Z1

Pergamon

0346-251X(95)00053-4

System, Vol. 24, No. 1, pp. 65–81, 1996
Copyright © 1996 Elsevier Science Ltd
Printed in Great Britain. All rights reserved
0346-251X/96 $15.00 + 0.00

AN ALGORITHM FOR DOMAIN KNOWLEDGE BASE ACQUISITION IN AN INTELLIGENT TUTORING SYSTEM: JAPANESE TRANSLITERATION RULES*

YUN-SUN KANG and ANTHONY A. MACIEJEWSKI

School of Electrical Engineering, Purdue University, West Lafayette, IN 47907, U.S.A.

This article describes an algorithm for developing a domain knowledge base that is used in a Japanese language intelligent tutoring system. The domain knowledge represents a model of the expertise that a student must acquire in order to be proficient at reading one of the distinct orthographies of Japanese, known as *katakana*. Whereas the effort required to memorize the relatively few *katakana* symbols and their associated pronunciations is not prohibitive, a major difficulty in reading *katakana* is associated with the phonetic modifications which occur when English words which are transliterated into *katakana* are made to conform to the more restrictive rules of Japanese phonology. The algorithm described here is able to generate a knowledge base of these phonological transformation rules automatically, that is subsequently used to assess a student's proficiency and then appropriately individualize the student's instruction. Copyright © 1996 Elsevier Science Ltd

INTRODUCTION

Interest in Japanese language instruction has risen dramatically in recent years, particularly for those Americans engaged in technical disciplines. However, the Japanese language is generally regarded as one of the most difficult languages for English-speaking people to learn. Whereas the number of individuals studying Japanese is increasing, there remains an extremely high attrition rate, estimated by some to be as high as 80% (Mills *et al.*, 1988). Much of this difficulty can be associated with the Japanese writing system. Japanese text consists of two distinct orthographies, a phonetic syllabary known as *kana* and a set of logographic characters, originally derived from the Chinese, known as *kanji*. The *kana* are divided into two phonetically equivalent but graphically distinct sets, *katakana* and *hiragana*, both consisting of 46 symbols and two diacritic marks denoting changes in pronunciation. The *katakana* are used primarily for writing words of foreign origin that have been adapted to the Japanese phonetic system. Due to the limited number of *katakana*, their relatively low visual complexity, and their systematic arrangement,

*This material is based upon work supported by the National Science Foundation under Grant No. INT-8818039 and in part by the NEC Corporation and a Purdue University Global Initiative Grant.

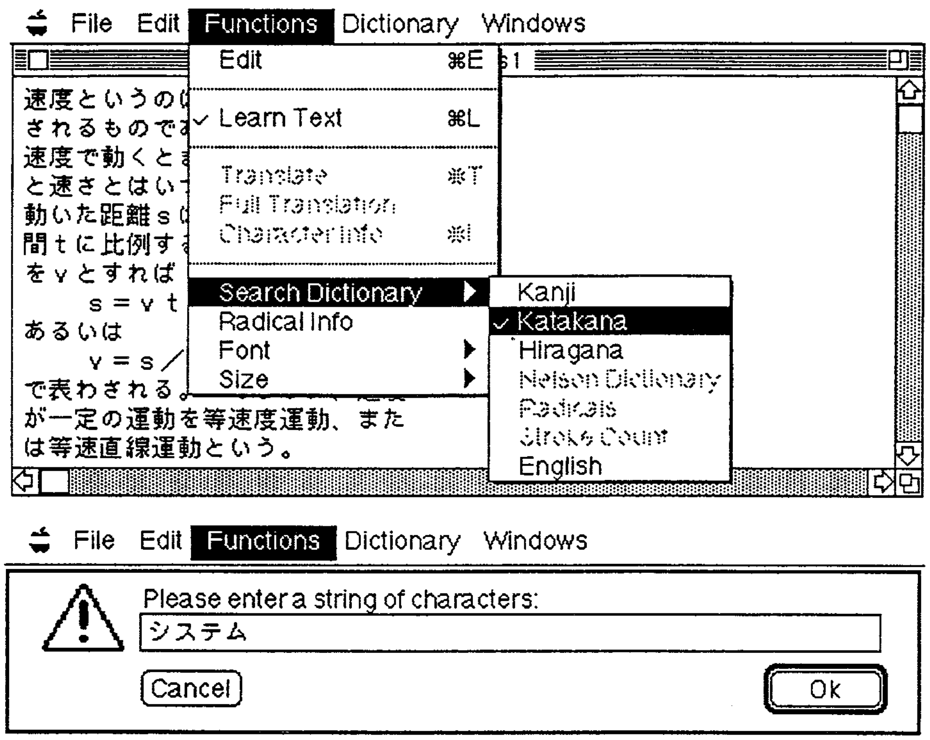
memorizing their pronunciations does not represent a significant barrier to the student of Japanese. If the student can also assimilate the phonological transformation that occurs, then the effort required to learn *katakana* yields significant returns to readers of technical Japanese because of the high incidence of terms derived from English and transliterated into *katakana*.

In this work, rules that clarify the phonological relationship between a *katakana* word and its English origin word are generated. The entries in a *katakana*-to-English dictionary (Kang and Maciejewski, 1992) are converted into their international phonetic alphabet (IPA) equivalents and an algorithm is used to match phonemes and to generate rules for phonemic modifications. This forms two rule bases, one for *katakana* to English and the other for English to *katakana*, which represent a model of the skill that a student must acquire in order to be proficient at reading *katakana*. This model is used by an intelligent tutoring system developed previously (Maciejewski and Leung, 1992) which assists a student who is learning to read technical Japanese. The remainder of this article is organized as follows: in the next section, the structure of intelligent tutoring systems is overviewed by comparing existing intelligent tutoring systems to the Japanese language intelligent tutoring system. The third section provides a brief introduction to the *katakana* writing system and to the rules of Japanese phonology. A data structure to represent a phonological transformation rule is described in the following section. In section five, an algorithm for generating a set of phonological transformation rules is described. An evaluation of the resulting rule set is provided in the following section. Finally, the conclusions of this work are presented in the last section.

