


**Direct Memory Access Speech-to-Speech
Translation**

A Theory of Simultaneous Interpretation

**Hideto Tomabechi, Hiroaki Kitano, Teruko Mitamura,
Lori Levin and Masaru Tomita**

May 1, 1989
CMU-CMT-89-111

A stylized graphic of the American flag, featuring horizontal stripes in a dark red color against a lighter red background. The stripes are slightly wavy and extend from the left edge of the cover towards the center.

**Carnegie
Mellon**

Direct Memory Access Speech-to-Speech Translation

A Theory of Simultaneous Interpretation

**Hideto Tomabechi, Hiroaki Kitano, Teruko Mitamura,
Lori Levin and Masaru Tomita**

May 1, 1989
CMU-CMT-89-111

*A short version of this paper is in the Proceedings of
the International Conference on Fifth Generation Computer Systems 1988
(FGCS-88), Tokyo, 1988. Parts of this paper have also appeared in the
Proceedings of the Fourth Conference of the European Chapter of the
Association for Computational Linguistics (European-ACL'89) and the
Proceedings of the Second European Workshop on Natural Language
Generation (European-NLG89).*

Copyright ©1989 by Hideto Tomabechi

Abstract

This paper describes the Phoneme-Based Direct Memory Access Translation System (Φ DMTRANS) which is a speech-to-speech translation system developed at the Center for Machine Translation at CMU. Φ DMTRANS utilizes a phonological and episodic/thematic memory network, and performs spreading-activation guided-marker-passing which is massively parallel in nature. Φ DMTRANS handles the problem of multiple hypothesizations of input phonetic streams through network memory-based encoding of knowledge for language-specific phonology and morphophonetics, as well as episodic/thematic memory that supplies contextual disambiguations of the input. Also, ambiguity resolution is performed using cost analysis and contextual priming through thematic marker propagations. Through dedicated marker passing scheme for generation markers, a concurrent generation is supported so that the system produces parts of output speech even before the end of the input sentence, thus modeling simultaneous interpretations. This model is ideal for massively parallel hardware architecture.

The following pages were formatted in \LaTeX , and reproduced from a camera-ready copy supplied by the authors. In order to obtain natural page breaks, footnotes are sometimes moved to the following page.

Acknowledgments

The authors would like to thank members of the Center for Machine Translation for fruitful discussions. We would also like to thank Dr. Morii of Matsushita Research Institute for his contribution of the speech recognition hardware used by our project. Eric Nyberg and Margalit Zabudowski were especially helpful in preparing the final version of this paper.

1 Introduction

Recently a few efforts have been made in the area of processing speech input to a natural language understanding system. These include the work of Hayes, *et al*[1986], Tomita[1986], Poesio&Rullent[1987], Saito&Tomita,[1988], Tomabechi&Tomita[1988a], and Hauptmann, *et al*[1988]. Among them, Tomabechi&Tomita and Hauptmann, *et al* use contextual information for disambiguation of speech inputs and therefore, since extra-sentential information is important in the speech input system, Φ DMTRANS shares this feature of the two systems. The uniqueness of Φ DMTRANS, however, is that:

- it uses a parallel spreading activation network from the phonetical level,
- morphophonetic and phonological knowledge is dynamically utilized during memory activity,
- the morphophonemic, episodic/thematic and pragmatic levels of processing are fully integrated.

Φ DMTRANS uses parallel processing, and our experiments with the prototype Φ DMTRANS at Center for Machine Translation at the Carnegie Mellon University show that Φ DMTRANS is a promising framework for translating speech input cross-linguistically in new generation parallel computers.

2 Some Background and History:

2.1 Recognize-and-Record

Φ DMTRANS is a "Phoneme-Based Direct Memory Access Translation" architecture which represents what we call the "recognize-and-record" paradigm of natural language processing usually grouped as DMA (Direct Memory Access) models. In this model, natural language understanding is viewed as a memory activity which identifies input with what is already known in memory as episodic (experiential) and thematic knowledge. This is contrasted with the traditional model of parsing, which we call the "build-and-store" paradigm, in which a syntactic parser (with the help of semantics) builds up a tree-style representation of an input sentence, and processing is done sentence by sentence with little (if any) interaction between parses. In other words, the DMA paradigm models the human mind in the sense that past linguistic and non-linguistic experiences are being remembered during the course of understanding the input, and each sentence recognized records a context that influences the processing of successive

inputs. On the other hand, in traditional (non-DMA) systems, each input sentence is parsed into syntactic trees, and semantics are used primarily as a tool for guaranteeing the right configuration of syntactic trees; normally, no long-term memory (such as experiential memory) is involved during the parse. Also, in these systems, the result of a parse is lost after the processing of each sentence.

2.2 A Brief History

The Direct Memory Access method of parsing originated in Quillian's [1968] notion of semantic memory, used in his TLC (Quillian [1969]), which led to further research in semantic network-based processing¹. TLC used breadth-first spreading marker-passing as an intersection search of two lexically pointed nodes in a semantic memory, leaving interpretation of text as an intersection of the paths. Thus, interpretation of input text was directly performed on semantic memory. DMA was not explored as a paradigm for parsing (except as a scheme for disambiguation) until mid 1980's when DMAPO (Riesbeck & Martin [1985]) followed by Tomabechi [1987a,b] developed the DMA paradigm into theories of parsing and translation respectively. These projects were part of the Yale AI Project and were aimed at building a DMA natural language system to be integrated with case-based reasoning systems developed under the XP (eXplanation Patterns) theory of Schank [1986]. Since DMA parsers work directly on memory through spreading activation, integration of natural language understanding with the experiential memory of the case-based system became possible. These DMA systems used a guided marker-passing algorithm to avoid the problem of an explosion of search paths, from which a dumb² (not guided) marker passing mechanism inherently suffers. P-markers (Prediction markers) and A-markers (Activation markers) are markers passed around in memory, adopting the notion of concept sequence which guides marker passing along the known ordering of concepts. Currently the most active DMA researches are done at Carnegie Mellon University. Recently, the paradigm was adopted as a scheme for a natural language interface for development of knowledge-based systems (Tomabechi & Tomita [1988b]). Also in Tomabechi, *et al* [1988], a massively parallel network of phonological memory was connected to the DMA recognition and generation memory and the first DMA speech-to-speech translation was demonstrated at Carnegie Mellon University. In

¹This includes the work of Fahlman [1979], Hirst & Charniak [1982], Small & Reiger [1982], Charniak [1983], Haun & Reimer [1983], Hirst [1984], Charniak [1986], Charniak & Santos [1987], Norvig [1987], and recent connectionist and distributed models such as Granger & Eiselt [1984], Waltz & Pollack [1984], Berg [1987], Bookman [1987].

²We call it 'dumb' when markers are passed everywhere (through all links) from a node. In a 'guided' scheme, markers are passed through specific links only.

Tomabechi&Tomita[1988a], DMA based contextual inference was integrated into a unification-based (LFG, Bresnan[1982]) phoneme-parser (Saito&Tomita[1988]) as a part of real-time speaker-independent speech-to-speech translation system³ (Tomabechi, *et al*[1989]).

3 Problems in Speech Input

3.1 Phonetics, Phonology and Morphology

The difficulty of parsing speech input is that unlike written text input, a parser receives multiple hypotheses as input for a particular voice input. This is partly due to current limitations on speech recognition systems, which are incapable of determining specific phonemes for each input and generally produce several possible segmentations of the hypothesized phonetic stream. It is not rare that a speech parser outputs 30 to 50 well-formed, semantically acceptable parse results for each independent sentence of a speech recognition device output.

For example, when testing the CMU-CMT speech parser (a phoneme-based Generalized-LR parser (Φ GLR, Saito&Tomita[1988])), the Japanese input "atamagaitai" ("I have a headache") was spoken into a speech recognition system⁴ (under ordinary office environment) and accepted by the integrated⁵ parser with 57 ambiguous interpretations. Each of the ambiguous interpretations are semantically legitimate, meeting the local restrictions set forth by case-frame instantiation restrictions. Below are some of the highly scored interpretations:

atamagaitai (I have a headache.)
 kazokuwaitai ((The) families want to stay.)
 kazokuheitai ((My) family is soldier(s).)
 kazokudeitai (I want to stay as (a) family.)
 asabanaidou (Love (make love) (every) morning and night.)
 asakaraikou (Go (come) (from) tomorrow morning.)
 kazokuwaikou ((The) families go.)
 asamadeikou (Go before morning, Come until morning.)
 okosanaika (Shall we wake (one) up?)
 okosumaika (Shall we not wake (one) up?)
 kazokuheikou ((The) family is disappointed.)
 kazokudeikou (Go with the family.)
 gohunaidou (Love (make love) for five minutes.)

³Which is currently accepted as the first real-time speech-to-speech translation system.

⁴Matsushita Research Institute's speech recognition hardware. The speech recognition system and the speech input enhanced LR parser are described in detail in Saito&Tomita[1988].

⁵By 'integrated', we mean concurrent processing of syntax and semantics during parsing as opposed to some parsing methods where syntax and semantics are separately processed.

ugokumaika (Shall I not move?)
 atukunaika (Is it not hot?)
 dokoeikou (Where shall we go?)
 dokodeikou (Where shall we come?)
 koupumadeikou (go to (the) cup.)

These are just some of the 57 disambiguations that were produced as acceptable readings by the speech understanding system given the input "atamagaitai". One problem that is typified here by the Φ GLR speech parser, and commonly shared by most existing speech understanding systems, is that these systems do not sufficiently utilize morphophonetic and phonological knowledge during recognition and understanding. We will be discussing such knowledge in Section 4, but to be precise, it is the kind of knowledge that, for example, dictates what type of phonetic and phonological variations are possible for each type of phonetic features specific to Japanese. Humans apparently utilize such knowledge in processing a sequence of phones, and we would like to model such processing, since speech input is not a sequence of independently-determined phones but a connected string of successive phones.

