High Performance Memory-Based Translation on IXM2 Massively Parallel Associative Memory Processor*

Hiroaki Kitano[1,2] and Tetsuya Higuchi[3]

Center for Machine Translation[1]
Carnegie Mellon University
Pittsburgh, PA 15213 U.S.A.
hiroaki@cs.cmu.edu

NEC Corporation[2]
5-33-1 Shiba, Minato-ku
Tokyo 108, Japan

Electrotechnical Laboratory[3]
1-1-4 Umezono, Tsukuba
Ibaraki 305 Japan
higuchi@etl.go.jp

Abstract

This paper reports experimental results of a high performance (real-time) memory-based translation. Memory-based translation is a new approach to machine translation which uses examples, or cases, of past translations to carry out translation of sentences. This idea is counter to traditional machine translation systems which rely on extensive use of rules in parsing, transfer and generation. Although, there are some preliminary reports on the superiority of the memory-based translation in terms of its scalability, quality of translation, and easiness of grammar writing, we have not seen any reports on its performance. This is perhaps, the first report discussing the feasibility and problems of the approach based on actual massively parallel implementation using real data. We also claim that the architecture of the IXM2 associative processor is highly suitable for memory-based translation tasks. Parsing performance of the memory-based translation system attained a few milliseconds per sentence.

1 Introduction

In this paper, we demonstrate that the *memory-based translation* model offers extremely high performance (real-time) on massively parallel machines, and exhibits a desirable scaling property. We will also show that the architecture of the IXM2 parallel associative processor, which extensively uses associative memory for storing and processing memory-base, offers much higher performance than other massively parallel architectures such as the CM-2 Connection Machine [Thinking Machine Corp., 1989].

The traditional approach to machine translation (MT) has been to rely on extensive rule application. This approach, however, exhibits several undesirable properties when a system grows to substantial size. There are several major problems in the current approach to machine translation which is widely recognized by researchers and industrial engineers engaged in this field. These are:

Performance: Performance of most existing machine translation systems is not good. It takes about a few seconds to a few minutes to translate one sentence. This performance is totally insufficient to carry out real-time spoken language translation or bulk text translation.

Scalability: Current machine translation systems are difficult to scale up because their processing complexity makes the systems' behavior almost intractable.

Quality: Intractability of a system's behavior combined with other factors lowers the quality of translations.

Grammar Writing: By the same token, grammar writing is very difficult since a complex sentence has to be described by the piecewise rules. It is a hard and time consuming task partly due to the intractability of the behavior when they are added into the whole system.

We propose memory-based translation to overcome these problems. All of these problems are equally important; however, as a conference paper, we do not have the space to discuss them. Instead, we will focus on the performance issues. Since memory-based translation was initially proposed to improve the quality of translation and to improve the productivity of grammar writing, there are some reports about experiments which shows the advantages of the memory-based translation over the traditional machine translation approach in these aspects. Those who are interested in the quality and the grammar aspect of this approach should refer to [Nagao, 1984], [Sato and Nagao, 1990], [Furuse et. al., 1990], and [Sumita and Iida, 1991].

In this paper, we will focus on the performance issue, particularly on parsing. The performance problem is one of the most serious problems of current machine translation systems, especially when machine translation is to be applied to spoken language translation and other applications which require real-time translation. When the translation process takes from a few seconds to a few minutes its is hopeless that any form of spoken language translation or any other real-time applications can be deployed either experimentally or commercially. However, most of the current parsing strategies require extensive computing time. ATR's translation system requires a translation process from 61 seconds to 119 seconds to complete, even for sentences in their conference registration task [Ogura et. al., 1989]. Our goal, therefore, is to develop a system which carries out translation in real-time, i.e. at milliseconds order. The real-time translation is the significant goal in MT research due to its potential breadth of applications. Besides its obvious application to spoken language processing, the ability to process natural language with high-performance has a large industrial impact. Its applications are wide spread including intelligent full text-retrieval, text classification, summarization, speech interface, large corpora processing, etc.

