Another Stride Towards Knowledge-Based Machine Translation

Masaru Tomita
Jaime G. Carbonell
Computer Science Department
Carnegie-Mellon University
Pittsburgh, PA 15213

12 April 1986

Abstract

Building on the well-established premise that reliable machine translation requires a significant degree of text comprehension, this paper presents a recent advance in multi-lingual knowledge-based machine translation (KBMT). Unlike previous approaches, the current method provides for separate syntactic and semantic knowledge sources that are integrated dynamically for parsing and generation. Such a separation enables the system to have syntactic grammars, language specific but domain general, and semantic knowledge bases, domain specific but language general. Subsequently, grammars and domain knowledge are precompiled automatically in any desired combination to produce very efficient and very thorough real-time parsers. A pilot implementation of our KBMT architecture using functional grammars and entity-oriented semantics demonstrates the feasibility of the new approach.[1]

1. Introduction

This paper introduces a new approach to knowledge-based machine translation for well-defined domains, integrating two recent advances in computational linguistics: entity-oriented parsing [16] and functional grammars [4, 19]. The entity-oriented formalism has several strengths in representing semantic knowledge for circumscribed domains, but has limitations in representing general syntactic knowledge. Functional grammar formalisms, such as lexical functional grammar (LFG) and functional unification grammar (UG), on the other hand, can represent general syntactic knowledge, but are severely limited in their ability to represent general semantic information. In our approach, the semantic and syntactic knowledge bases are developed separately in the entity-oriented and functional grammar formalisms, and a multi-stage *grammar precompiler* compiles them into a single knowledge base which contains both syntactic and semantic information in a form suitable for efficient real-time parsing. Our integrated approach copes with limitations of both entity-oriented and functional grammar formalisms, retaining the advantages of each. The approach is particularly well suited for machine translation, where knowledge of multiple languages must be represented in a uniform manner.

Knowledge-based machine translation (KBMT) [8] is the process of applying syntactic knowledge of the source language and semantic knowledge pertinent to the source text in order to produce a canonical language-free meaning representation, which may then be rendered in many different languages. The analysis process of producing a meaning representation is far more complex than that of using target-language knowledge to express the meaning representation in the target language, because the former is a many-to-one mapping, whereas the latter may be coerced into a one-to-one mapping.[2] Whereas KBMT is in principle far superior to conventional transfer grammar techniques requiring a human translator (the "posteditor") to clean up syntactic and semantic errors [5, 8], in practice semantic analysis requires fairly thorough coverage of the domain. This ravenous hunger for domain knowledge makes KBMT more practical for domains in which the development of the knowledge base can be amortized over very large numbers of texts to translate -- domains such as stocks and other security negotiations, doctor-patient communication, weather forecasts, banking transactions, financial reports, economic analyses, invoices and purchase orders, etc. Thus, KBMT is particularly well-suited for multi-lingual translation in high-volume well-defined semantic domains.

Whereas the technical feasibility of KBMT was proposed and demonstrated for limited domains by Carbonell, Cullingford and Gershman [5], its practical utility remained elusive. The entity-oriented approach factors linguistic and domain knowledge into separate data structures, thus making KBMT systems far more extensible and economically attractive than the earlier approaches. Moreover, recognizing that on occasion some esoteric domain knowledge necessary for semantic analysis will be lacking, we retain the possibility of interacting with a human user knowledgeable of the domain (but not of different target languages) to clarify any difficulties too complex for the domain semantics to handle, as illustrated in figure 1-1.

2. Background

Automating various forms of syntactic analysis has been a central concern of Computational Linguistics, producing methods ranging from context-free grammar interpreters [11, 25, 13], to ATNs [28], to unification grammars [18], and lexical-functional grammars [4]. The problem is that the production of accurate, unambiguous parses of the source text, tolerant of minor grammatical deviations, requires a fairly complete semantic model of the domain, and a method for bringing the semantic knowledge to bear in the parsing process. Semantically-oriented parsers have succeeded at integrating semantics with syntax, but only at the cost of intertwining both knowledge sources into the program

[1]The authors would like to acknowledge the other members of the the machine translation laboratory at CMU who contributed in various ways to the research described in this paper: Peggy Anderson, Philip Franklin, Alex Hauptmann, Marion Kee, Hiroaki Saito, Yuko Tomita and Teruko Watanabe.