3.2 Need for Contextual Knowledge

As we have seen in the preceding subsection, even with the semantic restrictions set forth by a syntax/semantics parser, we suffer from the problem of ambiguities that do not arise when the complete text is considered (i.e., 57 interpretations of "atamagaitai" in the preceding subsection were all acceptable syntactically and semantically only when not considering the context). This problem increases when the vocabulary of the speech understanding system enlarges and the variety of sentences that are accepted by the system expands. Although possible morphophonemic analyses of the speech input may be narrowed with the use of phonetic and phonological knowledge during speech understanding, we will still have large number of ambiguities for a specific phonetic stream.

In other words, local semantic restriction checks and phonetic/phonological narrowings are not sufficient for disambiguating continuous speech input, since an interpretation can be totally legitimate phonologically, syntactically, and semantically, but can mean something drastically different from what has been input into the speech recognition system (as well as being contextually inappropriate). The speech understanding system needs extra-sentential knowledge to choose an appropriate hypothesis for grouping phonetic segments and for selecting the appropriate word-sense of lexical entries. That is to say that the need for contextual knowledge in speech understanding systems is even more urgent than in text in-

put understanding systems; in a speech understanding system, the input can be interpreted in a way that is not possible in text input systems, and the input can still be acceptable to the local semantic restriction checks that integrated parsers perform within a sentence (such as slot-filler restriction checks of case-frame parsers).

4 Phonological Knowledge in Φ DMTRANS

Phonological knowledge is represented in Φ DMTRANS as weighted links connecting phonetic and phonemic nodes and functions stored in phonetic nodes capturing the physical and acoustic properties of sounds in a language (distinctive features) as well as environments that dynamically affect phonetic alterations. Phonological knowledge is used for providing the information to identify physical properties of articulated sounds instead of mental representations of each segment of words. Speakers have mental representation of sound systems, which are different from actual physical properties. Speakers of English feel /p/ in 'pin' and 'spin' are identical (and spelled the same in text inputs), but physically they are different sounds. /p/ in 'pin' is aspirated, represented in [ph], whereas /p/ in 'spin' is not aspirated represented in [p]. Both aspirated and unaspirated sounds do not differentiate the meaning in English, and they are predictable from a given environment. These units of phonetic segments are called phones. Thus, there are two levels of sound representation: a phonological level and a phonetic level.

Phonological rules convert phonological representations into phonetic ones. They can change, delete, or add segments. They can also coalesce or permute segments. In Japanese, high vowels become voiceless between voiceless consonants or after a voiceless consonant in the word final position.

$$\begin{array}{c}
 \text{V} \rightarrow [-\text{voice}] / \text{C} \\
 [+high] \quad \quad \quad / [-\text{voice}]
 \end{array}
 \quad \text{---} \quad
 \begin{array}{|c|}
 \hline
 \text{C} \\
 \hline
 [-\text{voice}] \\
 \hline
 \# \\
 \hline
 \end{array}$$

Phonological rules⁶ apply to classes of phonetically related segments. In order to capture the common features that certain phonological segments have, phonologists use distinctive features to represent them (Jakobson & Halle [1956]; Chomsky & Halle [1968]). For example, Japanese vowels are represented using the SPE system (Chomsky & Halle) in the following matrix.

⁶Phonological rules that dynamically affect processing due to phonological environments are captured via memory network representation (utilizing daemons in our system in 'FrameKit' (Nyberg [1988]) system) stored locally to each phones which was transformed from declarative description of rules originally supplied as phonological knowledge.

	i	e	a	o	u
high	+	-	-	-	+
back	-	-	+	+	+
low	-	-	+	-	-

The five vowels can be distinguished by using three kinds of features, and the matrix shows the phonemic relations. We can see the phonemic distance by counting the differences which are representable as weights:

	i	e	a	o	u
i	0	1	3	2	1
e		0	2	1	2
a			0	1	2
o				0	1
u					0

We can assume that lower distance numbers have higher confusion probabilities (i.e., higher weights). Therefore, when the input phone is [a], we can test from the segment which has a lower distance number, such as [a], then [o], and so on. With this matrix we can limit the test to close segments instead of testing all the segments⁷ and group close sounds in the network with certain thresholds. For consonants, distinctive feature matrix is more complex than our example of vowels and is provided in the Appendix 2 which is used as a base for encoding weights of the links.

The utilization of distinctive feature matrices described above; however, is a static knowledge that are encoded initially to the network (before parsing). We also need a scheme to dynamically assess the confusion of phones depending upon the phonetic environments that appear in the input speech. In Japanese, some speakers produce a glottal stop in a word initially before a vowel. In some speech recognition systems, the glottal stop may be interpreted as some voiceless stops, most likely /k/ because it is closer than others. The example of voiceless high vowel (specifically [u] and [i] in Japanese) between two voiceless consonants (or word final after voiceless consonant) is one case that we have seen in the phonological rule above. The method of capturing these types of phonological rules in our system is that we initially provide phonological environments and rules in a declarative form and the system precompiles the knowledge into functions stored in the phonetic nodes locally that are assessed every time the

⁷Since we use Matsusita Research Institute's Speech Recognition hardware, we adopt the phonemic system that the hardware recognizes. However, we have to note that some segments are not phonemes but are allophonic variants.

node is activated⁸ so that the phonemic activations are dynamically modified depending upon the phonetic environments on the speech input independent of the confusion matrices described above. This kind of phonological knowledge is thus encoded in the network for the dynamic phonetic activation changes, as well as the static confusion matrices that are pre-supplied and encoded as weighted links of the network along with the phonemic distances.

5 Contextual Knowledge in Φ DMTRANS

Φ DMTRANS uses an episodic/thematic memory network, similar to the ones described in Schank[1982] and Schank[1986], which is capable of dynamic modifications, inference and learning. Context in such a conceptual memory network can be represented as a grouping of concepts that are associated in a certain manner, i.e. an activation of one concept in memory triggers (or can potentially trigger) some other concepts in the memory network. To put it in another way, there is a relationship between concepts in which activation (recognition) of one concept reminds some other concept that it is related in a certain way. As we will see in detail in the following section, Φ DMTRANS uses the lexically-guided spreading activation mechanism for parsing. Context in this scheme is represented as what has been activated so far as 1) accepted concepts representing the previous sentences and 2) the concepts in the currently active concept sequences. These activations represent the recognition of what is being said so far and also represents what is likely to be heard under the current context. Readers may find our scheme of spreading activations similar to those researched by connectionists. However, we have not adopted connectionist associative architecture⁹ and

⁸The functions are stored as daemons in the nodes that are implemented via 'FrameKit' representations. For example, with the voiceless vowel between voiceless consonants example, the rule is originally supplied declaratively and then the declarative rule is precompiled as functions to be evaluated and stored locally in the phonetic node representing the voiceless vowel. At parsing time, when the voiceless vowel is hypothesized by the speech recognition hardware, i.e., receives the activation (A-Marker), then the functions stored in the node as the daemons are triggered and checks the environment (a lazy evaluation is used to attain the evaluation for both preceding and following nodes) and if the environment matches the precompiled knowledge for the voiceless vowel between voiceless consonants, then the voiced vowel phonetic nodes (i.e., [i] and [u] for Japanese) get activated and send activation to their phonemic nodes instead of activating the phonemic node for voiceless vowel.

⁹The connectionist associative model still lacks abilities to express complex relations between concepts and to perform variable binding (marker passing algorithm with structured markers can handle this) which are essential to handle linguistic phenomena such as metonymy as explained in Touretzky[1988].

back-propagation in our thematic conceptual clusters. Our spreading activations are guided and we do not spread everywhere.

6 Understanding in Φ DMTRANS

6.1 Phone Level Activity

Φ DMTRANS is the first DMA parser that works at the phonetic level. We will discuss the scheme of phonetic and phonological recognition in this subsection. First, Φ DMTRANS has as its nodes in the memory network nodes for phones and phonemes in each language. A phoneme may be realized as different phones in different phonetic environments. Several different phones may represent the same phoneme, for example phone [e] after dental and alveolar stops and affricates may represent phoneme /a/, in addition to phone [a] representing the phoneme /a/ in ordinary environments. In our memory network, each phone is connected to phonemes they represent via abstraction links. Also, each phoneme is connected by weighted phonological relation links to other phonemes. The weights of the links are determined by the strength of phonemic closeness based upon phonological distinctive feature thresholds as described in Section 4.

Above the phonemic nodes in the abstraction hierarchy are the lexical nodes, representing words. We have each lexical nodes in the memory network containing the phonemic sequence realizing the lexical entry in the given language. For example, in Japanese the lexical node "atama" (head) has the list <a t a m a> attached to it. So the structure linking phonetic node to lexical node is like this:

```

"atama" < lexical node
  <a t a m a> < phonemic sequence
              attached to "atama"
  /
  | -5--/u/ < phonological rel link with
  | /       distinctive feature weight
  | /
/a/ < phoneme node
  |
[a] < phone node

```

We have two types of markers (structured objects) passed around in memory. One is called P-Marker (for Prediction-Marker) and the other is called A-Marker (for Activation-Marker). P-Markers are passed along the phonemic sequences and A-Markers are passed above in the abstraction hierarchy (i.e., from phone to phoneme). Both markers contain information about which node originated the

marker passing. P-Markers also contain information about which was the immediately preceding node in the sequence. The algorithm for phonetic recognition is as follows. At the beginning of recognition, all the first elements of the phonemic sequences (such as /a/) are P-Marked by lexical nodes.

