Solarman", "Monty", and "Judy" mentioned earlier are sihlos. The whole SpeechWeb just consists of many sihlos.

Inside each sihlo there are two major components, a recognition grammar and an interpreter written by attribute grammar. When an end-user wants to access an application sihlo through a speech browser, the browser first downloads the recognition grammar from the remote application sihlo into the local computer. The recognition grammar downloaded is used by the speech browser to configure the speech recognizers to achieve higher recognition accuracy [Frost and Chitte 1999]. Then if the speech input matches the grammar requirement it is passed into the interpreter, which will return an answer back to the end-user speech browser and output synthesized voice.

4.4 Sample SIHLO code – the Solarman example (constructed by Frost)

4.4.1 the original recognition grammar of Solarman:

solar.gram

```
grammar solar;

public <s> = <linkingvb><termph>[<transvb>by] <termph>{sentence} |
                                        <quest1> <sent> {sentence} |
                                ( who | what ) <verbph> {sentence} |
                ( which | how many ) <nouncla> <verbph> {sentence} |
                                                <simple> {sentence} ;

<simple> = | ask them to be quiet
           | please introduce yourself
           | hello there
           | hello solar man
           | goodbye
           | goodbye solar man
           | fine thanks
```

```
            | thanks
            | thanks solar man
            | yes please
            | what is your mane
            | who are you
            | where do you live
            | what do you know
            | how old are you
            | who made you
            | what is your favorite band
            | who is the vice president at the university of windsor
            | who is the president at the university of windsor
            | who is the executive dean of science at the university
of windsor
            | who is the dean of science at the university of windsor
            | tell me a poem
            | know any poems
            | tell me a joke
            | know any jokes
            | who is judy
            | can i talk to judy
            | can i talk to solar man
            | who is monty
            | can i talk to monty;

<sent> = <termph> <verbph>;

<stermph> = <pnoun> | <detph>;

<termph> = <stermph> | <stermph> ( and | or ) <stermph>;

<verbph> = <transvbph> | <intransvb>;
<transvbph> = ( <transvb> | <linkingvb> <transvb> by ) <termph>;

<detph> = <det> <nouncla>;

<nouncla> = <adj> <cnoun> | <cnoun>;
```

```
<cnoun> = man | men | person | planet | planets | moon | moons;

<adj> = red;

<intransvb> = spin | spins | orbit | orbits | orbited | exist;

<det> = a | an | every | one | two | three;

<pnoun> = Earth | Jupiter | Mars | Mercury | Neptune | Pluto |
          Saturn | Uranus | Venus | Almathea | Ariel | Callisto |
          Charon | Deimos | Dione | Enceladus | Europa | Ganymede |
        Hyperion | Iapetus | Io | Janus | Jupitereighth |
        Jupitereleventh | Jupiterfourteenth | Jupiterninth |
         Jupiterseventh | Jupitersixth | Jupitertenth |
      Jupiterthirteenth | Jupitertwelfth | Luna | Mimas | Miranda |
          Nereid | Oberon| Phobos | Phoebe | Rhea | Saturnfirst |
          Tethys | Titan | Titania | Triton | Umbriel | Bernard |
            Bond | Cassini | Dollfus | Fountain | Galileo | Hall |
        Herschel | Huygens | Kowal | Kuiper | Larsen | Lassell |
         Melotte | Nicholson | Perrine | Pickering;

<transvb> = orbit | orbits | discover | discovered;

<linkingvb> = is | was | are | were;

<quest1>= did | do | does;

public <bye> = Good bye and go to sleep {bye};
```

The Solarman interpreter is constructed as an executable attribute grammar [Frost 1994]. Parts of the code of the Solarman interpreter are given in section 5.2.3.

5. USE OF GRAMMARS IN SPEECHWEB

The functions of SpeechWeb determine that it is necessary and important to use grammars. Speech browsers need the grammar to do speech recognition. Then each application sihlo needs to process queries and return the answers.

5.1 Discussion of Recognition Grammars to 'Restrict the search space'

A recognition grammar is a rule set used to restrict what can be recognized as an input data sequence. In 4.4.1 we listed the original speech-recognition grammar of Solarman. Obviously the role of the recognition grammar is to help speech recognition. Sihlos first must check each word and sentence to see if they match the language rule. This is the first step in language processing. Speech-recognition grammars may reside with each sihlo. It has been mentioned above that speech browser will download this recognition grammar from the sihlo to configure their speech recognizers. Each recognition grammar restricts the search space when spoken input is being processed [Frost 2002a]. Recognition grammars restrict the vocabulary and also the type and structure of sentences. In the industry many speech-recognition engines, for example, the ViaVoice engine IBM Company recommended, use recognition grammars to restrict the search space.

What is the key problem about these recognition activities? It is the accuracy problem!

5.2 Construction of interpreters as executable specifications of attribute grammars

5.2.1 Overview of attribute grammars

When Knuth was at the California Institute of Technology in 1968, he published a paper on Mathematical Systems Theory, with title "Semantics of Context-Free Languages" [Knuth 1968] (Also see [Knuth 1971]). This is a well-known milestone paper in the research of attribute grammars and most researchers think of it is the moment when attribute grammars were born. In his paper, Knuth introduced attribute grammars as a notation for specifying and implementing the static semantics of programming languages. After 22 years, in 1990, Knuth at Stanford University published another paper "The Genesis of Attribute Grammars" to review the history of attribute grammars [Knuth 1990].

Actually we may find the basic concepts of attribute grammars in most computer science textbook on compilers. Here we refer to the basic concepts of AGs from Knuth's paper [Knuth 1968]. The examples are from "Attribute Grammars: Definitions, Systems and Bibliography" [Deransart, P., Jourdan, M., and Lorho, B. 1988] and we also refer to Alblas' instruction [Alblas 1991a] and Paakki' survey [Paakki 1995].

An attribute grammar (AG) consists of three components, a context-free grammar G, a finite set of attributes A and a finite set of semantic rules R: AG = $\langle G, A, R \rangle$. G denotes the syntax of the target language. A and R specify the static semantics of the language.

A context-free grammar G is a quadruple: $G = \langle N, T, P, D \rangle$, where N is a finite set of non-terminal symbols; T is a finite set of terminal symbols; P is a finite set of productions; and $D \in N$ is the designated start symbol of G. An element in $V = N \cup T$ is called a grammar symbol. The productions in P are pairs of the form $X \rightarrow \alpha$, where $X \in N$ and $\alpha \in V^*$, i.e., the left-hand side X is a non-terminal, and the right-hand side α is a string of grammar symbols. An empty right-hand side (empty string) is represented by the symbol ε.

A finite set of attributes A (X) is associated with each symbol $X \in V$. The set A (X) is partitioned into two disjoint subsets, the inherited attributes I (X) and the synthesized attributes S (X). The start symbol and the terminal symbols do not have inherited attributes. Prior to the definition of attribute grammars, synthesized information and purely syntax-directed language processing were considered as the "proper" form of programming language semantics [Irons 1961; Steel 1966]. From a purely theoretical point of view, inherited attributes are unnecessary [Knuth 1968]. They are, however, useful in many practical situations, e.g., in representing symbol tables whose value is typically applied outside of the creating context. Inherited attributes also introduce an attractive conceptual balance into the compilation process, and that is why they are widely employed in systems that generate multi-pass evaluators over the attributed tree [Paakki 1995].

A production $p \in P$, $p: X0 \rightarrow X1 \ldots Xn$ $(n \geq 0)$, has an attribute occurrence $Xi.\alpha$, if $\alpha \in A(Xi)$, $0 = i = n$. A finite set of semantic rules Rp is associated with the production p with exactly one rule for each synthesized attribute occurrence $X0.\alpha$ and exactly one rule for each inherited attribute occurrence $Xi.\alpha$, $1 = i = n$. Thus Rp is a collection of rules of the form $Xi.\alpha = f(y1, \ldots, yk)$, $k = 0$, where (1) either $i = 0$ and $\alpha \in S(Xi)$, or $1 = i = n$ and $\alpha \in I(Xi)$; (2) each yj, $1 = j = k$, is an attribute occurrence in p; and (3) f is a function, called a semantic function, that maps the values of y1, ..., yk to the value of $Xi.\alpha$. In a rule $Xi.\alpha = f(y1, \ldots, yk)$, the

occurrence Xi.α depends on each occurrence yj, 1= j = k. By the definition, synthesized attributes are output by the left-hand-side symbols of productions, and inherited attributes are given to the right-hand-side symbols. (The synthesized attributes of terminal symbols are assumed to be externally defined.)

Example [Deransart, P., Jourdan, M., and Lorho, B. 1988]:

The following example exhibits previous definitions to define the conversion of a bit string into its decimal value (the original example is given in [Knuth 1968]). B, L, and D are standing respectively for bit, list of bits, and digital numbers.

G = {N, T, P, D}

N = {D, L, B}

T = {0, 1, .}

P = {D ::= L . L

D::= L

L ::= LB

L ::= B

B ::= 0

B ::= 1}

Attr = {l, v} Syn = {l, v}, where vX is the decimal value of the string derived from X. lX is the length of the string derived from X. Attr (D) = {v}, Attr (L) = {v, l}, Attr (B) = {v}. For example, the string 1101.01 receives the following structure (Figure 1).

Attribute definitions:

D ::= L1 . L2	$vD = vL1 + vL2 / 2^{lL2}$
D::= L	vD = vL
L1 ::= L2B	vL1 = 2(vL2) + vB, lL1 = lL2 +1
L ::= B	vL = vB, lL = 1
B ::= 0	vB = 0
B ::= 1	vB = 1

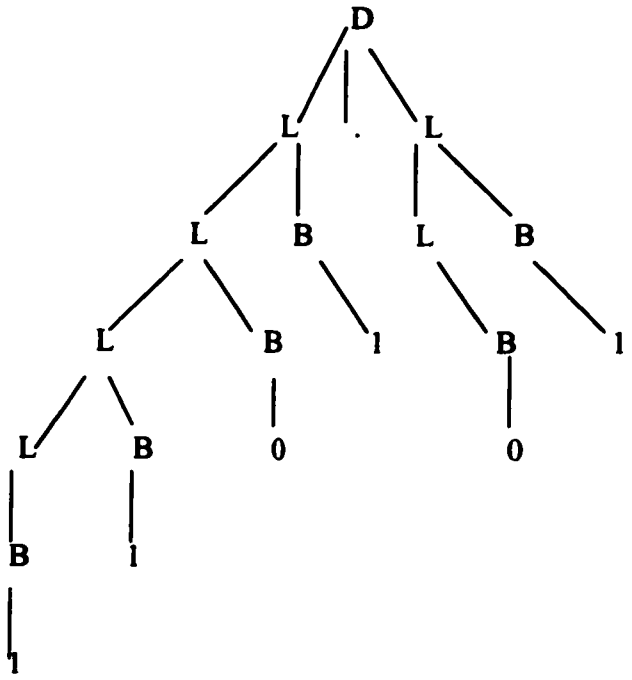

Figure 1

Figure 2 show that the binary number 1101.01 means 13.25 in decimal notation.

