Heterogeneous Computing for Example-Based Translation of Spoken Language

Eiichiro Sumita and Hitoshi Iida
ATR Interpreting Telecommunications Research Laboratories
2-2 Hikaridai, Seika, Soraku, Kyoto 619-02 JAPAN
phone: +81-7749-5-1301 fax: +81-7749-5-1308
e-mail: sumita@itl.atr.co.jp and iida@itl.atr.co.jp

Summary: Spoken language translation requires both (1) high accuracy and (2) a real-time response which are difficult to achieve using conventional technologies. To fulfill the first requirement, we have adopted an Example-Based Approach. It generates a target sentence by combining partial translations obtained by mimicking best-match partial translation examples. To fulfill the second requirement, this paper proposes using a Heterogeneous Computing Platform consisting of Multiple Instruction Multiple Data (MIMD) and Single Instruction Multiple Data (SIMD) parallel machines. Example-Based Approach is dominated by two processes, each of which is optimally accelerated by utilizing MIMD and SIMD, respectively, a) to build the source structure, and b) to retrieve the best-match examples. Experimental results show that Example-Based Approach is drastically speeded up with the Heterogeneous Computing Platform and has a performance sufficient for real-time response, even with a large vocabulary and a highly ambiguous sentence.

1. Introduction

Spoken language translation requires both high accuracy and a quick response. First, there is no doubt that translation accuracy is important, and there is a desire to minimize human intervention such as pre-, inter-, and post-interactions between user and machine. Second, a practical throughput is important to make real-time speech-to-speech translation viable. To fulfill the first requirement, the authors have proposed an Example-Based Approach. It generates a target sentence by combining partial translations obtained by mimicking best-match partial translation examples (hereafter, examples). It accurately performs structural disambiguation, target word selection, and whole translation. To fulfill the second requirement, this paper proposes using a Heterogeneous Computing Platform consisting of Multiple Instruction Multiple Data (MIMD) and Single Instruction Multiple Data (SIMD) parallel machines. In Example-Based Approaches, there are two dominant processes: Structure-Building, to build the source structure that covers an input sentence by combining examples; and Example-Retrieval, to retrieve the best-match examples. The Heterogeneous Computing Platform minimizes the total time for Example-Based Approach translation because Structure-Building can be parallelized best on a MIMD machine; Example-Retrieval can be parallelized best on a SIMD machine; and the cost of communication between Structure-Building and Example-Retrieval can be lessened to a negligibly minute degree by packing data. Experimental results show that Example-Based Approach can be drastically speeded up to a performance sufficient for real-time response, even with a large vocabulary (i.e., a huge example database) and a

highly ambiguous sentence. Consequently, the Example-Based Approach on an Heterogeneous Computing Platform meets the vital requirements of spoken language translation.

Section 2 introduces Example-Based Approaches and explains a sentence translation system using Example-Based Approach. Section 3 explains Structure-Building and Example-Retrieval in detail. Analyses of computational costs are also discussed. Section 4 describes the method and the speedup of parallelization of Structure-Building and Example-Retrieval on the Heterogeneous Computing Platform. Finally, Section 5 discusses related research.

2. Example-Based Approaches

The idea behind Example-Based Approaches and their notable features are introduced, and an algorithm for sentence translation using an Example-Based Approach is explained.

2.1. Idea and Features

Novel models for NLP have been studied in recent years. These methods have been called Example-Based Approaches because they rely upon linguistic examples derived from corpora, e.g., translation examples, and often utilize a best-match mechanism based on the semantic distance between linguistic expressions. In the early 1980s, Nagao presented the origin of Example-Based Approaches, "Translation by Analogy," based on the observation that a human being translates according to past translation experience (Nagao, 1984). Since the end of the 1980s, large corpora and powerful computational devices have allowed us to construct Nagao's model and to expand the model to deal with not only translation, but also other tasks such as parsing.

Example-Based Approaches surpass conventional approaches for NLP in several aspects. Here, we summarize observations made so far. Example-Based Approaches are accurate in a restricted domain if sufficient examples are prepared.[1] Example-Based Approaches can deal with well-known difficult problems such as target expression selection (e.g., function words (Sumita and Iida, 1992a; Sumita and Iida, 1992b), noun phrases (Sato, 1993a) and verb phrases (Sato, 1991)), and disambiguation of prepositional phrase attachment (Sumita et al., 1993a). They achieve high accuracy not only for these subproblems in machine translation, but also for sentence translation (Furuse et al., 1994). Example-Based Approaches are robust. Some deviations from conventional grammars that are specific to spoken language are handled well (Furuse and Iida, 1992). For example, particles such as **wa**, **o**, and **ni** are frequently omitted in spoken Japanese. These omissions are recovered well by Example-Based Approaches. Semantic distances used in many Example-Based Approaches are considered as a number indicating how much we can rely on the result. In our previous experiment (Sumita and Iida, 1992b) on the relationship between semantic distance and success rate, we found the tendency that the smaller the semantic distance, the better the quality. The

