Proceedings of International Conference
on Artificial Intelligence

Applying a Knowledge Acquisition Method in Intelligent Tutoring Systems*

Alan Liu Yen-Chih Kuo Arthur M.D. Shr

Department of Electrical Engineering
National Chung Cheng University
Ming-Hsiung, Chia-Yi, 621, Taiwan
aliu@ee.ccu.edu.tw

Abstract

Collecting the course material is an important task in intelligent tutoring systems (ITSs), and a tool for collecting such information is needed. We propose a knowledge acquisition method to do this work for an ITS which uses a knowledge base to store the teaching material. Knowledge acquisition can be basically divided to three research issues: knowledge elicitation, knowledge representation, and knowledge organization. By focusing on these three concepts, we explain how these can be implemented to an authoring system for ITSs.

1 Introduction

Intelligent tutoring systems (ITSs) use a great amount of information to present a subject for a student to learn. In the process of teaching, an ITS is capable of showing the course material to the student and to judge if the student has mastered the subject or not. Depending on whether the student has learned the subject right, a next lesson will be presented. How to construct the content of teaching is a challenging work. The teaching material to be collected include what subject to teach, what concept for the students to learn, what examples to use, and how to present the material.

The ITS model that we use consists of a user modeling module, a knowledge base, and a user interface. The user modeling module is to model different users into different user models, so that the system can interact with different users with tailored responses. The knowledge base have two parts — one for lessons to present and the other for ways to use the lessons. The user interface is to show the output produced by the knowledge base according to the user model and to gather the input from the user to the user modeling module and the knowledge base.

*This work was partially supported by the National Science Council, R.O.C. under Grant NSC85-2213-E-194-033

Thus, the task of collecting the teaching material can be viewed as a knowledge acquisition activity. Knowledge acquisition consists of three research issues: knowledge elicitation, knowledge representation, and knowledge organization. We propose a method of implementing these tasks into a system which can be used for an ITS for collecting the course material. This paper focuses more on the knowledge representation issue.

Some researchers have suggested that an environment for developing an ITS is needed [1, 2]. A method only focusing on constructing a knowledge base for an ITS is also proposed [3]. Our approach is focusing on constructing a knowledge base which contains the lessons to teach, the method to use the lessons, and the policies to choose the successive lessons.

We use the following three sections to present how we incorporate the concept of knowledge elicitation, knowledge representation, and knowledge organization in building a system for collecting the content of ITSs. The fifth section presents an example and our prototype which implements this approach. The final section gives a brief conclusion and future direction.

2 Knowledge Elicitation

The research in knowledge elicitation has been conducted for years [4, 5]. There are different techniques, such as interviewing, inductive learning, explanation-based learning. For our needs, we need to have a knowledge elicitation tool which can be used to collect information on teaching material and we have chosen interviewing. The reason for choosing interviews as our knowledge elicitation method is that it is simple enough to construct a rule-base system which can imitate a system developer to collect course material. Our approach takes the following three phases:

1. stepwise decomposition
2. concepts, examples, questions gathering

3. teaching strategies formation

The first phase is to decompose a given problem, so that the work in the second phase can be performed. The main idea is that the problem can be decomposed to sub-domains, and the sub-domains can be decomposed further to sub-sub-domains. This concept is similar to requirements analysis in software engineering, and the idea of having a knowledge-based support is presented in [6]. The knowledge base contains questions to ask user for decomposition of the problem.

The rules for guiding the user in defining the sub-domains are simple and organized as IF-THEN rules in the knowledge base. The rules are to ask the user if the problem can be categorized into sub-domains. If the user answers yes, then the original domain is decomposed according to the categories defined. The decomposition process is repeated until the user cannot come up with more categories. The categories and sub-categories are organized as a hierarchical tree, which is further discussed in the next section.

The second phase is to collect the content of the teaching material. The system uses a dialog box mentioned above to interact with the user for concepts, examples, and questions. The categories produced in the first phase will be followed to guide the user to enter the necessary information. First, a concept will be asked, and then the user is encouraged to enter as many examples or questions associated with the concept.

