Pacific Association for Computational Linguistics

A NEW APPROACH FOR PROVIDING NATURAL-LANGUAGE SPEECH ACCESS TO LARGE KNOWLEDGE BASES

R. A. FROST and S. CHITTE

Department of Computer Science, University of Windsor, Ontario, Canada

Constructing speech interfaces to large knowledge bases is difficult. Increasing the scope of the knowledge base often results in a decrease in speech-recognition accuracy. Re-engineering the input language to improve accuracy often necessitates non-trivial modification to the knowledge-base query processor. This problem is compounded if sophisticated techniques are used to analyze input to obtain context information to guide the speech recognizer. A partial solution to this problem has been developed. The knowledge base is divided into a collection of speech-accessible hyperlinked "sihlos" which are distributed over the Internet. Each sihlo has an associated grammar which is downloaded by the speech browser and used to configure the recognizer for that part of the knowledge base which is in scope. Sihlos are constructed as highly-modular executable specifications of attribute grammars. Re-engineering of the input language is still often required. However, concurrent modification to the sihlo query processor is now much easier. Experimentation with a prototype implementation has identified some linguistic issues which appear not to have been addressed elsewhere. The prototype provides a unique laboratory for the study of these issues.

Key words: Question-answering systems, natural-language speech interfaces, distributed databases, speech-recognition.

THE PROBLEM

Increasing the scope of a knowledge base usually requires that the query language be extended. This in turn, often results in reduced speech-recognition accuracy if a speech interface is used. This can be alleviated to some extent by re-engineering the input language using techniques such as:

1. Restricting the vocabulary (Moody 1996).
2. Restricting the input language to include only those utterances that are semantically as well as syntactically correct. This is achieved by coding semantic rules as syntax rules (Young, Hauptmann, Ward, Smith and Werner 1989; Seneff 1992).
3. Replacing problem words and phrases with equivalent words or phrases that are more easy to distinguish.
4. Modifying the input language so that the recognition search space at problematic points in the utterance is reduced. For example, recognition accuracy can be improved for syntax-directed speech recognizers if proper names are preceded by qualifying phrases which limit the search space at the problem point. For example, replacing `Hall` by `the person called Hall` reduces the search space from the total number of proper-names in the system to the number of names of people.

A major difficulty when re-engineering the input language is that it necessitates modification to the query processor. Unless the query processor has been constructed in a highly modular form, this in itself can be problematic.

Speech-recognition accuracy can also be improved by use of contextual information. The input can be analyzed in real-time for semantic clues in order to guide the recognizer. This approach works well when information from a dialogue with the user is available for analysis, or in extended dictation where previous utterances can be used to guide the recognizer. This approach is of limited value in knowledge-base query applications, where the analysis of a single query provides little help in recognizing the query itself. Context has to be provided through a different mechanism.

2. THE SOLUTION

The problem above can be alleviated by the integrated use of two concepts:

1. Implementing the knowledge base as a network of speech-accessible hyperlinlked objects called a SpeechNet, SpeechWeb, or HyperSpeechNet.
2. Implementing the query processors, associated with units of knowledge, as executable specifications of attribute grammars.

A SpeechWeb consists of a collection of speech-accessible hyperlinked objects called sihlos. Sihlos are distributed over a network. Each sihlo has an associated grammar which is downloaded by remote speech browsers which use the grammar to configure their speech recognizers to achieve higher recognition accuracy. Sihlos contain speech-activated hyperlinks to other sihlos on the network. It has been demonstrated (Frost 1999a; Frost and Chitte 1999) that this approach is viable and that SpeechWeb browsers with high-recognition accuracy and real-time response can be built using off-the-shelf speech-recognition technology.

It has been shown that highly modular and completely-declarative programs can be constructed as executable specifications of attribute grammars (Frost 1992). The advantages of the "attribute-grammar programming paradigm" have been described in detail in (Paakki 1995). This approach is particularly well-suited to the construction of natural-language database interfaces as discussed in (Frost 1999b).