OVERVIEW OF INTELLIGENT TUTORING SYSTEMS

Intelligent tutoring systems (ITSs) are computer programs that can individualize their instruction based on inferences about a student's knowledge. Whereas existing ITSs vary in architecture, they typically consist of at least four basic components (Mandl and Lesgold, 1988; Wenger, 1987): the expert knowledge module, the student model module, the tutoring module and the user interface module. The expert knowledge module provides the domain knowledge that the system intends to teach. The student model refers to the dynamic representation of a student's competence for the given domain. The tutoring module is the part of the ITS that designs and regulates instructional interactions with the student. Finally, the fourth component of ITSs is the user interface module, which controls interactions between the system and the student. More details on ITS structure and previously developed prototypes are available in Park *et al.*, 1987, Polson and Richardson, 1988, Rickel, 1989 and Yazdani, 1987.

The specific ITS being considered in this work is called the Nihongo Tutorial System which was designed to assist English-speaking scientists and engineers acquire Japanese reading proficiency in their technical area of expertise (Maciejewski and Leung, 1992). The architecture of this system closely resembles the general structure of ITSs outlined above. The domain knowledge database, which is the focus of this work, is prepared by a software module called the Parse Tree Editor that processes technical journal articles into instructional material by incorporating syntactic, semantic, phonetic and morphological information into a representation known as an augmented parse tree. The



速度というの
されるものであ
速度で動くとき
と速さとはい
動いた距離 s は
間 t に比例する
を v とすれば
　s = v t
あるいは
　v = s ／
で表わされる。
が一定の運動を等速度運動、また
は等速直線運動という。

Please enter a string of characters:
システム

Fig. 1. The user interface of the Nihongo Tutorial System. When the student does not comprehend a *katakana* word, he can ask the system to search the on-line dictionaries. The tutorial system saves student's queries and uses them to build the student model.

student model is updated based on a student's interactions with the system. The instructional interactions between the system and a student are regulated by the administrator module which matches the student's current level of Japanese proficiency and technical area of interest with the available instructional material produced by the Parse Tree Editor. The user interacts with the system through a graphical user interface to request information about the current instructional text or to obtain examples of material on the same or related concepts (see Fig. 1).

The expert knowledge module currently used by the Nihongo Tutorial System is designed based on a rule-based representation. This type of knowledge representation is commonly used in algorithmically tractable domains such as mathematics, physics and programming languages (Anderson, 1988), as well as in foreign language learning (Swartz, 1992). The design of the expert knowledge module is closely related to other components in an ITS, especially the student model module. The student model used by the Nihongo Tutorial System falls into the general class of student models known as "overlay" models (Carr and Goldstein, 1977). This commonly used type of model considers the student's knowledge to be a subset of the expert knowledge base. In the case of the Nihongo Tutorial System the expert knowledge contains the Japanese characters, vocabulary and the syntactic, morphological, and phonological transformation rules required to understand the Japanese text, along with a number that represents the probability that the student understands that particular piece of knowledge. The remainder of this work will only deal with the expert knowledge base associated with the phonological transformation rules required to understand Japanese text written in *katakana*.

KATAKANA AND JAPANESE PHONOLOGY

The Japanese lexicon contains an extremely large number of words originating from foreign languages. Whereas the proportion of words of Chinese origin in the lexicon is extremely large due to the profound cultural influence of China, words of English origin have dominated the class of loan words since the late 19th century: In a study of Japanese publications performed between 1956 and 1964, over 80% of the foreign words originated from English (Kokuritsu Kokugo Kenkyuujo, 1964). This process of adopting English words into the lexicon is particularly common for relatively new or specialized terms arising in technical literature. When adopting a word of foreign origin into Japanese, the original pronunciation of that word is typically transliterated into *katakana* which graphically represents all of the possible phonetic sequences in the Japanese language. It is this process of modifying English phonetic sequences to conform to the rules of Japanese phonology which presents English-speaking readers of *katakana* with difficulty in identifying a word's meaning. This is due to the fact that the rules of Japanese phonology are quite different from those of English. In particular, Japanese has only five single vowel sounds /a i u e o/ in contrast

Table 1. *Katakana* characters and their phonetic representations in the IPA symbols: Basic syllables

ア	イ	ウ	エ	オ
/a/	/i/	/u/	/e/	/o/
カ	キ	ク	ケ	コ
/ka/	/ki/	/ku/	/ke/	/ko/
サ	シ	ス	セ	ソ
/sa/	/ʃi/	/su/	/se/	/so/
タ	チ	ツ	テ	ト
/ta/	/ʧi/	/tsu/	/te/	/to/
ナ	ニ	ヌ	ネ	ノ
/na/	/ni/	/nu/	/ne/	/no/
ハ	ヒ	フ	ヘ	ホ
/ha/	/hi/	/fu/	/he/	/ho/
マ	ミ	ム	メ	モ
/ma/	/mi/	/mu/	/me/	/mo/
ヤ		ユ		ヨ
/ja/		/ju/		/jo/
ラ	リ	ル	レ	ロ
/ra/	/ri/	/ru/	/re/	/ro/
ワ				
/wa/				

to the large number of vowel sounds in English. These vowels, when combined with the nine Japanese consonants /k s t n h m j r w/ constitute 44 of the 46 basic sounds in Japanese which are traditionally organized as shown in Table 1. The pronunciation of each *katakana* character in the table is represented using the international phonetic alphabet (IPA) symbols. Additional variations of these basic syllables are presented in Table 2.

