

A New KE-Free Online ICALL System Featuring Error Contingent Feedback

Naoyuki Tokuda
Research & Development Center
SunFlare Company
Tokyo, Japan
E-mail: tokuda_n@sunflare.co.jp

Liang Chen
Computer Science Department
University of Northern British Columbia
Prince George, B.C., Canada
Email: chenl@unbc.ca

Abstract

As a first step to implementing a human language teacher, we have developed a new template-based online ICALL (intelligent computer assisted language learning) system capable of automatically diagnosing free-format translated inputs of learners, returning error contingent feedback. The system architecture adopted allows language teachers to build their expertise into the system by themselves without help from KEs (knowledge engineers), thus solving long-standing KE bottlenecks of conventional expert-systems-based ITS (intelligent tutoring system) or ICALL (Murray 1999). The core of the system comprises a unique FSA (finite state automaton)-based template knowledge base system, a robust and global HCS (heaviest common sequence)-based diagnostic engine, a POST(part-of speech-tagged) parser and related learners' model and an easy-to use VTAT (visual template authoring tool). To simplify the authoring task of often quite complex template patterns, we have developed two sets of simpler rules; the first group of rules allows language teachers to manipulate with ease the complex sentence patterns by constructing a template-template representation from which many separate templates can be extracted, and the second group of buggy rules can be used to generate syntactic bugs for learners automatically by replacing part of the correct syntactic rules with plausible buggy rules. Using study participants' responses extracted into the system templates, we present some convincing experimental verifications that the diagnostic engine is capable of providing error-contingent feedback and diagnosis applicable to a wide range of learners having various educational backgrounds.

Keywords: KE-Free ICALL, Longest Common Sequence Algorithm, Finite State Template Automaton, NBLT (network based language teaching).

I. INTRODUCTION

A. Increasing Need for Second Language Learning in Information Age

The increasing importance of language learning is accelerated by the dramatic spread of the Internet and accompanying globalization of every sector of our community. Since language is an essential means of communication, some popular language such as English might imminently emerge as a common language. An interesting insight emerges, however, if we look from the viewpoint of international language use; the recent statistics of IDC (<http://www.idc.com/>) shows that some 82% of Internet servers were in English, but this is changing very quickly. In fact, some recent estimates imply that the percentage of Web sites in English is already down to 54%, and is expected to drop further to a 30% level by the year 2005. The statistics show that a growing number of people throughout the world now use English as a second language, thus shifting the dynamics of the English language, devolving from the first language speakers to the second and foreign language speakers all around the world. For the first time in history, the number of the second language speakers of a language dominates that

of the first language speakers (Warschauer, 1999), confirming an ever-increasing need for language learning worldwide.

When regarded as a communicative means, demand for second language learning from the Internet is more far-reaching and innovative in its scope than merely an acute need for acquisition of languages. This is so because the language is built into what we call literacy, which implies mastering processes that are deemed valuable in particular societies, cultures, and contexts. In this globalized information age, so-called global literacy (Kawai, 2001) now must integrate IT (information technology) literacy into the conventional literacy that is dominated by the language-oriented communicative skills needed in a global community. The latter literacy requires language skills to “persuade” interested people, regardless of the languages being used. This implies that the CALL system under NBLT (network based language teaching) is a prerequisite to language learning as well. Warschauer (Warschauer, 1999, 2002), as well as Tokuda (Tokuda, 2002), examines language learning development in the Internet age from the points of view of literacy, international language use, and second language learning, and also from the digital divide point of view.

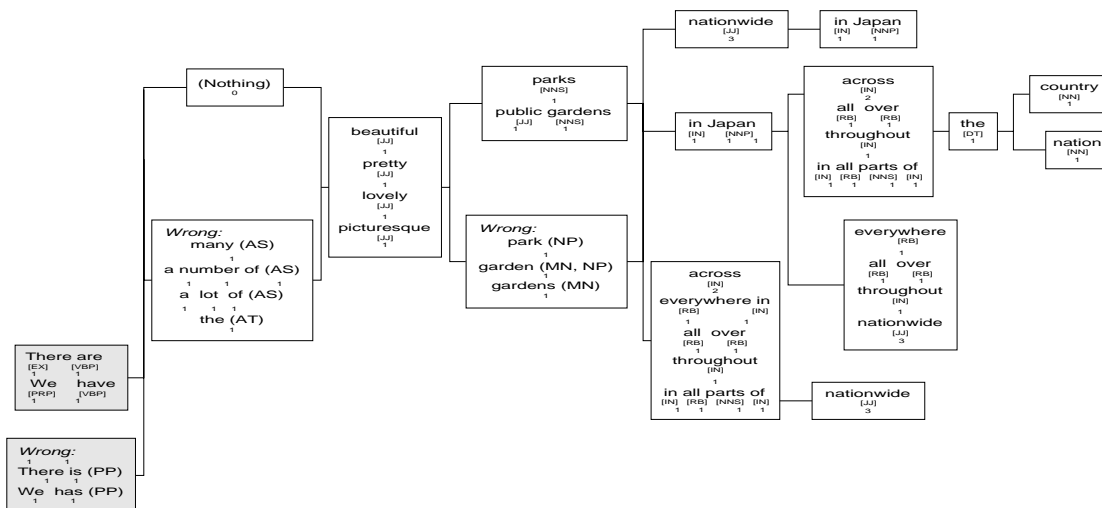
B. Difficult Tasks To Be Solved in ICALL

Starting from the oldest, mainframe-based PLATO developed in the 1960 at the university of Illinois, language teaching has a long history, compiling decades of human intelligence. Murray (Murray, 1999) gives an extensive review on authoring intelligent tutoring systems, including ICALL development, in the Internet infrastructure.