1. when the first input phone comes in (with this example, [a]) we put an A-Marker on (A-Mark) the phone node representing the phone (the node [a]).
2. when a node receives an A-Marker (i.e., if A-Marked) it sends an activation to (A-Marks) the node in its abstraction (i.e., phoneme /a/).
3. when an A-Marker and P-Marker meet, send a P-Marker to the next element of the sequence (i.e., since /a/ was P-Marked by the lexical node "atama", it sends a P-Marker in turn to /t/).
4. when the whole sequence is activated, then activate the root of the sequence (i.e., by repeating from 1. for [t], [a], [m], [a], the phonemic sequence <a t a m a> gets accepted and then we activate the lexical node "atama").

This is the basic cycle that is used in Φ DMTRANS. In the next subsection we discuss how the same algorithm is used for further processing at the sentential level, activating the episodic/thematic memory network. One thing we omitted in the above algorithm (for the sake of simplicity) is the way the phonological relation link is utilized in the activation of phones. Let us examine how this works:

When a certain phone (such as [t]) is activated, it not only activates its abstraction (such as the phoneme /t/) but also activates other phonemes that are related by the weighted links exceeding the given threshold. The weight of the phonological relation link is based upon distinctive feature study of each phone in the given language. For example, in Japanese the phoneme /t/ has the distinctive features 'alveolar' and 'stop' shared with the phoneme /d/, and link weight of 8 between them. So, if the threshold is given to be 5, when phone [t] is activated, both phonemes /t/ and /d/ are activated. This way, the phonological knowledge is encoded in the memory network as weighted links and is utilized during the spreading activation. Also, if the activated node contains the phonological rule application functions (i.e., stored as daemons, see footnote 7), and if the evaluation applies the rule and perform the dynamic alteration of the currently active phonetic node, then the phonemic nodes of the altered phone is activated capturing the phonetic changes in different environments which are not expressed in the static weighted links. Of course, because we have many lexical

entries that share similarity in attached phonemic sequences, and also because of activation of allophones (i.e., as we have seen both [a], and [e] may be under /a/), we have quite a significant number of simultaneously active phonemic sequences for a given stream of phones. This is where the strength of the parallel nature of our spreading activation mechanism is demonstrated. Since our memory network is a massively parallel network, the spreading activations for each concurrently active phonemic sequences will be parallelly performed.

6.2 Word Level and Sentential Level Activity

After a lexical node is activated through the acceptance of a whole phonemic sequence attached to a lexical node, we have similar spreading activations at the word level. We will not include the details of this processing in this paper because it is described in detail elsewhere (Tomabechi[1987b], Tomabechi-&Tomita[1988b], and Kitano, *et al*[1989]). One brief example would be the processing of the sentence "atamagaitai", which we saw before as a problematic input to other speech understanding systems. We use basically the same algorithm as we saw in the processing at phonetic level, except that each unit in the sequence is not a phoneme but a lexical node or a concept node and we call the sequence of such nodes concept sequences. An example of a concept sequence is <*BODY-LOCATION *PP[GA] *PAIN-SPEC> representing the sequence of concepts appear in "atamagaitai". The concept sequence can be regarded as a kind of subcategorization list (as in HPSG, Pollard&Sag[1987]) or as a generalized version of a phrasal lexicon (Becker[1975]) except that the sequence can be at higher levels in abstraction hierarchy as well as being episodic and thematic such as in MOPS and EXPLANATION PATTERNS (Schank[1982&1986]) encoding the knowledge for contextual processing.

We have nodes such as *HAVE-A-PAIN (representing the concept having a pain) and concept sequence such as <*BODY-LOCATION *PP[GA] *PAIN-SPEC> attached to the node (we call it root node if a concept sequence is attached to it). The elements of the sequence are the nodes in the memory network representing certain concepts¹⁰.

Φ DMTRANS uses three markers for parsing:

- An Activation Marker (A-Marker) is created when a concept is initially activated by a lexical item or as a result of *concept refinement*. It indicates

¹⁰*PP[GA] is a syntactic category representing the post-position "ga". This way, we can integrate syntactic knowledge as in subcategorization lists in syntactic theories as well. '*' preceding a concept name indicates that it is represented using our frame language 'FrameKit'.

which instance of a concept is the source of activation and contains relevant cost information. A-Markers are passed upward along is-a links in the abstraction hierarchy.

- A Prediction marker (P-Marker) is passed along a concept sequence to identify the linear order of concepts in the sequence. When an A-Marker reaches a node that has a P-Marker, the P-Marker is sent to the next element of the concept sequence, thus predicting which node is to be activated next.
- A Context marker (C-Marker) is placed on a node which has contextual priming.

Information about which instances originated activations is carried by A-Markers. The binding list of instances and their roles are held in P-Markers¹¹.

The following is the algorithm used in Φ DMTRANS parsing:

Let *Lex*, *Con*, *Elem*, and *Seq* be a set of lexical nodes, conceptual nodes, elements of concept sequences, and concept sequences, respectively.

Parse(S)

For each word *w* in *S*, do:
Activate(w),
 For all *i* and *j*:
 if *Active(N_i)* \wedge *N_i* \in *Con*
 then do concurrently:
 Activate(isa(N_i))
 if *Active(e_j.N_i)* \wedge *Predicted(e_j.N_i)* \wedge \neg *Last(e_j.N_i)*
 then *Predict(e_{j+1}.N_i)*
 if *Active(e_j.N_i)* \wedge *Predicted(e_j.N_i)* \wedge *Last(e_j.N_i)*
 then *Accept(N_i)*, *Activate(isa(N_i))*

Predict(N)

for all *N_i* \in *N* do:
 if *N_i* \in *Con*,
 then *Pmark(N_i)*, *Predict(isainv(N_i))*
 if *N_i* \in *Elem*,
 then *Pmark(N_i)*, *Predict(isainv(N_i))*
 if *N_i* \in *Seq*,

¹¹Marker parsing spreading activation is our choice over connectionist network precisely because of this reason. Variable binding (which cannot be easily handled in connectionist network) can be trivially attained through structure (information) passing of A-Markers and P-Markers.

then $Pmark(e_0.N_i), Predict(isainv(e_0.N_i))$
 if $N_i = NIL$,
 then *Stop*.

Activate

$I \leftarrow instanceof(c)$
 if $i = \phi$ then
 $createinst(c), Addcost, activate(c)$
 else
 for each $i \in I$
 do concurrently:
 $activate(c)$

Accept

if *Constraints* $\neq T$
 $Assume(Constraints), Addcost$
 $activate(isa(c))$

where N_i and $e_j.N_i$ denote a node in the memory network indexed by i and a j -th element of a node N_i , respectively.

$Active(N)$ is true iff a node or an element of a node gets an A-Marker.

$Activate(N)$ sends A-Markers to nodes and elements given in the argument.

$Predict(N)$ moves a P-Marker to the next element of the CSC.

$Predicted(N)$ is true iff a node or an element of a node gets a P-Marker.

$Pmark(N)$ puts a P-Marker on a node or an element given in the argument.

$Last(N)$ is true iff an element is the last element of the concept sequence.

$Accept(N)$ creates an instance under N with links which connect the instance to other instances.

$isa(N)$ returns a list of nodes and elements which are connected to the node in the argument by abstraction links.

$isainv(N)$ returns a list of nodes and elements which are daughters of a node N .

Some explanation would help understanding this algorithm:

1. Prediction.

Initially all the first elements of concept sequences (CSC - Concept Sequence Class) are predicted by putting P-Markers on them.

2. Lexical Access.

A lexical node is activated by the input word.

3. Concept Activation.

An A-Marker is created and sent to the corresponding CC (Concept Class) nodes. A cost is added to the A-Marker if the CC is not C-Marked (i.e. A C-Marker is not placed on it.).

4. Discourse Entity Identification

A CI (Concept Instance) under the CC is searched for.

If the CI exists, an A-Marker is propagated to higher CC nodes.

Else, a CI node is created under the CC, and an A-Marker is propagated to higher CC nodes.

5. Activation Propagation.

An A-Marker is propagated upward in the abstraction hierarchy.

6. Sequential Prediction.

When an A-Marker reaches any P-Marked node (i.e. part of CSC), the P-Marker on the node is sent to the next element of the concept sequence.

7. Contextual Priming

When an A-Marker reaches any Contextual Root node. C-Makers are put on the contextual children nodes designated by the root node.

8. Conceptual Relation Instantiation.

When the last element of a concept sequence receives an A-Marker, Constraints (world and discourse knowledge) are checked for.

A CSI is created under the CSC with packaging links to each CI. This process is called *concept refinement*.

The memory network is modified by performing inferences stored in the root CSC which had the accepted CSC attached to it.

9. Activation Propagation

A-Marker is propagated from the CSC to higher nodes.

*Concept refinement*¹² is an activity to locate the most specific node in memory, below the activated root node, which represents the specific instance of the input text. Such a node must have links to all the specializations (or instances) of the nodes that appeared in the concept sequence with relations that are equivalent to (or subclasses of) the relation links from the root node to the packaged nodes in the accepted concept sequence. The search for such a node underneath the root node is called *concept refinement*. This activity, which locates the concept that is identified with the specific input speech, is central to the understanding in the DMA parsing.

Processing of the example sentence "atamagaitai" is as follows: when the lexical node "atama" is activated after the acceptance of the phonemic sequence

¹²See also Lytinen[1984], Riesbeck&Martin[1985] and Tomabeck[1987a,b].