The poor performance of traditional machine translation systems (also applies to traditional parsers in general) can be attributed to the serial application of piecewise rules. Thus,

*This work is supported in part by the Pittsburgh Supercomputing Center under grant TRA900105P and IRI-910002P.

the time complexity of parsing algorithms has been one of the central issues of the research. The most efficient parsing algorithm known to date is Tomita's generalized LR parser which takes less than $O(n^3)$ for most practical cases[Tomita, 1986]. Although some attempt has been made to implement a parallel parsing process to improve its performance (such as [Tanaka and Numazaki, 1989]), the degree of parallelism attained by implementing a parallel version of the traditional parsing scheme is rather low. Plus, it degrades more than linearly as the length of the input sentence gets longer. Thus, no dramatic increase in speed can be expected unless a highly parallel algorithm is employed.

A highly parallel algorithm, however, requires extensive computational cost for search in memory when executed on serial machines, hence no significant improvement of performance can be expected. Massively parallel machines are the obvious choice to implement such highly parallel algorithms. Surprisingly, massively parallel machines such as CM-2 do not exhibit a high performance when the knowledge-base has a high average fanout factor which is the case in our application. This is due to its serial link constraint to propagate markers between processors. Our alternative to this problem is the use of the IXM2 architecture which extensively uses associative memory so that parallel marker-propagation to search memory can be carried out in constant time regardless of the fanout factor.

In this paper, we wish to demonstrate that the performance problem can be solved, or at least significantly mitigated, by using memory-based translation on massively parallel machines. We would also like to demonstrate that the machine architecture which we advocate in this paper provides a significantly better performance than do other massively parallel machines. Although it is generally argued that time-complexity can be transferred to space-complexity, and use of massively parallel machines can attain high performance, there is no previous study to experimentally prove this hypothesis. This paper is, perhaps, the first paper addressing this issue using actual experimental data.

2 Memory-Based Translation

The idea of using examples for translation was first proposed by [Nagao, 1984] as 'Translation by Analogy.' Recently, this approach began gaining increasing attention due to the problems in the traditional machine translation approach. There are already some papers which report preliminary idcas and on preliminary experiments ([Sato and Nagao, 1990], [Furuse et. al., 1990], and [Sumita and Iida, 1991]). This increasing interest in memory-based translation coincides with recent excitement about memory-based reasoning (MBR) [Stanfill and Waltz, 1986] and case-based reasoning (CBR) [Riesbeck and Schank, 1989]. The basic idea of memory-based reasoning places memory at the foundation of intelligence. It assumes that large numbers of specific events are stored in memory; and response to new events is handled by first recalling past events which are similar to the new input, and invoking actions associated with these retrieved events to handle the new input. This idea runs counter to most AI approaches which place rules or heuristics as the central thrust of reasoning.

The memory-based approach works well in machine translation, too. For example, in the ATR's conference registration corpus, almost all the sentences that involve the word 'would' can be expressed by a template `<I would like to *action>`, this is called the *Concept Sequence*. It is clear from the following KWIC view of the corpus:

English	Interlingua	Japanese
<I would like to *action>	*Sentence	<*action shitai nodesu>
<that would be *state>	*Sentence	<soreha *state desu>
<*office for *event>	*office	<*event *office>
<*action for *event>	*action	<*event ni *action>
<*action for *person>	*action	<*person ni *action>
<*action for *object>	*action	<*object ni *action>
<*object for *action>	*object	<*action *object>
<*object for *event>	*object	<*event no (you no) *object>
<*object for *object>	*object	<*object you no *object>
<*object for *person>	*object	<*person ni *object>
<hello>	*hello-tel	<moshimoshi>
<How's your business?>	*greeting-bus	<moukattemakka?>

Table 1: Concept Sequences for Source and Target Language

I would like to register for the conference.
I would like to take part in the conference.
I would like to attend the conference.
I would like to contribute a paper to the conference.
That would be helpful for me. Thank you very much.
I would like to know the details of the conference.
so I would like to cancel.
I would like to contribute a paper to the conference.
I would like to ask you about hotel accommodations for the conference.
Then I would like to make a reservation for the Hilton Hotel.