The basic construction scheme of most ITSs is based on that of the so-called expert system, where the role of KEs (knowledge engineers) is indispensable. It is true that more easy-to-use, efficient shells and authoring tools are developed to enable more nonprogrammers of domain experts to participate actively in system construction, but the labor intensity, which is increasing with the advancing level of intelligent systems, including knowledge representation, acquisition and interface construction, makes this implementation not only very difficult, but also prohibitively expensive. This difficulty is compounded in our domain of language teaching because we have to teach or learn natural languages. Unlike artificial programming languages, the substantial difficulty of a natural language can be tracked down to its inherent features, such as context sensitive grammar (Aho & Ullman, 1992; Allen, 1995; Shieber, 1986) and word sense disambiguation (Yarowsky, 1995). In addition to these difficulties, the ICALL must often deal with the additional difficulties of processing and diagnosing the ill-formed sentences of non-native students. Non-native students make unexpected errors, with which even the most powerful natural language processing systems fail to cope. We solved the problem by applying the well-established NLP tool for well-formed sentences, which regards differences between a student input and the nearest well-formed sentence as errors made by learners. See (Chen, Tokuda & Xiao, 2002) for detailed discussions. .

C. KE-free Scheme of Knowledge Representation and Acquisition

Several ICALL are built for training writing skills. Some are restricted to simple grammar aspects, such as spelling, punctuation (Sentence, 2000) or passive form writing (Virvou et al., 1997). More general systems are built for a German tutor (Heift et al., 2000), or for setting up an intelligent error diagnosis (Kruger and Hamilton, 1997). Some systems help learners by providing the overall text composition so that they help to structure a text and to organize it according to different schemes (Yang and Akahori, 2000). In most classical ITSs, knowledge is represented by rules, while knowledge is acquired from the vast search space of knowledge bases by exploiting many search techniques (Huang & Tokuda, 1996). Knowledge bases are constructed for representation and acquisition by KEs, who in turn consult closely with 'domain experts' (or experienced language teachers in our case) to extract expertise. As AI techniques advance more, the cost and difficulties involved in 'rule generation and solution space search techniques' often soar prohibitively (Murray, 1999), with the cost of KEs becoming a bottleneck



Error Messages:

- AS: an assumption has been made on the quantity of the noun
- AT: the article is not needed
- CM: a comma is needed
- CT: contraction is incorrect
- MN: meaning is incorrect
- NP: noun must be plural
- VS: verb must be singular, since subject is singular
- PR: preposition is incorrect
- PP: phrase must be plural

Figure 1 Example Template

B Error Classification and Error Messages

Detailed error classification and contingent error-specific messages are vital for remedial purposes and hence for completing an effective ICALL.

Errors are classified into 1) grammatical errors, 2) word and usage errors and 3) non-preclassified errors.

1) Grammatical errors

They are grouped into the following 9 subgroups having a similar hierarchical structure ending up with a total of 164 errors at the leaf level:

- | | | | | |
|--------------|---------------|---------------|----------------|------------|
| 1 Determiner | 2 Noun | 3 Punctuation | 4 Subject-verb | 5 Modifier |
| 6 Tense | 7 Preposition | 8 Negation | 9 Conjunction | |

2) Word and usage errors

They are classified into the following 9 groups:

- | | | | | |
|---------------------|-----------------|---------------|-----------------|--------------|
| 1 Wrong word | 2 Awkward word | 3 Misspelling | 4 Wrong meaning | |
| 5 Technically wrong | 6 Singular form | 7 Plural form | 8 Sexist | 9 Colloquial |

3) Non-preclassified errors.

Obviously we cannot deal with all of the possible errors made by students, even by our template mechanism. They can be grouped into the following three types:

- (1) Missing words: a part of the template is missing from the input sentence.
- (2) Extraneous words: a part of the input sentence is not included in the template.
- (3) All other non-classifiable errors.

A new classification called “non-preclassified errors” here adds flexibility to the system, providing an appropriate remediation. Obviously many of them can be regrouped and reclassified into new grammatical errors or word and usage errors, which obviously enhance the effective pedagogical tutoring. We discuss how we can turn non-preclassified errors into classified errors in section III-B on buggy rule generated templates.

To further reduce the possibility of useless confusion, we first pre-process misspelled words not available in dictionaries. Our basic strategy for misspelled words is the following:

1. When a word(s) is encountered in a student’s input sentence which is not found in the dictionary, use a Spellcheck module developed by a longest common sequence algorithm (Cormen, Leiserson & Rivest, 1990).
2. Find plausible candidate words from among all possible words of the templates having the first three longest common sequences.

3. Present the three most probable candidates of misspelled words for students to choose.

We have chosen to present the top candidate to the students, and this simple method seems to work very well.

The system performance depends critically on how accurately errors can be diagnosed and the error contingent messages can be presented to the learners. Compared with about 80 bugs of the Buggy Model (Huang & Tokuda, 1996), a total of about 170 distinct errors are used. As we show in section IV on system evaluation, the system works remarkably well when tested against control groups having an entirely different educational background from the study participant groups of the current template databases.

C. Heaviest Common Sequence Matching Algorithm

Our knowledge acquisition process exploits the template matching algorithm whereby an optimal pattern is sought from among all the paths formed within the template having the greatest *similarity* with a student's input sentence. Here similarity is measured by the heaviest or longest common sequence (Cormen, Leiserson & Rivest, 1990) between a student's input sentence and the possible template paths selected. The latter is a special case of the former when the weights of words are uniform among all words.

A common sequence of two sentences is defined as an ordered sequence of words, a_1, a_2, \dots, a_m , which appear in both of the sentences in an order of a_1 , then a_2 , then ..., then a_m . The definition of common sequence can be found in the book *Foundations of Computer Science* by A. V. Aho and J. D. Ullman (Aho & Ullman, 1992, pp. 321-327).

As each word or phrase in the template is assigned with weights, the heaviest common sequence of a path in the template and an input sentence is defined as the common sequence among all the possible common sequences whose total weights are the largest.

We now search for a heaviest common sequence in words and/or phrases of an input sentence from among all the possible valid paths of the template.

The heaviest common sequence of a template and an input sentence is defined as the sequence of words which has the heaviest total weight among the heaviest common sequences, each of which is obtained from one path of the template and the input sentence. We will show now how our heaviest common sequence template-matching algorithm works on input sentences and the FSA equivalent template structures.

The input to the system is a sentence consisting of a sequence of words, and the input is compared against a template comprising a finite set of sentences specified by a regular expression. A syntactically legal sequence of words in the regular expression is in fact a sentence of the template, and is called a path of the template. The words of the template can be associated with a positive real number called a weight of the word representing the relative importance of the word within the sentences.