[2]The analyzer needs to comprehend all possible syntactic variants of any semantic message in the analysis phase because it cannot control the form of its input, but to produce acceptable output, the generator need only render the the meaning in a well-defined standard surface form. Of course, to produce more expressive text, and to preserve syntactic as well as semantic invariance in the translation process, the generator must be expanded into a one-to-many mapping process comparable in complexity to that of the analyzer.

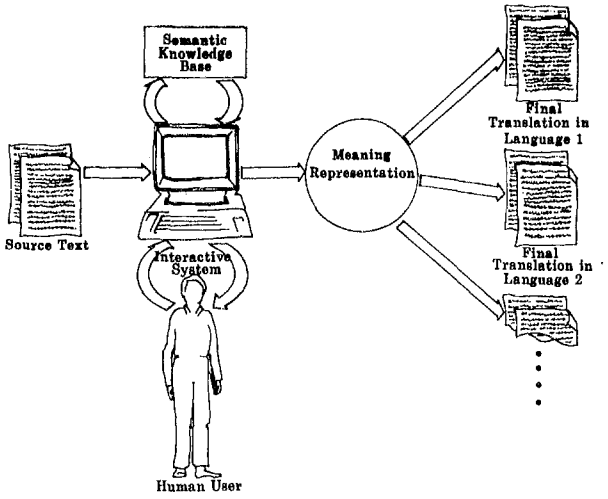

Figure 1-1: Knowledge-Based Interactive Machine Translation

itself in fairly non-extensible ways [23, 17, 2, 6]. Subsequent improvements have succeeded in factoring out much of the domain semantics, but leaving the syntactic interpretation as part of the recognition program rather than as an explicit external grammar [9, 14, 16].

In order to overcome these problems we have sought a method for *static separation* of the syntactic and semantic knowledge sources in the data structures, and *dynamic integration* to bring all relevant knowledge to bear in the process of parsing. Static separation has the advantage that as linguistic coverage increases, or new languages are added to the system, parsing (and translation) still function for all previous semantic domains. Conversely, if the semantic domains are extended, or new ones added, parsing and translation of texts in these domains will function for all previously entered languages. In contrast, earlier methods that mixed semantic and syntactic information required hand-crafted updates to all previous structures in order to integrate new grammatical extensions or new languages. With the possible exception of Lytinen [21], who attempted a rudimentary form of static separation and dynamic integration, this rather appealing principle has not heretofore been a primary design criterion of natural language parsers in general, much less full machine-translation systems.

Many of the syntactic analysis methods do not integrate well with semantic knowledge, especially knowledge that must be kept in separate data structures and integrated only by the precompiler at the run-time language intepretation process. Similarly, many of the semantic representation formalisms do not lend themselves well to dynamic integration with syntactic constraints at parse time. The best fit we have been able to achieve comes from precompiling syntactic and semantic knowledge into a single knowledge base which is used only at run-time, as described in the subsequent sections.

3. System Overview

Figure 3-1 shows the architecture of our current system. As mentioned in the previous section, we modularize domain-specific semantic knowledge and domain-independent (but language-specific) syntactic knowledge. We precompile semantic entities and LFG-style grammars into a single large grammar which is less perspicuous but more efficient. This merged grammar is further precompiled into a yet larger parsing table for added efficiency, enabling the run-time system to parse input text in a very efficient manner using the parsing algorithm recently introduced by Tomita [26, 25]. More on this issue shall be discussed in section 6.