Thus by having a few pairs of templates all the cases of sentences using 'would' in the corpus can be translated. In addition, only seven templates would translate all the cases in the corpus of the phrase filling `*action`. Although, at a glance, this seems to be a naive approach, the method essentially entails comparable or even superior translation capability than current MT systems. Given the fact that large-scale MT systems have a few thousand grammar rules in which most of them are for the exception handling of each specific case, the Memory-Based Translation is a straightforward and tractable approach to translation because it uniformally handles regular and irregular cases. A part of concept sequences for English-Japanese translation is shown in Table 1.

Unlike other machine translation systems which employ a single level of abstraction in parsing and generation, our model uses multiple levels of abstraction. For instance, the KBMT-89 system [Goodman and Nirenberg, 1991] uses Lexical Functional Grammar (LFG) as a basis for parsing and generation, but it does not use any phrasal lexicons or semantic-grammars along with the LFG rules. On the contrary, in our model, *specific case*, *generalized case* and *unification grammar* co-exist. This is illustrated in Figure 1. There, line (α) represents the process of translating a specific case, i.e. a representation of a particular source language sentence. The level of abstraction increases as we move up to line (β), which traces the translation of what we call "generalized cases" or conceptual representations (given as `<*person *want *to *circum>`). At the most abstract level, (γ), we rely on a unification-based grammar. Translation occurs at the lowest — the least abstract — possible level.

Advantages of using multiple levels of abstraction are the following. The approach:

1. Improves performance by performing translations whenever possible at a level closer to the surface; there is no need for expensive unification or constraint-based processes.
2. Ensures scalablity since new sentence structures can be handled simply by adding new concept sequences, or templates.

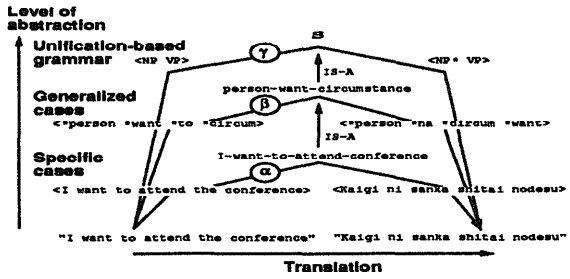

Figure 1: Translation paths at different levels of abstraction

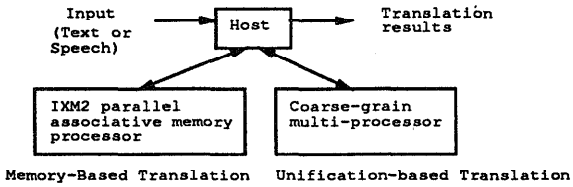

Figure 2: Diagram of the System Integrating Memory-Based and Unification Approach

3. Attains high performance with massively parallel machines because, in most cases, the translation can be done by finding specific or generalized cases during parsing and by invoking corresponding cases in the target language. This essentially converts time-complexity into space-complexity, which is not a problem with massively parallel machines.

When none of the cases is applicable for the input, the unification- or constraint-based process is invoked on the coarse-grain multi-processors as shown in figure 2. The memory-based translation carried out on the IXM2 will cover translation paths such as α and β in figure 1. A coarse-grain parallel machine or a high performance serial machine will cover the rest, γ in figure 1. We expect that the memory-based translation process (α and γ) covers most of cases (more than 99%).

3 The Massively Parallel Associative Processor IXM2

IXM2 is a massively parallel associative processor designed and developed at the Electrotechnical Laboratory [Higuchi et. al., 1991]. It is dedicated to semantic network processing using marker-passing.

IXM2 consists of 64 processors, called *associative processors*, which operate with associative memory, each of which has a memory capacity of 4K words by 40 bits. Each associative processor is connected to other associative processors through network processors.