The teaching strategies for the subject will be obtained in the third phase. The user is to decide what order each lesson should be presented, how the quizzes will be used, how the scores are to be interpreted, how the lessons should be reintroduced, etc. This information will be used by the ITS to present the lessons.

3 Knowledge Representation

The choice of a knowledge representation method depends on what knowledge to depict. In this research, we are interested in the lessons and the teaching methods which include how to present the teaching material, how to grade the student response, and how to choose the successive lessons. The material to present may include text, graphics, animation, and audio. The lessons are related to each other, and they can be categorized as concepts, examples, and questions. With these concerns, we chose the frame representation which can be implemented in an object-oriented approach, which provides the ability in modeling hierarchical and horizontal relations using classes.

We use three classes to organize the information described above — the concept class, the question class, and example class. These three classes are built on a common class, called the basic class. The following subsections explains what classes we use in order to represent those concepts.

3.1 Basic Class

The basic class is used for modeling most detailed information and it is combination of three classes — the KBfile class, the list class, and the node class. Data are organized as linked list and defined as the basic class. These classes are shown in Figure 1 which uses solid lines for the inheritance relationship and dashed lines for usage. The content of each class is discussed below.

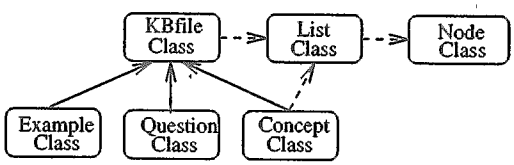

Figure 1: Basic class and other classes

The most primitive class is the node class, which defines the data structure to hold the information gathered. As shown in Figure 2, it defines two pointers — one pointing to another node and the other pointing to an object of KBfile class which contains the real data. The attribute `del_mark` is to mark if this node has been deleted or not, and the attribute `used_flag` is to record whether this node has been used or not.

Class name	Node
Attributes	next: Node pointer data_ptr: Node pointer del_mark: Node pointer used_flag: Node pointer
Operations	

Figure 2: Node Class

The list class is to represent a linked list. Shown in Figure 3, the list class consists of three pointers which point to different nodes like `head`, `tail`, and `current`. The attribute `end_status` tells whether the current position is at the end of the list. There are seven operations which are typical in operating on the linked list.

The KBfile class is to store the information for the knowledge base. As shown in Figure 4, its attribute is a list which is of the list class and its functions are basic file operations. The function `eof()` is to determine if the file reaches the end of file, and there are functions to delete a data, to recall a data just

Class name	List
Attributes	head: Node pointer current: Node pointer tail: Node pointer end_status: integer
Operations	current() end() delete() recall() reset() append(KBfile) skip()

Figure 3: List Class

deleted, to display the current data, to go to the n^{th} data from the current data, to skip the current data, and to list all the data.

Class name	KBfile
Attributes	aList: List
Operations	eof() delete() recall() display() go(integer) skip() list()

Figure 4: KBfile Class

3.2 Concept Class

The concept classes record the information associated with a particular concept to be taught. Along with the content of a lesson, the level of difficulty and relation with other concepts need to be stored. Some lessons have prerequisites and some lessons are independent.

Shown in Figure 5, the attributes in the concept class include `key` for storing the identification number for a lesson, `level` for three difficulty levels, `content` for actual lessons, `abs_relation` for the ID numbers of prerequisites, `gen_relation` for categorization with other lessons, `examples` for pointing to related examples, and `questions` for pointing to the related quizzes.

The functions `set` is for setting the content of a concept class, `print` for showing the content of a concept class, `append_blank` for creating a blank concept class, `add` for adding a new concept class (similar to performing `append_blank` and `set`), `replace`

Class name	Concept
Attributes	key: integer level: integer type: integer content: string abs_relation: integer gen_relation: integer examples: List questions: List
Operations	set() print() append_blank() replace() add() seek(integer)

Figure 5: Concept Class

for modifying the content of a concept class, and `seek` for finding a numbered lessons organized as concept classes.