<a t a m a>, then we activate the corresponding CC (Concept Class) node *HEAD and spread the activation upward in the abstraction hierarchy. One of the abstractions is the concept *BODY-LOCATION. At the beginning of understanding, we have all first elements of the concept sequences (CSCs) P-Marked (just as we did so with first elements of phonemic sequences). So *BODY-LOCATION was P-Marked by the root node *HAVE-A-PAIN. Therefore, when *BODY-LOCATION is activated from below, we have a collision of A-Marker and P-Marker. When the collision happens, we send a P-Marker to the next element of the concept sequence (i.e., *PP[GA]). This is continued and the last element *PAIN-SPEC gets accepted after acceptance of <i t a i>. So we activate the root node *HAVE-A-PAIN. One thing that happens (that we did not have at phonetic level) is that we perform the 'concept refinement'¹³, which is essentially what understanding in DMA means. It involves identifying the specific instance of the accepted root concept that represents the input to the understanding system. In our case, the memory searches for the node *HAVE-A-HEADACHE (or creates it if non-existent yet), that is underneath *HAVE-A-PAIN and packages the nodes *HEAD, *PP[GA], *PAIN-SPEC[UNSPEC] that are specific to the current input. Since concept sequences are generic and attached to relatively higher nodes in abstraction hierarchy, it is this concept refinement that specifies (or identifies) the specific input to the system. After concept refinement, we now have the node *HAVE-A-HEADACHE activated, and that is the result of the understanding. Of course, in the actual system, the spreading activation continues in a parallel manner because the concepts *BODY-LOCATION and *HAVE-A-HEADACHE (and the concepts in between them) may be a part of some other higher level concept sequences in abstractions such as scriptal and episodic memory packets.

6.3 Memory Network Modification

Several different incidents trigger the modification of the memory network during parsing:

- An individual concept is instantiated (i.e. an instance is created) under a CC when the CC receives an A-Marker and a CI (an instance that was created by preceding utterances) is not existent. This instantiation is a creation of a specific discourse entity which may be used as an existent instance in the subsequent recognitions.
- A concept sequence instance is created under the accepted CSC. In other words, if a whole concept sequence is accepted, we create an instance of

¹³Lytinen[1984] and Tomabechi[1987b] have detailed discussions of 'concept refinement'.

the sequence instantiating it with the specific CIs that were created by (or identified with) the specific lexical inputs. This newly created instance is linked to the accepted CSC with a instance relation link and to the instances of the elements of the concept sequences by links labelled with their roles given in the CSC.

- Links are created or removed in the CSI creation phase as a result of invoking inferences based on the knowledge attached to CSCs. For example, when the parser accepts the sentence *I went to the UMIST*, an instance of *I* is created under the CC representing *I*. Next, a CSI is created under PTRANS. Since PTRANS entails that the agent is at the location, a location link must be created between the discourse entities *I* and *UMIST*. Such revision of the memory network is conducted by invoking knowledge attached to each CSC.

Since modification of any part of the memory network requires some *workload*, certain costs are added to analyses which require such modifications.

6.4 Ambiguity Resolution Based on Cost Analysis

Ambiguity resolution in Φ DMTRANS is based on the calculation of the cost of each parse. Costs are attached to each parse during the parse process.

Costs are attached when:

1. A CC with insufficient priming is activated,
2. A CI is created under CC, and
3. Constraints imposed on CSC are not satisfied initially and links are created or removed to satisfy the constraint.

Costs are attached to A-Markers when these operations are taken because these operations modify the memory network and, hence, workloads are required. Cost information is then carried upward by A-Markers. The parse with the least cost will be chosen.

The cost of each hypothesis are calculated by:

$$C_i = \sum_{j=0}^n c_{ij} + \sum_{k=0}^m \text{constraint}_{ik} + \text{bias}_i$$

where C_i is a cost of the i -th hypothesis, c_{ij} is a cost carried by an A-Marker activating the j -th element of the CSC for the i -th hypothesis, constraint_{ik} is a

cost of assuming k-th constraint of the i-th hypothesis, and $bias_i$ represents lexical preference of the CSC for the i-th hypothesis. This cost is assigned to each CSC and the value of C_i is passed up by A-Markers if higher-level processing is performed. At higher levels, each c_{ij} may be a result of the sum of costs at lower-levels.

It should be noted that this equation is very similar to the activation function of most neural networks except for the fact our equation is a simple linear equation which does not have threshold value. In fact, if we only assume the addition of cost by priming at the lexical-level, our mechanism of ambiguity resolution would behave much like connectionist models without inhibition among syntactic nodes and excitation links from syntax to lexicon¹⁴. However, the major difference between our approach and the connectionist approach is the addition of costs for instance creation and constraint satisfaction. We will show that these factors are especially important in resolving structural ambiguities.

The following subsections describe three mechanisms that play a role in ambiguity resolution. However, we do not claim that these are the only mechanisms involved in the examples which follow¹⁵.

6.5 Contextual Priming

In our system, some CC nodes designated as Contextual Root Nodes have a list of thematically relevant nodes. C-Markers are sent to these nodes as soon as a Contextual Root Node is activated. Thus each sentence and/or each word might influence the interpretation of following sentences or words. When a node with C-Marker is activated by receiving an A-Marker, the activation will be propagated with no cost. Thus, a parse using such nodes would have no cost. However, when a node without a C-Marker is activated, a small cost is attached to the interpretation using that node.

In Tomabechi[1987a] the discussion of C-Marker propagation concentrated on the resolution of word-level ambiguities. However, C-Markers are also propagated to conceptual class nodes, which can represent word-level, phrasal, or sentential knowledge. Therefore, C-Markers can be used for resolving phrasal-level and sentential-level ambiguities such as structural ambiguities. For example, *atama ga itai* literally means, '(my) head hurts.' This normally is identified with the concept sequences associated with the *have-a-symptom concept class

¹⁴We have not incorporated these factors primarily because structured P-Markers can play the role of top-down priming; however, we may be incorporating these factors in the future.

¹⁵For example, in one implementation of DMTRANS, we are using time-delayed decaying activations which resolve ambiguity even when two CI nodes are concurrently active.

node, but if the preceding sentence is *asita yakuinkai da* ('There is a board of directors meeting tomorrow'), the *have-a-problem concept class node must be activated instead. Contextual priming attained by C-Markers can also help resolve structural ambiguity in sentences like *did you read about the problem with the students?* The cost of each parse will be determined by whether *reading with students* or *problems with students* is contextually activated. (Of course, many other factors are involved in resolving this type of ambiguity.)

Our model can incorporate either C-Markers or a connectionist-type competitive activation and inhibition scheme for priming. In the current implementation, we use C-Markers for priming simply because C-Marker propagation is computationally less-expensive than connectionist-type competitive activation and inhibition schemes¹⁶. Although connectionist approaches can resolve certain types of lexical ambiguity, they are computationally expensive unless we have massively parallel computers. C-Markers are a reasonable compromise because they are sent to semantically relevant concept nodes to attain contextual priming without computationally expensive competitive activation and inhibition methods.

6.6 Reference to the Discourse Entity

When a lexical node activates any CC node, a CI node under the CC node is searched for (Tomabechi[1987a], Tomabechi&Tomita[1988b]). This activity models reference to an already established discourse entity Webber[1983] in the hearer's mind. If such a CI node exists, the reference succeeds and this parse will be attached with no cost. However, if no such instance is found, reference failure results. If this happens, an instantiation activity is performed creating a new instance with certain costs. As a result, a parse using newly created instance node will be attached with some cost.

For example, if a preceding discourse contained a reference to a thesis, a CI node such as THESIS005 would have been created. Now if a new input sentence contains the word *paper*, CC nodes for *THESIS and *SHEET-OF-PAPER are activated. This causes a search for CI nodes under both CC nodes. Since the CI node THESIS005 will be found, the reading where *paper* means thesis will not acquire a cost. However, assuming that there is not a CI node corresponding to a sheet of paper, we will need to create a new one for this reading, thus incurring a cost.

¹⁶This does not mean that our model can not incorporate a connectionist model. The choice of C-Markers over the connectionist approach is mostly due to computational cost. As we will describe later, our model is capable of incorporating a connectionist approach.

We can also use reference to discourse entities to resolve structural ambiguities. In the sentence *We sent her papers*, if the preceding discourse mentioned Yoshiko's papers, a specific CI node such as YOSHIKO-PAPER003 representing Yoshiko's papers would have been created. Therefore, during the processing of *We sent her papers*, the reading which means we sent papers to her needs to create a CI node representing papers that we sent, incurring some cost for creating that instance node. On the other hand, the reading which means we sent Yoshiko's papers does not need to create an instance (because it was already created) so it is costless. Also, the reading that uses *paper* as a sheet of paper is costly as we have demonstrated above.

6.7 Constraints

Constraints are attached to each CSC. These constraints play important roles during disambiguation. Constraints define relations between instances when sentences or sentence fragments are accepted. When a constraint is satisfied, the parse is regarded as plausible. On the other hand, the parse is less plausible when the constraint is unsatisfied. Whereas traditional parsers simply reject a parse which does not satisfy a given constraint, following the scheme adopted in DMTRANS PLUS (Kitano, *et al*[1989]), Φ DMTRANS builds or removes links between nodes forcing them to satisfy constraints. A parse with such forced constraints will record an increased cost and will be less preferred than parses without attached costs.

The following example illustrates how this scheme resolves an ambiguity. As an initial setting we assume that the memory network has instances of 'man' (MAN1) and 'hand-gun' (HAND-GUN1) connected with a POSSES relation (i.e. link). The input utterance is: "Mary picked up an Uzzi. Mary shot the man with the hand-gun." The second sentence is ambiguous in isolation and it is also ambiguous if it is not known that an Uzzi is a machine gun. However, when it is preceded by the first sentence and if the hearer knows that Uzzi is a machine gun, the ambiguity is drastically reduced. Φ DMTRANS hypothesizes and models this disambiguation activity utilizing knowledge about world through the cost recording mechanism described above.