[1] An observation that translation quality improves as the number of examples increases was reported in our previous paper (Sumita and Iida, 1992b). Infinite examples are not required because best-match based on a thesaurus (Section 4.2) compensates for the notorious problem of low-frequency data. Let us examine experimental evidence here. In the pp-attachment problem, an Example-Based Approach that utilizes word cooccurrences, i.e., examples and a thesaurus, can achieve higher accuracy with a much smaller corpus than a statistically-based approach utilizing the frequency of word cooccurrences. Namely, words themselves are too fine-grained to get sufficient amounts of data.

central mechanism of Example-Based Approaches is language-independent. So far, we have implemented Japanese-to-English translation system and vice versa; (Sobashima et al., 1994) and Japanese-to-Korean translation system and vice versa. Not only Example-Retrieval, but also whole transfer process is shared by these four systems.

2.2. Mechanism of Sentence Translation using Example-Based Approach

Here, we describe a sentence translation model featuring an Example-Based Approach. A sentence is translated by combining partial examples in such a way that they jointly cover the sentence. Since the translation examples have a primary role and the whole process is controlled by transfer, we call our model Transfer-Driven Machine Translation (TDMT) (Furuse and Iida, 1992; Furuse et al., 1994; Furuse and Iida, 1994). In Example-Based Translation, there are two dominant processes: Structure-Building (building the source structure that covers an input sentence by combining examples); and Example-Retrieval (retrieving the best-match examples). Other low cost processes such as morphological analysis and target language generation are not focused on in this paper.

A translation example, a piece of the transfer knowledge, describes the correspondence between a source expression and target expressions. Each expression is represented by a pattern consisting of variables and constants (function words). The variables of a source pattern are accompanied by example words. For example, the correspondence between typical Japanese noun phrases of the pattern "X no Y" and English noun phrases is described as follows:[2]

X **no** Y => Y' **of** X' ((ronbun[*paper*], daimoku[*title*]),...),
Y' **for** X' ((hoteru[*hotel*], yoyaku[*reservation*]),...),
Y' **in** X' ((Kyooto[*Kyoto*], kaigi[*conference*]),...),

When TDMT translates the Japanese noun phrase "Oosaka[Osaka] **no** paatii[party]," it retrieves the best-match in the transfer knowledge, i.e., X no Y => Y' in X' (Kyooto[*Kyoto*], kaigi[*conference*]). According to this "best-match," TDMT generates "party **in** Osaka" by substituting English nouns for the Japanese nouns.

The top-level TDMT algorithm is as follows: produce possible source structures[3] in which source expressions are combined to cover the input **(Structure-Building);** transfer the source structures to the target structures by converting each source expression to the most appropriate target expressions using [a], [b] and [c] below.

For example, suppose the input Japanese sentence is as follows:

[2] In this example, capitals such as X and Y are variables for Japanese noun phrases; primed capitals such as X' and Y' are the English translations of X and Y, respectively; boldfaces such as **"no"** are function words, i.e., adnominal particles that correspond to English prepositions such as **of, for,** and **in.** An expression of the form j[*e*] represents a Japanese word, j, and the literally translated English word, *e*.

[3] Multiple structures are produced when ambiguity exists in syntactic relations between words in an input sentence. This is the main reason why Structure-Building costs are high, as will be explained in the next section. The most plausible structure is selected based on the total of the semantic distances obtained by Example-Retrieval.

"kaigi **no** toorokuryou **wa** annaisho **ni** kisaisa **re teimasu"**
[*conference*] [*of*] [*registrationfee*] [*sub*] [*announcement*] [*in*] [*be listed*]

The source structure shown in Figure 1 is produced by the combining of source expressions having patterns "X **no** Y", "X **wa** Y", "X **ni** Y", "X **re,**" and "X **teimasu**." The target structure shown in Figure 2 is produced according to the transfer knowledge. Finally, the following translation is obtained:

"The conference registration fee is listed in the announcement."

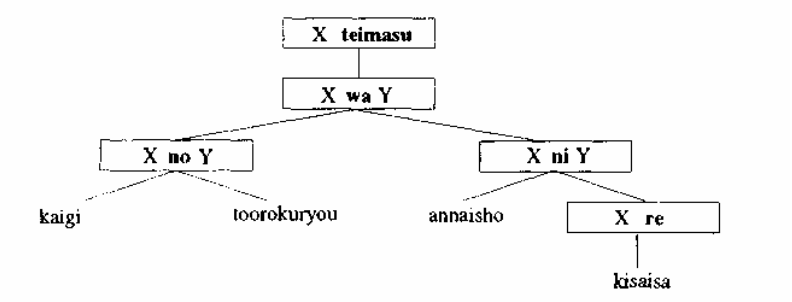

Figure 1: Source structure

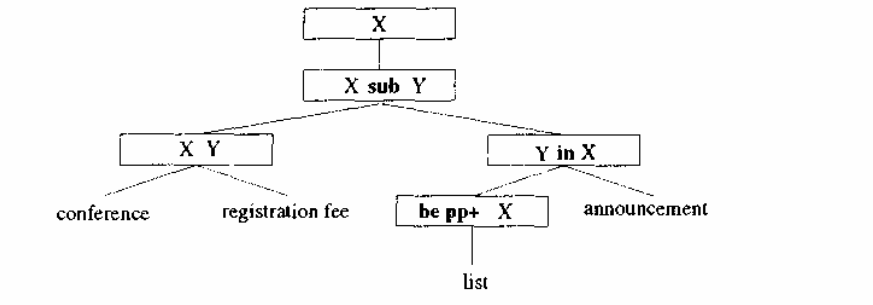

Figure 2: Target structure

The flow of transferring a source expression to the most plausible target expression is as follows:

[a] The semantic distance[4] from the input expression is calculated for **all** translation examples.
[b] The translation example with the **minimum-**distance is retrieved.
[c] The corresponding target expression associated with the retrieved translation example is used as the most plausible one for the input expression.

We call the combination of processes [a] and [b] **Example-Retrieval.**

[4] See the detailed explanation in Section 4.2.

3. Costs of Structure-Building and Example-Retrieval

This section analyzes the computational costs of Structure-Building and Example-Retrieval in a sequential implementation of a typical Example-Based Translation system, TDMT, that was introduced in the previous section. We investigate the figures of the prototype Japanese-to-English TDMT on a sequential machine, SPARCstation2, for 746 test sentences that are representative Japanese spoken sentences.[5] Figure 3 shows **the dominance of Example-Retrieval in translation.** The rate rises from about 0.2% to about 92.5%, and the average rate is about 70.6%. Moreover, the longer the translation time, the higher the rate. **The next most dominant part is Structure-Building.** As will be explained in the following subsections, Structure-Building cost is high when the input sentence is syntactically ambiguous and Example-Retrieval cost is high when the input sentence is syntactically ambiguous and/or the example database size is large.

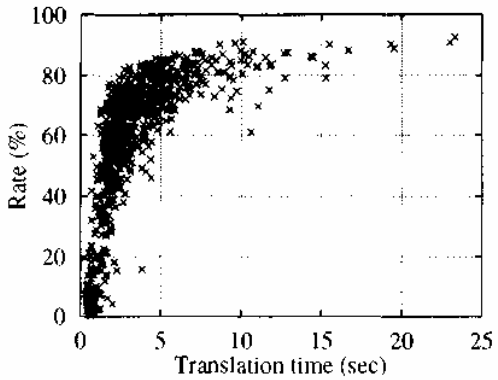

Figure 3: Rate for Example-Retrieval/Translation in sequential TDMT

3.1. Structure-Building Cost

A structure is built for the word sequence of an input sentence according to source patterns in a recursive top-down fashion. The patterns are classified into appropriate linguistic levels to hold down some explosions of structural ambiguity. They are applied from the highest level to the lowest level as follows:

(**I**) Look up the applicable patterns including the function words in the input sentence using the index from function words to patterns. Then, set LEVEL to the highest and execute (II).
(**II**) For all patterns on LEVEL, split the sequence into subsequences delimited by the function words, and bind the subsequences to the variables; execute (III) with respect to the variable bindings. Then, execute (IV).

[5] We have trained our prototype using a set of 825 sentences averaging about 9.0 words in length in our domain. These sentences cover broad variations of spoken expressions used in intermediate Japanese courses and include expressions for "request," "confirmation," "refusal," "permission," "obligation," "negation," and so on. These sentences were reviewed by Japanese linguists as well. Appendix I shows sample sentences for "request."

Of 825 sentences, we used 746 sentences (about 90.4%) in this experiment, excluding sentences translated by exact-match of whole sentences, e.g., "arigatoo-gozaimasu (Thank you very much)." Since exact-match sentences are instantly translated, there is no need to accelerate their translation.

(III) If the sequence is a word and is registered in the dictionary, return as success; otherwise, execute (II).
(IV) If LEVEL is the lowest, return fail; otherwise, decrement LEVEL and execute (II).

For example, the input sentence of Section 2.2, *"kaigi* **no** *toorokuryou* **wa** *annaisho* **ni** *kisaisa* **re teimasu**" is processed as follows: First, "X **teimasu"** is applied to the sequence; next, "X wa Y" is applied to the subsequence *"kaigi* **no** *toorokuryou* **wa** *annaisho* **ni** *kisaisa* **re";** then, "X **no** Y" and "X **ni** Y" are applied; finally, "X **re**" is applied.

Steps (II) to (IV) take care of syntactic ambiguity. To simplify this discussion, we take only a single level ambiguity into consideration. Let us study a sequence, "N1 **no** N2 ... **no** Ni," where N represents a noun.[6] For i=4, i.e., "A **no** B **no** C **no** D," we have five ambiguous source structures as shown in Figure 4. Similar kinds of ambiguous linguistic phenomena are also seen in other languages. A typical one is prepositional phrase attachment in English.