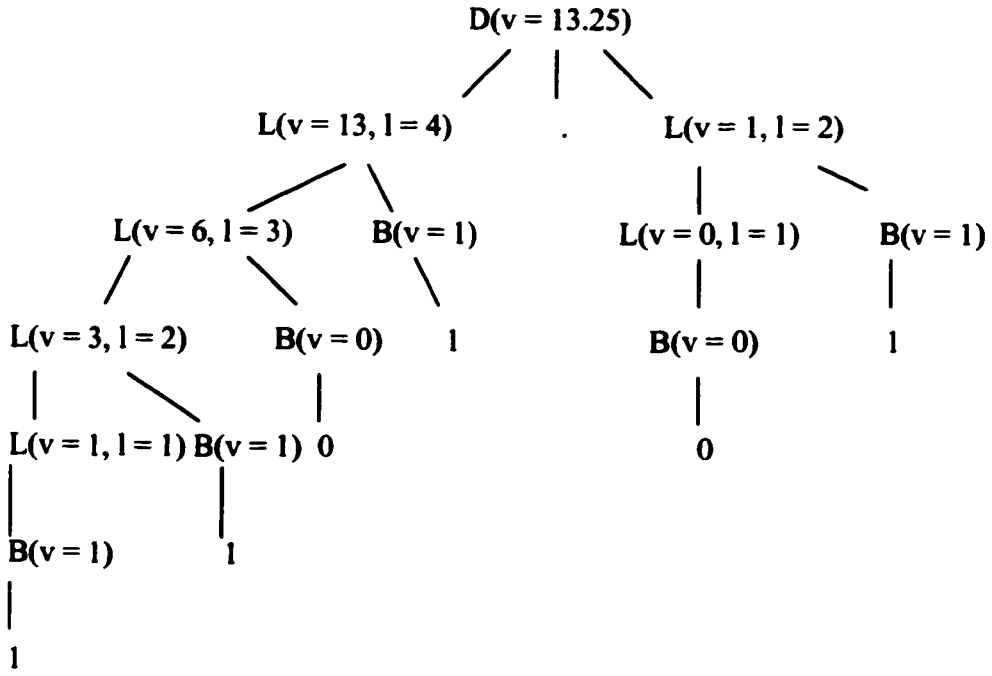

Figure 2

Actually the implementation of programming languages is the original and most widely-used application area of attribute grammars. Moreover they can also be used in many other fields. Paakki [Paakki 1995] has listed the following application areas: General software engineering [Shinoda and Katayama 1988], [Frost 1992] and [Lewi, J., De Vlaminck, K., Steegmans, E. and Van Horebeek, J. 1992]; reactive systems [Ding and Katayama 1993]; distributed programming [Kaiser and Kaplan 1993]; logic programming [Deransart and Maluszynski 1985; 1993]; static analysis of programs [Horwitz and Reps 1992]; databases [Ridjanovic and Brodie 1982]; nature language interfaces [Alexin, Gylmothy, Horvath, and Fabricz 1990] and [Frost 1994]; graphical user interfaces and visual programming [Hudson and King 1987] and [Crimi, Guercio, Pacini, Tortora and Tucci 1990]; pattern recognition [Tsai and Fu 1980] and [Trahanias and Skordalakis 1990]; hardware design [Jones and Simon 1986]; computer communication protocols [van de Burgt and Tilanus 1989] and [Chapman 1990] and combinatorics [Delest and Fedou 1992].

However, the applications mentioned above are not all be found in industry. Perhaps the main usage of attribute-grammar technology in industry is for solving compiler problems. In parsers, attribute grammars are used for mainly two tasks: First, for tree construction and similar tasks performed during parsing. Second, for name analysis within semantic analysis, which is performed on the abstract syntax tree. Grosch [Grosch 1999] has discussed in detail where and how attribute-grammars are used in parsers and provides a case study from the PL/I programming language.

5.2.2 Nature language processing: W/AGE overview

(a) W/AGE is an acronym for "Windsor Attribute Grammar programming Environment", which was developed by Frost in the mid-eighties, then extended by Frost and his research group at University of Windsor, Canada, in the 1990s [Frost 1994].

(b) W/AGE is a programming environment that was originally built to facilitate the investigation of approaches to the construction of natural-language database interfaces [Frost 1994]. Now, as a programming environment it enables natural-language recognizers, parsers and evaluators to be constructed as executable specifications of attribute grammars. Actually, the construction of natural-language processors from scratch is very difficult even for experienced programmers. In order to overcome this problem, W/AGE has been developed as a tool, which allows non-programmers to construct natural-language processors as executable specifications of attribute grammars, which define the syntax, and semantics of the input language.

(c) W/AGE consists of a set of higher-order functions that extend the standard environment of a pure lazy functional programming language, Miranda [Turner 1985], (which is a trademark of Research Software Ltd., England) and is available for Unix systems and PCs [Frost and Launchbury 1989]. The underlying parser, which is hidden from the W/AGE user, is based on a top-down recursive-descent backtracking search strategy. As discussed by Frost [Frost 2002c], this approach provides significantly more modularity than alternative techniques. Modularity is essential to simplify the construction, testing, and re-use of attribute grammars.

(d) Straightforward implementation of top-down parsing has exponential worst-case complexity. Frost and Szydlowski [Frost and Szydlowski 1996] have shown how memorization can be used to reduce this complexity to cubic order whilst maintaining pure-functional properties (and thereby declarativeness) of the parser. The parsing strategy is also based on a recursive programming technique [Frost 2002c] in which each component language processor returns a list of results, which are each processed by the subsequent component processor in turn. This approach not only simplified the construction of the W/AGE system, but also accommodates ambiguity.

(e) Miranda is a lazy (non-strict) functional programming language and is well suited for the construction of executable attribute grammars. Owing to the delayed computation, no attribute values are calculated until the parser has completed its task, even though the semantic attribute rules are closely associated with the syntax rules that direct the parser. This "laziness" is necessary in order to process complex queries in reasonable time.

(f) W/AGE has been used to build VLSI design transformers [Frost 1994], theorem provers [Frost and Karamatos 1989] [Frost 1990], and file processors, and a number of natural-language processors such as "solar man" in the SpeechWeb system. A text interface to the solar man query processor and a complete listing of the executable specification can be accessed at:

http://www.cs.uwindsor.ca/users/r/richard/miranda/wage_demo.html

(g) A W/AGE program takes a list of characters as input and returns a list of characters as output. In order to create a W/AGE program, the user has to define the language of the input strings. This definition includes the following eight main components [Frost 2002c]:

1) A set of rules defining the syntax of the input language.

2) A list of types of the semantic attributes (meanings) that are to be computed for different types of words and phrases of the input language.

3) A dictionary defining the words used in the input language. For each word, the user indicates the syntactic category, and the meaning of the word.

4) A set of syntax rules defining how compound phrases are made from simpler components.

5) A set of semantic rules that define how the meaning of a compound phrases is computed from the meanings of its components. Each semantic rule is associated with a syntax rule.

6) A set of definitions of the semantic operators/functions used in the semantic rules.

7) A set of definitions of data.

8) A set of definitions linking the integers representing entities to strings that can be used for output.

The resulting set of definitions constitutes a program which, when executed, will take a string as input and, if the string is a member of the language defined by the syntax rules, will output a string that is computed using the semantic rules.

(h) Advantages

W/AGE programs are completely declarative and the order of the definitions does not matter. W/AGE code is directly executable. Every production rule in the grammar can be executed independently of all others, except those which are used in its definition. This modularity facilitates construction and experimentation with the processor, and re-use of components. For example, new words can be defined as having the same meaning as phrases consisting of words and structures that have already been defined.

5.2.3 An attribute grammar interpreter for the Solar man sihlo

The full program (code) can be seen at the URL given in 5.2.2 (f). The program begins with some sample queries. It is followed by comments defining the syntactic structure of the queries. For example:

```
|| snouncla ::= cnoun | adjs | adjs cnoun
||
|| relnouncla ::= snouncla relpron joinvbph
||              | snouncla
||
|| adjs ::= adj | adj adjs
||
```

etc.

The grammar is followed by the definition of some synonyms for values used to represent entities etc.

```
entity          == num
entityset       == [entity]
string          == [char]
es              == entityset
b               == bool
```

These synonyms are now used in the declaration of attribute types (i.e. meaning types). These declarations state that the meaning of a sentence is a Boolean value, the meaning of a noun clause is a set of entities, the meaning of a termphrase is a function from entity sets to booleans, etc.

```
attribute ::=
 SENT_VAL                  b
| NOUNCLA_VAL              es
| VERBPH_VAL               es
| ADJ_VAL                  es
| TERMPH_VAL               (es -> b)
| DET_VAL                  (es -> es -> b)
| VERB_VAL                 bin_rel
| RELPRON_VAL              (es -> es -> es)
| NOUNJOIN_VAL             (es -> es -> es)
```

etc.

The next part of the program defines special symbols, which may appear in the input language with no space separating them from words.

```
reserved_words  = []
special_symbols = ['.', '?','\n']
```

Next is the dictionary.

```
dictionary =
[("man", "cnoun", [NOUNCLA_VAL set_of_men]),
 ("thing", "cnoun", [NOUNCLA_VAL set_of_things]),
 ("planets", "cnoun", [NOUNCLA_VAL set_of_planet]),
 ("sun", "cnoun", [NOUNCLA_VAL set_of_sun]),
 ("moon", "cnoun", [NOUNCLA_VAL set_of_moon]),
 ("blue", "adj", [ADJ_VAL set_of_blue]),
 ("red", "adj", [ADJ_VAL set_of_red]),
 ("exist", "intransvb", [VERBPH_VAL set_of_things]),
 ("spin", "intransvb", [VERBPH_VAL set_of_spin]),
 ("the", "det", [DET_VAL function_denoted_by_a]),
 ("a", "det", [DET_VAL function_denoted_by_a]),
 ("some", "det", [DET_VAL function_denoted_by_a]),
 ("any", "det", [DET_VAL function_denoted_by_a]),
 ("two", "det", [DET_VAL function_denoted_by_two]),
 ("Bernard", "pnoun", [TERMPH_VAL (test_wrt 55)]),
 ("venus", "pnoun", [TERMPH_VAL (test_wrt 10)]),
 ("discover", "transvb", [VERB_VAL (trans_verb rel_discover)]),
 ("person", "cnoun", meaning_of nouncla "man or woman"),
 ("discoverer", "cnoun", meaning_of nouncla "person who
   discovered something"),
 etc.
```

The next part of the program links the dictionary to the executable attribute grammar by defining basic interpreters such as cnoun to be the processors of all words whose syntactic type is given as "cnoun" in the dictionary.

```
pnoun       = pre_processed "pnoun"
cnoun       = pre_processed "cnoun"
adj         = pre_processed "adj"
det         = pre_processed "det"
etc.
```

The central part of the W/AGE program is a set of attribute grammar rules each of which is a syntax rule with set of associated semantic rules. For example:

```
snouncla = cnoun
           $orelse
           structure (s1 adjs ++ s2 cnoun)
           [a_rule 1( NOUNCLA_VAL $of lhs )
                   EQ intrsct1 [ADJ_VAL $of s1, NOUNCLA_VAL $of s2]]
```

etc.

In some cases there are no explicit semantic rules. For example:

```
termph = pnoun $orelse detph
```

This syntax rule states that a termphrase (termph) consists of a propernoun (pnoun) such as "mars", or a determiner phrase (detph) such as "a moon". No semantic rule is necessary in such cases, as the meaning of the term phrase is the meaning of the proper noun or the meaning of the determiner phrase.

The next part of the program defines the semantic operators that are referred to in the attribute grammar (the attribute evaluation functions). For example:

```
intrsct1 [ADJ_VAL x, NOUNCLA_VAL y] = NOUNCLA_VAL (intersect x y)
```

etc.

In the solar man application, the data are stored as part of the W/AGE program. In general, data can be stored in files or in a database system, provided that an appropriate interface has been developed. For many users who want to provide a natural-language interface to the type of data that they might include on a web page, encoding the data in the W/AGE program is the simplest approach. Entities in the solar man program are represented by integers. They could just as easily be represented by character strings. A sample of the solar man database is given below:

```
set_of_sun          = [8]
set_of_planet       = [9..17]
set_of_moon         = [18..53]
```

```
set_of_men          = [54..70]
set_of_woman        = []
set_of_red          = [12, 13, 14, 22]
set_of_blue         = [11, 14, 15, 16]
set_of_spin         = [8..53]

rel_orbit = [(9,8),(10,8),(11,8),(12,8),(13,8),
(14,8),(15,8),(16,8),(17,8),(18,11),
(19,12),(20,12),(21,13),(22,13),(23,13),
(24,13),(25,13),(26,13),(27,13),(28,13),
(29,13),(30,13),(31,13),(32,13),(33,13),
(34,13),(35,14),(36,14),(37,14),(38,14),
(39,14),(40,14),(41,14),(42,14),(43,14),
(44,14),(45,14),(46,15),(47,15),(48,15),
(49,15),(50,15),(51,16),(52,16),(53,17)]
```
etc.