From the above description of the *katakana* orthography, it is clear that certain inherent limitations are imposed on phonetic sequences in the Japanese language. In particular, Japanese does not allow any consonant clusters, except when a consonant is followed by

Table 2. *Katakana* characters and their phonetic representations in the IPA symbols: Additional variations of the basic syllables

Modified syllables

ガ	ギ	グ	ゲ	ゴ
/ga/	/gi/	/gu/	/ge/	/go/
ザ	ジ	ズ	ゼ	ゾ
/za/	/ʤi/	/zu/	/ze/	/zo/
ダ	ヂ	ヅ	デ	ド
/da/	/ʤi/	/zu/	/de/	/do/
バ	ビ	ブ	ベ	ボ
/ba/	/bi/	/bu/	/be/	/bo/
パ	ピ	プ	ペ	ポ
/pa/	/pi/	/pu/	/pe/	/po/

Consonants plus /ja/, /ju/, or /jo/

キャ	キュ	キョ	ギャ	ギュ	ギョ
/kja/	/kju/	/kjo/	/gja/	/gju/	/gjo/
シャ	シュ	ショ	ジャ	ジュ	ジョ
/ʃa/	/ʃu/	/ʃo/	/ʤa/	/ʤu/	/ʤo/
チャ	チュ	チョ			
/ʧa/	/ʧu/	/ʧo/			
ニャ	ニュ	ニョ			
/nja/	/nju/	/njo/	ビャ	ビュ	ビョ
ヒャ	ヒュ	ヒョ	/bja/	/bju/	/bjo/
/hja/	/hju/	/hjo/	ピャ	ピュ	ピョ
ミャ	ミュ	ミョ	/pja/	/pju/	/pjo/
/mja/	/mju/	/mjo/			
リャ	リュ	リョ			
/rja/	/rju/	/rjo/			

Mora consonants

ン	ッ
/N/	/Q/

a glide[1] or preceded by a moraic[2] consonant (Vance, 1987). In addition, consonants may not appear at the end of a sequence. These restrictions, together with the limited number of Japanese vowel sounds, result in the vast majority of phonological modifications which occur when transliterating an English word into *katakana*. By the same token, these resulting modifications are the source of difficulty for English-speaking readers of *katakana*.

It should also be noted that there are additional difficulties to comprehending *katakana* unrelated to the phonological processes involved. In particular, whereas it is true that the vast majority of loan words are created by the phonological process, foreign borrowings may also be modified by changes in form due to simplification, semantics, or Japanese coinage (Shibatani, 1990). For example, simplification frequently occurs with polysyllabic words such as "television" and "word processor" which are shortened to the *katakana* words *terebi* and *waapuro*, respectively. Changes in semantics have resulted in the *katakana* word *botan* being used to designate a touch tone type of telephone, whereas its phonetic origin is from the word button. Examples of coinage which result from combinations of existing loan words include *maikaa* (derived from my + car) and *maihoomu* (derived from my + home) which refer to privately owned cars and houses. All of these processes contribute to a student's difficulty in achieving reading proficiency in *katakana*; however, this work focuses only on the phonological modifications.

THE STRUCTURE OF PHONOLOGICAL TRANSFORMATION RULES

The phonological transformation rules that are determined by the rule-based system presented here consist of three elements: a source phoneme, a target phoneme and the context. The source phonemes are those that are derived from the *katakana* string (because this is the text that the student will be reading) and the target phonemes are those that are derived from the English text. The following standard notation is used to represent a phonological rule:

$$\mathrm{S} \rightarrow T \,/\, C_1 \_ \, C_2$$

where S is a phoneme of the source language, T is a phoneme of the target language, and C_1 and C_2 represent the context in which S may be replaced by T. The symbols C_1 and C_2 are either single phonemes or phonological categories such as vowels, consonants etc. Thus this rule implies that if the string C_1SC_2 occurs in the source language it may be transformed into the string C_1TC_2 in the target language. The phonological transformation rules are also associated with the probability of how likely a phoneme in a source language is to be transformed into a phoneme in a target language under a specific context. The probability of a rule is computed by dividing the number of occurrences of a rule by the total number of occurrences of all the rules that have the same source and the same context as that rule.

GENERATION OF PHONOLOGICAL TRANSFORMATION RULES

This section describes the algorithm designed to automatically determine a set of rules that govern the specific phonetic changes that occur in English text when it is transliterated into *katakana*. These rules represent the domain knowledge that the tutoring system must "learn" in order to provide efficient instruction. The process of rule generation is divided into three phases:

Phase I: Direct matching without using an existing rule base
The inputs into Phase I are a string of *katakana* and the English text from which it was derived. These two inputs are then converted into IPA symbols using Tables 1 and 2 for the *katakana* and the UNIX version of Webster's Seventh New Collegiate Dictionary (Gove, 1963) for the English. Phonological rules of the type described above cannot be directly generated from the IPA symbols of the raw input strings because there is not a correspondence between the *i*th phoneme of the source string and the *i*th phoneme of the target string. This is primarily due to the different constraints on consonant clusters in Japanese as compared to English. As mentioned previously, Japanese does not, in general, allow consonant clusters whereas English allows initial consonant clusters of two or three consonants, as in such words as "sky" and "spray", and either two, three, or four final consonants, as in "ask", "elks" and "glimpsed" (Catford, 1988). To compensate for these phonological differences, the sequence of IPA symbols for the English text string is modified by inserting the symbol "*", which has no pronunciation, between any consecutive consonant phonemes and after sequences that end with a consonant. This modification greatly improves the correspondence between phoneme symbols in the *katakana* string with the symbols in the English string. However, it does create a difficulty with the mora nasal consonant cluster that is allowed in Japanese. It has been observed that the mora nasal /N/ when followed by a consonant corresponds to either /m/, /n/, or /ŋ/ followed by a consonant in English. Therefore, in order to provide uniform treatment of these consonant clusters, the symbol "*" is also inserted into the *katakana* phoneme sequence between the mora nasal and the following consonant.