We now search for a common sequence in words of the sentence from among all the possible valid paths of the template. The longest common sequence is defined as the one having the largest number of the same words between the input sentence and the candidate path of the template. The weight of a common sequence is defined as the sum of the weights of all the words in the sequence. An algorithm to find the heaviest common sequence of a weight is called a heaviest common sequence algorithm, which is a modification to the longest common sequence algorithm (Cormen, Leiserson & Rivest, 1990).

The problem of finding an optimal path can be described as follows: Given the sentence and the template, find one path such that the weight of the heaviest common sequence of the path and the sentence is not less than that of any common sequence comprising any path of the template and the sentence. To be precise, we follow the following rule in choosing the final path candidate: Should two or more paths have the same weight in HCS, we choose the one having the least error weight; for the same error weight, we choose the one having the shortest path length; for the same length, we should choose the one having the fewest erroneous blanks. Details of the algorithm can be found in (Chen & Tokuda, 2003). The time complexity of this matching algorithm is $O(MN)$, where M represents the number of the arcs of the template and N represents the number of words of the sentence.

Let us work with Figure 1, which contains the templates for the translation of the following sentence: "Japan is dotted with beautiful gardens nationwide." Suppose an input sentence from

a student is: “In Japan dotted with lovely public gardens through nationwide.” The heaviest common sequence algorithm will find the path “Japan is dotted with lovely public gardens nationwide” as an optimal path from the templates of Figure 1. The heaviest common sequence of the path and the sentence is: “Japan .. dotted .. with .. lovely .. public .. gardens .. nationwide.”

By choosing this sentence as the most probable target sentence which a student has intended to type in, all the subsequent diagnosis and remedial procedures are considerably simplified, because all errors are now classified and contingent comments and repair schemes are specified. We think this global matching scheme of a heaviest common sequence is essential to our approach. The local matching scheme, on the other hand, can not process the case because there is not even a closely matching pattern to start tutoring.

III. SIMPLIFYING THE TEMPLATE CONSTRUCTION BY RULES

Developing a CALL system requires extremely complex processes far beyond the capability of conventional KE-dependent expert systems. Exploiting the power of FSA-based templates and easy-to-use authoring tools, we have developed a KE-free CALL system whose quality, however, depends strictly on the templates developed by the language teachers. To help the language teachers to develop good quality teaching contents, we will present below a new scheme for facilitating the otherwise labour-intensive template construction. Two different sets of rules are developed below. The first set of rules can be used by language teachers directly to construct the template-template concept, from which a set of templates can easily be extracted; the second set of rules helps language teachers to build non-pre-classified bugs into the templates, and in fact the rules may be kept running in “background.” We are implementing this system into our AZALEA system.

A. Template-Template Construction Rules

When possible erroneous expressions are included, the possible number of L2 (English) translations of one single L1 (Japanese) sentence is huge if we do not impose some constraints. To restrict the number of model translations, we ask the learners to use the key words they have learned in the lesson, so that at least the number of model translations and, hence, templates is reasonably constrained.

A template-template scheme is defined as an extended template, of which some of the nodes are assigned with rule-associated special symbols, allowing the template-template scheme to extract one or more of the templates as defined in the preceding sections.

Typically, a rule is related to a set of symbols, say $\{s_1, s_2, \dots, s_n\}$, of which each of the symbols is assigned to one or more nodes in the templates. These symbols are allowed to be assigned to one or several values, representing the style of the nodes that will appear in a template or templates extracted from the template-template. We call these symbols “label symbols.” The symbols related to a single rule are called “related symbols.” Related symbols should have certain restrictions, say that when $s_i=1$, s_k might have to be 2, for example. At the time when the value of one symbol s_i is decided by the values assigned to the set of other symbols, the choosing of the value of this symbol s_i is called the required choice of the other symbols.

Consider the following rules, which are simple enough for language teachers to understand:

Type A Rule: AP (appear)-NAP (not appear) Rule

In the template-template representation, suppose some nodes are marked with AP_i and some other nodes are marked with NAP_i (i being any integer, representing different Type A Rules). The rule imposes the condition such that a new expanded template can include either the nodes marked with AP_i or the nodes marked with NAP_i , but not both of them at the same time. We use $AP_i=0$ to denote the case in which the nodes marked with AP_i do not appear in a template. At this time, NAP_i have to be 1, meaning that the nodes marked with NAP_i will appear in the template. Thus, we can say that $NAP_i=1$ is the required choice of $AP_i=0$. With the same reason, when $NAP_i=0$, AP_i will have to be 1; and we can say that $NAP_i=0$ is the required choice of $AP_i=1$.

Type B Rule: PPR (personal pronoun) -PPRP (personal pronoun possessive) Rule

As in the Type A Rule, the template-template representation imposes the condition for nodes marked with PPR_i and other nodes marked with PPR_{Pi} (i being any integer) that there is a set of templates in each of which one personal pronoun appears in the nodes marked with PPR_i , and the corresponding possessive form of the pronoun appears in the nodes marked with PPR_{Pi} . The values of PPR_i and PPR_{Pi} would be chosen from the personal pronoun and personal pronoun possessive forms of pronouns. The required values of PPR_i when PPR_{Pi} is given, or the required values of PPR_{Pi} when PPR_i is given, can be defined by the natural language grammar of pronouns.

Type C Rule: AN (arbitrary number) Rule.

The Type C Rule imposes the condition that the nodes marked by AN can be chosen to be any arbitrary positive real number, i.e., AN_i can be any positive real number.

Type D Rule: CHO (choosing-one) rules.

The Type D Rule imposes the condition that at most one node marked by CHO_{ij} among all the nodes marked as $CHO_{i_1}, CHO_{i_2}, \dots, CHO_{i_k}$ (each different i representing a different set of nodes following the choosing-one) can appear in any of the templates extracted from the template-template. When $CHO_{ij}=0$, we are denoting that the node marked does not appear, while $CHO_{ij}=1$ means it does appear. Obviously, while one CHO_{i_k} is assigned as 1, all the other CHO_i must have values = 0.

We should note that the HCS matching algorithm should not be tried on the extracted templates independently, as we have done before. It is most efficient to organize the routes of paths for matching to the template-template construction directly. A considerable saving is expected in both temporal and special computational complexity by using the rules discussed.