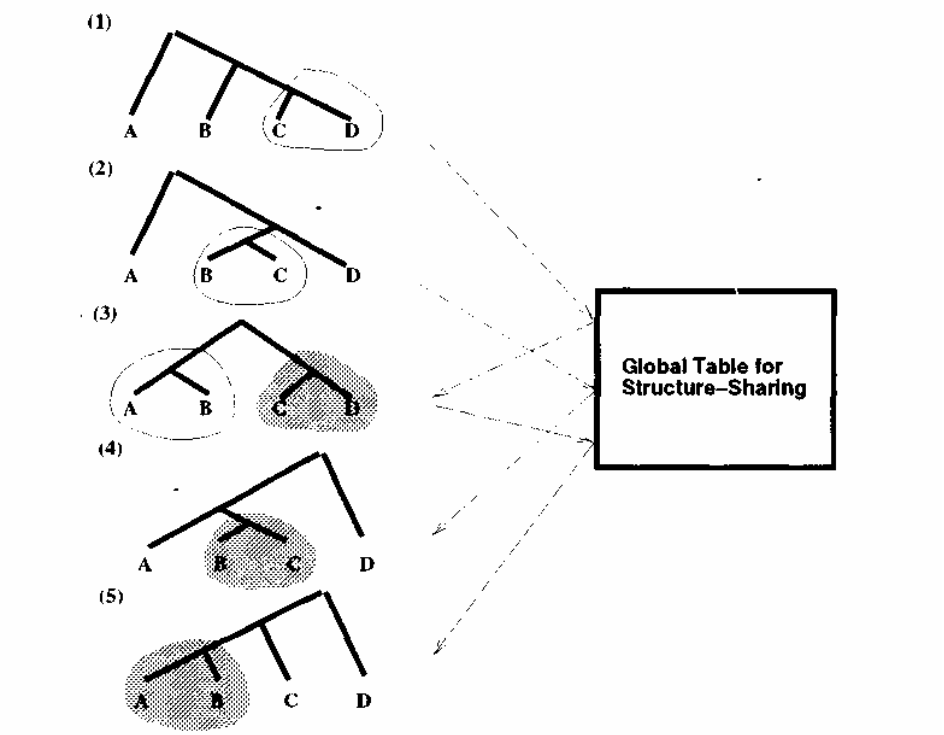

Figure 4: Structural Ambiguity of "A **no** B **no** C **no** D"

The current problem is similar to the Matrix-Chain Multiplication Problem (Cormen et al., 1990). Although naive implementation requires **exponential** time to the sequence length, memoizing the overlapping substructures[7] in a global table reduces the cost to a **cubic** time. However, it is, difficult for a sequential

[6] the same sequence is used throughout this paper because it is highly ambiguous, but the size of the ambiguity is easy to control by the length.

[7] In Figure 4, the substructures for "C **no** D" overlap in structures (1) and (3). In the same way, shaded substructures overlap surrounded substructures and should be shared to avoid unnecessary computation through the global table for structure-sharing.

machine to process a highly ambiguous sentence in a minute amount of time which is necessary for real-time response.

3.2. Example-Retrieval Cost

Example-Retrieval is slowed by two factors: structural ambiguity and the example database size. Structural ambiguity increases the number of Example-Retrieval calls; the example database size causes a problem even when the ambiguity is small. We focus on the second factor because **Example-Retrieval cost is linear to the example database size which is expected to be very large.** Let us estimate the example database size, *N*, for a large-scale system. In the prototype system, the vocabulary size is about 1,500 and *N* is 12,500. Assuming that *N* is in direct proportion to the square of the vocabulary size, then *N* rapidly increases. Hereafter, we take into consideration the case *N*=1,000,000, i.e. about 9*9 times larger than that of the prototype system. In the prototype using a SPARCstation2, the Example-Retrieval time for an average length sentence is about 2.5 seconds. The expected time for a large vocabulary system, at worst, is about 200 (= 2.5 x 1,000,000/12,500) seconds. This is clearly unacceptable because achieving a real-time response that does not disturb natural communication with a speech-to-speech translation system would be difficult.

4. Acceleration Using Heterogeneous Computing

As described in the previous section, Structure-Building and Example-Retrieval are dominant in Example-Based Approaches; however, they are, suitable for parallel processing in their own way. We propose accelerating them using different parallel devices, MIMD and SIMD processors. This section explains Structure-Building on MIMD, Example-Retrieval on SIMD, and communication between them and the overall performance.

Here, we sketch a basic distinction between MIMD and SIMD. SIMD supports data-parallelism where a processor is assigned to a data unit and all processors synchronously execute the same instruction. MIMD supports control-parallelism where a processor is assigned to a program unit and processors operate asynchronously and share access to a common memory. MIMD can simulate SIMD, but, in a less efficient way. We avoid such an inefficiency and form the optimal combination of these different architectures.

4.1. Structure-Building Acceleration on MIMD

We explain our parallelization of Structure-Building using the sample exemplified in Figure 4. The input sequence for Structure-Building is common data. Parallelism recurs when splitting the sequence at step (II) of the algorithm explained in Section 3.1. At the first recurrence, the sequence “A **no** B **no** C **no** D” is split into subsequences at three different positions, i.e., between A and B for structures (1) and (2), between B and C for structure (3), and between C and D for structures (4) and (5).