3.3 Example Class

Examples are often given to explain some concepts more clearly. Thus, examples are asked to be entered after the concepts are given to the system. Since the attributes are similar to the ones in a concept class, the attributes in a concept class can be used in the example class. The most important issue is to create a correct link between a concept class and an example class.

Class name	Example
Attributes	id: integer content: string conc_ptr: Concept pointer
Operations	set() print() append_blank() replace() add()

Figure 6: Example Class

Figure 6 shows that there are just three attributes for the example class, and almost exactly the same operators (only `seek` is missing) from the concept class is used. The attribute `id` is for the identification number for each example, `content` for recording an example, and `conc_ptr` for pointing to a lesson (concept) associated to this example.

3.4 Question Class

Questions play an important role in an ITS, because it provides the interaction between the students and the system. The system determines whether the students have mastered the lessons or not by the result of asking questions. Questions are associated with the concepts which are being taught. The answers need to be attached to the questions. In addition, the reasons and explanation for the answers are to be stored with the answers in order to provide the students with another chance for teaching.

Class name	Question
Attributes	id: integer level: integer content: string answer: string reason: string conc_ptr: Concept pointer
Operations	set() print() append_blank() replace() add()

Figure 7: Question Class

From Figure 7, there are 6 attributes in the question class. The attribute `id` is to store an identification numbers for a question, `level` for the difficulty level, `content` for a question, `answer` for the answer to the current question, `reason` for why the answer is this way, `conc_ptr` for what lesson this question is associated. The operators are similar to the ones in the concept class and the example class.

4 Knowledge Organization

The purpose of organizing the course material into a knowledge base is to make the lessons flexible in terms of teaching strategies. Our goal is to present the same material to different students in different ways, so that each student can receive personalized tutoring. The knowledge organization that we have is closely related to the decomposition work in the knowledge elicitation process. The knowledge base is organized as a tree consisting of groups as the leaf nodes. The groups are the categories for the original problem, and more detail pieces of knowledge are organized as a linear representation or a hierarchical representation in the groups. Figure 8 shows this concept. If there is a certain order needed to be followed, then a linear representation is used.

The tree model is a natural result of having an

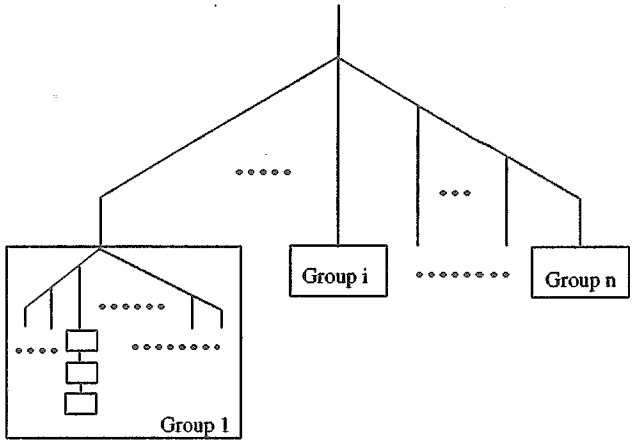

Figure 8: Knowledge organization

object-oriented structure as our model to store the information. By having this kind of organization, when a new concept is introduced, the new concept will be referred to the existing tree to see where it fits. This can help the work of knowledge elicitation by providing existing categories. It also provide a map for the new concept to be related to other concepts which are already stored in the tree.

5 Implementation

This section shows our prototype system which implements what we have discussed in the previous sections. An example on the usage of prepositions is chosen to show how different prepositions are organized and their examples are given.

5.1 Prototype System

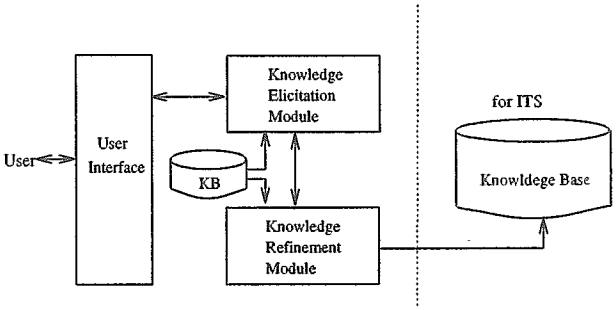

Figure 9: System overview

The purpose of implementing a prototype system is to show how our idea can be realized as a tool for collecting teaching material. The system can be divided into four parts: the user interface, the knowledge elicitation module, knowledge refinement module, and knowledge base. We have implemented the prototype on a IBM-PC compatible personal computer using