B. Buggy Rule generated templates

The simple rules of the preceding section are effective in introducing a simpler representation of otherwise complex templates, which in turn leads to reduced space and time savings in computation. To further facilitate reduction of the labour-intensive burden of the language teacher, we introduce the buggy rules-based template construction scheme below.

A buggy rule here is used to represent a way of generating common erroneous expressions. A buggy rule is in the form of:

$$H_1H_2\dots H_N \rightarrow R_1R_2\dots R_M,$$

where $H_1H_2\dots H_N$ represents a set of nodes in a correct path of any template-templates, or a set of grammatical part-of-speech tags representing a basic component of a correct expression.

$R_1R_2\dots R_M$ is the set of nodes which represent a typical erroneous expression whose correct form is $H_1H_2\dots H_N$.

Here are examples of simple buggy rules:

EX VBP \rightarrow EX VBZ

PRP VBP \rightarrow PRP VBZ

NNP VBZ \rightarrow NNP VBP

(Here EX—Existential, PRP—Personal pronoun, VBP—Verb for 1st and 2nd person present, VBZ—Verb, 3rd person singular present)

Simply apply the above buggy rules and the template-template construction rules in section III.A. We should be able to expand the following template-template representation (Figure 2) into a “fatter” template-template, and then extract the templates of Figure 1 from it.

This example shows that a language teacher could ignore some very common errors when he/she is constructing the templates, since buggy rules will then be able to expand the template so as to include these erroneous expressions.

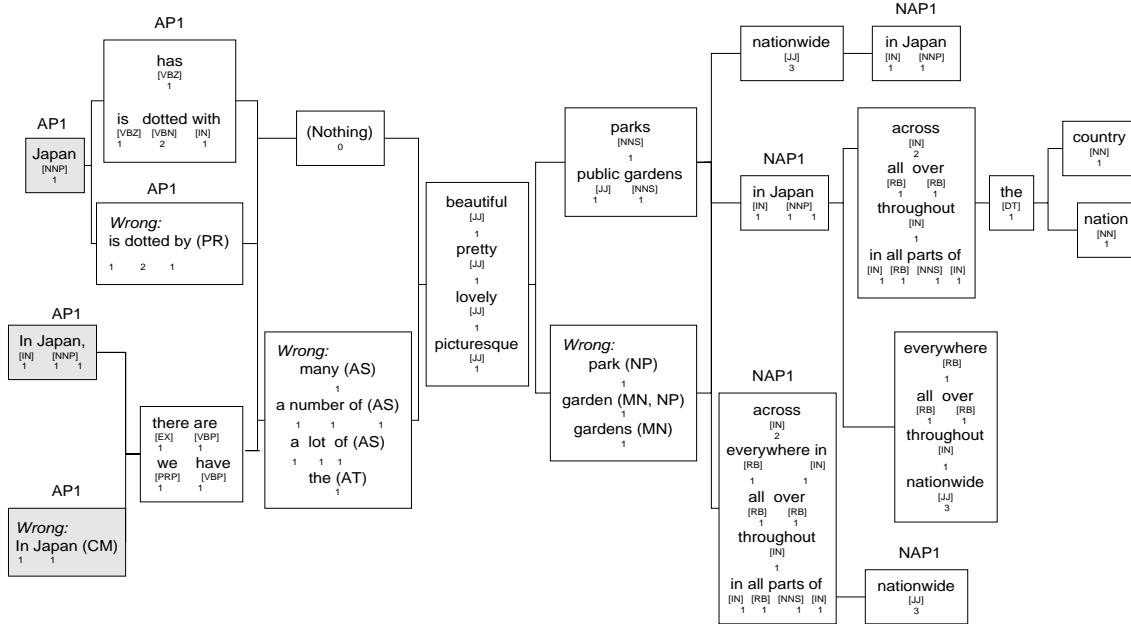


Figure 2. Buggy Rules-Generated Simplified Template-Template Construction

Because the buggy rules used here are independent of the tutoring courses or of the number of learners involved, they can be generated automatically with the program running in background. While they can be constructed manually, the buggy rules can also be effectively obtained from a frequency table of the minimum error subtree used as a part of the learners' model for a POST (part-of-speech-tagged) parser (see Tokuda, Chen and Sasai, 2002), US Patent Application No.10/072,954 and (Chen, Tokuda & Xiao 2002).

C. Template-template matching algorithm

As is evident from the discussion on single template-template of the preceding sections, we are able to extract many templates. Once the template-template has been obtained in an application involving a language translation tutoring system, the next step is to match an input sentence to each of all the possible templates, and then choose the closest path. We realize, however, that we are able to find a closest path from among all the valid paths of the templates that could be extracted from template-template with extracting rules (but without buggy rules) directly, without physically extracting all the templates from the template-template. We need to expand the buggy rules-embedded template-template first so that the template-template does not include any buggy rules before such a matching takes place.

Suppose a template-template has label symbols s_1, s_2, \dots, s_n to be associated with certain nodes of the transitional diagram; we know that the different templates extracted from the template-template can be obtained by assigning these symbols to nodes. We are able to denote each template extracted from the template by an n -tuple $\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}$, where the p_i 's are proper assignment to the symbols s_i . As has been discussed in an earlier section, p_i 's can be either numbers or words in accordance with the extracting rules used. The first step needed in the algorithm is to convert a template-template into its dual figure of an acyclic weighted finite digraph (directed graph), by simply expressing each node of the template as one or several arcs in the graph, adding arcs labeled ϵ with 0 weight for each empty node where applicable. Since the digraph is converted from the template-template, it contains many arcs associated with label symbols whose functions depend critically on the values assigned to the symbols. Accordingly given one such digraph, a completely different template can be extracted if a different set of label symbols is assigned to the arcs. That is to say, given such a digraph, we are able to obtain many digraphs, each of which corresponds to a template that can be extracted from the template-template. We hereafter call a digraph extracted from a template-template a "template-digraph."

We now define a procedure for finding the heaviest common sequences from among the common sequences of the paths of all the digraphs and an input sentence below.

The heaviest common sequence of the paths ended by any special node N in a digraph and an input sentence is defined as the sequence of words which has the heaviest total weight among all the heaviest common sequences, each of which is obtained from one path ended by N of the digraph and the input sentence.