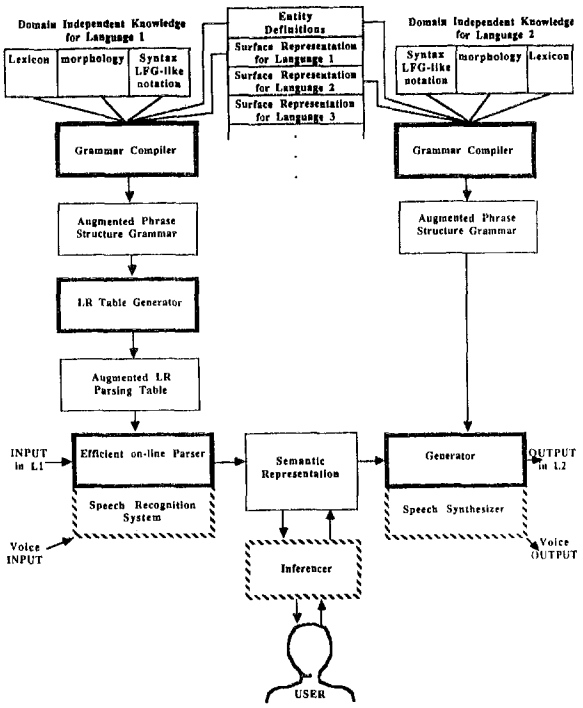

Figure 3-1: System Structure

4. The Entity-Oriented Approach

The entity-oriented approach to restricted-domain parsing was first proposed by Hayes [16] as a method of organizing semantic and syntactic information about all domain concepts around a collection of various entities (objects, events, commands, states, etc.) that a particular system needs to recognize. An entity definition contains information about the internal structure of the entities, about relations to other entities, about the way the entities will be manifested in the natural language input, and about the correspondence between the internal structure and multiple surface forms for each entity.

Let us consider the domain of doctor-patient conversations; in particular, the patient's initial complaint about some ailment. Entities in this domain include an event entity PATIENT-COMPLAINT-ACT and object entities PAIN, HUMAN and so on. A fragment of an entity-oriented grammar is shown in figure 4-1. The notation here is slightly simplified from that of Hayes. Sentences of different surface form that should be recognized as instantiations of this entity include:

I have a head ache
I have a burning pain in the chest.
I don't feel any pain.
Did you have a dull ache in your head?

```
[EntityName: PATIENT-COMPLAINT-ACT
 Type: STRUCTURED
 Agent: HUMAN            ; Semantic restriction on the agent.
 Pain: PAIN
 SurfaceRepresentation:
   [SyntaxType: SENTENTIAL
    Head: (have | feel)
    Subj: ($Agent)       ; $Agent and $Pain refer to the
    DObj: ($Pain)  ] ]   ; semantic cases above.
```

```
[EntityName: PAIN
 Type: STRUCTURED
 Location: BODY-PART     ; Semantic restriction on the location
 PainKind: PAIN-FEEL
 SurfaceRepresentation:
   [SyntaxType: NOUNPHRASE
    Head: (pain | ache)
    PP: ( [Prep: in
           Comp: ($Location) ]
        ) ,
    Adj: ( [AdjPhrase: (sharp | stabbing | acute | sudden)
            Component: PainKind
            Value: ACUTE ]
           [AdjPhrase: (dull | throbbing | diffuse | lasting)
            Component: PainKind
            Value: DIFFUSE ]
          )
    ]
]
```

Figure 4-1: Example Entity Definition

The final semantic representation of the sentence

"I have a dull ache in my chest"

produced by instantiating entities is shown in figure 4-2.

```
[cfname: MEDICAL-COMPLAINT-ACT
 type: SENTENTIAL
 agent: [cfname: PERSON
         name: *speaker*]   ; The "I" who has the chest ache.
 pain: [cfname: PAIN
        location: [cfname: BODY-PART
                   name: CHEST ]
        pain-kind: DIFFUSE]
]
```

Figure 4-2: Sample Semantic Representation: Instantiated Entities

The 'SurfaceRepresentation' parts of an entity guide the parsing by providing syntactic structures tied to the semantic portion of the entity. As the result of parsing a sentence (see figure 4-2), a composition of the semantic portion of the instantiated entities is produced. This knowledge structure may be given to any backend process, whether it be a language generator (for the target language), a paraphraser, a data-base query system, or an expert system.

The primary advantage of the entity-oriented grammar formalism hinges on its clarity of the sub-language definition (see Kittredge [20] for a discussion of sub-languages). Since all information relating to an entity is grouped in one place, a language definer will be able to see more clearly whether a definition is complete and what would be the consequences of any addition or change to the definition. Similarly, since syntactic and semantic information about an entity are grouped together, the former can refer to the latter in a clear and coherent way, both in the grammar production and in the run time system. This advantage is even more valuable in the application to multi-lingual machine translation. Because the semantic portions of the entities are totally language independent, we can use one set of entity definitions for all languages -- merely requiring that each entity have a multiple number of surface forms; one or more for each language. In this way, on can ensure that semantic coverage is consistent across all languages.

In addition to clarity and its multi-lingual extensibility, another advantage of the entity-oriented approach is robustness in dealing with extragrammatical input. Robust recovery from ill-formed input is a crucial feature for practical interactive language systems, but is beyond the immediate scope of this paper. See Carbonell and Hayes [7] for a full discussion on entity-based robust parsing.