This states, for example, that the entity represented by 9 orbits the entity represented by 8.

The last user-defined part of the solar man program is a lookup table linking integers representing entities to character strings for output. For example:

```
name_list = [("Bernard", 55),("Bond", 67),("venus", 10),
             ("Cassini", 65),("Dollfus", 63),("Fountain", 62),
             ("Galileo", 56),("Hall", 54),("Herschel", 64),
             ("Huygens", 66),("Kowal", 57),("Kuiper", 69),
etc.
```

The final part of the solar man program consists of a number of re-usable definitions of operators and functions that are used to format input and output.

6. PROBLEMS WITH SPEECH-RECOGNITION ACCURACY

In 1999, Frost demonstrated an earlier version of SpeechWeb (it was called SpeechNet [Frost 1999b]) at the Conference for the American Association for Artificial Intelligence [Frost 1999c]. SpeechWeb applications and theoretical research have had much progress since then [Frost 2002a]. The evolution of SpeechWeb gradually exposed two problems. One is an accuracy problem and another is a user-friendliness problem.

Firstly, SpeechWeb has poor speech-recognition accuracy. This is not a new problem within the speech-recognition field. Although research work in this field continues and much work has been done [Ogden and Bernick 1996] [Moore 1999], speech-recognition accuracy still is very limited. For example, in a simple solar system application, which involves planets, the moons that orbit them, and the people who discovered them, a simple grammar might admit the question "which man orbits kuiper" and therefore can mistakenly "recognize" this when the actual utterance is "which moon orbits Jupiter". Evidently this utterance is syntactically correct, but it is semantically incorrect owing to the fact that people cannot orbit other people at least in the context of this solar application [Frost 2002b].

Secondly, users need to know about links to navigate SpeechWeb. This is an end-user friendliness problem. And this is a special problem of SpeechWeb. The conventional web could show the links on the screen or use a top down, pop up manual to guide users navigating the web. But SpeechWeb must be designed for non-visual operation. There is no help except through speech. Solving this kind of

problem is especially important for SpeechWeb application development. We cannot imagine who will be interested in a Web, which is a hard to navigate.

Thirdly, recognition accuracy cannot only be improved by making Sihlos smaller, as this increases difficulty of navigation, i.e., an increase in recognition accuracy may lead to a decrease in user-friendliness. Each sihlo is only a professional "agent" in the narrow field. This is increasing the recognition accuracy of SpeechWeb by making the recognition grammar smaller. On the other hand, this also increases the difficulty of speech navigation in SpeechWeb, because there are too many sihlos in system. How can the system guide the end-user to navigate in SpeechWeb?

7. POSSIBLE SOLUTIONS TO IMPROVE RECOGNITION ACCURACY

7.1 Restrictions on the input language

The restrictions on the input language are the earliest and also most important method for improving speech-recognition accuracy. Several research investigations have been shown by Ogden and Bernick [Ogden and Bernick 1996] and Frost [Frost 2002b].

7.1.1 Restrictions on vocabulary

Restrictions on vocabulary should be divided into restricting the domain of the vocabulary and restricting the linguistically and phonetically types of the vocabulary.

(a) Restrictions on vocabulary domain: Several studies have shown that a restriction on vocabulary domain is effective [Ogden and Bernick 1996]. Examples are Kelly and Chapanis (1977), and Ford, Weeks and Chapanis (1980), and Michaelis (1980), and Ogden and Brooks (1983). Then, as Kelly and Chapanis, identified a 300-word vocabulary in a particular domain that allowed participants to communicate as effectively as participants who had an unrestricted vocabulary did, i.e. they used a vocabulary restricted to an empirically determined sub-set.

(b) Restrictions on the linguistic and phonetic types of vocabulary: For example [Frost and Chitte, 1999], in order to minimize the perplexity of the input language, noted that recognition systems need a careful selection of words. The perplexity of the input language denotes a measure of the number of alternative words that can follow a given word in the language, as one of the cases is a synonym. Typically, as the perplexity increases, i.e. there are more alternative words following a given word, the recognition accuracy will decrease. In addition, in order to reduce the number of clashes between phonetically similar words, a recognition system also needs a careful selection of words for restricting the use of homonym.

For example, consider the query: "which moon orbits Mars" we may denote it like:

Which moon orbits <noun>

The <noun> includes Mars and all nouns in the system. The perplexity of <noun> is very high (perplexity = number of nouns). If we use "which moon orbits the planet called Mars" instead, i.e.

Which moon orbits <category> called <planet>

In which <category> includes planets, moons, and sun. The perplexity is 3 (perplexity = number of category). And <planet> includes all planets in the solar system. The perplexity of <planet> is also low (perplexity = number of planets = 9). The reduction in perplexity results in a reduction in recognition errors.

7.1.2 Restrictions on syntax

Many research investigations on syntactic restrictions have been carried out. For example [Ogden and Bernick 1996], Hendler and Michaelis, Ogden and Brooks, and Jackson. (1) Hendler and Michaelis "restrictions on syntax" was just that the "grammar was selected to be easily processed by a computer"! (2) Ogden and Brooks developed a syntactic restriction in which the grammar restricted users to questions that first allowed for an optional action phrase (e.g. "What are..." or "List...") followed by a required phase naming a database retrieval object (e.g. "...the earnings...") which could optionally be followed by any number of phrases

describing qualifications (e.g. "...of the last two years"). (3) Jackson compared two restrictions on syntax. One was that an "English-like order with an action was followed by an object description (e.g. 'Find the VW ads')". The second one was that "reverse the order and specify the object first followed by an action (e.g. 'VW ads find')". But "syntax does not seem to matter for constrained languages like this".

Ogden and Bernick's results were that "The results of these laboratory studies of syntactic restrictions suggest that people adapt rapidly to these types of constraints when interacting with a computer". They didn't discuss more deeply.

7.2 Methods of Guess output and Feedback

In many other complex applications, the speech recognition engine may output several 'guesses', which are some text strings ranked in order of confidence. Each guess is an expression in the language defined by the recognition grammar, and the confidence level is determined by the extent to which the input utterance matches the phonetic structure of the grammar-defined expression. Then a semantic analyzer can post-process them to pick the most plausible guess. The semantic analyzer usually identifies the most plausible guess using knowledge captured from previous utterances, or general knowledge about the domain of discourse. Then it keeps on trying again for next most plausible guess until a good "match" is found [Frost 2002b].

According to Frost's investigation [Frost 2002b], in some research speech systems, semantic post-processing is used not only to help find the most plausible of the guesses returned by the recognizer, but also to provide feedback to modify the grammar used to recognize the next utterance. These speech recognition systems

using the feedback from the semantic analyzer modifies the current grammar in real time so that the recognition search space is dynamic during the user input utterance.

7.3 Problems with these approaches

In 7.1 although the restrictions on the input language are the earliest and most important method developed for improving speech recognition accuracy, the input language is unnecessarily and unnaturally restricted. The user is constrained in how a query is stated thereby reducing user-friendliness.

The methods of "Guess output" and "Feedback" sound very similar to human activity and it seems that this kind of "simulation" technique should be respected, but the implementing of "guess" methods complicates the task of building Sihlos, and implementing the "Feedback" method still is a research project.

8. AN ALTERNATIVE PROPOSED SOLUTION

8.1 Construct the application as a set of "small" hyper-linked sihlos (as in SpeechWeb) and then code semantic constraints as syntax in the recognition grammars.

Frost has developed a new approach, which involves the coding of semantic constraints as syntax in the recognition grammar [Frost 2002b]. The main idea of the approach is each non-terminal symbol in a recognition grammar has meaning with it. The particular "meaning" has a semantic domain. The non-terminal symbol of `<transvb>` has the meaning "transitive verb", and the semantic domain covers "discover", "discovers", "discovered", and "orbit", "orbits", "orbited" in this case. The new approach focuses on these semantic domains of the non-terminal symbols. And it divides and classifies some non-terminal symbols into smaller units according to smaller semantic domains such as `<animate_transvb>` and `<inanimate_transvb>`.

For example, a non-terminal symbol of noun phrase, `nounphrase`, includes the whole noun semantic domain including "animate" noun phrase such as man or woman and "inanimate" noun phrase such as moon or planet. But the "animate" noun is quite different from "inanimate" noun in many cases. "A man discovered a moon" is fine, but "a moon discovered a moon" is ridiculous. To avoid this kind of mistake and improve the accuracy of speech recognition, the new approach divides the non-terminal symbol of noun phrase, `nounphrase`, into two sub-non-terminal symbols of

noun phrase, `animatenounphrase` and `inanimatenounphrase`. The sub-non-terminal has a narrower semantic domain than the original one. "`animatenounphrase`" only covers the animate noun domain as man or woman. "`inanimatenounphrase`" only covers the inanimate noun domain as moon or planet.

Evidently, Frost's new approach is totally different from the restrictions on vocabulary and even restrictions on syntax. In Ogden and Bernick's investigation [Ogden and Bernick 1996], Hendler and Michaelis's constraint was just selecting to be easily processed by a computer. Ogden and Brooks' restriction was the "sentence structure constraint" (e.g. A followed by B, then followed by C). Jackson's constraint was also structure restrictions on syntax. Note that Frost's new approach reduces perplexity and subsumes the technique described in 7.1.1.b without restraining the input language unnatural.

8.2 An example of a semantic constraint

To clearly understand semantic constraints, we discuss a simple example. In the solar man application involving planets, the moons that orbit them, and the people who discovered them, we consider the following syntax rule, which accommodates any "which-type" utterance containing any noun phrase, such as "man", "moon", "planet" etc. followed by any verb phrase, such as "orbit mars", "discovered phobos and deimos", etc. For example:

```
question ::= "which" nounphrase verbphrase
```

This syntax rule admits the question "which man orbits kuiper". This utterance is syntactically correct, but it is semantically incorrect owing to the fact that people cannot orbit other people.

Frost states that semantically incorrect expressions can be removed from the recognition search space by coding appropriate semantic constraints in the syntax of the recognition grammar [Frost 2002b]. For example, the rule above can be replaced by the following rules in which "animatenounphrase" etc. could be defined to exclude utterances such as "which man orbits kuiper".

```
question::= "which" animatenounphrase animateverbphrase
        |"which" inanimatenounphrase inanimateverbphrase
```

"animatenounphrase" and "inanimatenounphrase" etc. are the "semantic constraints" as syntax in the recognition grammar.

8.3 How can the semantic constraints be coded in the syntax?

- Determine which non-terminals need to be constrained by their semantics when attempting to improve a recognition grammar or when designing a new one. For the example above, the non-terminals needing constraint are `nounphrase` and `verbphrase`.
- Determine the specific categories of phrases. In the example above, the detail categories of `nounphrase` could be `"animatenounphrase"` and `"inanimatenounphrase"`. The detail categories of `verbphrase` could be `"animateverbphrase "` and `"inanimateverbphrase"`.
- Identify the relationships or combinations of these specific categories of phrases. The example above could have four possible combinations:

1) `animatenounphrase animateverbphrase`

2) `animatenounphrase inanimateverbphrase`

3) `inanimatenounphrase animateverbphrase`

4) `inanimatenounphrase inanimateverbphrase`

But 2) and 3) do not make sense because "Which man orbits Mars?" or "Which moon discover moon?" is semantically wrong. The combinations 1) and 4) are the correct choices. Notice that the original combination:

`nounphrase verbphrase`

covered the all of case1) to 4). This is why the original syntax rule admits the question "which man orbits kuiper".

- Use the correct choice replacing the original syntax rules in recognition grammar.

8.4 Advantages of this approach

a) The input language is reduced in size with minimal effect on the user because the semantic constraints only remove senseless queries. In essence, this integrates some aspects of semantic post-processing directly in the recognition grammar. The technique provides a relatively simple method for improving recognition accuracy without unnaturally restricting the input language.

b) It should be easy to implement.