In addition to consonant clusters, one must also consider how to deal with non-identical vowel sequences in order to improve the correspondence between symbols in the source string and the target string. In Japanese, it is not obvious when to consider the second vowel of a sequence as a separate syllable as opposed to the second half of a diphthong. This creates a difficulty when trying to match the English vowels. Therefore, the approach adopted here is that all consecutive vowel sequences are treated as a single unit. This is true even if it is known that the English vowel sequence represents two separate syllables. Therefore, after the appropriate modification of the input strings by inserting "*" symbols, the strings are divided into units which consist of either a single consonant, a sequence of vowels or semi-vowels, or the symbol "*". If the number of partitions in the two strings are equal, then a set of phonological rules is generated.

The algorithm described above can be summarized by the following four steps:

(1) Convert the *katakana* input string and the English input string into IPA symbols.
(2) Modify the English IPA string by inserting a "*" symbol between any consecutive consonant phonemes and after a final consonant. Modify the Japanese IPA string by inserting a "*" symbol between the mora nasal /N/ and any following consonant.
(3) Divide both symbol strings into partitions which include either a single consonant, a consecutive sequence of vowels and/or semi-vowels, or the symbol "*".

(4) If the number of partitions in the source string is equal to that of the target string, generate a phonological rule which defines the transformation of a source partition into a target partition.

This process is illustrated in the following example, in which the, *katakana* word "*shisutemu*" is compared with the word "system" from which it was derived:

システム	→	/ʃisutemu/	→	/ʃ,i,s,u,t,e,m,u/
system	→	/sɪstəm/	→	/s,ɪ,s,*,t,ə,m,*/

The following eight rules are generated:

ʃ→s/#_i   i→ɪ/ʃ_s   s→s/i_u   u→*/s_t
t→t/u_e   e→ə/t_m   m→m/e_u   u→*/m_#

where the symbol "#" denotes a word boundary. Note that the somewhat obvious rules in which a phoneme is not changed are significant since the same phoneme may be modified in a different context. Additional information regarding the relative probability of occurrence for rules in an identical context is also maintained.

The next example illustrates how the vowel sequences are handled. Consider the *katakana* word "*iesu*" which was derived from the English word "yes":

イエス	→	/iesu/	→	/ie,s,u/
yes	→	/jes/	→	/je,s,*/.

The following three rules are generated:

ie→je/#_s   s→s/e_u   u→*/s_#.

The sequence of semi-vowels and/or vowels is not divided in the Phase I process. A rule that consists of multiple phonemes is decomposed into two different rules by Phase III which will be discussed later.[3] Finally, there are also cases in which the numbers of partitions in the source and target strings are not equal as shown in the following example using the *katakana* word "*ringu*":

リング	→	/riNgu/	→	/r,i,N,*,g,u/
ring	→	/ɹɪŋ/	→	/ɹ,ɪ,ŋ,*/.

The number of partitions were not equal in approximately 8% of the 850 example entries. When this occurs in Phase I, no phonological rules are generated and the modified phoneme string obtained from Step 2 of Phase I is used in a different technique called Phase II.

Phase II: Matching using an existing rule base
The dictionary entries that do not have an equal number of partitions during Phase I need to be processed using a different algorithm. Phase II is performed by using the rule

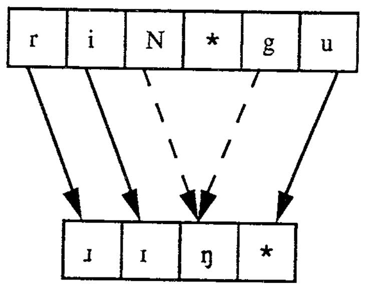


Fig. 2. An example of a dictionary entry that does not have the same number of partitions after applying Phase I. Applying three of the previously generated phonological transformation rules to the modified phoneme strings obtained from Step 2 of Phase I (shown in solid lines) allows Phase II to identify the new rule Ng→ŋ/i_u which is shown in dotted line.

base generated from the 92% of the entries that are successfully partitioned in Phase I. Before the algorithm applied in Phase II is described in detail, a simple example is considered. Because the partitioning technique does not provide a correlation between a *katakana* word and its English origin word, what one needs to do is to try and get some information about what pieces of those words are matched by either looking for the same phoneme or by applying a rule in the rule base. When the *katakana* word "*ringu*" is compared with the English word "ring", there are three phonological rules that match the input phoneme strings which are shown in solid lines in Figure 2. In this case, it is quite clear that the Japanese phonemes /Ng/ match the English phoneme /ŋ/, which is depicted in dotted lines. However, in most cases, many rules match input phoneme strings, and some matched rules can result in generating incorrect phonological rules. It is therefore necessary for the algorithm to be intelligent enough to make sure that only correct rules are produced.

The main idea of Phase II is to use the most likely phonological transformation rules generated from Phase I and to compare unmatched phonemes in the source and target inputs to generate additional phonological rules. In order to determine which rule is the most likely, the rule base generated from Phase I is used to consider a tree of all possible phoneme matches. The tree consists of nodes that contain either a phonological rule or a single phoneme and arcs that connect the nodes. The most likely tree is then constructed using an A* algorithm (Nilsson, 1982). This procedure is now considered in more detail.