The major limitation of entity-oriented grammars arises from the very same close coupling of syntax and semantics: all syntactic knowledge common across domains (or across entities within one domain) must be replicated by hand for each and every entity definition. Syntactic generalities are not captured. This problem is not merely an aesthetic one; it takes prodigious efforts for grammar developers to build and perfect each domain grammar, with little cross-domain transfer. How then, can one overcome this central limitation and yet retain all the advantages of semantic analysis in general and the entity-oriented approach in particular? The answer lies in decoupling the syntactic information at grammar development time -- thus having a general grammar for each language -- and integrating it via an automated precompilation process to produce highly coupled structures for the run-time system. Such an approach has been made possible through the advent of unification and functional grammars.

5. The Functional Grammar Formalism

Functional grammars, as presented by Kay [18], provide the key to automated compilation of syntactic and semantic knowledge. In essence, they define syntax in a functional manner based on syntactic roles, rather than by positions of constituents in the surface string. The functional framework has clear advantages for languages such as Japanese, where word order is of much less significance than in English, but case markings take up the role of providing the surface cues for assigning syntactic and semantic roles to each constituent. Moreover, functional structures integrate far more coherently into case-frame based semantic structures such as entity definitions.

Two well-known functional grammar formalisms are Functional Unification Grammar (UG) [19] and Lexical Function Grammar (LFG) [4]. In this paper, however, we do not distinguish between them and refer to both by the term "functional grammar". Application of the functional grammar formalism to machine translation is discussed in [19]. Attempts have being made to implement parsers using these grammars, most notably in the PATR-II project at Stanford [22, 24]. However, these efforts have not been integrated with external semantic knowledge bases, and have not been applied in the context of KBMT systems.

There are two main advantages of using the functional grammar formalism in practical machine translation systems:

- A system implemented strictly within the functional grammar formalism will be *reversible*, in the sense that if the system maps from A to B then, to the same extent, it maps from B to A. Thus, we do not need to write separate grammars for parsing and generation. We merely compile the same grammar into an efficient uni-directional structure for parsing, and a different uni-directional structure for generation into that language.

- Functional grammar formalisms such as UG and LFG are well-known among computational linguists, and therefore need not be trained (with some justifiable resistance) to write grammars in arcane system-specific formalisms.

The general problem in parsing with functional grammars is implementation inefficiency for any practical application. Although much work has been done to enhance efficiency [24, 22], the functional grammar formalisms are considered far less efficient than formalisms like ATNs [28] or (especially) context-free phrase structure grammars. We resolve this efficiency problem by precompiling a grammars written in a the functional grammar (together with a separate domain semantics specification) into an augmented context-free grammars, as described in the following section.

6. Grammar Precompilation and Efficient On-Line Parsing

The previous two sections have described two kinds of knowledge representation methods: the entity-oriented grammar formalism for domain-specific but language general semantic knowledge ,and the functional grammar formalism for domain-independent but language specific syntactic knowledge. In order

to parse a sentence in real time using these knowledge bases, we precompile the semantic and syntactic knowledge, as well as morphological rules and dictionary, into a single large morph/syn/sem grammar. This morph/syn/sem grammar is represented by a (potentially very large) set of context-free phrase structure rules, each of which is augmented with a Lisp program for *test* and *action* as in ATNs[3] A simplified fragment of a morph/syn/sem grammar is shown in figure 6-1.

```
patient-complaint-act-1-S  --> patient-NP  complaint-act-1-VP
((cond ((equal (not (getvalue '(x1: agr:)))
               (getvalue '(x2: agr:)))
       (return nil)))
 (setvalue '(x0: semcase:) (getvalue '(x2: semcase:)))
 (setvalue '(x0: semcase: agent:) (getvalue '(x1: semcase:)))
 (setvalue '(x0: syncase:) (getvalue '(x2: syncase:)))
 (setvalue '(x0: syncase: subj:) (getvalue '(x1:)))
 (return (getvalue '(x0:))))


complaint-act-1-VP  -->  complaint-act-1-V
((setvalue '(x0: semcase:) (getvalue '(x1: semcase:)))
 (setvalue '(x0: syncase: pred:) (getvalue '(x1:)))
 (setvalue '(x0: agr:) (getvalue '(x1: agr:)))
 (setvalue '(x0: form:) (getvalue '(x1: form:)))
 (return (getvalue '(x0:))))


complaint-act-1-V  -->  ACHE-V
((setvalue '(x0: semcase: cfname:) 'PATIENT-COMPLAINT-ACT)
 (setvalue '(x0: agr:) (getvalue '(x1: agr:)))
 (setvalue '(x0: form:) (getvalue '(x1: form:)))
 (return (getvalue '(x0))))
```