In the same way, parallelism rapidly increases when recurrence deepens and/or the input is long; however, the processors are finite. To manage this problem, first, a **pool of processors** are setup and a processor is **dynamically** assigned when needed. Structure-Building assigns a processor to process a subsequence when a processor remains; otherwise Structure-Building continues to process the subsequence by itself. When the processing of a subsequence is completed, the processor is returned to the pool and will be reused to process other subsequences.

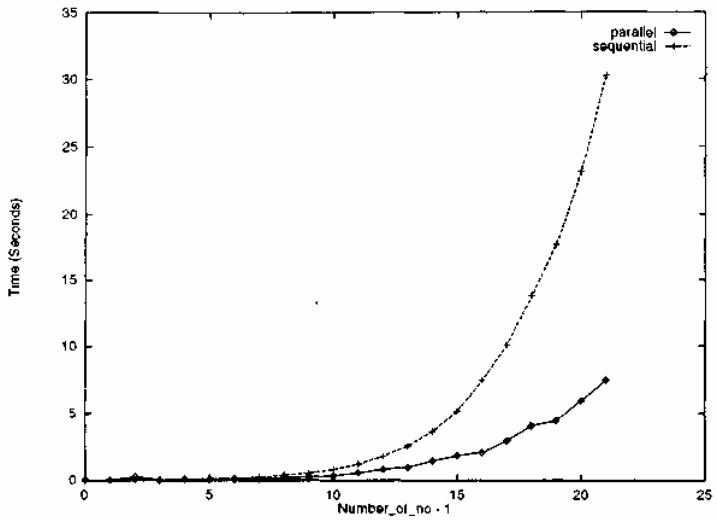

Figure 5: Speedup of Structure-Building over sequential implementation

The global table for structure-sharing is accessed by all processors; thus it is locked mutually exclusively when a new substructure is written in it. As the experimental result demonstrates, this overhead does not cancel the effect of parallelization. Figure 5 is an early report on the effectiveness of our parallelization.

4.2. Example-Retrieval Acceleration on SIMD

Example-Retrieval is best for SIMD because step [a], calculating the semantic distance, can be done by the same program independent of all data; and step [b], finding the minimum-distance example, can be swiftly done as a parallel reduction.

Moreover, we can accelerate Example-Retrieval by virtue of our definition of semantic distance. TDMT utilizes the **Semantic Distance Calculation** proposed in (Sumita and Iida, 1992b) to retrieve the best-match examples. The semantic distance between examples and input is reduced to the distance between words. Each word is assigned a *k*-digit *l*-scale code, which clearly represents the thesaurus hierarchy. In the thesaurus (Ohno and Hamanishi, 1984) used in our experiment, a 3-digit decimal code is assigned. The semantic distance between codes is calculated according to Table 1. Exhaustive search is speeded up by an indexing technique that suppresses unnecessary computation. We use thesaurus codes as an index of the example array. According to Table 1, if $CI_l \neq CE_l$ (most of the examples), we need not compute the distance because it is always 1; otherwise, we need to check the example in more detail to compute the distance between the input and the examples.

Example-Retrieval was first extensively studied on an Associative Processor, then as explained below, on several high performance machines.

Example-Retrieval Acceleration on Associative Processors An Associative Processor is the processing element of the massively parallel machine, IXM2 (Higuchi et al., 1991), which has shown that a large Associative Memory works effectively as a SIMD device for AI applications. The Associative Memory not only features storage operations, but also logical operations such as retrieving by content.

The first experiment was conducted on a single Associative Processor with

Table 1: Semantic distance between thesaurus codes.

Condition	Example	Distance
[c1] $CI_1CI_2CI_3 = CE_1CE_2CE_3$	**347** , **347**	0
[c2] $CI_1CI_2 = CE_1CE_2, CI_3 \neq CE_3$	**347** , **346**	1/3
[c3] $CI_1 = CE_1, CI_2 \neq CE_2$	**347** , **337**	2/3
[c4] $CI_1 \neq CE_1$	347 , 247	1

NB) $CI_1CI_2CI_3$(input) and $CE_1CE_2CE_3$(example).

1,000 examples of a single pattern (Sumita et al., 1993b). In the experiment, the Associative Processor outperformed several state-of-the-art high performance machines.

The second experiment was conducted on 10 Associative Processors connected in a tree configuration, with 12,500 examples of multiple patterns (Oi et al., 1994; Higuchi et al., 1994; Sumita et al., 1995). The algorithm implements Example-Retrieval by distributing examples onto multiple Associative Processors, retrieving examples on each Associative Processor in parallel, and merging all results. Example-Retrieval on multiple Associative Processors exhibits clear scalabilty: because the time for the semantic distance calculation, step [a], does not increase because calculations are independently done, and the communication time among Associative Processors, step[b], is only a few percent of the Example-Retrieval time because it is controlled by the tree depth and small coefficient[8], even if the example database size is increased by nearly two figures from the prototype system.[9]