An associative processor consists of an IMS T800 transputer, 8 associative memory chips, RAM, link adapters, and associated logic. When operated at 20 MHz clock, T800 attains 10 MIPS [Inmos, 1987]. Each associative memory chip is a 20 Kbit CAM (512 words $\times$ 40 bits). The IXM2 has 64 such processors, thus attaining 256K parallelism which is far larger than 64K parallel of the Connection Machine [Hillis, 1985]. This high level of parallelism allows us to implement

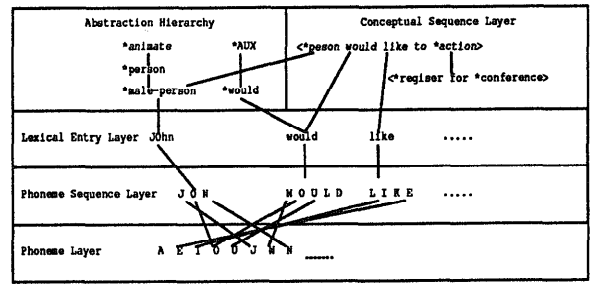

Figure 3: Overall Architecture of the Parsing Part

practical memory-based systems. The design decision to use associate memory chips driven by 32 bit CPUs, instead of having thousands of 1-bit CPUs, is the major contributing factor for performance, processor efficiency, and cost performance.

Network processors are used to handle communication between associative processors. There is one top-level network processor which deals with communication among the lower-level network processors, and 8 lower-level network processors each of which is connected to 8 associative processors. Unlike most other massively parallel architectures which use N-cube connections or cross-bar connections, IXM2 employs a full connection so that communication between any two processors can be attained by going through only 2 network processors. This full connection architecture ensures high communication bandwidth and expandability which are critical factors in implementing real-time applications. Each interconnection attains high speed serial links (20 Mbits/sec) which enable the maximum transfer rate per link at the speed of 2.4 Mbytes/sec.

4 ASTRAL: An Experimental System

ASTRAL[1] is an implementation of the memory-based translation on IXM2. The overall architecture is shown in figure 3. The memory consists of four layers: a phoneme sequence layer, a lexical entry layer, abstraction hierarchy, and a concept sequence layer.

Phoneme Layer: Phonemes are represented as nodes in the network, and they are connected to each instance of phoneme in the phoneme sequence layer. Weights are associated to links which represent the likelyhood of acoustic confusion between phonemes.

Phoneme Sequence Layer: The phoneme sequence of each word is represented in the form of a network. This part is shown in figure 4.

Lexical Entry Layer: The lexical entry layer is a set of nodes each of which represents a specific lexical entry.

Abstraction Hierarchy: The class/subclass relation is represented using IS-A links. The highest (the most general) concept is **all* which entails all the possible concepts in the network. Subclasses are linked under the **all* node, and each subclass node has its own subclasses. As a basis of the ontological hierarchy, we

[1]ASTRAL is an acronym for the *Associative model of Translation of Language*.

```
link(first,ax31,about).
link(last,t34,about).
link(instance_of,ax31,ax).
link(destination,ax31,b32).
link(instance_of,b32,b).
link(destination,b32,aw33).
link(instance_of,aw33,aw).
link(destination,aw33,t34).
link(instance_of,t34,t).
```

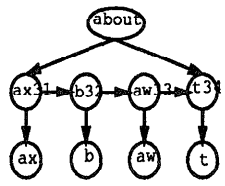

Figure 4: Network for 'about' and its phoneme sequence

use the hierarchy developed for the MU project [Tsujii, 1985], and domain specific knowledge has been added.

Concept Sequence: Concept sequences which represent patterns of input sentences are represented in the form of a network. Concept sequences capture linguistic knowledge (syntax) with selectional restrictions.