Figure 6-1: A Compiled Grammar Fragment

Once we have a grammar in this form, we can apply efficient context-free parsing algorithms, and whenever the parser reduces constituents into a higher-level nonterminal using a phrase structure rule, the Lisp program associated with the rule is evaluated. The Lisp program handles such aspects as construction of a semantic representation of the input sentence, passing attribute values among constituents at different levels and checking semantic and syntactic constraints such as subject-verb agreement. Recall that those Lisp programs are generated automatically by the grammar precompiler from LFG f-structures and semantic entities. Note also that the Lisp programs can be further compiled into machine code by the Lisp compiler.

We adopt the algorithm introduced by Tomita [25, 26] as our context-free parsing algorithm to parse a sentence with the morph/syn/sem grammar. The Tomita algorithm can be viewed as an extended LR parsing algorithm [1]. We compile further the morph/syn/sem grammar further into a table called the *augmented LR parsing table*, with which the algorithm works very efficiently.

The Tomita algorithm has three major advantages in the application of real-time machine translation systems:

- The algorithm is fast, due to the LR table precompilation; in several tests it has proven faster than any other general context-free parsing algorithm presently in practice. For instance, timings indicate a 5 to 10 fold speed advantage over Earley's algorithm in several experiments with English grammars and sample sets of sentences.

- The efficiency of the algorithm is not affected by the size of its grammar, once the LR parsing table is obtained. This characteristic is especially important for our system, because the size of the morph/syn/sem grammar will be very large in practical applications.

- The algorithm parses a sentence strictly from left to right, proving all the *on-line* parsing advantages describe below.

The *on-line* parser starts parsing as soon as the user types in the first word of a sentence, without waiting for the end of a line or a sentence boundary. There are two main benefits from on-line parsing:

- The parser's response time can be reduced significantly. When the user finishes typing a whole sentence, most of the input sentence has been already processed by the parser.

- Any errors, such as mis-typing and ungrammatical usages, can be detected almost as soon as they occur, and the parser can warn the user immediately without waiting for the end of the line.

Thus, on-line parsing provides major advantage for interactive applications (such as real-time parsing, immediate translation of telex messages, and eventual integration sith speech recognition and syntesis systems), but is transparent when operating in batch-processing mode for long texts. More discussion of on-line parsing can be found in Chapter 7 of Tomita [25].

7. Future Directions

The twin advantages of the KBMT approach and the reversible functional grammars, applied to f-structures and semantic entity definitions, are 1) to provide a measure of extensibility that cannot be achieved via the conventional transfer grammar approach, and 2) to enable efficient real-time parsing via multi-stage precompilation. A further advantage over traditional transfer grammars becomes evident when one considers the translation problem from a more global perspective. In order to translate between any pair of N languages, our approach requires the development of only N bi-directional grammars (one per language). On the other hand, the conventional transfer approach requires that a new grammar be developed for each *pair* of languages and for each *direction* of the translation. Thus, to achieve the same number of bi-directional translations, requires on the order of N^2 transfer grammars. This calculation yields over 5,000 transfer grammars vs 72 functional/entity grammars to translate among the 72 most commonly spoken languages today. Recall that in addition to the economy of development argument, the KBMT paradigm produces meaning-invariant translations for those domains whose semantics have been successfully codified.

Although we have made significant inroads in the establishment of knowledge-based machine translation as a viable and superior alternative to the transfer grammar methodology, much of the difficult work remains before us. The integration of entity-oriented semantic representations and a generalized functional grammar, coupled with grammar precompilers, on-line parsers and generator provide a significant improvement over the first successful attempts to perform knowledge-based machine translation [10, 5]. The improvements are based on extensibility and uniformity of the semantic and syntactic knowledge sources, providing static separation and dynamic "run-time" integration. Our initial implementations convince us that this approach may hold the key to practical KBMT.