First, the rule base is scanned for all rules with the context and the source phoneme that match any of the phonemes in the source input. There may be no rules that match some phonemes and others may be matched by multiple rules. For those rules that match the source string, the target string is then searched for the target phoneme. The target input may contain the target phoneme in several locations. Each of them is used as a different node in the tree. In addition to the rules that match the input phoneme strings, each phoneme of the source input is also used as a node in the tree. These nodes represent the source of potential new phonological transformation rules if they are included in the most likely tree. Two different types of nodes are thus created: one type that contains a phoneme and the other type that contains a phonological transformation rule. A search begins by expanding the start node of the tree, i.e. generating a node consisting of the first phoneme of the source input as well as all of the rules that match that phoneme as successor nodes. The search sequentially continues by using the following phonemes of the source input until the last phoneme or the matched rule including that phoneme is expanded. The portions of the tree that contain the most likely rules are then generated by using the A* algorithm. In the A* algorithm all of the remaining possible search paths are stored in the list in each step, and the paths are kept in order based on the heuristic information that determines which path is the most promising. This information is represented by the evaluation function

$$f(n) = g(n) + h(n) \tag{1}$$

where $g(n)$ is the cost of the path in the tree from the start node s to the current node n and $h(n)$ is an estimate of $h^*(n)$ which is the cost of the maximal cost path from n to a goal node. The arc cost associated with a node that contains a matched rule is assigned to be the probability of that rule occurring, and the arc associated with a node that contains just a phoneme is assigned a value of zero. The heuristic function used in this work is defined as the largest path cost of the rules that are possible successor nodes of the node n, i.e. the rules that match the phonemes following node n. Using this definition it is clear that h is an upper bound on h^*, which satisfies the condition for the A* algorithm.

The performance of this algorithm is improved by incorporating basic assumptions of the transliteration process into the heuristic function $h(n)$. For example, one knows that the *katakana* characters that are generated by the transliteration process must be in the same order as the characters in the original English word. Therefore, in order for a node that contains a rule to be actually generated, the target phoneme of any of its ancestor rule nodes must precede the target phoneme of the node in the target input. This information is also used for implementing the heuristic function in the A* algorithm. One additional detail that must be included in this A* search algorithm is that one does not allow a path that would cause a phoneme in the target string not to correspond with any phoneme in the source string. This technique greatly reduces the number of possibly incorrect rules that could be generated.

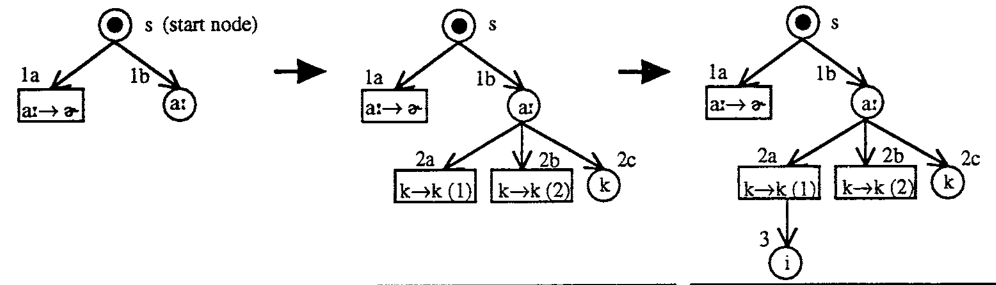
The following example illustrates how the A* algorithm just discussed is applied to the dictionary entry "architecture" which is transliterated into the *katakana* word "*aakitekucha*" (see Fig. 3). A search begins by expanding the start node. There is only one rule that matches the first phoneme of the source string as listed in Fig. 3(a). Consequently, two nodes are generated as successor nodes of the start node as depicted in Fig. 3(b). Note that a node in a circle denotes a phoneme from the source input string and a node in a box represents a phonological rule that is matched with the input strings. After the start node is expanded, the values of the functions g and h are computed for each node, which are listed in the table in Fig. 3(b). Because node 1a contains a phonological rule, it has a larger value of g. However, its value of h is zero because there are no rules that can be a successor of node la. Consequently, node 1b has a larger value of f so this node is selected for expansion. By the same technique, the search continues (as shown in Fig. 3(c) and (d)) until the node that contains the last phoneme of the source string is expanded. The complete search tree is depicted in Fig. 4(a). During the search, if any node that contains a rule violates the sequential transliteration condition, the node is pruned from the tree, which is denoted in Fig. 4(a) by a shaded box. After

(a) Rules from existing rule base that match both source and target phonemes for the transformation /aːkitekutʃa/ → /aɪkətektʃɚ/

Number	Rule†				Probability
1	aː	→	ɚ	/ (C|#)_(C|#)	100.0
2	k	→	k	/ (V|#)_(V|j)	100.0
3	t	→	t	/ (V|#)_V	100.0
4	e	→	e	/ (C|#)_C	100.0
5	k	→	k	/ (V|#)_(V|j)	100.0
6	u	→	*	/ C_(C|#)	100.0
7	ʧ	→	ʧ		45.8
8	a	→	a		0.7
9	a	→	ə	/ (C|#)_(C|#)	100.0

†The rule format $A \rightarrow B/C_1\_C_2$ implies that the string C_1AC_2 may be replaced by C_1BC_2. All lower case letters in the table represent themselves. The upper case letters and special characters have the following meanings: C: any consonant, V: any vowel, #: sequence boundary, *: null character, $(x_1|x_2)$: either x_1 or x_2. Note also that the phonemes /aː/ and /ʧ/ are counted as a single phoneme.

(b) The first step (c) The second step (d) The third step



Node	1a	**1b**
g(n)	100.0	0.0
h(n)	0.0	645.8
f(n) = g(n)+ h(n)	100.0	**645.8**

Node	1a	**2a**	2b	2c
g(n)	100.0	100.0	100.0	0.0
h(n)	0.0	545.8	245.8	545.8
f(n)	100.0	**645.8**	345.8	545.8

Node	1a	2b	2c	**3**
g(n)	100.0	100.0	0.0	100.0
h(n)	0.0	245.8	545.8	545.8
f(n)	100.0	345.8	545.8	**645.8**