Figure 4 shows a part of the network. The figure shows a node for the word 'about', and how the phoneme sequence is represented. The left side of the figure is a set of IXM instructions to encode the network in the right side on the IXM2 processor. Refer [Higuchi et. al., 1991] for details of the mapping of semantic networks to IXM2. We have encoded a network including phonemes, phoneme sequences, lexical entries, abstraction hierarchies, concept sequences which cover the entire task of the ATR's conference registration domain [Ogura et. al., 1989]. The vocabulary size is 405 words in one language, and at least over 300 sentences in the corpus have been covered. The average fanout of the network is 40.6. The weight value has not been set in this experiment in order to compare the performance with other parsers which do not handle stochastic inputs. In the real operation, however, a fully tuned weight is used. The implementation in this paper is different from [Kitano and Higuchi, 1991] which simply stores flat sequences of syntactic categories. The implementation in this paper uses a hiearchical memory networks thereby attaining a wider coverage with smaller memory requirements[2].

The table for templates of the target language is stored in the host computer (SUN-3/250). The binding-table of each concept and concept sequence, and specific substrings are also created. When the parsing is complete, the generation process is invoked on the host. It is also possible to compute distributively on 64 T800 transputers. The generation process is computationally cheap since it only retrieves and concatenates substrings (which is a lexical realization in the target language) bound to conceptual nodes following the patterns of the concept sequence in the target language. The system implemented in this paper is based on the generation algorithm described in [Kitano, 1990b] with substantial modifications to meet hardware constraints.

The algorithm is simple. Two markers, activation markers (A-Markers) and prediction markers (P-Markers) are used to control the parsing process. A-Markers are propagated through the memory network from the lexical items which are activated by the input. P-Markers are used to mark the next possible elements to be activated. This algorithm is similar to the basic framework of the ΦDMDIALOG speech-to-speech translation system [Kitano, 1990a], and inherits the basic notion of the direct memory access parsing (DMAP) [Riesbeck and Martin, 1986]. The parsing algorithm can process context-free grammar (CFG) and augmented CFG using constraints (in effect, augment CFG is Context Sensitive Grammar due to constraints added to CFG). Part of the parsing process is analogous to the Earley-type shift-reduce parser. To help understanding, *shift* and *reduce* have been labeled where appropriate. However, the basic operation is highly parallel. Particularly, it exhibits the data-parallel nature of the operation due to simultaneous operations for all the data in the memory. A general algorithm follows (only a basic framework is shown. Some extra procedures are necessary to handle CFG and Augmented CFG.):

1. Place P-Markers at all first elements of Concept Sequence.
2. Activate Phoneme Node.
3. Pass A-Markers from the Phoneme Node to Nodes of Phoneme Sequences.
4. If the A-Marker and a P-Marker co-exist (this is called an *A-P-Collision*) at an element in the Phoneme Sequence, then the P-Marker is moved to the next element of the Phoneme Sequence. (Shift)
5. If the A-P-Collision takes place at the last element of the phoneme sequence, an A-Marker is passed up to the Lexical Entry. (Reduce) Else, Goto 2.
6. Pass the A-Marker from the lexical entry to the Concept Node.
7. Pass the A-Marker from the Concept Node to the elements in the Concept Sequence.
8. If the A-Marker and a P-Marker co-exist at an element in the Concept Sequence,
   then the P-Marker is moved to the next element of the Concept Sequence (Shift).
9. If an A-P-Collision takes place at the last element of the Concept Sequence, the Concept Sequence is temporarily accepted (Reduce),
   and an A-Marker is passed up to abstract nodes.
   Else, Goto 2.
10. If the Top-level Concept Sequence is accepted, invoke the generation process.