The integration of the SpeechWeb and executable attribute grammar concepts would appear to hold much promise. Both the knowledge base and the interfaces to it are highly modular. This provides:

1. "Context-sensitive" reconfigurability of speech-recognizers using grammars relevant for units of knowledge that are in scope.
2. Modifiability for the purpose of improving speech-recognition accuracy through re-engineering of input languages.
3. Scalability for increasing the scope of the knowledge base without affecting recognition-accuracy or access to existing knowledge modules.
4. Openness — allowing widescale speech access to knowledge and to tools which facilitate the addition of new knowledge units anywhere on the network.

3. THE CURRENT IMPLEMENTATION OF SPEECHWEB

SpeechWeb currently consists of a collection of speech-accessible hyperlinked objects called sihlos which reside on servers on the Internet. Each sihlo consists of a grammar defining its input language, a language processor, and a set of voice properties. When a speech-browser first contacts a remote sihlo, it downloads the grammar and voice properties in order to configure its speech-recognizer and voice synthesizer. User input, if recognized, is converted to text by the browser, which forwards the text to the sihlo. The response is returned to the speech browser which converts it to synthesized voice output.

In the current implementation, communication is based on http and the cgi-bin protocol. This allows sihlos to be stored on Web servers and accessed by remote speech browsers anywhere where access to the Internet is available.

The prototype speech browser is written in Java and uses IBM's Via Voice speech-recognition engine and IBM's implementation of Java Speech APIs (Application Interfaces). The prototype executes on a regular PC.

The following is a transcript of a speaker–independent continuous-speech dialogue with a prototype SpeechWeb, that took place in March 1999, as part of demonstrations given to high-school students visiting the University of Windsor in Ontario, Canada. The computer successfully recognized all speech inputs except "what do you know , solar man?" which required two attempts.

```
y: Hello, is Monty here?
c: Hi, yes, I am Monty.
y: Hello Monty.
c: Hello, how are you?
y: Fine thanks.
c: Good so am I.
y: How old are you?
c: Older than you think but younger than my friend solar man.
y: Who is solar man?
c: He knows  about the solar system.
y: Can I talk to solar man?
c: Yes I will get him for you
HYPERLINK FOLLOWED TO THE SLOLAR MAN SIHLO,
 SOLAR MAN GRAMMAR DOWNLOADED,
 AND CHANGE OF VOICE TO THAT OF AN OLDER MAN
c: Hello I am solar man.
y: What do you know, solar man?
c: I know about the planets, moons and people who discovered them.
y: Which moons were discovered by Hall?
c: Phobos and Deimos.
y: Which planet is orbited by Deimos?
c: Mars.
y: How many moons were discovered by hall or Kuiper?
c: Four.
y: Do you know any poems?
c: No, but my friend Judy does.
y: Can I talk to Judy?
c: 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
c: Hello, I am Judy, would you like to hear a poem?
y: Yes, I would like to hear a poem.
c: JUDY READS A SHORT POEM
```

4. CONSTRUCTING SIHLOS AS EXECUTABLE SPECIFICATIONS

It has been shown that if the speech-recognizer and the query processor are both implemented as executable specifications, this facilitates the re-engineering of the input language to improve recognition accurracy (Frost 1995; Frost and Haddad 1998). Consequently, it is appropriate to construct the sihlo interpreters as executable specifications.

The `solar man` sihlo is constructed entirely as an 800–line executable specification of an attribute grammar in a programming language called W/AGE developed at the University of Windsor (Frost 1994). It can answer several thousand questions.

Solar man computes answers to queries using a computational strategy that is loosely based on Richard Montague's approach to the semantics of natural language (Dowty, Wall and Peters 1981). Montague's theory is ideally suited for implementation as an executable attribute grammar as his approach links semantic denotations directly with syntactic constructs.