During the processing of the first sentence, Φ DMTRANS creates instances of 'Mary' and 'Uzzi' and records them as active instances in memory (i.e., MARY1 and UZZI1 are created). In addition, a link between MARY1 and UZZI1 is created with the POSSES relation label. This link creation is invoked by triggering side-effects (i.e., inferences) stored in the CSC representing the action of 'MARY1 picking up the UZZI1'. We omit the details of marker passing (for A-, P-, and C-

Markers) since it is described detail elsewhere (particularly in Tomabechi[1987b]).

When the second sentence comes in, an instance MARY1 already exists and, therefore, no cost is charged for parsing 'Mary'¹⁷. However, we now have three relevant concept sequences (CSC's¹⁸):

CSC1: <*AGENT *SHOOT *OBJECT>

CSC2: <*AGENT *SHOOT *OBJECT *WITH *INSTRUMENT>

CSC3: <*PERSON *WITH *INSTRUMENT>

These sequences are activated when concepts in the sequences are activated in order from below in the abstraction hierarchy. When the "man" comes in, recognition of CSC3:(*PERSON *WITH *INSTRUMENT) starts. When the whole sentence is received, we have two top-level CSCs (i.e., CSC1 and CSC2) accepted (all elements of the sequences recognized). The acceptance of CSC1 is performed through first accepting CSC3 and then substituting CSC3 for *OBJECT.

When the concept sequences are satisfied, their constraints are tested. A constraint for CSC2 is (POSSES *AGENT *INSTRUMENT) and a constraint for CSC3 (and CSC1, which uses CSC3) is (POSSES *PERSON *INSTRUMENT). Since 'MARY1 POSSESS HAND-GUN1' now has to be satisfied and there is no instance of this in memory, we must create a POSSESS link between MARY1 and HAND-GUN1. A certain cost, say 10, is associated with the creation of this link. On the other hand, MAN1 POSSESS HAND-GUN1 is known in memory because of an earlier sentence. As a result, CSC3 is instantiated with no cost and an A-Marker from CSC3 is propagated upward to CSC1 with no cost. Thus, the cost of instantiating CSC1 is 0 and the cost of instantiating CSC2 is 10. This way, the interpretation with CSC1 is favored by our system.

7 Generation in Φ DMTRANS

Φ DMTRANS has two types of generation scheme: the explanation based generation and the marker propagation based generation (Kitano[1989], Kitano&-Tomabechi[ms a]). Since the explanation based generation has been described elsewhere (particularly in Tomabechi[1987a,b]), we will be concentrating on the description of our marker propagation based generations.

¹⁷Of course, 'Mary' can be 'She'. The method for handling this type of pronoun reference was already reported in Tomabechi[1987a,b] and we do not discuss it here.

¹⁸As we can see from this example of CSC's, a concept sequence can be normally regarded as a subcategorization list of a VP head. However, concept sequences are not restricted to such lists and are actually often at higher levels of abstraction representing MOP-like sequences.

Two markers that are specific to generation are used. These are: (1) Generation Markers (G-Markers), which each contain a surface string and an instance which the surface string represents and (2) Verbalization Markers (V-Markers), which anticipate and keep track of verbalization of surface strings. G-Markers are created at the lexical-level and each contain a surface string and an instance to which the surface string refers. G-Markers are passed up through the memory network. At a certain point of processing, surface strings of G-Markers are concatenated following the order of CSC and a final string of the utterance is created. When an incremental sentence production is performed, V-Markers record part of sentences which are already verbalized and anticipate next possible verbalization strings.

The algorithm is shown below:

Let *Lex*, *Con*, *Elem*, and *Seq* be a set of lexical nodes, conceptual nodes, elements of concept sequences, and concept sequences, respectively.

Translate(S)

For each phoneme *p* in *S*, do concurrently:

Activate(p),

 For all *i* and *j*:

 if *Active(N_i)* \wedge *N_i* \in *Lex*

 for each *n_m* in *trans(N_i)*

 then

CreateGMarker(n_m),

GActivate(isa(n_m))

 if *Active(N_i)* \wedge *N_i* \in *Con*

 then do concurrently:

Activate(isa(N_i)),

MakeTrans(N_i)

 if *Active(e_j.N_i)* \wedge *Predicted(e_j.N_i)* \wedge \neg *Last(e_j.N_i)*

 then *Predict(e_{j+1}.N_i)*

 if *Active(e_j.N_i)* \wedge *Predicted(e_j.N_i)* \wedge *Last(e_j.N_i)*

 then *Accept(N_i)*, *Activate(isa(N_i))*

MakeTrans(N_i)

 For each *n_m* in *TL(N_i)*

 do concurrently:

 if *N_i* \in *Lex*

CreateGMarker(n_m), *GActivate(isa(n_m))*

 if *N_i* \in *CSC*

InstantiateUtterance(n_m), *GActivate*(*isa*(n_m))

```

GActivate( $e_j.N_i$ )
  GMark( $e_j.N_i$ )
  if VMarked( $e_j.N_i$ )  $\wedge$  Unambiguous( $e_j.N_i$ )
    then
      Verbalize( $e_j.N_i$ )
      if  $\neg$ Last( $e_j.N_i$ )
        VMark( $e_{j+1}.N_i$ )
      if Last( $e_j.N_i$ )
        GActivate(isa( $N_i$ ))
    else
      GActivate(isa( $e_j.N_i$ ))

```

where N_i and $e_j.N_i$ denote a node in the memory network indexed by i and a j -th element of a node N_i , respectively. *Active*(N) is true iff a node or an element of a node gets an A-Marker. *Activate*(N) sends A-Markers to nodes and elements given in the argument. *Unambiguous*(N) tests if the role of the node is unambiguous. *Verbalize*(N) produces phonological realization of the concept and it is send to the speech synthesizer. *CreateGMarker*(N) creates a G-Marker containing a surface string and an instance referred to. *TL*(N) returns a list of nodes connected under N by target language links. *InstantiateUtterance*(N) concatenates the surface string of the G-Markers corresponding to each element of the CSC and stores the result in a new G-Marker passed up to higher abstraction links. *Predict*(N) moves a P-Marker to the next element of the CSC. *Predicted*(N) is true iff a node or an element of a node gets a P-Marker. *Pmark*(N) puts a P-Marker on a node or an element given in the argument. *Last*(N) is true iff an element is the last element of the concept sequence. *Accept*(N) creates an instance under N with links which connect the instance to other instances. *isa*(N) and *trans*(N) return a list of nodes and elements which are connected to the node in the argument by abstraction links and translation links, respectively. *isainv*(N) returns a list of nodes and elements which

are daughters of a node N.

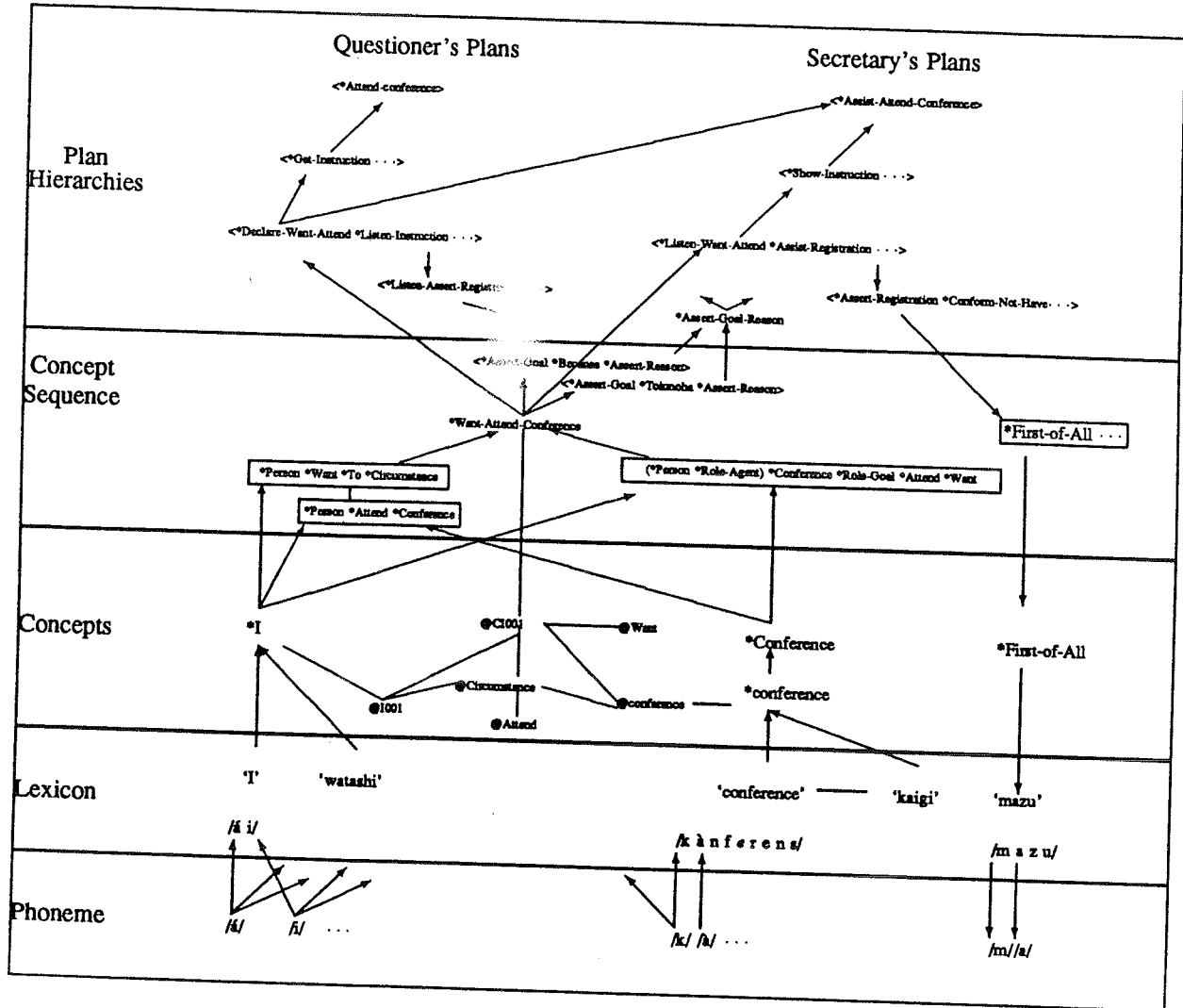


Figure 1: A Process of Parsing, Generation and Prediction

Let us now explain with a simple example (Figure-1). In the following explanation, we will assume consecutive interpretation mode which translations are made after a whole sentence is parsed. Simultaneous interpretation mode is described in the next section. Suppose the input is *I want to attend the conference*; parsing