5 Performance

We carried out several experiments to measure the system's performance. Figure 5 shows the parsing time against sentences of various lengths. Parsing at milliseconds order is attained. PLR is a parallel version of Tomita's LR parser. The performance for PLR was shown only to provide a general idea of the speed of the traditional parsing models. Since machines and grammars are different from PLR and our experiments, we can not make a direct comparison. However, its order of time required, and exponentially increasing parsing time clearly demonstrate the problems inherent in the traditional approach. The memory-based approach on IXM2 (MBT on IXM2) shows a magnituate faster parsing performance. Also, its parsing time increases almost linearly to the length of the input sentences, as opposed to the exponential increase seen in the PLR. Notice that this graph is drawn in log scale for the Y-axis. CM-2 is slow in speed, but exhibits similar characteristics with IXM2. The speed is due to PE's capabilities and machine architecture, and the fact that CM-2 shows a similar curveture indicates the benefits of

[2]An alternative method of covering wider inputs is to use similarity-based matching as seen in [Sumita and Iida, 1991]. Combining such an approach with our model is feasible.

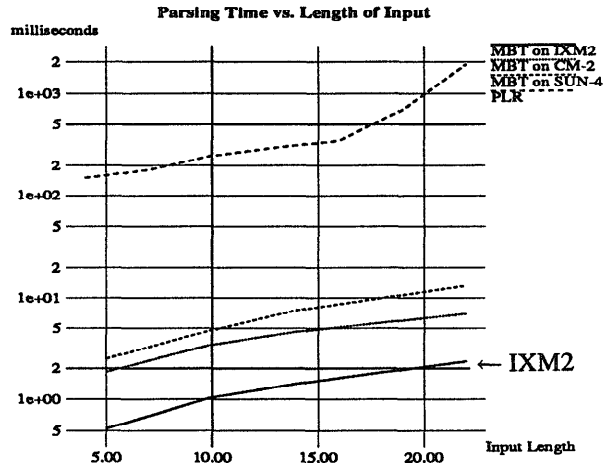

Figure 5: Parsing Time vs. Length of Input

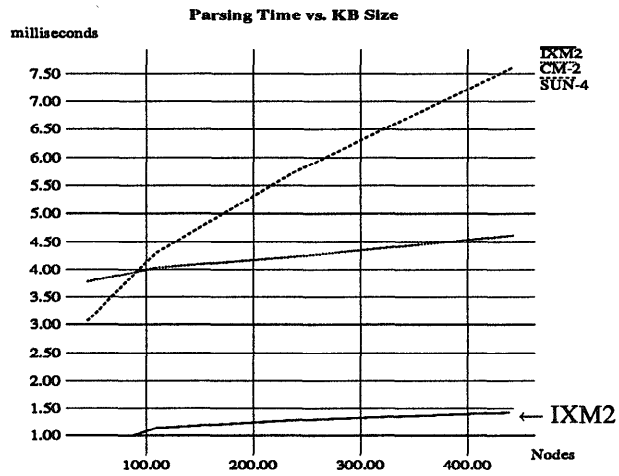

Figure 6: Parsing Time vs. KB Size

the MBT. The SUN-4 shows a similar curve, too. However, because the SUN-4 is a serial machine, its performance degrades drastically as the size of the KB grows, as discussed below.

Scalability is demonstrated in figure 6. The parsing time of a sentence with 14 input symbols is shown for various sizes of KBs. The size of the KB is measured by the number of nodes in the network. The performance degradation is less than linear due to the local activation of the algorithm. This trend is the opposite of the traditional parser in which the parsing time grows beyond linear to the size of the grammar KB (which generally grows square to the size of grammar rules, $o(G^2)$) due to a combinatorial explosion of the serial rule applications. CM-2 shows a similar curve with IXM2, but is much slower due to the slow processing capability of 1-bit PEs. The SUN-4 has a disadvantage in a scaled up KB due to its serial architecture. Particularly, the MBT algorithm involves an extensive set operations to find nodes with the A-P-Collision, which is suitable for SIMD machines. Serial machines need to search the entire KB which lead to the undesirable performance as shown in the figures in this section.