Example-Retrieval Acceleration on different architectures We have compared several accelerations of Example-Retrieval with bulk (100,000) examples of a single pattern on different architectures, i.e., sequential, MIMD and SIMD (Sumita et al., 1994). By doing so, generally speaking, we can find the architecture response relationship that allows us to find the most suitable architecture for a given response time requirement. This paper discussed four accelerating strategies that take into consideration the performance of processor *M* (MIPS) and the number of processors p, to meet a goal of 100,000 examples per millisecond.[10] The discussion is based on the results obtained for three state-of-the-art machines; DEC alpha/7000, KSRl (KSR, 1992), and MP-2 (MasPar, 1992), which are representative of the three architectures - sequential, MIMD and SIMD - respectively. Strategy (1), i.e., to increase M, seems to hit a wall[11] because the speedup slope is already very small at 200 MIPS. Strategy (2), i.e., only to increase p, cannot go beyond the overhead because the speedup slope is already very small at 80 processors. Strategy (3), i.e., to increase p with higher M, is feasible if the overhead can be approximately decreased in inverse proportion to

[8] The transmitted data include only the semantic distance and the locations of minimum-distance examples; thus, the amount is very small.

[9] The example database size, 1,000,000, is 9*9 times larger than that of prototype system. The number of Associative Processors required to load an example database of this size is 800 and the tree depth is 6.

[10] Achieving this goal means that we have succeeded in accelerating Example-Retrieval to a sufficient speed much less than the utterance time.

[11] To attain the goal, we should have 20,000 MIPS. Increasing *M* to 20,000 MIPS, however, is hopeless because it will take considerable time as explained below. MIPS is increasing at the rate of about, 35% per year (Hennessy and Patterson, 1990), and we already have a 200-MIPS processor; thus, we can get a greater than 20,000-MIPS processor by 2010.

the increase in M.[12] As of now, only strategy (4), i.e., to increase *p* drastically at the expense of *M,* that is, SIMD, achieves our goal.

4.3. Communication and Total Performance

As explained so far, TDMT consists of Structure-Building. Example-Retrieval, and other processes. Structure-Building is suitable for MIMD, Example-Retrieval is best for SIMD, and the other processes may or may not match parallel processing.

Here, we concentrate on the communication between Structure-Building and Example-Retrieval. If the communication cannot attain a sufficient speed, we should be reconciled to another choice of using MIMD for both Example-Retrieval and Structure-Building, abandoning the best performance of Example-Retrieval on SIMD. Two machines should be connected through a standard network, e.g., ETHERNET, FDDI, and so on. Empirical studies have clearly revealed the following behavior of ETHERNET: (1) when the size of communicated data is under 1 K byte, the communication time is almost constant: about 10 milliseconds; (2) when the size is at 1 K byte, the communication time begins to increase; and (3) when the size is over 10 K byte, the time is proportional to the size. The size of data for a single Example-Retrieval call is at most tens of bytes because the parameters for Example-Retrieval are only the pattern and the bound words, and the return values of Example-Retrieval are only the semantic distance and the locations of minimum-distance examples. Therefore, packing multiple Example-Retrieval calls into a single communication whose size is under 1 K byte is the best solution for our purpose. Just like Structure-Building and Example-Retrieval, the communication depends on the ambiguity of the input. However, there is no fear that the communication will cause a bottleneck of a Heterogeneous TDMT because the communication consumes only a small amount of time as shown in the following experiments.

An early result of our small-scale experimental Heterogeneous TDMT is summarized in Table 2. The input is "dewa [*then*] kaigi [*conference*] **wa** [*sub*] 2 gatsu 18 nichi [*February* l8*th*] tsumari [*i.e.*] raishuu [*next week*] **no** [*of*] getsuyoobi [*Monday*] **kara** [*from*] hajimaru [*begin*] **ndesune** [*ending for confirmation*]," one of the slowest out of our 746 test sentences. This result does not look so striking because for an input and example database of this size, a 4.6 times faster sequential machine would achieve the same result. However, our goal is to establish a technology scalable to a large example database and/or a highly ambiguous sentence. This goal is difficult to achieve by the same algorithm on a sequential high performance machine. For example, with a 10-times larger example database, the parallel Example-Retrieval time does not increase; thus, parallel Example-Retrieval is about 127 times faster than sequential Example-Retrieval; With a 100-times larger example database, the parallel Example-Retrieval time is multiplied by about 1 only, thus, parallel Example-Retrieval is about 276 times faster than sequential Example-Retrieval.

Combining the results so far. we expect a total translation time that achieves a real-time response even with a highly ambiguous sentence and/or a large example database.

[12] The next generation of MIMD machines, or a cluster of high-performance workstations, will meet the requirement. Compared to KSR1, KSR2 which has 2-times faster processors, has performed Example-Retrieval 2-tirnes faster; however, our goal requires a 20-times faster performance.