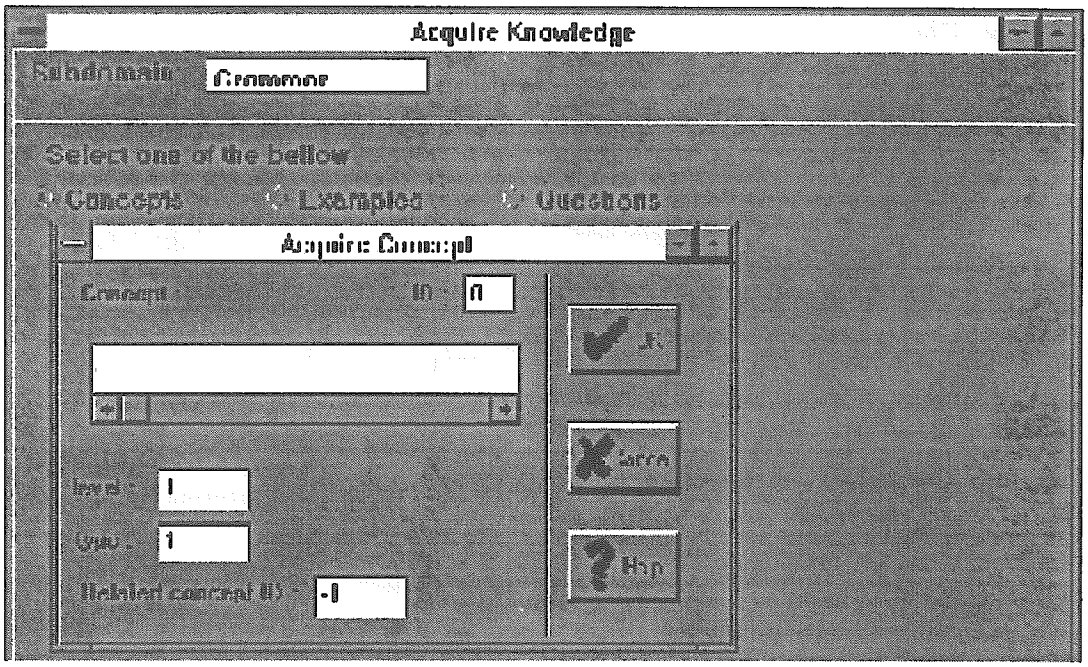

Figure 10: Dialog Box

C++ for MS-Windows 3.1. Figure 9 shows the relations between different components. The knowledge elicitation module interacts with the user through the user interface. The information collected through the module is then passed to the knowledge refinement module for checking if the information is good for the ITS to use. In these activities, a knowledge base is there to aid the work. After going through the refinement the information will be store in the knowledge base to be used by the ITS.

The user interface displays what the system wants the user to input. By using a dialog box shown in Figure 10 for the user to response, we ease the user from guessing what to enter and what to type. The knowledge elicitation module is responsible for collecting concepts, examples, questions, and strategies. The lessons are constructed from a collection of concepts which are organized as hierarchical trees. The knowledge elicitation module allows the user to enter a concept and requires the user to enter examples and questions associated to that concept. Concepts, examples, and questions are domain dependent; whereas strategies are not entirely domain dependent. According to the organization of the concepts, a default set of teaching strategies can be initially built. Teaching strategies are focused on how to present concepts, examples, and questions. With a user modeling approach, we can provide different strategies depending on different students. Thus, the system asks the user to enter the relations among the concepts and the dependencies between concepts.

The knowledge refinement module currently can only detect if duplicate concepts are entered. However, by having this module independent, we can expand this module into checking the consistency of the information entered. The knowledge base contains each concept along with examples and questions associated with it. In addition, the teaching strategies are also included in the knowledge base.