The following is a brief overview of the solar man code. A complete listing, and a text-based interface to solar man is available for investigation at the following URL:

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

Using W/AGE, one begins by declaring the types of the attributes denoted by expressions of the input language. For example:

```
attribute  ::=  SENT_VAL            boolean
              | NOUNCLA_VAL         entity_set
              | etc
```

The meanings of "base" words is now defined by relating them to database relations, or functions in set theory. For example:

```
 [("man",       "cnoun",     [NOUNCLA_VAL set_of_men]),
  ("venus",     "pnoun",     [TERMPH_VAL (test_wrt 10)]),
  ("discover",  "transvb",   [VERB_VAL   (trans_verb rel_discover)]),
  ("or",        "verbphjoin",[VBPHJOIN_VAL   union]),
```

Other "non-base" words can be added to the dictionary by defining then in terms of phrases whose meanings can be computed by recursive application of the executable attribute grammar (or some component of it). For example:

```
("person",   "cnoun",    meaning_of nouncla "man or woman"),
  etc.
```

The body of the attribute grammar is then defined by writing down syntax rules (productions) and attribute computation rules (semantic rules) associated with them. For example, the following code states how the `NOUNCLA_VAL` of a simple noun clause `snouncla` is computed by intersecting the meanings of the common noun `cnoun` and adjectives `adjs` from which it is composed.

```
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]]
```

The input language can be extended further by defining new syntactic constructs as re-writes of constructs whose meaning can be computed by recursive application of the attribute grammar or components of it. For example, the following simple re-write rule causes inputs such as "How many moons are there?" to be re-written into "How many moons exist?"

```
non_core_transformed_verbph  = rewrite_are_there $orelse rewrite_did
rewrite_are_there = rewriting 2.1
                    OF (s1 linkingvb ++ s2 (the_word "there"))
                    AS (the_phrase "exist")
etc.
```

The semantic rules which are used to compute attributes of a complex expression from the attributes of its components are now defined in terms of basic functions:

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

Finally, the semantic functions that are used to relate the meanings of words to set and relational operators are defined. For example:

```
test_wrt e s        = member s e
make_trans_vb rel p = [x | (x, image_x) <- collect rel; p image_x]
termph_and p q      = g where g x = (p x) & (q x), etc.
```

Language processors that are constructed in this way are highly modular. Component interpreters such as `snouncla` can be used independently provided that their components are also available.

Current work (Frost and Chitte 1999) has shown that a similar approach can be implemented in object-oriented languages through the creation of "executable grammar objects". For example, a recognizer class can be declared and extended to the subclasses of "basic recognizer", "alternative recognizer", and "sequence recognizer". The viability of this approach has been demonstrated through an implementation in Java. The approach is not quite as elegant as in functional programming languages, but does allow recognizers to be constructed as executable specifications which are very similar in structure to the grammars of the languages to be processed.

5. LINGUISTIC ISSUES

The work described in this paper raises two linguistic issues:

1. What rules should guide the designer of a sihlo in how best to incorporate speech-activated hyperlinks to other sihlos? The example session illustrated the use of very simple speech prompts: the input "Can I talk to Solar man?" caused the sihlo to send data to the speech browser causing it to change access to the `solar man` sihlo, download `solar man`'s grammar and change voice appropriately. The user was alerted to `solar man`'s existence through the earlier response `Older than you think but younger than my friend solar man`. Some work has been done on the hyperlinking of recorded speech segments (Arons 1991). However, no work appears to have been done on developing theories to support the construction of collections of speech-accessible hyperlinked objects that support user-friendly speech browsing.
2. The construction of both the speech browser and the natural language query processors as highly modular executable specifications facilitates the construction, modification and extension of speech-accessible knowledge bases. However, no theory exists to guide one in designing an appropriately-structured collection of sihlos. Sihlos should be "small" to improve speech-recognition accuracy. But they should not be too small otherwise user input

will be "out of scope" more often than not. Use of sihlos which act as indexes, and the careful categorization and distribution of knowledge across sihlos will also help, but there remains a related problem: when new knowledge is added, speech-activated hyperlinks must be added to other sihlos in order to integrate that knowledge. Depending on the name of the new sihlo (or the means with which a hyperlink to it is to be prompted), the recognition accuracy of other sihlos which link to it may be degraded. This problem is related to the growth in the number of links on a Web HTML page. However, that problem is simpler — just replace the page with an index linked to two or more smaller pages which contain the information on the original page. It is not clear how this can be best achieved with speech browsing.

SpeechWeb and the use of modular attribute grammars provides an ideal environment in which to conduct empirical studies related to the development of theories for use of context in speech recognition and theories for the design of modular hyperlinked speech-accessible knowledge structures.