Fig. 3. An illustration of the rule generation algorithm used in Phase II for the example *katakana* word "*aakitekucha*" which was transliterated from the word architecture. The table in (a) shows a list of all rules from the existing rule base (generated in Phase I) that are due to matching phonemes in the source and target inputs. Note that multiple rules are matched with the same source input phoneme (such as rules 8 and 9) and that the target input contains the same target phoneme in different locations (for example, the phoneme /k/ when using rules 2 and 5). The figures in (b) through (d) show the portions of a tree that are generated by the A* algorithm for the example from the first step to the third step in Phase II. The tables in (b) through (d) list the values of the evaluation function *f*(*n*) for all possible nodes to be expanded from each step. The node on the list having the largest value of the evaluation function *f*(*n*) is selected for expansion (shown in boldface). Note that the evaluation function *f*(*n*) is calculated by adding up the values of *g*(*n*) and *f*(*n*) where the value of *g*(*n*) is computed by summing the arc costs encountered while tracing the parent nodes from the node *n* to the start node *s* and the heuristic information *h*(*n*) is defined as the largest total of probabilities of the matched rules that can be a successor node of the node *n*.

the path from the start node to the goal node is found, additional phonological transformation rules are generated by comparing the phonemes from source and target inputs that are not matched by any rule in the path of a tree as shown in Fig. 4(b). The procedure of Phase II can be summarized as follows:

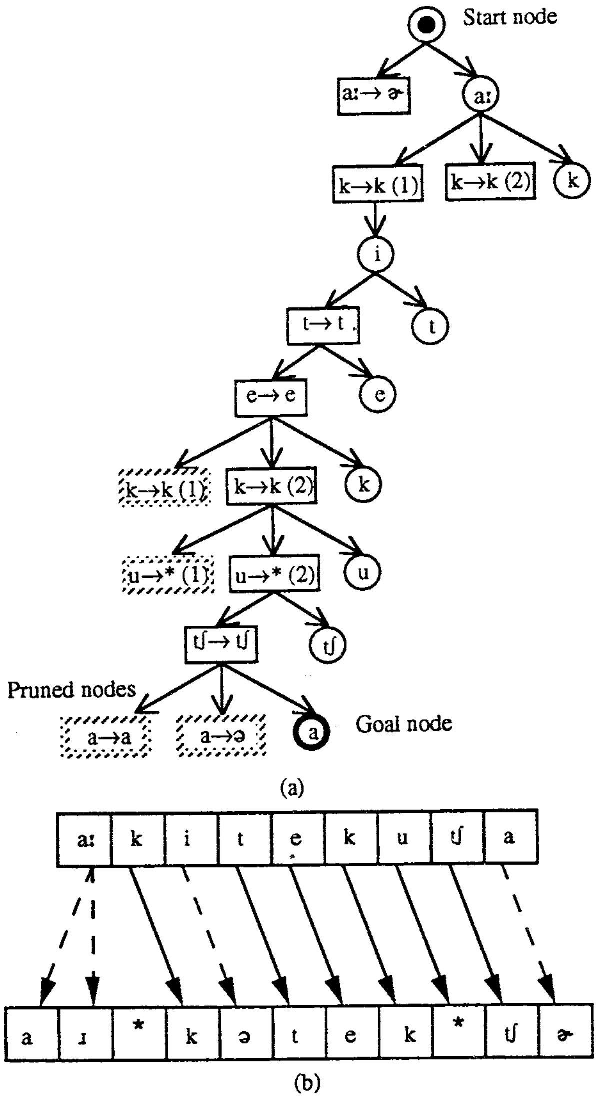


Fig. 4. An example of generating phonological rules in Phase II. The figure in (a) shows the configuration of the complete tree for the IPA strings /aːkitekuʧa/ and /aɹkətekʧɚ/ in Phase II where a node in a circle denotes a phoneme from the source input and a node in a box represents a phonological rule that is matched with the input strings (shown in solid lines in (b)). During the search, if any node that contains a rule violates the sequential transliteration condition, the node is pruned from the tree, which is denoted by a shaded box. Using six of the phonological transformation rules allows Phase II to identify the new rules aː→aɹ/#_k, i→ə/k_t, and, a→ɚ/ʧ_# which are shown in dotted lines in (b).

(1) Using the *katakana* and English IPA strings modified from Step 2 in Phase I as the source and target inputs, scan the rule base for all rules that match any of the phonemes in the source input. For those rules that match a phoneme in the source string, the target string is then searched for the target phoneme.
(2) Construct the tree by using the A* algorithm where an arc for a node that contains a phonological rule is assigned the cost that is equal to the probability of that rule occurring.

(3) Generate a phonological transformation rule by comparing the phoneme sequences in the source and target inputs that are not matched by any rule from the start node to the goal node in the path of the tree.

The rules generated in Phases I and II are then further processed in Phase III:

Phase III: Rule decomposition
Multiple phoneme rules that are generated in Phase I and Phase II are considered for decomposition into several single phoneme rules in Phase III of the rule base acquisition algorithm. In Phase I these multiple phonemes occur due to the fact that a sequence of semi-vowels and/or vowels is not divided into partitions because of insufficient information about Japanese diphthongs. These multiple phoneme sequences may therefore be a part of the source or target of a rule. In Phase II, rules are generated by comparing phonemes of the source and target strings which are not matched by any rule generated in Phase I. In this case, the source and target of a rule may also be a sequence of any phonemes, because in the worst case it is possible that no rule matches the input phoneme strings. These multiple phoneme rules are candidates for decomposition into rules where either the source or the target has a single phoneme. In order to prevent any incorrect rule from being generated in this process, the rule decomposition process is divided into three steps using progressively weaker restrictions in each step.

In the first step, if a multiple phoneme rule can be decomposed into existing single phoneme rules, the rule is replaced by the existing rules. For example, if there are rules "i→j" and "e→e" in the rule base, the rule "ie→je/#_s" is replaced by the two rules "i→j/#_e" and "e→e/i_s". Any multiple phoneme rule that fails to be decomposed into existing single phoneme rules is considered in the second step. In this step the source and target of a multiple phoneme rule are regarded as input phoneme strings, and are compared using the same technique used in Phase II except that the context for a rule in the rule base is not checked. Step 2 also continues the restriction that the generation of a silent rule is prohibited, which greatly helps prevent any incorrect rule from being generated. In contrast, during the third step silent rules are allowed. Otherwise the procedure of the third step is identical to that of the second step. The rule decomposition process can be summarized as follows:

Decompose multiple phoneme rules under the following conditions:

(1) if there exist single phoneme rules that can exactly replace the multiple phoneme rule
(2) using the A* algorithm and the heuristic information defined in Phase II
(3) using the same technique as Step 2, but allowing a silent rule.