6 Summary and Discussions

In this paper, we described a *massively parallel memory-based translation* on the IXM2 associative processor. We have shown, using data obtained from our experiments, that the *massively parallel memory-based translation* is a promising approach to implement a high-performance real-time parsing system. Major claims and observations made from our experiments include:

- The massively parallel memory-based translation attains real-time translation when implemented on a massively parallel machine. Our experiments using the IXM2 associative memory processor show that parsing is completed on the order of a few milliseconds, whereas the traditional parsers requires a few seconds to even a few minutes. The main reason for this performance is the data-parallel nature of the memory-based translation paradigm where a parallel search is carried out for all sentence patterns (represented as conceptual sequences). In addition, the parsing time grows only linearly (or sublinearly) to the size of the inputs ($\leq o(n)$), whereas traditional parsers generally require $o(n^3)$. The system not only attains milli-second order parsing performance, but also exhibits a desirable scaling property. The parsing time required grows only sublinearly to the size of the knowledge-base loaded. This scaling property is the real benefit of using a massively parallel machine. Also, we can state that the memory-based approach is promising for large-scale domains.
- The effectiveness of the IXM2's architecture for large-scale parallelism has been confirmed. In the memory-based translation, a large set of sentence patterns are stored in associative memory. In natural language processing, each phoneme, word, and concept appear in various places due to the vast combinatorial possibilities of sentence production. This is particularly true for the memory-based translation because surface, near-surface, and conceptual sequences are used, which are more specific than most grammar rules. Because of this representation level, the average fanout of the semantic network which represents linguistic knowledge is large. The network used in this experiment has an average fanout of 40.6. The IXM2 has an architectural advantage in processing networks with a large fanout. An additional experiment verifies the advantage of the IXM2 architecture for this type of processing. Given a network with a different fanout, the IXM2 has an overwhelming performance over other machines as average fanout becomes larger (Figure 7). While other machines degrade its performance, the IXM2 keeps a constant time to complete the propagation of the markers to all nodes linked to the source of activation. This is due to the use of associative memory in IXM2. For memory-based natural language processing, this approach is extremely powerful because semantic networks for natural language processing tend to have a large fanout factor as seem in the example in this paper.

One of the major contributions of this paper, however, is that we have shown that the time-complexity of the natural language processing can be transferred to space-complexity, thereby drastically improving the performance of the parsing when executed on massively parallel machines. This

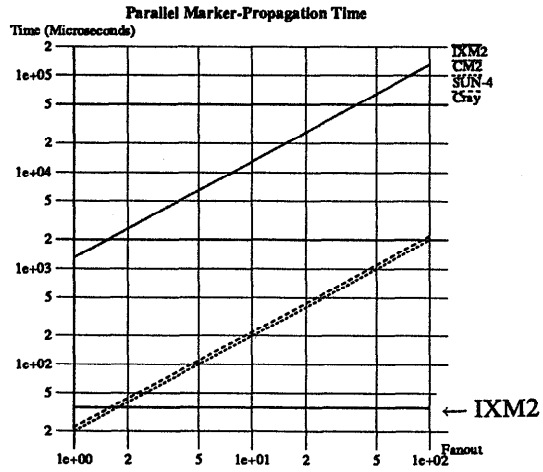

Figure 7: Parallel Marker-Propagation Time vs. Fanout

assumption is the basic thrust of the memory-based and case-based reasoning paradigm. This point has been clearly illustrated by comparing a version of Tomita's LR parsing algorithm and the memory-based parsing approach. Traditional parsing strategies exhibited an exponential degradation due to extensive rule application, even in a parallel algorithm. The memory-based approach avoids this problem by using hierarchical network which compiles grammars and knowledge in a memory-intensive way. While many AI researchers have been speculatively assuming the speed up by massively parallel machines, this is the first report to actually support the benefit of the memory-based approach to natural language processing on massively parallel machines.

In addition, we have shown that the difference in architectures between massively parallel machines significantly affects the total performance of the application. The IXM2 is significantly faster than the CM-2, mainly due to its parallel marker-passing capability of associative memory.

Acknowledgements

We would like to thank the members of the Center for Machine Translation at Carnegie Mellon University, particularly Jaime Carbonell and Masaru Tomita, and the members of Electrotechnical Laboratory for discussions and support. Also, we would like to thank Hitoshi Iida and Akira Kurematsu at the ATR Interpreting Telephony Research Laboratories for allowing us to use their corpus, and their continuous encouragement.