The work of collecting information can be seen in Figure 11. The flow chart displays that a new knowledge base is created when the work is just started. The knowledge elicitation part has three phases in decomposition, sub-domain collection, and strategies gathering. The knowledge base is also checked to see if it needs to be updated after new information is organized as new knowledge.

After the course material is collected and organized in a knowledge base, it is used by our tutoring system which uses a user modeling technique to present to the student as lessons. As a personal tutor to the user, the main work of the tutoring system can be summarized below:

- If a student is new, then his personal file with default values will be created.
- From the student record, the system will choose a suitable lesson to start.

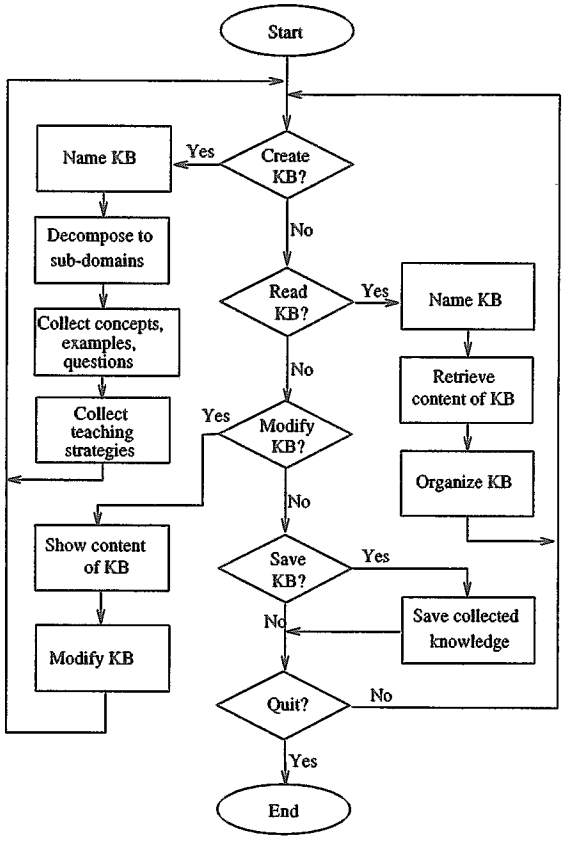

Figure 11: Flow of knowledge acquisition

- After teaching some concepts, the system will give some quizzes to check the learning situation.
- According to the results of evaluation, the system will modify the student's learning ability and teaching plan.
- After all lessons in the course plan have been presented, the record will be updated and the session is over.

This system overview is shown on Figure 12. The first phase is to initialize the student model. It is followed by an instruction phase. The evaluation phase follows that, and finally the user modeling phase takes place. A knowledge base containing the teaching material supplies the necessary information to the instruction and the evaluation modules. The student record is kept in a file, and a student model and the student information is kept during the execution.

Tutoring is divided into course instruction, test evaluation, and result analysis. By dividing the functions further, course instruction can be treated as content and example presentation. Test evaluation can be seen as quizzing and answer checking. Result ana-

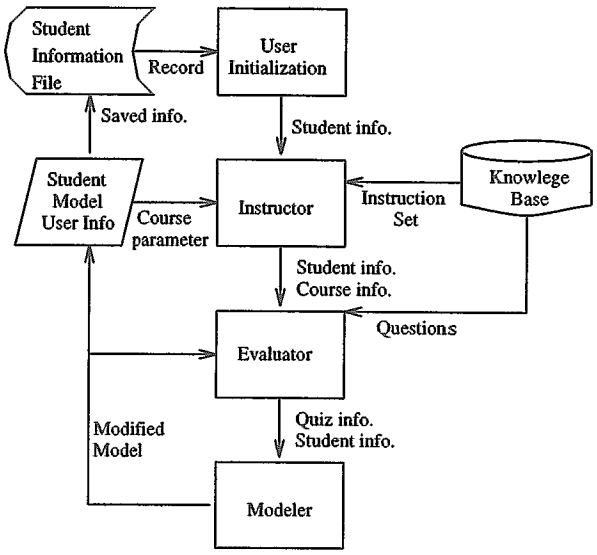

Figure 12: Overview of the tutoring system

lysis is to analyze student features, concept levels, and course strategies. This kind of categorization is implemented in the classes.