Table 2: Total Performance (Seconds) of Small-Scale Prototype

	Sequential (KSR2)	Heterogeneous	Speedup
Structure-Building	0.84	(KSR2) 0.20	4.2
Example-Retrieval	4.20	(MP-2) 0.33	12.7
Communication	0	(ETHERNET) 0.13	N.A.
Total	5.56	1.22	4.6

5. Related Research

Speech-to-Speech Translation Challenging research on speech-to-speech translation began in the mid-1980s. Such research has brought about several prototype systems (Morimoto et al., 1993; Kitano, 1991; Waibel et al., 1991; Rayner et al., 1993; Hatazaki et al., 1992). However, no large-vocabulary system capable of responding in real-time has emerged. Speech-to-speech translation consists of three processes, i.e., speech recognition, spoken language translation and speech synthesis. There are two possible models for speech-to-speech translation: (1) a simultaneous model where processes start while overlapping each other; and (2) a sequential model where processes start after their preceding process completes. Unfortunately, state-of-the-art NLP technologies do not allow us to adopt the simultaneous model; thus, as we have shown in this paper each component in the sequential model should be accelerated as much as possible.

Massively Parallel Natural Language Processing Up to now, some systems using massively parallel machines in the field of natural language processing, such as a parsing system (Kitano and Higuchi, 1991b) and translation systems, e.g., ASTRAL (Kitano and Higuchi, 1991a), MBT3n (Sato, 1993b), have been proposed. They have demonstrated good performance; nonetheless, they differ from our proposal. For the first two systems, although they use Associative Processors, they use a different mechanism for their natural language tasks. They do not calculate semantic distance, but propagate markers through a semantic network. For the last system, it deals with a translation subproblem: translating not sentences, but noun phrases (technical terms). It uses a different mechanism based on matching and similarity on a MIMD machine.

In contrast, our proposal deals with sentence translation not by a single architecture, but by a Heterogeneous Computing Platform consisting of MIMD and SIMD machines.

6. Concluding Remarks

An Example-Based Approach using Heterogeneous Computing for spoken language translation has been proposed. According to our previous experiments, the translation quality of Example-Based Approaches is good. Heterogeneous Computing drastically accelerates the Example-Based Approach and gives it a desirable scalability against vocabulary size and ambiguity size. Consequently, an Example-Based Approach on Heterogeneous Computing meets the vital requirements for spoken language translation by breaking through the limitations of conventional technologies.

Example-Based Approach on Heterogeneous Computing has desirable features that will be suitable for integration with speech recognition: high accuracy, robustness, the output of a reliability score, and a quick response. Tightly coupling

our model and speech recognition might make real-time speech-to-speech translation possible in the future.

References

Cormen, Thomas H., Leiserson, Charles E., and Rivest, Ronald L. (1990). *Introduction to Algorithms.* The MIT Press.

Furuse, O. and Iida, H. (1992). "Cooperation Between Transfer and Analysis in Example-Based Framework". In *Proc. of COL1NG'92,* pages 645 651, July.

Furuse, O. and Iida, H. (1994). "Constituent Boundary Parsing for Example-Based Machine Translation". In *Proc. of COLING'9~4,* August.

Furuse, O., Sumita. E., and Iida, H. (1994). "Transfer Driven Machine Translation Utilizing Empirical Knowledge". *Transactions of Information Processing Society of Japan,* 35(3):414-425, March.

Hatazaki, K., Yoshida, K., Okumura, A., Mitome, Y., Watanabe, T., Fujimoto, M.. and Narita, K. (1992). "A Japanese-English bidirectional automatic interpretation system: INTERTALKER". In *44th convention of IPSJ ,* 6P-5, March.

Hennessy, John L. and Patterson, David A. (1990). *Computer Architecture: A Quantitative Approach.* Morgan Kaufmann.

Higuchi, T.. Furuya. T., Handa, K., Takahashi, N., Nishiyama, H., and Kokubu, A. (1991). "IXM2 : A Parallel Associative Processor". In *Proc. of the 18th International Symposium on Computer Architecture,* May.

Higuchi, T., Handa, K., Takahashi, N., Furuya, T., Iida, H., Sumita, E., Oi, K., and Kitano. H. (1994). "The IXM2 Parallel Associative Processor for AI". In *IEEE Computer,* pages 53-63, November.

Kitano, H. and Higuchi, T. (1991a). "High Performance Memory-Based Translation on IXM2 Massively Parallel Associative Memory Processor". In *Proc. of AAAI'91,* volume 1, pages 149-154, July.

Kitano, H. and Higuchi, T. (1991b). "Massively Parallel Memory-Based Parsing". In *Proc. of IJCAI'91,* pages 918-924.

Kitano, H. (1991). "ΦDM-Dialog: An Experimental Speech-to-Speech Dialog Translation System". *IEEE Computer,* 24(6):36-50, June.

KSR. (1992). *KSRI Technical Summary.* Kendall Square Research Corp.

MasPar. (1992). *The Design of the MasPar MP-2 A Cost Effective Massively Parallel Computer.* MasPar Computer Corp.

Morimoto, T., Takezawa, T.. Yato, E., Sagayama, S., Tashiro, T., Nagata, M., and Kurematsu. A. (1993). "ATR's Speech Translation System: ASURA". In *Proc. of EUROSPEECH'93,* pages 1291 1294, September.