Furthermore, we let $N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}$ represent the paths of all the digraphs extracted from a template-digraph, but ending at node N_i , where the symbol s_i is assigned with value p_i ($i=1, 2, \dots, n$). We call an n-tuple $\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}$ a label of node N_i . Here we should assume there is no contradiction of the rules when we set s_1 as p_1 , s_2 as p_2 , ..., s_n as p_n . We call such a label $\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}$ a contradiction-free label.

The heaviest common sequence of a node labeled $N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}$ and an input sentence is defined as the sequence of words which has the heaviest total weight among all the heaviest common sequences, each of which is obtained as the heaviest common sequence of one digraph extracted from the digraph-template with the nodes marked with the contradiction-free label $\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}$.

Notice that some nodes can not appear simultaneously in one digraph extracted from a digraph-template, as in a node labeled AP2 and a node labeled NAP2, in one digraph. As the result, a rule-breaking label such as $N_i(\dots, AP2, 1, \dots, NAP2, 1, \dots)$ should not be allowed in any calculation scheme of the common sequences of a node in the digraph-template and an input sentence.

The following algorithm describes the procedure for calculating the heaviest common sequence of a template-template and an input sentence. In the following calculation, “ λ ” is used as a very special value of the label symbols, whereby its value remains unspecified up to a certain stage of the calculation:

1. Turn the template-template into a template-digraph, of which the directed edges (transitions) are labelled by the corresponding words in the template.
2. Topologically sort all the nodes of the digraphs into N_1, N_2, \dots, N_t , such that for each pair of nodes N_i and N_j , there is no transition from N_j to N_i , when $j > i$.
3. Add an empty node N_0 to the digraph, and add an arc from N_0 to all the starting nodes in the template-template.
4. Set $CM(N_0, M_0) = 0$.
5. For $i=0$ to t , do the following steps:
6. If there is at least one arc to N_i which is associated with a symbol, and the s related labels would never exist after the node N_i , then for all i 's, for $j=0$ to m , do:
Check all $CM(N_i\{\dots\}, M_j)$, for any label $\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}$, if no s-related label appears in $\{s_1, s_2, \dots, s_n\}$ and there exist at least an $\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, sx_1, px_1, sx_2, px_2, \dots, sx_h, px_h\}$, where sx_1, sx_2, \dots, sx_h are s-related labels, such that $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, sx_1, px_1, sx_2, px_2, \dots, sx_h, px_h\}, M_j)$ is defined. Define $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_j)$ as the largest $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, sx_1, px_1, sx_2, px_2, \dots, sx_h, px_h\}, M_j)$, and undefine all the $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, sx_1, px_1, sx_2, px_2, \dots, sx_h, px_h\}, M_j)$ that had already been defined.
7. For $j=0$ to m , do the following steps:
8. For each of the nodes N_k to which there is an arc from N_i , do the following:
 - (1) If arc $N_i N_k$ has no label, check all the $CM(N_i\{\dots\}, M_j)$, $CM(N_i(\dots), M_{j+1})$, $CM(N_k\{\dots\}, M_j)$, $CM(N_k(\dots), M_{j+1})$ that have been defined. Define $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_{j+1})$ as the maximum of the following data if one of $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_j)$, $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_{j+1})$, $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_j)$, $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_{j+1})$ has already been defined:
 - $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_j)$ if it has already been defined
 - $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_j) + W(N_i N_k)$ if $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_j)$ had already been defined and the arc $N_i N_k$ is matched with W_k
 - $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_{j+1})$ if it has already been defined
 - $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_j)$, if it has already been defined

- $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_{j+1})$ if it has already been defined
- (2) If arc $N_i N_k$ is associated with symbol s , check all the $CM(N_i(\dots), M_j)$, $CM(N_i(\dots), M_{j+1})$, $CM(N_k(\dots), M_j)$, $CM(N_k(\dots), M_{j+1})$ that have been defined.
- (i) If the node label $\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, \lambda\}$ is a contradiction-free label, and at least one of the following has already been defined:
- $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_j)$, $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_{j+1})$,
 - $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_j)$, $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_{j+1})$,
 - $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, \lambda\}, M_j)$, $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, \lambda\}, M_{j+1})$,
 - $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, \lambda\}, M_j)$, $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, \lambda\}, M_{j+1})$;
- Define $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, \lambda\}, M_{j+1})$ as the maximum from one of the above defined data.
- (ii) Given label $\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}$, suppose that the label $\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, p\}$ is a contradiction-free label, and that at least one of the following is true:
- $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_j)$ has been defined and setting s to p is a required choice of $\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}$ or $N_i N_k$ is matched to M_{j+1} after p is assigned to s .
 - $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_{j+1})$, has been defined and setting s to p is a required choice of $\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}$.
 - $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_j)$, has been defined and setting s to p is a required choice of $\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}$.
 - $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, p\}, M_{j+1})$, has been defined.
 - $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, p\}, M_j)$, has been defined.
 - $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, p\}, M_{j+1})$, has been defined.
 - $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, p\}, M_j)$, has been defined.
 - $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, p\}, M_{j+1})$ has been defined.
- Define $CM(N_k\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, p\}, M_{j+1})$ as the maximum from one the above defined data and the following data:
- $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_j) + W(M_{j+1})$ if $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n\}, M_j)$ is defined and $N_i N_k$ is matched to M_{j+1} after p is set to s .
 - $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, p\}, M_j) + W(M_{j+1})$ if $CM(N_i\{s_1, p_1, s_2, p_2, \dots, s_n, p_n, s, p\}, M_j)$ is defined and $N_i N_k$ is matched to M_{j+1} after p is assigned to s .
- And also change those s_i from λ to a required choice of the value, subject to the condition of p 's being assigned as s .

Among all the $CM(N_x, M_m)$ already defined with N_x being a final vertex, the largest $CM(N_x, M_m)$ will be the weight of the heaviest common sequence of the template-template and the path. Notice that, in the above algorithm, we set a kind of back link to the one we selected whenever we select $CM(N(\dots), M(\dots))$ from several candidates. By tracing the back link, we would be able to obtain the path of the extracted template which has the heaviest common sequence with the input sentence immediately, as you have found the weight of the largest common sequence of the template-template and the path.