9. AN EXPERIMENT

To check the effect of improving accuracy by using this new approach, we performed a series of tests using a SpeechWeb sihlo. These tests were carried out independently by three users. Users performed the same testing schedule, used the same hardware devices in same software environment, and applied the same query questions. The grammars tested were: (1) The original Solar man grammar, which is the grammar before adding semantic constraints, (see APPENDIX I for the code); (2) the new grammar, which is the grammar after adding semantic constraints. (See APPENDIX II for the code). A total of 25 queries were used in the test. For each query, the tester must repeat six times and record the six results on the test data-sheet.

9.1 The grammar before semantic constraints

See 4.4.1 the original recognition grammar of solar man is given on page 12 of this report. This is the first version grammar in our test.

9.2 The grammar after semantic constraints

```
grammar solar02;
public <s> = <linkingvb> <terphrase_verbphrase> {sentence}|
                                  is <pnoun> <pnoun> {sentence}|
                              is <pnoun> a <nouncla> {sentence}|
            is <pnoun> a <nouncla> or a <nouncla> {sentence}|
                                  <quest1> <sent> {sentence}|
                          ( who )<animate_verbph> {sentence}|
                       ( what )<inanimate_verbph>{sentence}|
              ( which | how many )<nouncla_verbph>{sentence}|
                                        <simple> {sentence};

<simple> = | ask them to be quiet
           | please introduce yourself
           | hello there
           | hello solar man
           | goodbye
           | goodbye solar man
           | fine thanks
           | thanks
           | thanks solar man
           | yes please
           | what is your name
           | who are you
           | where do you live
           | what do you know
           | how old are you
           | who made you
           | what is your favorite band
           | who is the vice president at the university of Windsor
           | who is the president at the university of Windsor
           | who is the executive dean of science at the university
of Windsor
           | who is the dean of science at the university of Windsor
           | tell me a poem
           | know any poems
           | tell me a joke
           | know any jokes
```

```
            | who is judy
            | can i talk to judy
            | can i talk to solar man
            | who is monty
            | can i talk to monty;

<terphrase_verbphrase> =
                    <nonhuman_termph_planet> <transvb_by_termph> |
       <nonhuman_termph_moon> <animate_transvb> by <human_termph>;

<transvb_by_termph> = <animate_transvb> by <human_termph> |
            <inanimate_transvb> by <nonhuman_termph_moon> ;

<sent> = <human_termph> <animate_verbph> |
         <nonhuman_termph_moon> <inanimate_verbph_active> |
         <nonhuman_termph_planet> <inanimate_verbph_passive> ;

<nouncla_verbph> = <human_nouncla> <animate_verbph> |
                 <nonhuman_nouncla_moon> <animate_verbph_passive>|
               <nonhuman_nouncla_moon> <inanimate_verbph_active>|
             <nonhuman_nouncla_planet> <inanimate_verbph_passive> ;

<inanimate_verbph> = <inanimate_verbph_active> |
                     <inanimate_verbph_passive> ;

<human_stermph> = <human_pnoun> | <human_detph> ;

<nonhuman_stermph_planet> = <nonhuman_pnoun_planet> |
                            <nonhuman_detph_planet> ;

<nonhuman_stermph_moon> = <nonhuman_pnoun_moon> |
                          <nonhuman_detph_moon> ;

<human_termph> = <human_stermph> |
           <human_stermph> ( and | or ) <human_stermph> ;

<nonhuman_termph_planet> = <nonhuman_stermph_planet> |
 <nonhuman_stermph_planet> ( and | or ) <nonhuman_stermph_planet> ;
```

```
<nonhuman_termph_moon> = <nonhuman_stermph_moon> |
     <nonhuman_stermph_moon> ( and | or ) <nonhuman_stermph_moon> ;

<animate_verbph> = <animate_transvbph> ;

<animate_verbph_passive> = <linkingvb> <animate_transvb>
                                        by<human_termph>;

<inanimate_verbph_active> = <inanimate_transvbph_active> |
                                              <intransvb>;

<inanimate_verbph_passive> = <inanimate_transvbph_passive> |
                      <intransvb> | <inanimate_transvb> sun ;

<animate_transvbph> = <animate_transvb> ( <nonhuman_termph_planet>|
                                       <nonhuman_termph_moon> );

<inanimate_transvbph_active> = <inanimate_transvb>
                                <nonhuman_termph_planet>;

<inanimate_transvbph_passive> = <linkingvb> <inanimate_transvb>
                                     by <nonhuman_termph_moon> ;

<human_detph> = <det> <human_nouncla> ;

<nonhuman_detph_planet> = <det> <nonhuman_nouncla_planet> ;

<nonhuman_detph_moon> = <det> <nonhuman_nouncla_moon> ;

<human_nouncla> = <adj> <human_cnoun> | <human_cnoun> ;

<nonhuman_nouncla_planet> = <adj> <nonhuman_cnoun_planet> |
                                  <nonhuman_cnoun_planet> ;

<nonhuman_nouncla_moon> = <adj> <nonhuman_cnoun_moon> |
                                <nonhuman_cnoun_moon> ;
```

```
<nouncla> = <human_nouncla> | <nonhuman_nouncla_planet> |
                                <nonhuman_nouncla_moon> ;

<human_cnoun> = man | men | person ;
<nonhuman_cnoun_planet> = planet | planets ;
<nonhuman_cnoun_moon> = moon | moons ;

<adj> = red ;
<intransvb> = spin | spins | orbit | orbits | orbited | exist ;

<animate_transvb> = discover | discovers | discovered ;
<inanimate_transvb> = orbit | orbits | orbited ;

<linkingvb> = is | was | are | were ;

<quest1> = did | do | does ;

<det> = a | an | every | one | two | three | four | five ;

<nonhuman_pnoun_planet> = earth | Jupiter | mars | mercury |
                          neptune | pluto | saturn | uranus | venus ;

<nonhuman_pnoun_moon> = almathea | ariel | callisto | charon |
                    deimos | dione | enceladus | europa | ganymede |
                hyperion | iapetus | io |janus | jupitereighth |
              jupitereleventh | jupiterfourteenth | jupiterninth |
               jupiterseventh | jupitersixth | jupitertenth |
            jupiterthirteenth | jupitertwelfth |luna |mimas |
              miranda | nereid| oberon | phobos | phoebe | rhea |
         saturnfirst | tethys | titan | titania | triton | umbriel;

<human_pnoun> = bernard | bond | cassini | dollfus | fountain |
                galileo | hall | herschel | huygens | kowal |
                 kuiper | larsen | lassell | melotte |
                nicholson | perrine | pickering ;

<pnoun> = <nonhuman_pnoun_planet> | <nonhoman_pnoun_moon> |
                                                <human_pnoun> ;
```

```
public <bye> = good bye and go to sleep {bye};
```

9.3 Details of the semantic constraints

From the original grammar to the new grammar, we have added the following semantic constraints (Left column are the non-terminals before semantic constraint, and right column are the non-terminals semantic-constrained).

Non-terminal in original grammar	Non-terminal in the revised grammar
<stermph>	<human_stermph> <nonhuman_stermph_planet> <nonhuman_stermph_moon>
<termph>	<human_termph> <nonhuman_termph_planet> <nonhuman_termph_moon>
<verbph>	<animate_verbph> <animate_verbph_passive> <inanimate_verbph_active> <inanimate_verbph_passive>
<transvbph>	<animate_transvbph> <inanimate_transvbph_active> <inanimate_transvbph_passive>
<detph>	<human_detph> <nonhuman_detph_planet> <nonhuman_detph_moon>
<nouncla>	<human_nouncla>

	<nonhuman_nouncla_planet> <nonhuman_nouncla_moon>
<cnoun> = man | men | person | planet | planets | moon | moons ;	<human_cnoun> = man | men | person ; <nonhuman_cnoun_planet> = planet | planets ; <nonhuman_cnoun_moon> = moon | moons ;
<transvb> = discover | discovers | discovered | orbit | orbits | orbited ;	<animate_transvb> = discover | discovers | discovered ; <inanimate_transvb> = orbit | orbits | orbited ;
<pnoun>	<human_pnoun> <nonhuman_pnoun_planet> <nonhoman_pnoun_moon>

9.4 The list of queries used

The following queries were used in our tests:

1. Which moons orbit Charon or Mars? *
2. Who discovered a moon or a planet? *
3. Which moons orbit a planet or a moon? *
4. Which moon was discovered by Huygens?
5. Is Mars a moon or a planet?
6. Was every planet discovered by a person?
7. Which planet is orbited by Hyperion?
8. Which planet is orbited by Titan?
9. Which planet is orbited by Dione?

10. Which planet is orbited by Tethys?
11. Which planet is orbited by Mimas?
12. Which planet is orbited by Satumfirst?
13. Which planet is orbited by Jupiterthirteenth?
14. Which moons were discovered by Dollfus?
15. Which moons were discovered by Nicholson?
16. Who discovered Jupiter?
17. How many moons orbit Venus?
18. How many moons orbit Jupiter?
19. How many moons orbit Pluto?
20. How many moons orbit Mercury?
21. How many moons orbit Satum?
22. How many moons orbit Mars?
23. How many moons orbit Earth?
24. Is Phobos a planet?
25. What moon exists? *

* Note: The "star" set of questions should not be recognized, as it is grammatically incorrect.

9.5 The data-sheet for collection from one user experiment

See APPENDIX III, IV, and V for the data-sheets of User 1, User 2, and User 3. The symbols Q1 ... Q25 denote the questions from 1 to 25 (see section 9.4) in the query list used for this test. In the second column of the data-sheet, the numbers from 1 to 6 denote the attempts for each question.

9.6 Data Collected from Different Users

		USER 1		USER 2		USER 3	
		Correct	Wrong	Correct	Wrong	Correct	Wrong
Original grammar	Total	115	11	90	36	110	16
	(Total/126)%	91.27%	8.73%	71.43%	28.57%	87.30%	12.70%
Revised grammar	Total	126	0	104	22	120	6
	(Total/126)%	100%	0%	82.54%	17.46%	95.24%	4.76%

Data Collected from Different Users

10. ANALYSIS OF THE EXPERIMENTAL RESULTS AND THE APPROACH

10.1 The improvement of speech-recognition accuracy

The data collected from different users (in section 9.6 on pp 46) has shown the result that we had hoped for. Speech-recognition accuracy has been improved by coding semantic constraints as syntax in the recognition grammar.

For user1, a total of 126 answers of 126 attempts are correct for the revised grammar, which semantic-constrained. The accuracy is 100%. It is 8.73% higher than the accuracy of the original grammar, which is the grammar before the semantic constraint and has 91.27% recognition accuracy. Another way to look at the data is that the error rate decreased from 8.73% to 0. This is a 100% decrease in errors.

For user2, a total of 104 answers of 126 attempts are correct for the revised grammar. Although the accuracy is 82.54%, it is 11% higher than the accuracy of the original grammar, which has 71.43% recognition accuracy and a total of 90 answers of 126 attempts are correct. The error rate decreased from 28.57% to 17.46%. This is a 39% decrease in errors.

For user3, a total of 120 answers of 126 attempts are correct for the revised grammar. The accuracy is 95.24%, it is 7.94% higher than the accuracy of the original grammar, which has 87.30% recognition accuracy and a total of 110

answers of 126 attempts are correct. The error rate decreased from 12.70% to 4.76%. This is a 62.5% decrease in errors.

10.2 Main errors in the data sheets of the revised grammar

(1) The users' data-sheets show that both grammars got the wrong results on the 25th question. The 25th query was "What moon exists?" Obviously, this should not be recognized. This is not a proper question. Usually it should be "Which moons exist?" The original grammar has totally wrong answers. But for the revised grammar, 3 answers (user1), and 4 answers (user2) of 6 gave attempts, as "Please repeat, I do not understand", which was correct. Anyway we finally excluded this query in our count because of this is not system problem.