is performed by activating each phoneme node and passing up activation through the memory network. Multiple hypothesis for a possible word would be activated. Each lexical node corresponds to each hypothesis is activated and passing up activation through the memory network. The generation process runs concurrently. When the phoneme sequence *a I* is entered, corresponding phoneme nodes and lexical nodes which encode phonological realization of the lexical entry gets activated. If the lexical node has translation links to any Japanese lexical nodes, these Japanese lexical nodes will be activated. Since there is no translation link in the lexical node, activation is simply passed up and conceptual nodes, *I and *Person (superclass of *I), gets activated as a part of the parsing process. At this time, the program searches for a Japanese lexical entry for the concept *I* and finds *Watashi*. A G-Marker is created and includes an instance (@I001) and a surface string (*watashi*). The G-Marker is passed up through the memory network. This process takes place for each word in the sentence. P-Markers are initially located on the first element of CSCs for the source language (In this example, a P-Marker is at *Person of <*Person *Want *To *Circumstance> and <*Person *Attend *Conference>). V-Markers are located in the first element of CSCs for the target language. V-Markers are important in the simultaneous interpretation mode which will be described in the next section. When *Person gets an A-Marker, the P-Marker is moved to *Want. When *Want gets an A-Marker then the P-Marker is moved to *To and then to *Circumstances. The word *attend, the, conference* would activate each element of the CSC (<*Person *Attend *Conference>). Thus, the CSC is accepted and further an A-Marker propagation will activate *Circumstance. Instantiation takes place when this sequence is accepted, i.e. the P-Marker is placed on the last element of the sequence and that element gets an A-Marker. As a result, a CI is created under the CC (*Want-Attend-Conference) which is a root node of the accepted CSC, and linked with relevant instances. Since a CSC for a Japanese expression, <(*Person *Role-Agent) *Conference *Role-Goal *Attend *Want>, is stored under *Want-Attend-Conference, a surface string for an utterance can be created by concatenating surface strings stored in each G-Marker on each conceptual node (linearization). For instance, G-Markers on *Conference and *Attend contain surface strings *kaigi* and *sanka*, respectively. A result of the concatenation is *Kaigi ni sanko shitai*.¹⁹ When we only require a single sentence to be generated, we can use this sentence. This surface string is, again, stored in a G-Marker and passed up to discourse-level CSCs in order to generate multiple sentences. Activation is further passed up the discourse knowledge layer and *Declare-Want-Attend and *Listen-Want-Attend

¹⁹Notice that optional elements, (*Person *Role-agent), are not concatenated to produce subject-ellipsis sentence.

are activated. As a result of this activation, the next possible utterances *Assert-Registration, *First-of-All, and $\langle /m a z u / \rangle$ are predicted (shown as downward arrows).

7.1 Concurrent Generation while Parsing

It is important to note that a massively parallel approach allows the generation process to be executed concurrent to parsing. Of course, formulation of each part of the sentence takes place after it is processed and its meaning is determined. However, it is concurrent in the sense that the generation process does not wait until the entire parse is completed so that the translated utterance is generated incrementally²⁰. Lexical selection and partial production of utterances are conducted while parsing is in progress. Thus, in some input, a part of the utterance can be generated before parsing of the entire sentence is completed. We do this by verbalizing a surface string or phonological realization of the instance whose role is determined, i.e. not ambiguous, and wait verbalization of ambiguous instances until they get disambiguated. The part of the sentence which is verbalized is recorded in a V-Marker and the V-Marker is moved to the next possible verbalization elements. This will avoid redundant verbalization. Only the element with a V-Marker can be verbalized in order to ensure consistency of the produced sentence.

Figure-2 indicate a temporal relationship between a series of words given to the system and incremental generation in the target language. An incremental translation and generation of the input, *I want to attend the conference because I am interested in Interpreting Telephony*, results in two connected Japanese sentences; *(Watashi wa) kaigi ni sanku shitai. Toiunoha (watashi ha) tuuyaku denwa ni kyoumi ga arukara desu..* During the analysis of utterances, there is a point for which the semantics of a part of the sentence is determined. For instance, *I (@I001)* can only be a agent of the action when *want* is analyzed. At this time, *watashi ha* (I Role-Agent) can be verbalized. A V-Marker which is initially located on *Person is now moved to *Conference, which is the next element of the CSC. The next verbalization will not be triggered until *because* comes in because the role of *I want to attend the conference* in the discourse is still ambiguous. At this moment, *Want-Attend-Conference gets activated and its superclass node including *Assert-Goal get activated. A P-Marker is now placed on *Because in a CSC ($\langle *Assert-Goal *Because *Assert-Reason \rangle$)

²⁰Unlike an incremental generation of Kempen&Hoenkamp[1987] which assigns procedure to each syntactic category, our algorithm uses markers to carry information. Also, concepts to be expressed are incrementally determined as parsing progress.

Input Utterance	Translation
I want to attend the conference	watashi ha (I Role-Agent; This is ellipsed in the actual translation)
because	kaigi ni sanku shitai (want to attend the conference)
I am interested in	toiunoha (because) watashi ha (I Role-Agent; This is ellipsed in the actual translation)
Interpreting Telephony	tuuyaku denwa ni kyōumi ga arukara desu (interested in Interpreting Telephony)

Figure 2: Simultaneous Interpretation in Φ DMTRANS

and a V-Marker is still placed on *Assert-Goal of the corresponding Japanese CSC (<*Assert-Goal *Touiuunoha *Assert-Reason>). Activation of *Because will determine the role of *Want-Attend-Conference. The word *because* acts as a clue-word which divides assertion of the speaker's goal and her/his reasons for it as represented in the CSC. in the discourse. Verbalization is now triggered; i.e. *I want to attend the conference* is vocalized, and the V-Marker is moved the next element of the CSC. When a whole sentence is parsed, the entire meaning is made clear and the rest of the sentence is verbalized. This example illustrate generation in the target language at the earliest possible point. Although this is perfectly legitimate Japanese, this is not the best Japanese translation as far as styles are concerned. In order to overcome this problem, we made an option to generate translation after waiting until a whole clause or a sentence is parsed so that it provides options to generate the best style under the given discourse situation. One option is to reverse the order of clauses, i.e. *Assert-Reason then *Assert-Goal, which realizes perfectly fluent Japanese. One other example is when a topicalization of the first clause is demanded, translation will be (*Watashi ga*) *kaigi ni sanku shitai noha (watashi ga) tuuyaku denwa ni kyōumi ga arukara desu* (The reason why I want to attend the conference is that I am interested in interpreting telephony). Such constraints on generation scheduling are provided mostly from discourse knowledge and temporal constraints. Our model is basically a language

independent so that it is applicable to Japanese to English translation. For a sentence such as *Tuuyaku denwa ni kyoumi ga arimasu node kaigi ni sanku shitai nodesuga*, ((I am) interested in interpreting telephony, so (I) want to attend the conference), our algorithm can generate *I am interested in interpreting telephony* before *kaigi ni sanku shitai nodesuga* (I want to attend the conference) is parsed.

7.2 Lexical Selections and Linearization Pattern Selections

Both generation lexicons and linearization patterns are selected by using either C-Markers or the connectionist network.

The C-Marker passing scheme puts C-Markers to contextually relevant nodes when a conceptual root node is activated. A G-Marker which goes through a node without C-Marker will be added with larger cost than others. A pattern of linearization is selected among CSCs linked with CC nodes representing the input utterance such as *Want-Attend-Conference. The CC node is determined through the cost-based ambiguity resolution scheme during parsing (Kitano, *et al*[1989]). When there are multiple hypothesis for the specific CC node; i.e. when multiple CSCs are linked with the CC, we adopt an idea of the cost-based disambiguation to select one of them. We will add up the cost of each G-Marker used for each linearization combined with pragmatic constraints which may be assigned to each CSC and the preference for each CSC. The hypothesis with least cost will be selected as a translated result.

The Connectionist Network will be adopted with some computational costs. When a connectionist network is fully deployed, every node in the network is connected with weighted links. A competitive excitation and inhibition process is performed to select one hypothesis (Waltz&Pollack[1985]). Our cost-based model can be used to determine initial activation strength of the each node. When a lexical node is activated, associated CCs are activated according to the weights of connections. Activation of CCs will propagate to lexical nodes of the target language. A-Markers are created when CCs are activated and G-Markers are created when any lexical node of the target language is activated. Although both A-Markers and G-Markers carry cost information, its actual value changes over time according to the change in the activation level of the lexical and conceptual nodes. Activation levels of each lexical node can be decreased when (1) stronger inhibitions are given, (2) excitation from the CSC is decreased due to the cost attached to the semantic constraint for the hypothesis, and (3) excitation level of the corresponding CC node is decreased. Similarly, activation levels of CSCs and CCs change over time affecting and affected by excitation and inhibition of other nodes. Final interpretation and translation in the target language are selected

through a winner-take-all mechanism.