IV. SYSTEM EVALUATION

The present templates are constructed using the responses of about 200 members of **Study Participant Group 1** (SPG1) who have been engaged either in professional translation activities or in a retraining stage of technical translation after their college education. To test the effectiveness of the present ICALL, we have tested the system against a control having an entirely different background. We have now chosen a new **SPG2** of 100 students who are now freshmen (male or female) at a typical Japanese university. We show below the test translations obtained from the new study participants.

A. Test Problems for participants in the new study

The test problems are shown below.

After reading the following text giving the key sentence patterns, translate the underlined sentences into English. In translating the sentences, you may use dictionaries.

English Text:

Following the postwar rehabilitation of the mid-1940s, Japan gave priority to industrial progress in the 1950s and 1960s. As a result, the country achieved rapid economic growth, but this brought about a number of problems. One of these problems was pollution. For example, many chemical plants were rapidly constructed in Yokkaichi during this period. It was found that disease was caused by the sulfur oxides discharged from the plants.

Keyword 1:

the mid-XXXXs/the XXXXs

XXXX 年代中期 / XXXX 年代

Example Sentence

自動車産業は 1960 年代から急速に成長した。

Translation

The automotive industry has grown rapidly since the 1960s.

Problem No. 3-1-3:

コンピューター産業は 1970 年代と 1980 年代に成長しつづけた。

Keyword 2:

the late XXXXs/the early XXXXs XXXX 年代後期 / XXXX 年代初期

Example Sentence:

原油価格は 1970 年代初期に石油危機のあと急騰した。

Translation:

Crude oil prices soared after the Oil Crisis in the early 1970s.

Problem No. 3-2-3:

この会社は、1980 年代後期から 1990 年代初期にかけて事業を拡大した。

Keyword 3:

give priority to ~

~ を優先する

Example Sentence:

運転手は安全を優先させなければならない。

Translation:

Drivers must give priority to safety.

Problem No. 3-3-3:

政府は社会福祉を優先させなければならない。

Keyword 4:

bring about ~

~ もたらす

Example Sentence:

不景気はいくつかの深刻な問題をもたらした。

Translation:

problems.

The economic recession has brought about a number of serious

Problem No. 3-4-3:

自由化は競争をもたらした。

Keyword 5:

it was found that ~

~ が分かった

Example Sentence:

故障の原因は、オペレーターの誤りだったことが分かった。

Translation:

It was found that an operator error was the cause of the failure.

Problem No. 3-5-3:

この製品の品質は、温度制御によって改良できることが分かった。

Figure 3. Test problem sheet

The problem numbers, such as No. 3-5-3, are the numbers given in the demonstration version of the ICALL given in our homepage <http://azalea.sunflare.co.jp/>. The results of the evaluation are given below.

B. Template Statistics

The system performance naturally depends not only on the efficiency of the built-in diagnostic algorithm, but also on the data size of the templates we embed into the system, such as the number of model translations and the number of transitional states we build in, which in turn determine how many combinations of valid or erroneous invalid sentences are provided in the system. So far only four out of 20 sections of the present ICALL have been completed, each section having ten exercise problems.

TABLE I. Number of Model Translations and Total Finite (Transitional) States Built-In.

Template	Translation	Finite States	Template	Translation	Finite States
01-1-2	11	1412	02-1-3	4	828
01-1-3	9	1938	02-2-2	9	1488
01-2-2	7	808	02-2-3	4	412
01-2-3	6	584	02-3-2	9	2589
01-3-2	9	2301	02-3-3	5	1141
01-3-3	8	2344	02-4-2	8	497
01-4-2	7	755	02-4-3	8	2464
01-4-3	6	1671	02-5-2	9	2140
01-5-2	11	636	02-5-3	8	2071
01-5-3	9	1892	Total Averages	7.25	1438.45
02-1-2	7	798			

The number of model translations refers to the number of typical model translations which the language teacher has decided to build in, while the number of total finite states refers to that of the transitional states we build in. We emphasize that the former number refers to the number of basic templates to which other combinations can be added as valid translations by links of transitional states. The latter reflects the possible number of diagnoses we can provide in this system, including correct sentences as well as errors of students. On the average, we provide 7 model patterns of translations and 1420 finite states. That is to say, for each model sentence, we provide about 200 combinations of diagnostic messages.

C. Performance Analysis

We have collected 100 responses from the new test study participants, while we had about 200 responses from the original study participants. The analysis results are shown in Table 2, where the test results of the two groups are compared.

TABLE II. Test Results of the ICALL

Problem Number	02-2-3 SPG 2	02-2-3 SPG 1	02-3-3 SPG 2	02-3-3 SPG 1	02-4-3 SPG 2t	02-4-3 SPG 1
No. of Responses	100 (100%)	163 (100%)	100 (100%)	163 (100%)	100 (100%)	160 (100%)
A1	3 (3%)	97 (60%)	8 (8%)	59 (36%)	0 (0%)	22 (14%)
A2	7 (7%)	23 (14%)	5 (5%)	51 (31%)	2 (2%)	41 (26%)
B1	77 (77%)	39 (24%)	82 (82%)	53 (33%)	73 (73%)	91 (57%)
B2	1 (1%)	2 (1%)	1 (1%)	0	5 (5%)	5 (3%)
TooManyErrors	12 (12%)	2 (1.2%)	4 (4%)	0	20 (20%)	1 (1%)

In Table II, Study Participant Group 1 (SPG1) denotes the study participants comprising about 200 (strictly, 160 or 163) professional or semi-professional translators or those taking such courses at SunFlare Translation Academy, while SPG 2 denotes the new study participants consisting of 100 freshmen at a typical Japanese university, of whom 52 belong to the Engineering School and 48 to the School of International Studies. The distinct difference between the two groups can be traced to their educational background in English composition. The latter group very seldom practice speaking or writing English in the current Japanese educational system. The results of evaluation are given in Table 2 for problems of Templates 02-2-3, 02-3-3, 02-4-3 and the two groups for comparison.