(2) About the first 3 queries, "Which moons orbit Charon or Mars?", "Who discovered a moon or a planet?", and "Which moons orbit a planet or a moon?", we see that the objectives of these sentence ("Charon or Mars" and "a moon or a planet") consist of "planet" and "moon". The revised grammar has separated the "behavior" of "moon" and "planet" by the semantic constraints. The revised grammar can only handle one case ("moon" or "planet"), not both of them, once in one sentence. These first 3 queries have involved both "moon" and "planet" in one sentence. This is really "out of grammar". It returned the totally wrong results (user1, user2, and user3). We also excluded these queries in our count because of this is inappropriate for these experiments.

10.3 Analysis of remaining errors

These remaining errors come from the user2 mostly and user3. After investigating the details of user2's data sheets from hand draft page 51 to 100 in the APPENDIX VI, and user3's data sheets from hand draft page 101 to 150 in the APPENDIX we discovered why so many errors made.

(1) Pronunciation problems

Look at the hand draft page 80 in APPENDIX VI, for the question 5, "Is Mars a moon or a planet?", user2 can't say "M" correctly. He uses the "L" instead of the "M" pronunciation. "Mars" was changed to "Lassell". The similar case also happened at the question 15, "Which moons were discovered by Nicholson?" in hand draft page 90 (APPENDIX VI). "Nicholson" was changed to "a person". And also in question 18 (page 93), 19 (page 94), and 20 (page 95) in APPENDIX VI, "or" in "orbit" was changed to "ar" in "are discovered". Also look at the hand draft page 130 in APPENDIX VI, for the question 5. "Mars" was changed to "Mimas" in three answers of six attempts. This is a pronunciation problem too.

(2) The background noise

The background noise is another mistake reason. These cases happened at Q4 (page 79), Q9 (Page 84), Q14 (page 89), Q15 (page 90), Q16 (page 91), Q20 (Page 95), Q22 (page 97), Q24 (page 99) in APPENDIX VI.

(3) Some errors are fixable problems

Look at the hand draft pages 131 and 149. The "a person" was changed to "two person", and "is phobos a planet" was changed to "is phobos jupiterninth". This kind of problems although were made by pronunciation, it maybe fixable problems by grammars.

(4) A table for each mistake

The "page x" in the right most of column comes from the hand draft pages of APPENDIX VI.

Actual query	Q4. Which moon was discovered by Huygens	Page79
Mistaken as	Was jaius discover by huygens and huygens	Attempt 1
Possible reason	The background noise.	
Actual query	Q5. Is Mars a moon or a planet	Page 80
Mistaken as	Is io lassell a moon or a planet	Attempt 1
Possible reason	User's pronunciation. The background noise.	
Actual query	Q5. Is Mars a moon or a planet	Page 80
Mistaken as	Is hall lassell a moon or a planet	Attempt 2
Possible reason	User's pronunciation. The background noise.	
Actual query	Q5. Is Mars a moon or a planet	Page 80
Mistaken as	Is lassell deimos or a planet	Attempt 3
Possible reason	User's pronunciation. "M" changed to "L"	
Actual query	Q5. Is Mars a moon or a planet	Page 80
Mistaken as	Is lassell a moon or a planet	Attempt 4
Possible reason	User's pronunciation. "M" changed to "L"	
Actual query	Q5. Is Mars a moon or a planet	Page 80
Mistaken as	Is lassell umbriel or a planet	Attempt 5, 6
Possible reason	User's pronunciation. "M" changed to "L"	
Actual query	Q9. Which planet is orbited by dione	Page 84
Mistaken as	Which planet exist	Attempt 2
Possible reason	User's pronunciation. "is" changed to "ex"	
Actual query	Q9. Which planet is orbited by dione	Page 84
Mistaken as	Which red planet is orbited by dione	Attempt 4
Possible reason	Speech was too slow	
Actual query	Q14. Which moons were discovered by dollfus	Page 89
Mistaken as	Which red moons were discovered by dollfus	Attempt 5

Possible reason	Speech was too slow	
Actual query	Q15. Which moons were discovered by nicholson	Page 90
Mistaken as	Which moons were discovered by a person	Attempt 1
Possible reason	User's pronunciation	
Actual query	Q15. Which moons were discovered by nicholson	Page 90
Mistaken as	Which person discovered urarus	Attempt 2
Possible reason	The background noise, user's pronunciation	
Actual query	Q16. Who discovered jupiter	Page 91
Mistaken as	Which men discovered jupiter	Attempt 1
Possible reason	Speech was too slow	
Actual query	Q17. How many moons orbit venus	Page 92
Mistaken as	How many planet are orbits by iapetus	Attempt 1
Possible reason	The background noise, user's pronunciation	
Actual query	Q18. How many moons orbit jupiter	Page 93
Mistaken as	Does io orbits jupiter	Attempt 1
Possible reason	The background noise	
Actual query	Q18. How many moons orbit jupiter	Page 93
Mistaken as	How many planet are orbit jupitereight	Attempt 5
Possible reason	"or" pronunciation be changed to "at"	
Actual query	Q19. How many moons orbit pluto	Page 94
Mistaken as	How many moons are discover by galileo	Attempt 2
Possible reason	"or" pronunciation be changed to "at"	
Actual query	Q20. How many moons orbit mercury	Page 95
Mistaken as	How many moons are discover by perrine	Attempt 2
Possible reason	The background noise	
Actual query	Q20. How many moons orbit mercury	Page 95
Mistaken as	How many moons exist	Attempt 3
Possible reason	The background noise	

Actual query	Q22. How many moons orbit Mars	Page 97
Mistaken as	How many moons orbited earth and Mars	Attempt 2
Possible reason	The background noise	
Actual query	Q24. is phobos a planet	Page 99
Mistaken as	is phobos jupiterninth	Attempt 1
Possible reason	The background noise	
Actual query	Q24. is phobos a planet	Page 99
Mistaken as	is phobos hyperion	Attempt 2
Possible reason	The background noise, user's pronunciation	
Actual query	Q5. Is Mars a moon or a planet	Page130
Mistaken as	Is Mimas a moon or a planet	Attempt1,2,6
Possible reason	Maybe user pronunciation	
Actual query	Q6. Was every planet discovered by a person	Page131
Mistaken as	Was every planet discovered by two person	Attempt 4
Possible reason	(Maybe fixable with agreement rules)	
Actual query	Q19. How many moons orbit pluto	Page144
Mistaken as	What orbit pluto	Attempt 6
Possible reason	Maybe user "stutter".	
Actual query	Q24. Is Phobos a planet?	Page149
Mistaken as	Is Phobos jupiterninth.	Attempt 6
Possible reason	(Fixable by grammar)	

10.4 How does the approach compare with other approaches?

1) In this example application there is little scope to reduce the vocabulary so we cannot compare with that strategy.

2) The following example compares the approach with the use of "categories" in the query. For example, consider the query: "Does Phobos orbit Mars?" we may denote it like:

Does <noun> orbit <noun> (1)

The <noun> includes all nouns in the system. The perplexity of <noun> is very high (perplexity = number of nouns). If n denotes the number of noun, we have n + n = 2n perplexity for (1) under the un-constraint (original) grammar in the worst case.

We also consider another query: "Does the moon called Phobos orbit the planet called Mars?" This is an exactly same question with (1). It is using of "categories" of restricting vocabulary domain and restricting on the sentence structure:

Does <category> called <moon> orbit <category> called <planet> (2)

In which <category> includes planets, moons, and sun. The perplexity is three. The <moon> includes all moons in the solar system. The perplexity of <moon> equals to the number of moons. The <planet> includes all planets in the solar system. The perplexity of <planet> equals to the number of planets that is nine. Assume c denotes the number of category, m denotes the number of moons, p denotes the number of planets. So the perplexity of (2) is: c + m + c + p = 3 + m + 3 + 9 = 15 + m. Since n = c + m + p + other names, "15 + m" (the perplexity of (2)) is less than 2n (the perplexity of (1)).

Now, in our experiment, by the revised grammar, we don't need query like (2), which is unnaturally even if it has low perplexity that means high accuracy. We query as simple as (1) and we could obtain as low perplexity as (2). Because of the constraints, the perplexity of the first <noun> in (1) is m, and the second is p + 1 (the number of planets and the sun). This means that the perplexity of (1) equals to only m + p +1 = 10 + m in worst case if using the revised grammar.

11. EXPERIMENT II

- An experienced user (Dr. Frost) with 11 utterances that are known to be error-prone.
- Error rate:

Original grammar –16 errors out of 66 attempts (error rate = 24%)

Revised grammar – 6 errors out of 66 attempts (error rate = 9%)

The actual data results of experiment II can be found from hand draft page 151 to 172 in APPENDIX VI. The following lists the eleven queries of experiment II.

1) Was Callisto discovered by Kowal or Kuiper?
2) Was Callisto discovered by Huygens or Kowal?
3) Is Saturn orbited by two moons?
4) What orbits Earth or Jupiter?
5) What orbits Earth and Jupiter?
6) What orbits Earth and a planet?
7) What orbits Saturn and a planet?
8) What orbits Earth or a planet?
9) What orbits Mercury or a planet?
10) What orbits Uranus or a planet?
11) Which moons orbit Saturn or Mars?

12. RELATED WORK

Frost's research on SpeechWeb is the directly related to this study. Since the 1990's Frost has done many tasks to develop SpeechWeb applications. Especially in year 2002, the paper [Frost 2002b] investigated the situation of speech-recognition accuracy improvement, and then developed this new approach, coding semantic constraints as the syntax in the recognition grammar, for improving speech-recognition accuracy.

Moore also discussed the speech-recognition accuracy improvement in his paper [Moore 1999]. Moore's investigation of the methods of speech-recognition accuracy improvement helps us understand various models and approaches. Moore makes two important observations: (i) adding semantic constraints to CFGs increases their size considerably and complicates construction and analysis of the CFGs. (ii) qualitative language models cannot recognize out-of-grammar utterances. Moore suggests various methods to overcome these problems. Moore's observations and proposed solutions are appropriate for "large-language" interfaces.

There appears to be little other work that investigations the coding of high-level semantics in syntax to improve speech-recognition accuracy. The following are some examples of such work.

Kaiser, Johnston, and Heeman, in 1999, presented a method for constructing a predictive, robust finite-state parser for speech-recognition's semantic parser, and contrasted it with both FSAs and the PHOENIX system, which is a robust chart-

based semantic parser. They also demonstrated the use of this parser within a small prototype system, built within the CSLU toolkit's RAD environment. Their work is still in the early stages. Anyway this is the latest work related this area except Frost [Kaiser, Johnston, and Heeman 1999].

Atwell and Demetriou, in 1994, surveyed the art of computational semantic methods in speech-recognition research. A taxonomy classifying the approaches adopted in the literature is divided into six main categories: semantic networks, semantic grammars, caseframes, statistical, unification-based and neural networks. After the overview and discussion they concluded that there is no adequate theory (in 1994) for semantic constraints for speech-recognition [Demetriou and Atwell 1994b]. Then they focused on the large vocabulary semantic network for computerized speech-recognition [Demetriou and Atwell 1994a] and [Atwell and Kevitt 1994c].

13. CONCLUSIONS AND FUTURE WORK

The new approach has really shown its power to improve speech-recognition accuracy in "small-language" applications. The experiments clearly present the effectiveness of the new approach.

Future work:

(1) Investigate the robustness of the resulting SpeechWeb. We need to build lots of related sihlos.

(2) Test system with many more utterances. And new and more detailed work on semantic constraints should be done. For example, we may consider the "single" and "plural" constraints as:

```
<det_single> = a | an | every | one ;
<det_plural> = two | three | four | five ;
<linkingvb_single> = is | was ;
<linkingvb_plural> = are | were ;
<quest1_single> = does ;
<quest1_plural> = did | do ;
<animate_transvb_single> = discovers | discovered ;
<animate_transvb_plural> = discover | discovered ;
<inanimate_transvb_single> = orbits | orbited ;
<inanimate_transvb_plural> = orbit | orbited ;
```

and also gender constraints, and so on. This will make the SpeechWeb more accurate.