8 Future Possibilities

8.1 Incorporation of Connectionist Model

Our model can incorporate connectionist processing for morphophonemic activations and ambiguity resolutions. In a connectionist network activation of one node triggers interactive excitation and inhibition among nodes. Nodes which get more activated will be primed more than others. When a parse uses these more active nodes, no cost will be added to the hypothesis. On the other hand, hypotheses using less activated nodes should be assigned higher costs. There is nothing to prevent our model from integrating this idea. However, currently the computational cost for such a network is rather prohibitive for real-time implementations. Readers should also be aware that DMA is a guided marker passing algorithm in which markers are passed only along certain links whereas connectionist models allow spreading of activation and inhibition virtually to any connected nodes. We hope to integrate the DMA and the connectionist models on a massively parallel hardware to demonstrate a real-time translation. We are currently planning to integrate the Φ DMTRANS with the CMU/ATR neural-net speech recognition system (Waibel, *et al*[1989]) where our phonemic activations are triggered from the lower layers of acoustic networks. The Φ DMTRANS will be receiving a bundle of micro-acoustic features directly from the hidden layers of the speech-recognition neural network instead of simply receiving a stream of hypothesized phonemes. The network will have a uniform architecture from the lowest acoustic signal processing to the higher discourse level knowledge. The predictions from the various levels (discourse, domain specific, syntactic, phonological, etc.) are propagated down to the lowest layers of the phoneme recognitions for higher accuracy in recognitions and understanding.

8.2 Syntactic Constraint Propagations

Currently the sole syntactic knowledge in the Φ DMTRANS network is the linear sequence of concepts represented as CSCs. In order to handle a tighter syntactic processing two schemes have been tried. The first scheme is to utilize an external syntactic recognizer to handle the syntactic knowledge as reported in Tomabechi-&Tomita[1988a]. This scheme has an advantage of utilizing our existing grammar (in LFG formalisms) that has been developed for our large-scale MT projects. One problem with this scheme, however, is that the unification operation used

for the syntactic recognition is sequential in nature and there will be little gain from massive parallelism (Knight[1988]). Since we are assuming a massively parallel architecture as a basis of the Φ DMTRANS activity, this nature of unification operation is discouraging.

The second method we have been experimenting with is what we call Massively-Parallel Structured-Marker Passing (MSP) algorithm under our Massively-Parallel Constraint Propagation (MCP) scheme (Tomabechi&Tomita[ms]) and Head-driven Massively-parallel Constraint Propagation (HMCP) scheme (Tomabechi&Levin[ms]). Under these schemes, the markers that are propagated contain specific constraints to be satisfied at different levels of abstractions with different types of marker collisions. Currently we are experimenting with incorporating HPSG (Pollard&Sag[1987]) based syntactic constraint propagations in order to handle syntactic constraints such as case and agreement. The advantage of this scheme is that we will be able to handle a large proportion of syntactic knowledge formulated under the unification-based grammar without the use of unifications replaced by a MSP algorithm which is inherently massively parallel in nature.

8.3 Utterance-Case Acquisitions

We call specific patterns of utterances recorded in memory 'utterance-cases' (Kitano&Tomabechi[ms b]). It is different from *phrasal lexicon* in the sense that an 'utterance-case' is a memory structure that is integrated in the network providing specific constraints for certain combination of concepts and surface strings. A large depository of such 'utterance-cases' will constrain the configurational possibilities of specific concepts and morphological realizations in a contextually sensitive manner. An 'utterance-case' will work to provide a restrictions during a sentence understanding and generation where syntactic, semantic and even pragmatic knowledge of discourse may not provide enough constraints.

Such a constraint is necessary especially for parsing noisy input and for generations. In parsing noisy streams of input, the retrieval of a specific 'utterance-case' may provide a strong constraint on a specific phonemic configuration. In terms of generations, since syntactic and semantic knowledge are maintained without pragmatic restrictions, most generation programs in existence over-generate. In other words, sentences that are grammatical and acceptable in terms of syntax/semantics but are unacceptable to native speakers may be over-generated by these generators. Under the MCP model described in the previous subsection, during a parsing, a specific 'utterance-case' may be acquired that contains the current syntactic, semantic and contextual environment that may be anchored to specific morphological realizations. The constraints associated with the 'utterance-case'

may be propagated during the future parsing and generation so that an experience of parsing one sentence can positively influence the parsing and generations in the future.

8.4 Dedicated Hardware

As we have seen, the spreading activation guided marker passing algorithm is massively parallel in nature. It leads to our understanding that Φ DMTRANS is ideal for the new generation computer architectures where massively parallel processings are supported from the hardware level (such as FMNN machines, Tomabechi&Kitano[1989]). We currently have a version of Φ DMTRANS on MULTILISP parallel lisp environment; however, we would like to see the system running on much more massively parallel architecture²¹ which can support the parallelism of every phonemic and concept sequence recognition performed concurrently at all levels of abstractions and triggered by multiple morphophonic, phonological and semantic hypothesizations of continuous speech inputs. Also, a dedicated VLSI chip for our algorithm is another viable possibility. Actually a part of the DMA algorithm has been implemented by using special VLSI chips (Kitano[1988]).

9 Conclusion

We have reported an integration of phonological and contextual knowledge in speech understanding in a massively parallel spreading activation marker passing network. As we have seen, the method of marker passing spreading activation is uniform from the phone level up to abstract thematic structures. Because a phonetic input stream can be hypothesized in multiple ambiguous and semantically acceptable ways, we have seen the necessity of both phonological knowledge and contextual knowledge participating during the course of direct memory access translation. We have also described the ambiguity resolution scheme based on cost analysis through 1) reference and instance creation; 2) constraint satisfaction. Also we have seen that C-Marker passing is another method of contextual disambiguation.

Parallel processing of concurrently active phonemic and conceptual sequences seems solely attainable in a DMA style spreading activation architecture. In the traditional build-and-store model, since the result of parsing is lost after the pro-

²¹Such as neuro-computer type architectures and connection machine (Hillis[1985]) type architectures.

cessing of each sentence, the context for subsequent translations is hardly ever established, whereas in our DMA model, context is naturally recorded as what is left in memory after understanding previous sentences as well as what is being recognized as parts of currently active concept sequences. With the explanatory generation mechanism added, the Φ DMTRANS model of translating a speech input is an extremely viable option for future massively parallel computers, in which massively parallel processing activity is hardware-supported²². The marker-passing based DMA architecture is also straightforwardly supported by the Frequency Modulation Neural-Network machine architecture (Tomabechi&Kitano[1989]) in which we can implement the massively parallel network from the hardware-level to the abstract concept-level in a uniform architecture.

²²Thus, although the massively parallel machines are yet to come to be utilized by the end-users of speech translation systems, we have shown the theory and an implementation that whenever personal massively parallel computers are available, we hope to see Φ DMTRANS running for business-people and for travelers in their personal machines.

APPENDIX 1: Implementation

Speech recognition hardware was by Matsushita Research Institute and is used in our system by the courtesy of the Institute. In addition to the firmware written control codes, the low-level control program is written in 'C' for the device hardware. Current implementation of Φ DMTRANS runs real-time²³ on HP9000 AI Workstations and is written in HP CommonLisp. The object-code of the speech recognition control programs is directly called from inside the CommonLisp code. Also, non-real-time²⁴ versions are implemented on IBM-RTs using CMU-CommonLisp and MULTILISP. The parallelism of spreading activation is simulated using lazy evaluations in CommonLisp versions. Parallelism in the MULTILISP version is supported at the operating system level on 'Mach' (Rashid, *et al*[1987]) at CMU. MULTILISP is described in Halstead[1985], which is a parallel lisp developed at MIT for Concert multi-processors and is now implemented on the distributed operating system 'Mach' at CMU. Because MULTILISP is a true parallel lisp, the MULTILISP version of Φ DMTRANS runs on any parallel hardware that supports MULTILISP. MULTILISP has already been implemented on several types of parallel computers including Concert, Multi-vaxens and Encores.

The speech synthesis module is a commercial product built by DEC called DECTALK. It is capable of producing different types of voices (female, male, young, old, etc.) and at varying pitches and can receive either phonemes or text inputs. Since the generator outputs the text output, the input to DECTALK is a text input and it produces the synthesized human voice.

APPENDIX 2: Distinctive Feature Matrix Using SPE

Below is the distinctive feature matrix used in our system for Japanese:

	p	t	(c)	k	b	d	g	(*)	s	z	r
cons	+	+	+	+	+	+	+	+	+	+	+
syll	-	-	-	-	-	-	-	-	-	-	-
son	-	-	-	-	-	-	-	+	-	-	+
high	-	-	+	+	-	-	+	+	-	-	-
back	-	-	-	+	-	-	+	+	-	-	-
low	-	-	-	-	-	-	-	-	-	-	-

²³By 'real-time' we mean that what is spoken into the microphone is translated into sentences in the target language with a negligible delay.

²⁴Non-real-time on IBM-RTs simply because hardware connections between RTs and the speech recognition hardware are not currently supported and therefore, processings are done via network.

cor	-	+	+	-	-	+	-	-	+	+	+
voice	-	-	-	-	+	+	+	+	-	+	+
cont	-	-	-	-	-	-	-	-	+	+	-
nasal	-	-	-	-	-	-	-	+	-	-	-
	m	n	=	w	j	h	i	e	a	o	u
cons	+	+	+	-	-	-	-	-	-	-	-
syll	-	-	+	-	-	-	+	+	+	+	+
son	+	+	+	+	+	-	+	+	+	+	+
high	-	-	+	+	+	-	+	-	-	-	+
back	-	-	+	+	-	-	-	-	+	+	+
low	-	-	-	-	-	-	-	-	+	-	-
cor	-	+	-	-	-	-	-	-	-	-	-
voice	+	+	+	+	+	-	+	+	+	+	+
cont	-	-	-	+	+	+	+	+	+	+	+
nasal	+	+	+	-	-	-	-	-	-	-	-

APPENDIX3: Sample Runs

The audio tape-recording of sample runs of our system is available from the authors by request.