Diagnosis group A in Table 1 refers to the group of responses having exact matches with the finite states of the template classification. A1 in particular refers to the group of excellent

students, whose translations have matched perfectly with those of the model translations provided, while A2 refers to those whose errors in translations have exact matches with those of the finite states of the templates. On the other hand, B1 refers to the correctly diagnosed errors, although they do not have exact matches with the finite states of the templates. B2 refers to those which are not well diagnosed in our system. This implies that a student may be able to learn a correct translation by our system, but a human teacher will most likely need to provide better comments. A classification of “TooManyErrors” will be given if the student input has more than half of the template paths or more than 2/3 of the words in the input sentence have misspellings. More often than not, this message is given when a student gives up the solution because of some other inability.

A distinct difference in the statistics of A1 very clearly shows the superiority of SPG1 in English composition competence over SPG2. As naturally expected from the data input of SPG1, the statistics for A2 and B1 show that the translations of Study Participant Group 1 have a better chance of correct diagnosis by the FSA of the present templates. On the other hand, the dominance of the numbers B1 over B2 for Study Participant Group 2 implies that the tutoring ability of the present system is remarkably robust and excellent, because the HCS (LCS) algorithm correctly identifies the most likely paths of the student input from among the templates and provides an error-contingent effective tutoring comment. We emphasize that the analysis was done by us manually for each participant one by one.

We give below the cases of B1, B2 and “TooManyErrors” in Figures 4-6. Notice that, in Figure 4, although a correct translation “... brought failure.” will be obtained by simply following the comments of the system, an experienced language teacher would suggest the student to change “failure in ended” to “ended in failure,” which is closer to the original erroneous expression. In both the B1 and B2 cases, the feedback is too generic and not error contingent. To solve this problem, we should add more frequent unclassified errors into a template by making use of the buggy rules for template-templates given in Section III.B.

Model translation: In recent days, many different models have become available.
 Input sentence: In the recent days, many different model is become available.
 Diagnosis given: In [the](1) recent days, many different [model](2) [is](3) become available.
 Comment (1): When referring to general classes of things, do not use "the" before plural countable nouns (boats, motors, meetings, countries). This part is not needed.
 Comment (2): This noun should be plural. This part should be: models
 Comment(3): This part should be: have

Fig. 4. Example 1 of B1. (Template 01-1-3)

Model translation 1: The work on the new material hasn't been a success.
 Model translation 2: The research into the new materials ended in failure.
 Model translation 3: Our research into new materials went unsuccessfully.
 Model translation 4: We were unsuccessful in our studies on the new materials.
 Input sentence: The research onto new materials failure in ended.
 Diagnosis given: The research [onto](1) new materials [XXX](2) failure [in ended](3) .
 Comment (1): To translate the phrase ~に関する研究, use "the research into ~" or "the research on ~." This part should be written as follows: into / about / concerning / on / regarding
 Comment (2): Something is missing. This part should be: brought
 Comment (3): This part is not needed.

Fig. 5. Example 2 of B2. (Template 1-3-2)

Model translation 1: Galileo's theory exerted a major influence on physics.
 Model translation 2: Physics was greatly affected by Galileo's theory.
 Model translation 3: Physics was influenced by Galileo's theories to a great extent.
 Input sentence: Galileo's theory has big Physik's influence.
 Diagnosis given: TooManyErrors, Try again.

Fig. 6. TooManyErrors Example. (Template 1-5-2)

D. ANALYSIS OF DIAGNOSIS

The diagnosis and subsequent comments provided are huge in volume, but we have checked them one by one manually. We think our current version of ICALL operates with remarkable robustness and accuracy when tested against control study participants having entirely different backgrounds from those of the study participants from whom the template database is extracted.

V. VISUAL TEMPLATE AUTHORIZING TOOL AND OPERATING ENVIRONMENT OF ICALL

A. VTAT (Visual Template Authoring Tool)

To help a language teacher input and edit the templates and comments or error messages associated with the templates, we have developed a visual authoring tool called VTAT (Visual Template Authoring Tool). Using this tool, a native speaker can draw the templates on the screen by simple drag and drop operations, constructing the links of connections between words, and managing and linking the comments to the proper position of the templates. The tool consists of three parts: Template Canvas window, Node Editor window and Comments Editor window. The Template Canvas window is used to draw and construct the template structure. The Node Editor window is used to define words and phrases and their properties of finite state automaton. The Comments Editor window is used to organize the comments database. A detailed description can be found in Tokuda and Chen (2001).

B. System Environment

Our application was first built on a SUN Sparc Station using Sunpro C++, based on the object-oriented database Object Store of ODI Inc. (Barry, 1996; Bertino & Martino, 1994) for templates and other linguistic information. Object Store has a built-in persistent class library `os_Collection`, which is very useful for representing a variety of data types in structures such as `os_Set`, `os_Bag`, and `os_List`. It is quite efficient for storing, retrieving and using persistent data, just like using them in the computer's transient memory.

To extend the applicability of the system, all the system has now been transported to MS Windows. The new version of VTAT has a capability of group editing in the WEB environment. Details of VTAT will be described elsewhere.

VI. CONCLUDING REMARKS

Advances in human-computer interfaces allow considerable human-like flexibility to be adopted into the ICALL systems. Most e-learning platforms allow only a very restricted form of input, such as multi-choice problems or fill-in-the-blank types. Few language learning systems allow free format input. Heift & Nicholson's German Tutor (Heift & Nicholson, 2001) allows free format input in WEB environments for introductory German courses where a learner's model and a parser operate on the Prolog server of the Server machine, while Java is used for the client side. In addition, Toole & Heift developed the *Tutor Assistant*, a web-based authoring tool for an ICALL system for English as a Second Language (ESL), which is designed to be usable by language instructors with little or no experience of ICALL systems and ICALL authoring tools, thus considerably simplifying the authoring tasks. They demonstrate that separating the tasks of specifying the expert module and developing learning content is productive in the domain of ICALL systems.

As we have confirmed with the new study participants, the present ICALL has turned out to be remarkably efficient and robust. Our ICALL has the following characteristics:

1. Enhanced pedagogical effects as well as the marked simplicity of template construction of the system are ensured by adopting Fries's teaching method (Fries, 1945), which is effective in controlling an exponential explosion of the template construction.
2. Efficient error diagnosis and also ease of exploiting the expertise of experienced language teachers are easily implemented into the current ICALL system by adopting a string matching algorithm.
3. The system can be used from the beginner level to the advanced level, providing basic writing competence.