Our pilot system operates in a subdomain of doctor-patient communications, selected for its relative syntactic richness, but fairly self-contained semantics. We have selected English and Japanese as our initial source and target languages, although we are also starting to investigate Spanish, French, German and Italian. Moreover, we are striving to produce a system requiring minimal if any clarification from the source-language user in his or

[3] To be exact, each rule has two Lisp programs; one for parsing and the other for generation. These programs are synthesized automatically by the precompiler in order to test semantic and syntactic constraints, including as long-distance dependencies, and to assign constituents their appropriate semantic and syntactic roles.

her own language, and no aid whatsoever from a human translator or "posteditor" who knows both languages. We intend to grow this pilot system in several dimensions, including achieving a measure of completeness in subdomain coverage, adding one or two more languages, moving to a second and perhaps a third domain, and tailoring our implementation for relative efficiency of operation by completing the development of our multi-phase precompilers.

In addition to continued construction and extension of the pilot system -- the vehicle through which we are testing our theoretical tenets -- we are pursing the following objectives:

- ***Bi-directionality*** -- As discussed above, functional grammars are theoretically bi-directional, but such a property has not yet been proven in practice for large scale systems. Our approach is not to interpret the bi-directional grammars directly, but rather to compile them into much more efficient (and different) parsing and generation grammars. The latter endeavor still requires empirical validation.

- ***Incremental Compilation*** -- In order to expedite the grammar development and testing cycle, we are contemplating incremental compilation for new additions or recent changes into large existing grammars rapidly. Although the compilation process has proven successful in earlier parsers we have built [3, 27], incremental compilation introduces new technical problems.

- ***User extensibility*** -- A longer range research topic is to provide a structured interface whereby a user of the KBMT system could add domain knowledge (entities) and dictionary entries without requiring any knowledge of the internal structure of the system. Extending the lexicon is, of course, much simpler than extending the domain semantics. All such extensions would work in concert with existing domain knowledge, lexicon, and grammar.

- ***Robustness*** -- The recognition of ill-structured language is very important, especially for the short-text domains we envision for our system (telex messages, banking transactions, doctor-patient dialogs, etc.). We have built selective-relaxation methods that integrate semantic and syntactic constrains before in the MULTIPAR system [7, 12], but have not yet investigated their application or extension into the functional/entity paradigm selected here.

- ***Speech Compatibility*** -- A long-term objective is to integrate speech recognition and generation with on-line real-time machine translation. A parallel project at CMU is integrating speaker-independent continuous-speech recognition with a case-frame semantic parser of English [15]. We expect results of that investigation, which is already moving towards the precompilation parsers discussed here, to pave the way towards eventual translation of spoken language.

We expect that these and other developments will require a continued focused research effort over the coming years.[4] We claim only to have taken one more stride in the long march towards the theoretical and practical development of fully-automated knowledge-based machine translation.

8. References

1. Aho, A. V. and Ullman, J. D., *The Theory of Parsing, Translation and Compiling,* Prentice-Hall, Englewood Cliffs, N. J., , Vol. II, 1972.

2. Birnbaum, L. and Selfridge, M., "Conceptual Analysis in Natural Language," in *Inside Computer Understanding,* R. Schank and C. Riesbeck, eds., New Jersey: Erlbaum Assoc., 1980, pp. 318-353.

3. Boggs, W. M. and Carbonell, J. G., Kee, M. and Monarch, I., "The DYPAR-I Tutorial and Reference Manual," Tech. report, Carnegie-Mellon University, Computer Science Department, 1985.

4. Bresnan, J. and Kaplan, R., *Lexical-Functional Grammar: A Formal System for Grammatical Representation,* MIT Press, Cambridge, Massachusetts, 1982, pp. 173-281.

5. Carbonell, J. G., Cullingford, R. E. and Gershman A. G., "Steps Towards Knowledge-Based Machine Translation," *IEEE Trans. PAMI,* Vol. PAMI-3, No. 4, July 1981.

6. Carbonell, J. G. and Hayes, P. J., "Dynamic Strategy Selection in Flexible Parsing," *Proceedings of the 19th Meeting of the Association for Computational Linguistics,* 1981.

7. Carbonell, J. G. and Hayes, P. J., "Recovery Strategies for Parsing Extragrammatical Language," *American Journal of Computational Linguistics,* Vol. 9, No. 3-4, 1983, pp. 123-146.