6. RELATED WORK

Some excellent award-winning work has been done on building Web interfaces for visually-challenged users, e.g. (Zajicek, Powell and Reeves 1999). Although that work is relevant to the use of speech-recognition technology, the objectives are somewhat different to the work described here which presents an alternative way of storing and accessing knowledge for speech access (rather than speech-access to knowledge already stored in hypertext pages).

Other related work includes the "Hyperspeech" project (Arons 1991). That work investigated "techniques for presenting 'speech as data', allowing a user to navigate by voice through a database of recorded speech without any visual clues. The ideas being developed can be applied to create a generalized form of interaction with unstructured speech data. Applications for such a technology include the use of recorded speech, rather than text, as a brainstorming tool or personal memory aid. A talking system would allow a user to create, organize, sort, and filter audio notes under circumstances where a traditional graphical interface would not be practical". In the Hyperspeech project, access to segments of speech itself is the focus of attention, rather than speech access to knowledge.

A non-visual speech browser has been developed by (Morley, Petrie, O'Neil and McNally 1998) but uses techniques other than speech to navigate the knowledge base. Surprising little other work appears to have been done on speech-access to networks of objects.

Sihlos could be constructed with the help of "parser-generators" such as Cup developed by Scott Hudson which is written in an object-oriented language, and which produces parsers and evaluators implemented in an object-oriented language. Cup is now maintained at Princeton. Detailed information can be found at

`http://www.cs.princeton.edu/~appel/modern/java/CUP/CUPman.HTML`

Although parser generators are appropriate for use in many applications, they have three major shortcomings: the language processors which they generate are not modular, their interface with the rest of the application is usually non-trivial, and they typically do not accommodate grammars for ambiguous languages.

The shortcomings of parser-generators makes them somewhat unsuitable for constructing certain types of sihlos and they can therefore be thought of as complementing the use of executable grammar objects.

In addition to parser-generators, systems have been built which construct evaluators automatically by compiling attribute grammars which define both the syntax and the semantics of the language to be processed. One of the first attribute-grammar compilers to be developed using the object-oriented technology is LISA (Language Implementation System based on Attribute grammars), which is written in, and produces language processors in, C++ (Zumer, Kortar and Merick, 1997). Such systems may be appropriate for constructing sihlos.

Techniques related to executable grammar objects have been presented by other researchers. For example: (Moreira and Clark, 1996) have developed a method which integrates formal description techniques with standard object-oriented analysis methods. In their approach, the specifications are executable and prototyping can be used to validate the specification against the requirements, thereby enabling the early detection of inconsistencies, omissions and ambiguities in the requirements definition. (Peake and Salzman, 1997) have extended the functional approach to modular parsing by adding the object-oriented constructs of class inheritance and dynamic method dispatch. Their approach differs from the use of executable grammar objects as it extends the functional approach rather than adapting it for use in an object-oriented language.

7. CONCLUDING COMMENTS

The cgi-bin protocol only supports sessionless communication. To overcome this, SpeechWeb is currently being re-implemented using the Common Object Request Broker Architecture (CORBA 1999). This will enable dialogue with sihlos.

The W/AGE programming language is currently implemented as a set of library functions added to the functional programming language Miranda. To make W/AGE more accessible, it is currently being re-implemented in Java using executable grammar objects.

The goal of the work described in this paper is to augment the Web with mechanisms by which a large body of knowledge can be generated and made available for non-visual access through user-friendly speech browsers.

ACKNOWLEDGEMENTS

Many people at the University of Windsor have contributed to the project. In particular, the authors wish to acknowledge Tarek Haddad, who developed an earlier speech interface to Web pages, Barbara Szydlowski who wrote a text-based Web-interface to solar-man, Dr. Ono Tjandra who patiently explained the advantages and disadvantages of CORBA, and Walid Mnaymneh, Maunzer Batal, and Stephen Karamatos who provided technical help throughout the project.

The author also wishes to acknowledge the support received from NSERC in the form of an individual research grant.

REFERENCES

ARONS, B. 1991. Hyperspeech: Navigating in Speech-Only Hypermedia. In Proceedings of Hypertext '91, San Antonio, TX, Dec. 15-18, ACM, New York, 1991:133-146.