References

- [1] Becker, J.D. (1975) 'The phrasal lexicon'. In *Theoretical Issues in Natural Language Processing*.
- [2] Berg, G. (1987) 'A Parallel Natural Language Processing Architecture with Distributed Control'. In *Proceedings of the CogSci-87*.
- [3] Bookman, L.A. (1987) 'A Microfeature Based Scheme for Modelling Semantics'. In *Proceedings of the IJCAI-87*.
- [4] Bresnan J. (Ed.) (1982) *The Mental Representation of Grammatical Relations*. MIT Press.
- [5] Charniak, E. (1983) 'Passing Markers: A theory of Contextual Influence in Language Comprehension'. *Cognitive Science* 7.
- [6] Charniak, E. (1986) 'A neat theory of marker passing'. In *Proceedings of the AAAI-86*.
- [7] Charniak, E. and Santos, E. (1987) 'A Connectionist Context-Free Parser Which is not Context-Free, But Then It is Not Really Connectionist Either'. In *Proceedings of the CogSci-87*.
- [8] Chomsky, N., and Halle, M. (1968) *The Sound Pattern of English*. New York: Harper and Row.
- [9] Fahlman, S. (1979) *NETL: A system for representing and using real-world knowledge*. The MIT Press.
- [10] Granger, R. and Eiselt, K. (1984) 'The parallel organization of lexical, syntactic, and pragmatic inference processes' In *Proceedings of the First Annual Workshop on Theoretical Issues in Conceptual Information Processing*.
- [11] Hahn, U. and Reimer U. (1983) *World expert parsing: An approach to text parsing with a distributed lexical grammar*. Technical Report, Universitat Konstanz, West Germany.
- [12] Halstead, R. (1985) 'Multilisp: A language for Concurrent Symbolic Computation'. In *ACM Trans. on Prog. Languages and Systems*.
- [13] Hauptmann, A., Young, R. and Ward, W. (1988) 'Using Dialog-Level Knowledge Sources to Improve Speech Recognition'. In *Proceedings of the AAAI-88*.

- [14] Hayes, P., Hauptmann, A., Carbonell, J. and Tomita M. (1986) 'Parsing Spoken Language: A Semantic Caseframe Approach'. In *Proceedings of COLING-86*.
- [15] Hillis, Daniel W. (1985) *The Connection Machine*. The MIT Press.
- [16] Hiraoka, S., Morii, S., Hoshimi, M. and Niyada, K. (1986) 'Compact Isolated Word Recognition System for Large Vocabulary.' In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP86)*.
- [17] Hirst, G. and Charniak, E. (1982) 'Word Sense and Slot Disambiguation'. In *Proceedings of AAAI-82*.
- [18] Hirst, G. (1984) 'A Semantic Process for Syntactic Disambiguation'. In *Proceedings of AAAI-84*.
- [19] Hyman, L. (1975) *Phonology: Theory and Analysis*. Holt, Rinehart and Winston.
- [20] Jakobson, R. and Halle, M. (1956) *Fundamentals of Language*. Mouton.
- [21] Kempen, G. and Hoenkamp, E. (1987) 'An Incremental Procedural Grammar for Sentence Formulation'. *Cognitive Science* 11, 201-258.
- [22] Knight, K. (1988) *Unification: A Multi-Disciplinary Survey* CMU MS Thesis.
- [23] Kitano, H. (1988) 'Multilingual Information Retrieval Mechanism using VLSI'. In *Proceedings of RIAO-88*.
- [24] Kitano, H. (1989) 'A Massively Parallel Model of Natural Language Generation for Interpreting Telephony: Almost Concurrent Processing of Parsing and Generation'. In *Proceedings of the Second European Workshop on Natural Language Generation*.
- [25] Kitano, H., Tomabechi, H and Levin, L., (1989) 'Ambiguity Resolution in Φ DMTRANS'. In *Proceedings of the fourth Conference of the European Chapter of the Association for Computational Linguistics*. (1989)
- [26] Kitano, H. and Tomabechi, H., Manuscript. (ms a) *A Model of Natural Language Generation under Massively Parallel Computational Model*. Carnegie Mellon University.

- [27] Kitano, H. and Tomabechi, H., Manuscript. (ms b) *Acquisition of Utterance-Cases under Massively-Parallel Structured-Marker Passing Scheme: Avoiding over-generations with dynamic acquisitions of utterance-cases*. Carnegie Mellon University.
- [28] Lytinen S. (1984) *The organization of knowledge in a multi-lingual, integrated parser*. Ph.D. thesis Yale University.
- [29] Morii, S., Niyada, K., Fujii, S. and Hoshimi, M. (1985) 'Large Vocabulary Speaker-independent Japanese Speech Recognition System.' In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP85)*.
- [30] Norvig, P. (1987) 'Inference in Text Understanding'. In *Proceedings of the AAAI-87*.
- [31] Nyberg, E. (1988) *The FrameKit User's Guide Version 2.0*. CMU-CMT-88-107. Carnegie Mellon University.
- [32] Poesio, M. and Rullent, C. (1987) 'Modified Caseframe Parsing for Speech Understanding Systems'. In *Proceedings of the IJCAI-87*.
- [33] Pollard, C. and Sag, A. (1987) *An Information-based Syntax and Semantics*. Vol 1. CSLI.
- [34] Quillian, M.R. (1968) 'Semantic Memory'. In *Semantic Information Processing*, ed. Minsky, M. MIT Press.
- [35] Quillian, M.R. (1969) *The teachable language comprehender*. BBN Scientific Report 10.
- [36] Rashid, R., A. Tevanian, M. Younge, D. Youge, R. Baron, D. Black, W. Bolosky and J. Chew (1987) 'Machine-Independent Virtual Memory Management for Paged Uniprocessor and Multiprocessor Architectures'. CMU-CS-87-140. Carnegie Mellon University.
- [37] Riesbeck, C. and Martin, C. (1985) *Direct Memory Access Parsing*. Yale University Report 354.
- [38] Saito, H. and Tomita, M. (1988) 'Parsing Noisy Sentences'. In *Proceedings of the COLING-88*.
- [39] Schank, R. (1982) *Dynamic Memory: A theory of learning in computers and people*. Cambridge University Press.

- [40] Schank, R. (1986) *Explanation Patterns: Understanding mechanically and creatively*. Lawrence Erlbaum Associates, Publishers.
- [41] Small, S. and Reiger, C. (1982) 'Parsing and comprehending with word experts (a theory and its realization)'. In *Strategies for natural language processing* Eds. Lenhart G. and Ringle M. Lawrence Erlbaum.
- [42] Tomabechi, H. (1987a) 'Direct Memory Access Translation'. In *Proceedings of the IJCAI-87*.
- [43] Tomabechi, H. (1987b) *Direct Memory Access Translation: A Theory of Translation*. CMU-CMT-87-105, Carnegie Mellon University.
- [44] Tomabechi, H. and Tomita, (1988a) 'The Integration of Unification-based Syntax/Semantics and Memory-based Pragmatics for Real-Time Understanding of Noisy Continuous Speech Input'. In *Proceedings of the AAAI-88*.
- [45] Tomabechi, H. and Tomita, M. (1988b) 'Application of the Direct Memory Access paradigm to natural language interfaces to knowledge-based systems'. In *Proceedings of the COLING-88*.
- [46] Tomabechi, H. and Tomita, M. Manuscript. 'Massively Parallel Constraint Propagation: Parsing with Unification-based Grammar without Unification'. Center for Machine Translation, Carnegie Mellon University.
- [47] Tomabechi, H. Mitamura, T. and Tomita, M. (1988) 'Direct Memory Access Translation for Speech Input: A Massively Parallel Network of Episodic/Thematic and Phonological Memory. In *Proceedings of the International Conference on Fifth Generation Computer Systems 1988 (FGCS'88)*.
- [48] Tomabechi, H., Saito, H. and Tomita, M. (1989) 'SPEECHTRANS: An Experimental Real-Time Speech-to-Speech Translation System'. In *Proceedings of the Spring Symposium of the AAAI 1989*.
- [49] Tomabechi, H. and Kitano, H. (1989) 'Beyond PDP: the Frequency Modulation Neural Network Architecture'. In *Proceedings of the IJCAI-89*.
- [50] Tomabechi, H. and Levin, L. Manuscript. 'Head-features and subcategorization as interacting constraints in associative memory'. Center for Machine Translation, Carnegie Mellon University.

- [51] Tomita, M. (1986) 'An Efficient Word Lattice Parsing Algorithm for Continuous Speech Recognition'. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP86)*.
- [52] Touretzky, D (1988) 'Beyond Associative Memory: Connectionists Must Search for Other Cognitive Primitives'. In *Proceedings of the 1988 AAAI Spring Symposium Series. Parallel Models of Intelligence*.
- [53] Waibel, A., Hanazawa, T., Hinton, G., Shikano, K. and Lang, K., "Phoneme Recognition Using Time-Delay Neural Networks," *IEEE, Transactions on Acoustics, Speech and Signal Processing*, March, 1989.
- [54] Waltz, D. and Pollack, J. (1984) 'Phenomenologically plausible parsing'. In *Proceedings of the AAAI-84*.
- [55] Waltz, D. L. and Pollack, J. B., 'Massively Parallel Parsing: A Strongly Interactive Model of Natural Language Interpretation.' *Cognitive Science* 9(1): 51-74, 1985.
- [56] Webber, B. L., 'So what can we talk about now?', in *Computational Models of Discourse*, (Eds. M. Brady and R.C. Berwick), MIT Press, 1983.