The following examples present the successful completion of the rule decomposition process. When the rule base has the rules "i→I" and "a→ə", the rule "ia→Iə" is decomposed into the existing rules in the first step. Figure 5 illustrates how the rule "aijaɪ̆→aIə˞" is decomposed into three different rules using all the steps in the rule decomposition process. Because the rule cannot be decomposed into existing single phoneme rules, the first step does not decompose the rule any further. In the next step, the rule base is scanned and a rule matches the source and target of the rule. The rule

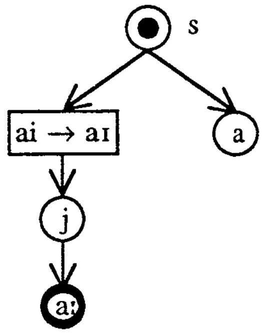
Phase I

ハイヤー ⇒ /haijaː/ ⇒ /h, aijaː/
higher ⇒ /haɪɚ/ ⇒ /h, aɪɚ/

Phase III: Rule Decomposition

step 1: Not decomposed

step 2: The phonological rule aijaː→aɪɚ is decomposed into the two rules ai→aɪ and jaː→ɚ.



step 3: The phonological rule jaː→ɚ is decomposed into the two rules j→* and aː→ɚ.

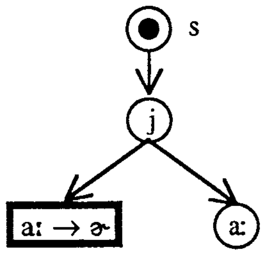


Fig. 5. An example of three progressively less restrictive steps used in Phase III to decompose the multiple phoneme rules. The phoneme sequence "aɪ" is typically regarded as a single phoneme.

"aijaː→aɪɚ" is decomposed into the two rules "ai→aɪ" and "jaː→ɚ". The rule "ai→aɪ" is not decomposed any more since the English phoneme sequence "aɪ" is typically regarded as a single phoneme and "ai" consists of phonemes that belong to the same category. The rule "jaː→ɚ" is also not decomposed because a silent rule is prohibited. Finally, the rule "jaː→ɚ" is decomposed into the two rules "j→*" and "aː→ɚ" in Step 3.

RESULTS

This article has described an algorithm used to generate a domain knowledge base automatically, which is used for a model of the skill that a student must acquire in order to be proficient at reading *katakana*. The algorithm was tested on 850 entries from a *katakana*-to-English dictionary generated previously (Kang and Maciejewski, 1992). When their IPA strings were divided in order to balance the number of syllables in both the Japanese and English entries in Phase I, approximately 92% of the 850 entries of the dictionary had the same number of partitions and were correctly matched. The comparison of the divided phoneme strings generated a set of 180 phonological rules. The remaining 8% of the dictionary entries were analysed using the algorithm in Phase II by using the rule base generated in Phase I. When these 70 entries were tested in Phase II, only one incorrect phonological rule was generated, which occurred in the following case: the *katakana* word "*haiaraki*" vs its English origin word "hierarchy".[4] The use of the A* algorithm with the defined heuristic information resulted in a 71% reduction on average in the number of nodes generated when compared to the breadth-first search algorithm, which consequently made the Phase II algorithm run 20 times faster. The 204 rules generated in Phases I and II were modified by Phase III. Approximately 40% of the rules had a multiple phoneme source or target. These multiple phoneme rules were decomposed into several single phoneme rules using the existing rules. In order to prevent any incorrect rule from being generated, this process was divided into three steps using progressively less restrictive conditions in each step. The complete resulting rule base consisted of 146 rules (Kang and Maciejewski, 1992), a sampling of which are listed in Table 3.

To verify that this rule base represents an accurate model of the knowledge required to read *katakana*, it was evaluated on test data consisting of 2234 *katakana* words and their

Table 3. Examples of the phonological rules created by matching the IPA equivalent of the *katakana* word with that of its English origin

	Data		Example			
Rule*	Probability (%)	Number of occurrences	*Katakana* word	English word	IPA (JAP)	IPA (ENG)
*→w	63	5	シーケンス	sequence	ʃiːk(*)ensu	sikwens
a→ə	36	156	ディジタル	digital	didʒitaru	dIdʒətəl
a→ɚ	24	103	コンピュータ	computer	koNpjuːta	kəmpjutɚ
a→æ	34	145	セラミック	ceramic	seramiQku	səɹæmIk
aː→ɚ	69	43	サーマル	thermal	saːmaru	θɚməl
b→v	34	44	バルブ	valve	barubu	vælv
dʒ→z	19	10	ビジー	busy	bidʒiː	bIzi
e→ə	28	58	ドキュメント	document	dokjumeNto	dakjəmənt
h→f	21	8	ヒューズ	fuse	hjuːzu	fjuz
i→ə	18	63	サービス	service	saːbisu	sɚvəs
i→I	48	165	レジスタ	resistor	redʒisuta	ɹIzIstɚ
o→*	50	208	キーボード	keyboard	kiːboːdo	kiboɹd(*)
o→ə	20	81	セッション	session	seQʃoN	seʃən
o→a	15	64	プロセス	process	purosesu	pɹases
r→l	57	270	ライン	line	raiN	laIn
s→θ	4	11	レングス	length	reNgusu	leŋθ
ʃ→s	47	33	システム	system	ʃisutemu	sIstəm
ʧ→t	42	14	チューブ	tube	ʧuːbu	tjub
u→*	91	609	チェック	check	ʧeQku	ʧek(*)

*The context for these rules is not shown here for the sake of clarity.