(3) Investigate providing users with knowledge of constraints. We need to make the SpeechWeb application more friendly. We could provide users with knowledge of constraints.

(4) Create "Secretary" sihlo and "Searcher" sihlo to assist user navigation. (Case A) a user is in the small scope of SpeechWeb, for example, a company building. We could design a "Secretary" sihlo who knows about the basic features or functions of all of sihlos in this building. When the user doesn't know who can answer his questions, he only needs saying "Who may help me?" or "help me, please", then the Secretary sihlo will talk with him and indicate who may help him. (Case B) a user is in the extended (or unlimited) scope of SpeechWeb as convenance web. We could design a "Searcher" sihlo who knows many links and list of sihlos. The user may talk with Searcher sihlo first to find someone (sihlo) who then talks with goal sihlo.

(5) Investigate more ways to constrain the recognition grammars. For example: "which moon orbits the planet called Mars" (which moon orbits the A called B). The "A" and "B" both could be a moon, a planet, or the Sun, even the nine planets of Solar system. So, the combination of A and B may be over 20 branches! How can we constrain this case?

REFERENCES

Alexin, Z., Gylmothy, T., Horvath, T. and Fabricz, K.1990. Attribute grammar specification for a natural language understanding interface. In *Proceedings of the International Conference on Attribute Grammars and their Applications*. Lecture Notes in Computer Science, vol. 461. Springer-Verlag, New York, 313-326.

Alblas, H. 1991a. Introduction to attribute grammars. In *Attribute Grammars, Applications and Systems*. Lecture Notes in Computer Science, vol. 545. Springer-Verlag, New York, 1-15.

Chapman, N. P. 1990. Defining, analysing and implementing communication protocols using attribute grammars. *Formal Aspects Comput.* 2, 359-392.

Crimi, C., Guercio, A., pacini, G., Tortora, G. and Tucci, M. 1990. Automating visual language generation. *IEEE Trans. Softw. Eng.* 16, 10, 1122-1135.

Delest, M. P. and Fedou, J. M. 1992. Attribute grammars are useful for combinatorics. *Theor. Comput. Sci.* 98, 1, 65-76.

Demetriou, G. and Atwell, E S. 1994a. A semantic network for large vocabulary speech recognition in: Evett, L & Rose,T (editors) *Computational Linguistics for Speech and Handwriting Recognition AISB'94 Workshop*, pp. 21-28 University of Leeds/AISB.

Demetriou, G. and Atwell, E S. 1994b. Semantics in speech recognition and understanding : a survey in: Evett, L & Rose,T (editors) *Computational Linguistics for Speech and Handwriting Recognition AISB'94 Workshop University of Leeds/AISB.*

Atwell, E S; McKevitt, P. 1994c. Pragmatic linguistic constraint models for large-vocabulary speech processing in: McKevitt, P (editors) *Integrating Speech and Natural Language Processing : AAAI94 Workshop Proceedings*, pp. 58-64 AAAI Press.

Deransart, P., Jourdan, M., and Lorho, B. 1988. Attribute Grammars: Definitions, Systems and Bibliography. *Lecture Notes in Computer Science*, vol. 323, Springer-Verlag, New York.

Ding, S. and Katayama, T. 1993. Attribute state machines for behavior specification of reactive systems, In *Proceedings of the 5th International Conference on Software Engineering and Knowledge Engineering* (SEKE'93) (San Francisco). Knowledge Systems Institute, 695-702.

Deransart, P. and Maluszynski, J. 1993. A Grammatical View of Logic Programming. *The MIT Press*, Cambridge, Mass.

Deransart, P. and Maluszynski, J. 1985. Relating logic programs and attribute grammars. *J. Logic Program.* 2, 2, 119-155.

Frost, R. A. 2002a. SpeechWeb: A Web of Natural-Language Speech Applications. *Proceedings of the AAAI'02 Intelligent Systems Demonstrations (ISD)*, U. of Alberta, Edmonton.

Frost, R. A. 2002b. Improving Speech-Recognition Accuracy by Coding Semantic Constraints in the Syntax of the Recognition Grammar. *Unpublished paper.*

Frost, R. A. 2002c. W/AGE: The Windsor Attribute Grammar Programming Environment. *Proceedings of the 2002 IEEE Symposia on Human Centric Computing Languages and Environments --HCC' 02.* Arlington, Virginia, 2002.

Frost, R. A. 2001. Constructing Language-processors as Executable Grammar Objects. (Submitted to AI2002 on 18 November 2001).

Frost, R. A. 1999a. Improving the efficiency of executable grammar objects. *University of Windsor, Department of Computer Science Technical Report CS* 1999-01.

Frost, R. A. 1999b. SpeechNet: A network of hyper linked speech-accessible objects. *IEEE Proceedings of WECWIS '99: International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems.* Santa Clara, CA. Apr. 1999; 116-121.

Frost, R. A. 1999c. A Natural-Language Speech Interface Constructed Entirely as a Set of Executable Specifications. *Proceedings of the AAAI'99 Intelligent Systems Demonstrations,* Orlando, Fla. July 1999.

Frost, R. A. 1994. W/AGE: The Windsor Attribute Grammar Programming Environment. *Schloss Dagstuhl International Workshop on Functional Programming in the Real World. Seminar Report 89; 16.05-20.05.94 (9420). Editors Robert Giergerich and John Hughes.*

Frost, R. A. 1992. Constructing programs as executable attribute grammars. *Computer Journal* 35, 4, 376-387

Frost, R. A. 1990. Constructing Programs in a Calculus of Lazy Interpreters. ACM/ *SIGSOFT International Workshop on Formal Method in Software Development.*

Napa, California, May 1990. ACM/SIGSOFT Software Engineering Notes 15 (4) pp.30-41.

Frost, R. A. and Chitte, S. 1999. A New Approach For Providing Natural-Language Speech Access to Large Knowledge Bases. *Proceedings of the Pacific Association of Computational Linguistics Conference PACLING'99.* University of Waterloo, August, 1999, 82-89.

Frost, R. A. and Karamatos, S. 1989. Use of the W/AGE CASE Tool in Artificial Intelligence. *IEEE Computer Society International Workshop on Tools for Artificial Intelligence* at Herndon, Virginia.

Frost, R. A. and Launchbury, J. 1989. Constructing natural language interpreters in a lazy functional language. *The Computer Journal* 32 (2) pp.108-121.

Frost, R. A. and Saba, W. 1990. A Database Interface Based on Montague's Approach to the Interpretation of Natural Language. *Intl. J. Man-Machine Studies* 33 pp.149-176.

Frost, R. A. and Szydlowski, B. 1996. Memorizing Purely Functional Top-down Backtracking Language Processors. *Science of Computer Programming* (27), pp.263-288.

Grosch, Josef 1999. Are Attribute Grammars Used in Industry? *Second Workshop on Attribute Grammars and Their Applications*, WAGA99.

Horwitz, S. and Reps, T. 1992. The use of program dependence graphs in software engineering. In *Proceedings of the 14th International Conference on Software Engineering*. IEEE Computer Society press, Los Alamitos, Calif., 392-411.

Hudson, S. E. and King, R.1987. Implementing a user interface as a system of attributes. In *Proceedings of the ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments.* ACM SIGPLAN Not. 22, 1, 143-149.

Irons, E. T. 1961. A syntax directed compiler for Algol 60. *Commun. ACM* 4, 1, 51-55.

Jones, L. G. and Simon, J. 1986. Hierarchical VLSI design systems based on attribute grammars. In *Conference Record of the 13th ACM Symposium on Principles of Programming languages* ACM, New York, 58-69.

Kaiser, E. C., Johnston, M. and Heeman, P.A. 1999. Profer: Predictive, Robust Finite-State Parsing For Spoken Language. *ICASSP-99*, vol. 2, pp.629-632, Phoenix, AZ.

Knuth, D. E. 1990. The genesis of attribute grammars. In *Proceedings of the International Conference on Attribute Grammars and their Applications.* Lecture Notes in Computer Science, vol. 461. Springer-Verlag, New York, 1-12.

Knuth, D. E. 1971. Semantics of context-free languages: correction.. *Math. Syst. Theory* 5, 1, 95-96.

Knuth, D. E. 1968. Semantics of context-free languages. *Math. Syst. Theory* 2, 2, 127-145.

Kaiser, G. and Kaplan, S. M. 1993. Parallel and distributed incremental attribute evaluation algorithms for multiuser software development environments. *ACM Trans. Softw. Eng. Method.* 2, 1, 47-92.

Lewi, J., De Vlaminck, K., Steegmans, E. and Van Horebeek, J. 1992. *Software Development by LL(1) Syntax Description.* John Wiley & Sons, New York.

Moore, R. C. 1999. Using Natural-Language Knowledge Sources in Speech Recognition. *In Keith Ponting, editor, Speech Pattern Processing. Springer-Verlag.*

Moore, R., Appelt, D., Dowding, J., Gawron, J.M. and Moran, D. 1995. Combining Linquistic and Statistical Knowledge Sources in Natural-Language Processing for ATIS in *Proceedings of the Spoken Language Systems Technology Workshop, Austin, Texas, pp.* 261-264.

Ogden, W. C. and Bernick, P. 1996. Using Natural Language Interfaces. *Handbook of Human-Computer Interaction.* M. Helander Ed. Elsevier Science Publishers B.V. (North-Holland).

Paakki, J. 1995. Attribute grammar paradigms -- a high-level methodology in language implementation. *ACM Computing Surveys,* vol. 27, No. 2, 196-255.

Ridjanovic, D. and Brodie, M. L. 1982. Defining database dynamics with attribute grammars. *Inf. Process. Lett.* 14, 3, 132-138.

Shinoda, Y. and Katayama, T. 1988. Attribute grammar based programming and its environment. In *Proceedings of the 21st Hawan International Conference on System Sciences.* IEEE Computer Society Press, 612-620.

Steel, T. B., Jr. (Ed) 1966. Formal language description languages for computer programming. In *Proceedings of the IFIP Working Conference on Formal Language Description Languages.* North-Holland, Amsterdam.

Turner, D. A. 1985. A Lazy Functional Programming Language With Polymorphic Types. *Proc. IFIP Int. Conf. On Functional Programming Language and Computer Architecture*. Nancy, France. Springer Lecture Notes 201.

Tsai, W. and Fu, K. S. 1980. Attributed grammar--A tool for combining syntactic and statical approaches to pattern recognition. *IEEE Trans. Syst. Man Cybernet.* 10, 873-885.

Trahanias, P. and Skordalakis, E. 1990. Syntactic pattern recognition of the ECG. *IEEE Trans. Patt. Anal. Machine Intell.* 12, 7, 648-657.

van de Burgt and Tilanus 1989. Attributed ASN.1. In *Proceedings of the 2nd International Conference on Formal Description Techniques for Distributed Systems and Communication Protocols*. North-Holland, Amsterdam, 298-310.

Woods, W. A. 1970. Transition Network Grammars for Natural Language Analysis. *Communications of the ACM*, Vol. 13, No.10, pp.591-606.