AUGUSTEIJN, L. 1993. Functional Programming, Program Transformations and Compiler Construction. Phillips Research Laboratories. ISBN 90–74445–04–7.

CORBA: `http://www.egr.msu.edu/~thakkarv/corba.HTML`

Cup:`http://www.cs.princeton.edu/~appel/modern/java/CUP/CUPman.HTML`

DOWTY, D. R., WALL, R. E. and PETERS, S. 1981. Introduction to Montague Semantics. D. Reidel Publishing Company, Dordrecht, Boston, Lancaster, Tokyo.

FROST, R. A., and CHITTE, S. 1999. Sihlos: Speech-Accessible Hyperlinked Objects. Submitted to OOPSLA 99.

FROST, R. A. 1999a. SpeechNet: A network of hyperlinked speech-accessible objects. Proceedings of teh IEEE WECWIS International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems. Joint Workshop of (3rd RTDB and 2nd DARE), San Jose, April 1999: 71–76.

FROST, R. A. 1999b. A natural language speech interface constructed entirely as a set of executable specifications. Accepted for AAAI Intelligent Systems Demonstrations. Orlando July 1999.

FROST, R. A. and HADDAD, T. 1998. Engineering and re-engineering a speech interface to the Web. Proceedings of the 9th International Conference on Computing and Information, ICCI'98. University of Manitoba, June 1998: 237–244.

FROST, R. A. 1995. Use of executable specifications in the construction of speech interfaces. Proceedings of the IJCAI Workshop on Developing AI Applications for the Disabled, Montreal 1995.

FROST, R. A. 1994. W/AGE The Windsor attribute grammar programming environment. Schloss Dagstuhl International Workshop on Functional Programming in the Real World. 1994.

FROST R. A. 1992. Constructing programs as executable attribute grammars. The Computer Journal 35(4):376 — 389.

JOHNSON, S. C. 1975. YACC – Yet Another Compiler Compiler, CS Technical Report #32, Bell Telephone Laboratories, Murray Hill, NJ.

LEERMAKERS, R. 1993. The Functional Treatment of Parsing. Kluwer Academic Publishers, ISBN 0–7923–9376–7.

MOODY, T. S. 1988. The effects of restricted vocabulary size on voice discourse structures. Ph.D. Thesis, North Carolina State University.

MOREIRA, A, and CLARK, R 1996. Adding rigor to object-oriented analysis. IEE Software Engineering Journal 11 (5):270–280.

MORLEY, S., PETRIE, H., O'NEILL, A., McNALLY, P. 1998. Auditory Navigation in Hyperspace: Design and Evaluation of a Non-Visual Hypermedia System for Blind Users. The Third International ACM SIGCAPH Conference on Assistive Technologies ASSETS '98, April 15-17, 1998, Marina del Rey, CA USA

PAAKKI, J. 1995. Attribute grammar paradigms — a high-level methodology in language implementation, ACM Computing Surveys 27(2):196–255.

PEAKE. I. and SALZMANN, E. 1997. Support for modular parsing in software reengineering. Proceedings of the International Workshop on Software Technology and Engineering Practice, STEP Jul 14–18 1997:58–66.

PEREIRA, F. and WARREN, D. H. D. 1978. Definite Clause Grammars compared with Augmented Transition Networks. Technical Report, Department of Artificial Intelligence, University of Edinburgh.

SENEFF, S. 1992. TINA: A natural language system for spoken language applications. Computational Linguistics March 1992:61–86.

YOUNG, S. R., HAUPTMANN, A. G., WARD W. H., SMITH, E. T. and WERNER, P. 1989. High level knowledge sources in usable speech recognition systems. CACM 32 (2):183–194

ZUMER, V., KORBAR, N. and MERNICK, M. 1997. Automatic implementation of programming languages using object oriented approach, Journal of Systems Architecture 43 (1):203–210.

ZAJICEK, M., POWELL, C., REEVES, C., 1999, 'Web search and orientation with BrookesTalk,' California State University Northridge, CSUN '99, Technology and Persons with Disabilities, Los Angles.