English origins collected from the CD-ROM version of the Sci Terms Dictionary (Ministry of Education, Science, and Culture of Japan, 1988). When their IPA strings were compared by considering all of the possible transformations, the knowledge base identified 98.2% of the required transformation rules so that approximately 90.4% of the *katakana* phoneme strings were successfully transformed into the same IPA string of the corresponding English origin. The remaining 9.6% of the dictionary entries were analysed, and it was found that 89.8% of the entries not successfully transformed were due to the absence of only one phonological rule required for complete transliteration, where 64.4% of these missing rules were vowel rules. This is not surprising because there is a very poor correspondence between Japanese and English vowels due to the existence of only five vowel phonemes in Japanese. In fact, it has been previously shown that infrequently occurring vowel transformation rules have little or no effect on a student's ability to comprehend *katakana* (Maciejewski and Kang, 1994). Therefore, the knowledge base of phonological transformation rules generated here can be effectively used to enable the tutoring system to assess a student's proficiency and to individualize their instruction.

CONCLUSIONS

The goal of this work was the development of a domain knowledge base that could be used in a Japanese language ITS. The domain knowledge represents a model of the expertise that a student must acquire in order to be proficient at reading one of the distinct orthographies of Japanese, known as *katakana*. It was shown that the algorithm presented here was able to generate a knowledge base of the phonological transformation rules automatically, which represent a model of the inverse transformation from Japanese phonology back to the original English pronunciation. The knowledge base was subsequently used to generate overlay student models for an ITS and to analyse the effects of rule frequency on student acquisition rules (Maciejewski and Kang, 1994).

NOTES

[1]A transitional sound made as the vocal organs move towards or away from an articulation.

[2]A minimal unit of rhythmical time equivalent to a short syllable.

[3]The phonological rule ie →/#_s will be decomposed into i→j/#_e and e→e/i_s by Phase III.

[4]When the *katakana* and English IPA strings modified from Step 2 in Phase I, i.e., /haiaraːki/ and /haiərar*ki/, were matched, the phoneme /ai/ was incorrectly associated with /aiər/, and the phoneme /aː/ with "*".

REFERENCES

ANDERSON, J. R. (1988) The expert module. In Polson, M. C. and Richardson, J. J. (eds), *Foundations of Intelligent Tutoring Systems*, pp. 21–53. Hillsdale, NJ: Lawrence Erlbaum Associates.

CARR, B. and GOLDSTEIN, I. (1977) *Overlays: A Theory of Modeling for Computer Aided Instruction*. Cambridge, MA: MIT, Artificial Intelligence Laboratory.

CATFORD, J. C. (1988) *A Practical Introduction to Phonetics*. New York, NY: Oxford University Press.

GOVE, P. B. (ed.) (1963) *Webster's Seventh New Collegiate Dictionary*. Springfield, MA: G. and C. Meriam Company.

KANG, Y.-S. and MACIEJEWSKI, A. A. (1992) Data on English to Japanese transliteration of technical terminology. *Technical Report TR-EE 92-34*, Purdue University, West Lafayette, IN, USA.

KOKURITSU KOKUGO KENKYUUJO (1964) *Gendai-zasshi 90shu no yoogo yooji (3)*. Number 25.

MACIEJEWSKI, A. A., and KANG, Y.-S. (1994) A student model of katakana reading proficiency for a Japanese language intelligent tutoring system. *IEEE Trans Syst Man Cybern* SMC-**24**, 1347–1357.

MACIEJEWSKI, A. A. and LEUNG, N. K. (1992) The Nihongo Tutorial System: An intelligent tutoring system for technical Japanese language instruction. *J Comp Assist Lang Learn and Instruct Consort* **9**, 5-25.

MANDL, H. and LESGOLD, A. (eds) (1988) *Learning Issues for Intelligent Tutoring Systems*. New York, NY: Springer-Verlag.

MILLS, D. O., SAMUELS, R. J. and SHERWOOD, S. L. (1988) Technical Japanese for scientists and engineers: Curricular options. *Technical Report MITJSTP WP 88-02*, MIT, Cambridge, MA.

MINISTRY OF EDUCATION, SCIENCE AND CULTURE, Japan (ed.) (1988) *Japanese Scientific Terms*. Ministry of Education, Science, and Culture of Japan.

NILSSON, N. J. (1982) *Principles of Artificial Intelligence*. New York, NY: Springer-Verlag.

PARK, O.-C., PEREZ, R. S. and SEIDEL, R. J. (1987) Intelligent CAI: Old wine in new bottles, or a new vintage? In Kearsley, G. (ed.), *Artificial Intelligence and Instruction*, pp. 11–45. Reading, MA: Addison Wesley.

POLSON, M. C. and RICHARDSON, J. J. (eds) (1988) *Foundations of Intelligent Tutoring Systems*. Hillsdale, NJ: Lawrence Erlbaum Associates.

RICKEL, J. W. (1989) Intelligent computer-aided instruction: A survey organized around system components. *IEEE Trans Syst Man Cybern* SMC-**19**, 40–57.

SHIBATANI, M. (1990) *The Languages of Japan*. New York, NY: Cambridge University Press.

SWARTZ, M. L. (1992) Issues for tutoring knowledge in foreign language intelligent tutoring systems. In Swartz, M. L. & Yazdani, M. (eds), *Intelligent Tutoring Systems for Foreign Language Learning*, pp. 219–233. Berlin: Springer-Verlag.

VANCE, J. T. (1987) *An introduction to Japanese Phonology*. Albany, NY: State University of New York Press.

WENGER, E. (1987) *Artificial Intelligence and Tutoring Systems*. Los Altos, CA: Morgan Kaufmann.

YAZDANI, M. (1987) Intelligent tutoring system, An overview. In Lawler, R. W. and Yazdani, M. (eds), *Artificial Intelligence and Education (Vol. 1)*, pp. 183–201. Norwood, NJ: Ablex.