APPENDIX I

Solar man grammar (original grammar)

```
grammar solar;

public <s> = <linkingvb> <termph> [ <transvb>  by ] <termph> (sentence)|
                <questl> <sent> (sentence)|
             (who | what )<verbph>(sentence)|
                (which | how many ) <nouncla> <verbph> (sentence)|
                <simple> (sentence);
<simple> =  | ask them to be quiet
            | please introduce yourself
            | hello there
            | hello solar man
            | goodbye
            | goodbye solar man
            | fine thanks
            | thanks
            | thanks solar man
            | yes please
            | what is your name
            | who are you
            | where do you live
            | what do you know
            | how old are you
            | who made you
            | what is your favorite band
            | who is the vice president at the university of windsor
            | who is the president at the university of windsor
            | who is the executive dean of science at the university of windsor
            | who is the dean of science at the university of windsor
            | tell me a poem
            | know any poems
            | tell me a joke
            | know any jokes
            | who is judy
            | can i talk to judy
            | can i talk to poems
            | who is monty
            | can i talk to monty;

<sent> = <termph> <verbph>;

<stermph> = <pnoun> | <detph>;

<termph> = <stermph> | <stermph> ( and | or ) <stermph> ;

<verbph> = <transvbph> | <intransvb> ;

<transvbph> = ( <transvb> | <linkingvb> <transvb> by ) <termph>;

<detph> = <det> <nouncla> ;

<nouncla> = <adj> <cnoun> | <cnoun> ;

<cnoun> = man | men | person | planet | planets | moon | moons;

<adj> =   red ;

<intransvb> = spin | spins | orbit | orbits | orbited | exist ;

<det> = a  | an | every | one | two | three ;

<pnoun> = earth | jupiter | mars | mercury | neptune | pluto | saturn | uranus | ve
nus |
          almathea | ariel | callisto | charon | deimos | dione | enceladus | europ
a |
          ganymede | hyperion | iapetus | io |janus | jupitereighth | jupitereleven
th |
```



```
          jupiterfourteenth | jupiterninth | jupiterseventh | jupitersixth |jupiter
tenth |
          jupiterthirteenth | jupitertwelfth |luna |mimas | miranda | nereid| obero
n | phobos |
          phoebe | rhea | saturnfirst | tethys | titan | titania |triton | umbriel
|
          bernard | bond | cassini | dollfus | fountain | galileo | hall | herschel
 |
            huygens | kowal | kuiper | larsen | lassell | melotte | nicholson | per
rine | pickering ;

 <transvb> = orbit | orbits | discover | discovered ;

 <linkingvb> = is | was | are | were ;

 <questl> = did | do | does ;



public <bye> = good bye and go to sleep (bye);
```

APPENDIX II

Solar man grammar 2 (revised grammar)

```
 grammar solar02;
public <s> = <linkingvb> <termphrase_verbphrase> (sentence)|
             is <pnoun> <pnoun> (sentence) |
             is <pnoun> a <nouncla> (sentence) |
             is <pnoun> a <nouncla> or a <nouncla> (sentence) |

                  <questl> <sent> (sentence)|
          ( who )<animate_verbph> (sentence)|
        ( what )<inanimate_verbph>(sentence)|
( which | how many )<nouncla_verbph>(sentence)|

                            <simple> (sentence);



<simple> =  | ask them to be quiet
            | please introduce yourself
            | hello there
            | hello solar man
            | goodbye
            | goodbye solar man
            | fine thanks
            | thanks
            | thanks solar man
            | yes please
            | what is your name
            | who are you
            | where do you live
            | what do you know
            | how old are you
            | who made you
            | what is your favorite band
            | who is the vice president at the university of windsor
            | who is the president at the university of windsor
            | who is the executive dean of science at the university of windsor
            | who is the dean of science at the university of windsor
            | tell me a poem
            | know any poems
            | tell me a joke
            | know any jokes
            | who is judy
            | can i talk to judy
            | can i talk to solar man
            | who is monty
            | can i talk to monty;




<termphrase_verbphrase> = <nonhuman_termph_planet> <transvb_by_termph> |
       <nonhuman_termph_moon> <animate_transvb> by <human_termph>;



<transvb_by_termph> = <animate_transvb> by <human_termph> |
                 <inanimate_transvb> by <nonhuman_termph_moon> ;



<sent> = <human_termph> <animate_verbph> |
         <nonhuman_termph_moon> <inanimate_verbph_active> |
         <nonhuman_termph_planet> <inanimate_verbph_passive> ;
<nouncla_verbph> = <human_nouncla> <animate_verbph> |
```



```
                  <nonhuman_nouncla_moon> <animate_verbph_passive>|
               <nonhuman_nouncla_moon> <inanimate_verbph_active>|
            <nonhuman_nouncla_planet> <inanimate_verbph_passive> ;
<inanimate_verbph> = <inanimate_verbph_active> |
                     <inanimate_verbph_passive> ;
<human_termph> = <human_pnoun> | <human_detph> ;
<nonhuman_termph_planet> = <nonhuman_pnoun_planet> |
                           <nonhuman_detph_planet> ;
<nonhuman_termph_moon> = <nonhuman_pnoun_moon> |
                         <nonhuman_detph_moon> ;
<human_termph> = <human_termph> |
          <human_termph> ( and | or ) <human_termph> ;
<nonhuman_termph_planet> = <nonhuman_termph_planet> |
   <nonhuman_termph_planet> ( and | or ) <nonhuman_termph_planet> ;
<nonhuman_termph_moon> = <nonhuman_termph_moon> |
       <nonhuman_termph_moon> ( and | or ) <nonhuman_termph_moon> ;


<animate_verbph> = <animate_transvbph> ;
<inanimate_verbph_active> = <inanimate_transvbph_active> | <intransvb>;
<inanimate_verbph_passive> = <inanimate_transvbph_passive> |
                     <intransvb> | <inanimate_transvb> sun ;


<animate_verbph_passive> = <linkingvb> <animate_transvb> by <human_termph>;


<animate_transvbph> = <animate_transvb> ( <nonhuman_termph_planet>|
                                          <nonhuman_termph_moon> );

<inanimate_transvbph_active>=<inanimate_transvb><nonhuman_termph_planet>;
<inanimate_transvbph_passive> = <linkingvb> <inanimate_transvb>
                                by <nonhuman_termph_moon> ;
<human_detph> = <det> <human_nouncla> ;
<nonhuman_detph_planet> = <det> <nonhuman_nouncla_planet> ;
<nonhuman_detph_moon> = <det> <nonhuman_nouncla_moon> ;
<human_nouncla> = <adj> <human_cnoun> | <human_cnoun> ;
<nonhuman_nouncla_planet> = <adj> <nonhuman_cnoun_planet> |
                                  <nonhuman_cnoun_planet> ;
<nonhuman_nouncla_moon> = <adj> <nonhuman_cnoun_moon> |
                                <nonhuman_cnoun_moon> ;

<nouncla> = <human_nouncla> | <nonhuman_nouncla_planet> |
            <nonhuman_nouncla_moon>;
<human_cnoun> = man | men | person ;
<nonhuman_cnoun_planet> = planet | planets ;
<nonhuman_cnoun_moon> = moon | moons ;
<adj> = red ;
<intransvb> = spin | spins | orbit | orbits | orbited | exist ;

<animate_transvb> = discover | discovers | discovered ;
<inanimate_transvb> = orbit | orbits | orbited ;
<linkingvb> = is | was | are | were ;
<questl> = did | do | does ;
<det> = a | an | every | one | two | three | four | five ;
<nonhuman_pnoun_planet> = earth | jupiter | mars | mercury | neptune |
                  pluto | saturn | uranus | venus ;
<nonhuman_pnoun_moon> = almathea | ariel | callisto | charon | deimos |
                  dione | enceladus | europa | ganymede | hyperion |
                  iapetus | io |janus | jupitereighth |
                  jupitereleventh | jupiterfourteenth |
                  jupiterninth | jupiterseventh | jupitersixth |
                  jupitertenth | jupiterthirteenth |
                  jupitertwelfth |luna |mimas | miranda | nereid|
                  oberon | phobos | phoebe | rhea | saturnfirst |
                  tethys | titan | titania | triton | umbriel ;
```

```
<human_pnoun> = bernard | bond | cassini | dollfus | fountain |
                galileo | hall | herschel | huygens | kowal |
                 kuiper | larsen | lassell | melotte |
                nicholson | perrine | pickering ;

<pnoun> = <nonhuman_pnoun_planet> | <nonhuman_pnoun_moon> |
          <human_pnoun> ;

public <bye> = good bye and go to sleep (bye);
```

APPENDIX III

User 1 data-sheet

		Original Grammar		Revised Grammar	
		Correct	Wrong	Correct	Wrong
Q1*	1	OK			WR
Q1*	2	OK			WR
Q1*	3		WR		WR
Q1*	4		WR		WR
Q1*	5		WR		WR
Q1*	6		WR		WR
Q2*	1		WR		WR
Q2*	2		WR		WR
Q2*	3		WR		WR
Q2*	4		WR		WR
Q2*	5		WR		WR
Q2*	6		WR		WR
Q3*	1	OK			WR
Q3*	2		WR		WR
Q3*	3		WR		WR
Q3*	4		WR		WR
Q3*	5		WR		WR
Q3*	6	OK			WR
Q4	1	OK		OK	
Q4	2	OK		OK	
Q4	3	OK		OK	
Q4	4	OK		OK	
Q4	5	OK		OK	
Q4	6	OK		OK	
Q5	1		WR	OK	
Q5	2		WR	OK	
Q5	3	OK		OK	
Q5	4		WR	OK	
Q5	5	OK		OK	
Q5	6	OK		OK	
Q6	1	OK		OK	
Q6	2	OK		OK	
Q6	3	OK		OK	

Q6	4	OK		OK	
Q6	5	OK		OK	
Q6	6	OK		OK	
Q7	1	OK		OK	
Q7	2	OK		OK	
Q7	3	OK		OK	
Q7	4	OK		OK	
Q7	5	OK		OK	
Q7	6	OK		OK	
Q8	1	OK		OK	
Q8	2	OK		OK	
Q8	3	OK		OK	
Q8	4	OK		OK	
Q8	5	OK		OK	
Q8	6	OK		OK	
Q9	1	OK		OK	
Q9	2	OK		OK	
Q9	3	OK		OK	
Q9	4	OK		OK	
Q9	5	OK		OK	
Q9	6	OK		OK	
Q10	1	OK		OK	
Q10	2	OK		OK	
Q10	3	OK		OK	
Q10	4	OK		OK	
Q10	5	OK		OK	
Q10	6	OK		OK	
Q11	1	OK		OK	
Q11	2	OK		OK	
Q11	3	OK		OK	
Q11	4	OK		OK	
Q11	5	OK		OK	
Q11	6	OK		OK	
Q12	1	OK		OK	
Q12	2	OK		OK	
Q12	3	OK		OK	
Q12	4	OK		OK	
Q12	5	OK		OK	
Q12	6	OK		OK	
Q13	1	OK		OK	
Q13	2	OK		OK	
Q13	3	OK		OK	
Q13	4	OK		OK	
Q13	5	OK		OK	
Q13	6	OK		OK	
Q14	1	OK		OK	
Q14	2	OK		OK	
Q14	3	OK		OK	
Q14	4	OK		OK	
Q14	5	OK		OK	
Q14	6	OK		OK	
Q15	1	OK		OK	
Q15	2	OK		OK	

Q15	3	OK		OK	
Q15	4	OK		OK	
Q15	5	OK		OK	
Q15	6	OK		OK	
Q16	1		WR	OK	
Q16	2		WR	OK	
Q16	3		WR	OK	
Q16	4		WR	OK	
Q16	5		WR	OK	
Q16	6	OK		OK	
Q17	1	OK		OK	
Q17	2	OK		OK	
Q17	3	OK		OK	
Q17	4	OK		OK	
Q17	5	OK		OK	
Q17	6	OK		OK	
Q18	1	OK		OK	
Q18	2	OK		OK	
Q18	3	OK		OK	
Q18	4	OK		OK	
Q18	5	OK		OK	
Q18	6	OK		OK	
Q19	1	OK		OK	
Q19	2	OK		OK	
Q19	3	OK		OK	
Q19	4	OK		OK	
Q19	5	OK		OK	
Q19	6	OK		OK	
Q20	1	OK		OK	
Q20	2	OK		OK	
Q20	3	OK		OK	
Q20	4	OK		OK	
Q20	5	OK		OK	
Q20	6	OK		OK	
Q21	1	OK		OK	
Q21	2	OK		OK	
Q21	3	OK		OK	
Q21	4	OK		OK	
Q21	5	OK		OK	
Q21	6	OK		OK	
Q22	1	OK		OK	
Q22	2	OK		OK	
Q22	3	OK		OK	
Q22	4	OK		OK	
Q22	5		WR	OK	
Q22	6	OK		OK	
Q23	1		WR	OK	
Q23	2	OK		OK	
Q23	3	OK		OK	
Q23	4		WR	OK	
Q23	5	OK		OK	
Q23	6	OK		OK	
Q24	1	OK		OK	