References

[Becker, 1975] Becker, J. D., *The Phrasal Lexicon*, BBN, Technical Report, 3081, 1975.

[Furuse et. al., 1990] Furuse, O., Sumita, E., Iida, H., "A method for realizing Transfer-Driven Machine Translation," *Workshop on Natural Language Processing*, IPSJ, 1990 (in Japanese).

[Goodman and Nirenberg, 1991] Goodman, K., Nirenberg, S., *Knowledge Based Machine Translation: A Case Study*, Kluwer Academic, 1991 (In Press).

[Higuchi et. al., 1991] Higuchi, T., Kitano, H., Handa, K., Furuya, T., Takahashi, N., and Kokubu, A., "IXM2: A Parallel Associate Processor for Knowledge Processing," *Proceedings of the National Conference on Artificial Intelligence (AAAI-91)*, 1991.

[Hillis, 1985] Hillis, D., *The Connection Machine*, The M.I.T. Press, 1985.

[Inmos, 1987] Inmos, *IMS T800 Transputer*, 1987.

[Kitano, 1990a] Kitano, H., "ΦDMDIALOG: A Speech-to-Speech Dialogue Translation System," *Machine Translation*, 5, 301-338, 1990.

[Kitano, 1990b] Kitano, H., "Parallel Incremental Sentence Production for a Model of Simultaneous Interpretation," *Current Research in Natural Language Generation*, Dale, R., Mellish, C., Zock, M.,(Eds.), Academic Press, 1990

[Kitano and Higuchi, 1991] Kitano, H. and Higuchi, T., "Massively Parallel Memory-Based Parsing," *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI-91)*, Sydney, 1991.

[Nagao, 1984] Nagao, M., "A Framework of a Mechanical Translation between Japanese and English by Analogy Principle," *Artificial and Human Intelligence*, Elithorn, A. and Banerji, R. (Eds.), Elsevier Science Publishers, B.V. 1984.

[Ogura et. al., 1989] Ogura, K., Sakano, T., Hosaka, J., and Morimoto, T., *Spoken Lnaguage Translation Experimental System from Japanese to English*, TR-I-0102, ATR Interpreting Telephony Research Laboratories, 1989.

[Riesbeck and Martin, 1986] Riesbeck, C. and Martin, C., "Direct Memory Access Parsing," *Experience, Memory, and Reasoning*, Lawrence Erlbaum Associates, 1986.

[Riesbeck and Schank, 1989] Riesbeck, C. and Schank, R., *Inside Case-Based Reasoning*, Lawrence Erlbaum Associates, 1989.

[Sato and Nagao, 1990] Sato, S. and Nagao, M., "Toward Memory-based Translation," *Proceedings of COLING-90*, 1990.

[Stanfill and Waltz, 1986] Stanfill, C. and Waltz, D., "Toward Memory-Based Reasoning," *Communications of the ACM*, 1986.

[Sumita and Iida, 1991] Sumita, E., and Iida, H., "Experiments and Prospects of Example-Based Machine Translation," *Proceedings of ACL-91*, 1991.

[Tanaka and Numazaki, 1989] Tanaka, H. and Numazaki, H., "Parallel Generalized LR Parsing based on Logic Programming," *Proceedings of the First International Workshop on Parsing Technologies*, Pittsburgh, 1989.

[Thinking Machine Corp., 1989] Thinking Machine Corp., *Model CM-2 Technical Summary*, Technical Report TR89-1, 1989.

[Tomita, 1986] Tomita, M., *Efficient Parsing for Natural Language*, Kluwer Academic Publishers, 1986.

[Tsujii, 1985] Tsujii, J., "The Roles of Dictionaries in Machine Translation," *Jouhou Shori*, (Information Processing) Information Processing Society of Japan, Vol. 26, No. 10, 1985 (In Japanese).