We certainly do not expect the present system to provide facilities of good quality translations where context-sensitive semantic understanding is essential. Because any natural language can be decomposed into an FSA-based template automaton system, we believe the system can be applied to a language learning system of any language.

A basic implementation of the ICALL can be checked through the following home page: <http://azalea.sunflare.co.jp/>

ACKNOWLEDGMENTS

We wish to thank many people who participated in the developmental phase of the project, including Mr. Huang Liang for developing the VTAT, Dr. Qing Zhong for transporting it into the MS Windows system, and Professor Lorian Derooujer for cooperating with us in the new study participant test in this experiment.

REFERENCES

- Aho, A. V., and Ullman, J. D. (1992). *Foundations of Computer Science*, Computer Science Press.
- Allen, J. (1995). *Natural Language Understanding*, The Benjamin/Cummings Publishing Company Inc., pp. 53-60, pp. 83-123 and pp. 227-295.
- Barry, D. (1996), *The Object Database Handbook*, John Wiley & Sons, pp. 29-45.
- Bertino, E., and Martino, L. (1994), *Object-Oriented Database Systems Concepts and Architectures*, Addison-Wessley, pp. 232-242.
- Chen, L. and Tokuda, N. (2003), Bug Diagnosis by String Matching: Application to ILTS for Translation, *CALICO Journal*, Vol. 20, No. 2, pp.227-244.
- Chen, L., Tokuda, N. and Xiao, D. (2002). A POST Parser-Based Learner Model for Template-Based ICALL for Japanese-English Writing Skills, *Computer Assisted Language Learning*, Vol. 15, No.4, pp.357-372.
- Cormen, T. H., Leiserson, C. E. and Rivest, R. L. (1990). *Introduction to Algorithms*, MIT Press, Cambridge, Mass.
- Fries, C. C. (1945). *Teaching and Learning English as a Foreign Language*, University of Michigan Press.
- Heift, T., & Nicholson, D. (2001). Web Delivery of Adaptive and Interactive Language Tutoring. *International Journal of Artificial Intelligence in Education*, 12, see also http://cbl.leeds.ac.uk/ijaied/abstracts/Vol_12/heift.html
- Hopcroft, J. E., and Ullman, J. D. (1979). *Introduction to Automata Theory, Language, and Computation*, Addison-Wessley Publishing Co.
- Huang, L. (1999). *Users Manual for VTAT V1.1*, Computer Science Department, Utsunomiya University.
- Huang, Z., and Tokuda, N. (1996). A Syntactical Approach to Diagnosing Multiple Bugs in an Intelligent Tutoring System, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 26, No. 2, pp. 280-285.
- Kawai, H. (2001), "Japan's Goal in the 21st Century: The Frontier Within: Individual Empowerment and Better Governance in the New Millennium," Final Report to Japanese Government (see also <http://www.kantei.go.jp/jp/21century/report/pdfs/index.html>).
- Kr"uger, A., and Hamilton, S. (1997). RECALL: individual language tutoring through intelligent error diagnosis. *ReCALL*, 9(2).

- Murray, T. (1999). Authoring Intelligent Tutoring Systems: An Analysis of the State of the Art, *International Journal of Artificial Intelligence in Education*, vol. 10, pp. 98-129
- Sekine, S., and Grishman, R. (1996), A Corpus-based Probabilistic Grammar with Only Two Non-terminals, *4-th International Workshop on Parsing Technology*, pp. 216-223.
- Sentance, S. (1997) A Rule Network for English Article Usage Within an Intelligent Language Tutoring System. *Computer Assisted Language Learning*, 10 (2).
- Shieber, S. M. (1986). An Introduction To Unification-based Approach To Grammar, *CSLI Lecture Notes 4*: Chicago U. Press, Chicago.
- Tokuda, N. (2002). Guest Editor's Introduction; New Developments in Intelligent CALL Systems in a Rapidly Internationalized Information Age, Special Issue on Recent Development in ICALL, *Computer Assisted Language Learning*, Vol 15, No. 4, pp. 319-327.
- Tokuda, N., and Chen, L. (2001), An Online Tutoring System for Language Translation, *IEEE Multimedia*, Vol. 8, No. 3, pp. 46-55.
- Tokuda, N., Chen, L. and Sasai, Y. (2002), System and Method for Accurate Grammar Analysis Using A Learners' Model and Part-of-Speech Tagged (POST) Parser, US Patent Application No. 10/072, 954.
- Toole, J., and Heift, T. (2002). The *Tutor Assistant*: An Authoring Tool for an Intelligent Language Tutoring System. To appear in Special Issue on Recent Development in ICALL, Volume 15, No. 4, *Journal of CALL* .
- Toole, J., & Heift, T. (2001). Generating Learning Content for an Intelligent Language Tutoring System. *Proceedings of NLP-CALL Workshop at the International Conference on Artificial Intelligence in Education*, pp. 1-8.
- Virvou, M., Maras, D., and Tsiriga, V. (2000), Student Modelling in an Intelligent Tutoring System for the Passive Voice of English Language. *Educational Technology & Society Journal*, 3 (4).
- Yang, J. C., and Akahori, K. (2000) A Discourse Structure Analysis of Technical Japanese Texts and Its Implementation on the WWW. *Computer Assisted Language Learning*, 13 (2)
- Yarowsky, D. (1995). Unsupervised Word Sense Disambiguation Rivaling Supervised Methods, *Proceedings of ACL-95*, pp. 189-196.
- Warschauer, M. (1999). Millennialism and Media: Language, Literacy, and Technology in the 21st Century, *Keynote speech at World Congress of Applied Linguistics*, Tokyo.
- Warschauer, M. (2002). What Is the Digital Divide, <http://www.gse.uci.edu/markw/dd.pdf>
- Warschauer, M., and Kern, R. (eds.) (2000). *Network-based Language Teaching: Concept and Practice*, Cambridge Applied Linguistics Series, Cambridge University Press, 2000.
- White, C. (1996), Using Object Databases on the World-Wide Web, *Info DB (USA)*, Vol. 10, No. 1.