Q24	2	OK		OK	
Q24	3	OK		OK	
Q24	4	OK		OK	
Q24	5	OK		OK	
Q24	6	OK		OK	
Q25*	1		WR		WR
Q25*	2		WR		WR
Q25*	3		WR	OK	
Q25*	4		WR	OK	
Q25*	5		WR	OK	
Q25*	6		WR		WR
Total		115	11	126	0
(Total/126)%		91.27%	8.73%	100%	0%

APPENDIX IV

User 2 data-sheet

		Original Grammar		Revised Grammar	
		Correct	Wrong	Correct	Wrong
Q1*	1	OK			WR
Q1*	2	OK			WR
Q1*	3	OK			WR
Q1*	4		WR		WR
Q1*	5		WR		WR
Q1*	6		WR		WR
Q2*	1		WR		WR
Q2*	2		WR		WR
Q2*	3		WR		WR
Q2*	4		WR		WR
Q2*	5		WR		WR
Q2*	6		WR		WR
Q3*	1		WR		WR
Q3*	2	OK			WR
Q3*	3		WR		WR
Q3*	4		WR		WR
Q3*	5		WR		WR
Q3*	6		WR		WR
Q4	1		WR		WR
Q4	2	OK		OK	
Q4	3		WR	OK	
Q4	4		WR	OK	
Q4	5		WR	OK	
Q4	6	OK		OK	
Q5	1		WR		WR
Q5	2		WR		WR
Q5	3		WR		WR
Q5	4		WR		WR
Q5	5		WR		WR
Q5	6		WR		WR
Q6	1		WR	OK	
Q6	2	OK		OK	
Q6	3		WR	OK	
Q6	4		WR	OK	

Q6	5	OK		OK	
Q6	6	OK		OK	
Q7	1	OK		OK	
Q7	2	OK		OK	
Q7	3	OK		OK	
Q7	4	OK		OK	
Q7	5	OK		OK	
Q7	6	OK		OK	
Q8	1	OK		OK	
Q8	2	OK		OK	
Q8	3	OK		OK	
Q8	4	OK		OK	
Q8	5	OK		OK	
Q8	6	OK		OK	
Q9	1	OK		OK	
Q9	2	OK			WR
Q9	3	OK		OK	
Q9	4	OK			WR
Q9	5	OK		OK	
Q9	6	OK		OK	
Q10	1	OK		OK	
Q10	2	OK		OK	
Q10	3	OK		OK	
Q10	4	OK		OK	
Q10	5	OK		OK	
Q10	6	OK		OK	
Q11	1	OK		OK	
Q11	2	OK		OK	
Q11	3	OK		OK	
Q11	4	OK		OK	
Q11	5	OK		OK	
Q11	6	OK		OK	
Q12	1	OK		OK	
Q12	2	OK		OK	
Q12	3	OK		OK	
Q12	4	OK		OK	
Q12	5	OK		OK	
Q12	6	OK		OK	
Q13	1	OK		OK	
Q13	2	OK		OK	
Q13	3	OK		OK	
Q13	4	OK		OK	
Q13	5	OK		OK	
Q13	6	OK		OK	
Q14	1	OK		OK	
Q14	2	OK		OK	
Q14	3	OK		OK	
Q14	4	OK		OK	
Q14	5	OK			WR
Q14	6	OK		OK	
Q15	1	OK			WR
Q15	2	OK			WR
Q15	3	OK		OK	

Q15	4	OK		OK	
Q15	5	OK		OK	
Q15	6	OK		OK	
Q16	1		WR		WR
Q16	2	OK		OK	
Q16	3	OK		OK	
Q16	4		WR	OK	
Q16	5		WR	OK	
Q16	6		WR	OK	
Q17	1	OK			WR
Q17	2	OK		OK	
Q17	3	OK		OK	
Q17	4	OK		OK	
Q17	5	OK		OK	
Q17	6	OK		OK	
Q18	1	OK			WR
Q18	2	OK		OK	
Q18	3	OK		OK	
Q18	4	OK		OK	
Q18	5	OK			WR
Q18	6	OK		OK	
Q19	1		WR	OK	
Q19	2		WR		WR
Q19	3	OK		OK	
Q19	4		WR	OK	
Q19	5		WR	OK	
Q19	6		WR	OK	
Q20	1	OK		OK	
Q20	2	OK			WR
Q20	3		WR		WR
Q20	4	OK		OK	
Q20	5		WR	OK	
Q20	6		WR	OK	
Q21	1	OK		OK	
Q21	2	OK		OK	
Q21	3	OK		OK	
Q21	4	OK		OK	
Q21	5	OK		OK	
Q21	6	OK		OK	
Q22	1		WR	OK	
Q22	2	OK			WR
Q22	3		WR	OK	
Q22	4		WR	OK	
Q22	5		WR	OK	
Q22	6		WR	OK	
Q23	1	OK		OK	
Q23	2	OK		OK	
Q23	3	OK		OK	
Q23	4	OK		OK	
Q23	5	OK		OK	
Q23	6	OK		OK	
Q24	1		WR		WR
Q24	2		WR		WR

Q24	3		WR	OK	
Q24	4		WR	OK	
Q24	5		WR	OK	
Q24	6		WR	OK	
Q25*	1		WR		WR
Q25*	2		WR		WR
Q25*	3		WR	OK	
Q25*	4		WR	OK	
Q25*	5		WR	OK	
Q25*	6		WR	OK	
Total		90	36	104	22
(Total/126)%		71.43%	28.57%	82.54%	17.46%

APPENDIX V

User 3 data-sheet

		Original Grammar		Revised Grammar	
		Correct	Wrong	Correct	Wrong
Q1*	1		WR		WR
Q1*	2		WR		WR
Q1*	3		WR		WR
Q1*	4		WR		WR
Q1*	5		WR		WR
Q1*	6		WR		WR
Q2*	1		WR		WR
Q2*	2		WR		WR
Q2*	3		WR		WR
Q2*	4		WR		WR
Q2*	5		WR		WR
Q2*	6		WR		WR
Q3*	1		WR		WR
Q3*	2		WR		WR
Q3*	3		WR		WR
Q3*	4		WR		WR
Q3*	5		WR		WR
Q3*	6		WR		WR
Q4	1	OK		OK	
Q4	2	OK		OK	
Q4	3	OK		OK	
Q4	4	OK		OK	
Q4	5	OK		OK	
Q4	6	OK		OK	
Q5	1	OK			WR
Q5	2		WR		WR
Q5	3	OK		OK	
Q5	4		WR	OK	
Q5	5		WR	OK	
Q5	6		WR		WR
Q6	1	OK		OK	
Q6	2	OK		OK	
Q6	3	OK		OK	

Q6	4		WR		WR
Q6	5	OK		OK	
Q6	6	OK		OK	
Q7	1	OK		OK	
Q7	2	OK		OK	
Q7	3	OK		OK	
Q7	4	OK		OK	
Q7	5	OK		OK	
Q7	6	OK		OK	
Q8	1	OK		OK	
Q8	2	OK		OK	
Q8	3	OK		OK	
Q8	4		WR	OK	
Q8	5	OK		OK	
Q8	6	OK		OK	
Q9	1	OK		OK	
Q9	2	OK		OK	
Q9	3	OK		OK	
Q9	4	OK		OK	
Q9	5	OK		OK	
Q9	6	OK		OK	
Q10	1	OK		OK	
Q10	2	OK		OK	
Q10	3	OK		OK	
Q10	4	OK		OK	
Q10	5	OK		OK	
Q10	6	OK		OK	
Q11	1	OK		OK	
Q11	2	OK		OK	
Q11	3	OK		OK	
Q11	4	OK		OK	
Q11	5	OK		OK	
Q11	6	OK		OK	
Q12	1	OK		OK	
Q12	2	OK		OK	
Q12	3	OK		OK	
Q12	4	OK		OK	
Q12	5	OK		OK	
Q12	6	OK		OK	
Q13	1	OK		OK	
Q13	2	OK		OK	
Q13	3	OK		OK	
Q13	4	OK		OK	
Q13	5	OK		OK	
Q13	6		WR	OK	
Q14	1	OK		OK	
Q14	2	OK		OK	
Q14	3	OK		OK	
Q14	4	OK		OK	
Q14	5	OK		OK	
Q14	6	OK		OK	
Q15	1	OK		OK	
Q15	2	OK		OK	

Q15	3	OK		OK	
Q15	4	OK		OK	
Q15	5	OK		OK	
Q15	6	OK		OK	
Q16	1	OK		OK	
Q16	2	OK		OK	
Q16	3	OK		OK	
Q16	4	OK		OK	
Q16	5	OK		OK	
Q16	6	OK		OK	
Q17	1	OK		OK	
Q17	2	OK		OK	
Q17	3	OK		OK	
Q17	4	OK		OK	
Q17	5	OK		OK	
Q17	6	OK		OK	
Q18	1	OK		OK	
Q18	2	OK		OK	
Q18	3	OK		OK	
Q18	4	OK		OK	
Q18	5	OK		OK	
Q18	6	OK		OK	
Q19	1	OK		OK	
Q19	2	OK		OK	
Q19	3	OK		OK	
Q19	4	OK		OK	
Q19	5	OK		OK	
Q19	6	OK			WR
Q20	1	OK		OK	
Q20	2	OK		OK	
Q20	3	OK		OK	
Q20	4	OK		OK	
Q20	5	OK		OK	
Q20	6	OK		OK	
Q21	1		WR	OK	
Q21	2		WR	OK	
Q21	3		WR	OK	
Q21	4		WR	OK	
Q21	5		WR	OK	
Q21	6	OK		OK	
Q22	1	OK		OK	
Q22	2		WR	OK	
Q22	3		WR	OK	
Q22	4	OK		OK	
Q22	5		WR	OK	
Q22	6	OK		OK	
Q23	1	OK		OK	
Q23	2	OK		OK	
Q23	3	OK		OK	
Q23	4	OK		OK	
Q23	5	OK		OK	
Q23	6	OK		OK	
Q24	1	OK		OK	

Q24	2	OK		OK	
Q24	3	OK		OK	
Q24	4	OK		OK	
Q24	5	OK		OK	
Q24	6		WR		WR
Q25*	1		WR		WR
Q25*	2		WR		WR
Q25*	3		WR		WR
Q25*	4		WR		H
Q25*	5		WR		N
Q25*	6		WR		N
Total		110	16	120	6
(Total/126)%		87.30%	12.70%	95.24%	4.76%

APPENDIX VI

Actual data sheets

The actual data sheets from the three test users are holden by School of Computer Science in University of Windsor. The data sheets are total of 172 pages (see the hand draft number from page1 to page 172), which have been photocopied into 86 sheets. The order of page1 to page 150 shows the sequence from user1 to user3. The page1 to page 50 are from user1, which including the original grammar test sheets from page1 to page25 and the revised grammar test sheets from 26 to 50. Analogously, from page 51 to 75 is user2's original grammar test sheets, page 76 to 100 is the revised grammar test sheets of user2, and page101 to 125 is user3's original grammar test data and the page126 to 150 is the revised grammar test results for user3. In every 25-hand draft pages, the queries sequence is strictly following the order of queries showed at the section 9.4 (pp 44). Each hand draft page has only one query and 6 attempts. The data sheets from 151 to 172 show the experiment II.

VITA AUCTORIS

NAME:	Linan Li
PLACE OF BIRTH:	Changsha, CHINA
EDUCATION:	University of Windsor, Windsor, Ontario 1998-2000 Bachelor of Science in Computer Science
	University of Windsor, Windsor, Ontario 2000-2002 Master of Science in Computer Science