Nagao, M. (1984). "A Framework of a Mechanical Translation between Japanese and English by Analogy Principle". In Elithorn, A. and Banerji, R., editors, *Artificial and Human Intelligence,* pages 173-180. North-Holland.

Ohno, S. and Hamanishi, M. (1984). *Ruigo-Shin-Jiten.* Kadokawa.

Oi, K., Sumita, E., Furuse, O., Iida, H., and Higuchi, T. (1994). "Real-Time Spoken Language Translation Using Associative Processors". In *Proc. of 4th ANLP,* volume 1, pages 101-106, October.

Rayner, M., Alshawi, H., Bretan, I., Carter, D., Digalakis, V., Gamback, B, Kaya, J., Karlgren, J., Lyberg, B., Pulman, S., Price, P., and Samuelsson, C. (1993). "A Speech to Speech Translation System Built from Standard Components". In *Proc. of the Workshop on Human Language Technology,* pages 217-222. ARPA, March.

Sato, S. (1991). *Example-Based Machine Translation.* Ph.D. thesis, Kyoto University, September.

Sato, S. (1993a). "Example-Based Translation of Technical Terms". In *Proc. of the TMI'93,* pages 58-68, August.

Sato, S. (1993b). "MIMD Implementation of MBT3". In *Proc. of the Workshop on Parallel Processing for Artificial Intelligence,* pages 28-35. IJCAI'93, August.

Sobashima, Y., Furuse, O., Akamine, S., Kawai, J., and Iida, H. (1994). "A Bidirectional, Transfer-Driven Machine Translation System for Spoken Dialogues". In *Proc. of COLING'94,* August.

Sumita, E. and Iida, H. (1992a). "Example-Based NLP Techniques - A Case Study of Machine Translation". In *Proc. of Statistically-Based NLP Techniques Workshop (AAAI'92),* August.

Sumita, E. and Iida, H. (1992b). "Example-Based Transfer of Japanese Adnominal Particles into English". *IEICE TRANS. INF. & SYST.,* E75-D(4):585-594, April.

Sumita, E., Furuse, O., and Iida, H. (1993a). "An Example-Based Disambiguation of Prepositional Phrase Attachment". In *Proc. of TM1'93,* pages 80-91, July.

Sumita, E., Oi, K., Furuse, O., Iida, H., Higuchi, T., Takahashi, N., and Kitano, H. (1993b). "Example-Based Machine Translation on Massively Parallel Processors". In *Proc. of IJCAI'93,* volume 2. pages 1283 1288, August.

Sumita, E., Nisiyama, N., and Iida, H. (1994). "The Relationship Between Architectures and Example-Retrieval Times". In *Proc. of AAAI'94,* volume 1, pages 478-483, August.

Sumita, E., Oi, K., Furuse, O., Iida, H., and Higuchi, T. (1995). "Example-Based Machine Translation using Associative Processors". *Journal of Natural Language Processing,* 2(3).

Waibel, A., Jain, A., McNair, A., Saito, H., Hauptmann, A., and Tebelskis, J. (1991). "JANUS: A Speech-to-speech Translation Using Connectionist and Symbolic Processing Strategies". In *Proc. of ICASSP'91,* volume 2, pages 793-796, May.

Appendix I: Sample Spoken Sentences Handled by TDMT

Here, we show some variations of "request" sentences in our test corpus.

- hoteru o shookai shite itadakemasuka. ⇒ Would you recommend me a hotel?
- nittei nitsuite oshiete itadakitainodesuga. ⇒ I would like you to tell me about your schedule.
- saido shorui o o-okuri negaemasuka. ⇒ Would you please send me the papers again?
- denwa-bangoo o o-negaishimasu. ⇒ What's your phone number, please?
- ginkou-furikomi de o-shiharai kudasai. ⇒ Please pay by bank transfer.
- shiryoo o okutte hoshiinodesuga. ⇒ I would like you to send me the materials.
- o-namae o osshatte kudasai. ⇒ Could you please tell me your name?

Appendix II: Specifications of Machines Used in the Experiments

The feasibility of accelerating Example-Retrieval was studied by an experiment conducted with an IXM2, consisting of 10 Associative Processors connected in a trinary tree. As specified in Table 3, each Associative Processor has a 4K-word Associative Memory and an INMOS T801 Transputer. The Associative Memory is mapped into the memory space of the Transputer. A single Associative Processor allows 4K search/write operations in parallel.

Table 3: Components of an Associative Processor

• 4 K x 40 bit Associative Memory • INMOS T801 Transputer, 12.5 MIPS 4 serial links, 10 Mbits/sec

The following experiments were conducted using several state-of-the-art machines (sequential, MIMD and SIMD machines) which are shown[13] in Table 4.

Table 4: Specifications of Other Machines

Machine	*M*	*p*	*M*p*	*Architecture*
DEC alpha/7000	200	1	200	Sequential
KSR1	40	80	3200	MIMD
KRS2	80	30	2400	MIMD
MP-2	4	8000	32000	SIMD

[13] *All figures here are rounded off to simplify the comparison.*