8. Carbonell, J. G., and Tomita, M., "Knowledge-Based Machine Translation, The CMU Approach," in *Theoretical Issues in Machine Translation,* Nirenberg, S., ed., Cambridge, U. Press, 1986.

9. Carbonell, J. G., "Discourse Pragmatics in Task-Oriented Natural Language Interfaces," *Proceedings of the 21st annual meeting of the Association for Computational Linguistics,* 1983.

10. Charniak, E. and Wilks, Y., *Computational Semantics,* Amsterdam: North Holland, 1976.

11. Earley, J., "An Efficient Context-free Parsing Algorithm," *Communication of ACM,* Vol. 6, No. 8, February 1970, pp. 94-102.

12. Fain, J., Carbonell, J. G., Hayes, P. J. and Minton, S. N., "MULTIPAR: a Robust Entity-Oriented Parser," *Proceedings of Seventh the Cognitive Science Society Conference,* Irvine, CA, 1985, pp. 110-119.

13. Gazdar, G., *Phrase Structure Grammar,* D. Reidel, 1982, pp. 131-186.

14. Hayes, P. J. and Carbonell, J. G., "A Natural Language Processing Tutorial," Tech. report, Carnegie-Mellon University, Computer Science Department, 1983.

15. Hayes, P. J., Hauptmann, A., Carbonell, J. G., and Tomita, M., "Parsing Spoken Language: a Semantic Caseframe Approach," *Proceedings of COLING-86,* 1986.

---

[4] Since the development of this new-generation technology for knowledge-based machine translation promises to be a major new direction to the field, but requires substantial resources to grow from theoretical conception to large-scale application, we are starting the ***International Center for Machine Translation*** at CMU. The center is dedicated to research and development of new techniques for machine translation and their engineering into substantial demonstration systems. Its first major project is the investigation of the functional grammar/entity-oriented approach to KBMT, as presented in this paper. Persons interested in acquiring more information about the center may do so by contacting one of the authors.

16. Hayes, P. J., "Entity-Oriented Parsing," *10th International Conference on Computational Linguistics,* Stanford, July 1984, pp. 212-217.

17. Hendrix, G. G., Sacerdoti, E. D. and Slocum, J., "Developing a Natural Language Interface to Complex Data," Tech. report Artificial Intelligence Center., SRI International, 1976.

18. Kay, M., "Functional Grammar," *Fifth Annual Meeting of the Berkeley Linguistic Society,* MIT Press, Berkeley, California, February 1979, pp. pp. 142-158.

19. Kay, M., "Functional Unification Grammar: A Formalism for Machine Translation," *10th International Conference on Computational Linguistics,* Stanford, July 1984, pp. 75-78.

20. Kittredge, R. and Lehrberger, J., *Sublanguages: Studies of Language in Restricted Semantic Domains,* deGruyter, Berlin, 1981.

21. Lytinen, S., *The Organization of Knowledge in a Multi-lingual, Integrated Parser,* PhD dissertation, Yale University, November 1984.

22. Pereira, F. C. N., "A Structure-Sharing Representation for Unification-Based Grammar Formalisms," *23rd Annual Meeting of the Association for Computational Linguistics,* Chicago, July 1985, pp. 137-144.

23. Riesbeck, C., "Conceptual Analysis," in *Conceptual Information Processing,* R. C. Schank, ed., Amsterdam: North-Holland, 1975, pp. 83-156, ch. 4.

24. Shieber, S. M., "Using Restriction to Extend Parsing Algorithms for Complex-Feature-Based Formalisms," *23rd Annual Meeting of the Association for Computational Linguistics,* Chicago, July 1985, pp. 145-152.

25. Tomita, M., *Efficient Parsing for Natural Language: A Fast algorithm for Practical Systems,* Kluwer Academic Publishers, Boston, MA, 1985.

26. Tomita, M., "An Efficient Context-free Parsing Algorithm for Natural Lnaguages," *9th International Joint Conference on Artificial Intelligence (IJCAI85),* August 1985.

27. Tomita, M., *An Efficient Context-free Parsing Algorithm for Natural Languages and Its Applications,* PhD dissertation, Computer Science Department, Carnegie-Mellon University, May 1985.

28. Woods, W. A., "Transition Network Grammars for Natural Language Analysis," *CACM,* Vol. 13, 1970, pp. pp.591-606.