5.2 Example

We have chosen the topic of teaching prepositions in this example. The source of the content is from different textbooks from elementary English classes in the junior high school level in Taiwan. We followed the three phases for knowledge elicitation mentioned in the previous section:

Phase 1: Decomposing the original problem into sub-problems (categories) — *time, place, direction, situation*

Phase 2: Gathering concepts, examples, questions

Phase 3: Forming teaching strategies — to order the concepts.

In the first phase of knowledge elicitation, the system asks what subject to teach and gets the answer "preposition". The user is then asked to decompose the subject into smaller domains. There are four categories introduced — time, place, direction, situation. After these four categories are produced, the rest is to fill up the categories with the real prepositions and their usages, so the second phase can be started. The content of the teaching material is collected in the second phase. The third phase focuses on obtaining the teaching strategies for the subject.

In order to produce a course plan, a teaching strategy needs to be known before hand. The lessons, examples, and quizzes on a subject is collected, but

how these should be used will depend on the teaching strategy. The relations between the lessons and the subjects need to be defined. The strategies are recorded as rules in the knowledge base which uses the student model to redesign the course plan dynamically.

The teaching strategies are just a guideline for the big picture, but the detail steps of what lessons, examples, quizzes are to be presented in what order will be determined according to each student. A parameter on the student level is kept in the student model to help the number of the lessons to be introduced in a course plan. The mistakes a student make and how much time he spends on responding are also factors to evaluate a student. These are also recorded in a student model and can be used to refine the teaching strategies.

After the knowledge elicitation phase is completed, we have all the lessons represented as objects and organized as nodes in a tree. Figure 13 shows the tree. The branches bear the labels of four sub-domains and the nodes have the ID numbers for each concept. For this topic we have gathered a total of 36 concepts and 77 examples along with more than 100 questions. Table 1 displays a part of what we have gathered. Since the questions are too many to list, they are not included in the table.

For example, the first concept in the first category *time* is the preposition *at* for the concept *exact time*. An example for this concept is given as *School begins at 8:30*. The second concept in the category *time* is the preposition *in* for the sub-concept *within a time period*. The sentence *I will be back in an hour* is given as an example. The same preposition *at* is also used in the concept *place* which may have an example, *He lives at 4018 Grant Street*. In fact, the preposition *at* can also be categorized as the concept *direction* which has an example as *The boy threw a stone at the dog*. There are other concepts given to each category for *time*, *place*, *situation*, and *direction*.

Table 1: Content for course in prepositions

ID	Type	Category	Content
1	concept	time	At — exact time
	example	time	School begins at 8:30.
	example	time	Please arrive at 11 o'clock. (at exact time)
2	concept	time	In — within a time period
	example	time	I will be back in an hour.
3	concept	time	On — exact date
	example	time	He went to London on Sunday.
4	concept	time	Before, By, In, Within — future event
5	concept	time	Before, By — open period
	example	time	Please arrive by 11 o'clock. (no later than)
	example	time	Please arrive before 11 o'clock. (earlier than)
6	concept	time	In, Within — during a period of time
	example	time	Please finish your work in (within) two hours.
...	...	...	...
12	concept	place	In, On, At — address
13	concept	place	In — country or city
	example	place	He lives in Chicago.
14	concept	place	On — street
	example	place	He lives on Grant Street.
15	concept	place	At — street address
	example	place	He lives at 4018 Grant Street.
...	...	...	...
28	concept	direction	For, From — associated with action
29	concept	direction	For — toward a direction
	example	direction	The train is bound for Taichung.
30	concept	direction	From — originated
	example	direction	The train is from Taichung.
31	concept	direction	At — toward a target
	example	direction	The boy threw a stone at the dog.
...	...	...	...
32	concept	situation	With
	example	situation	I accept your invitation with pleasure.

6 Conclusion

Our research in developing a tool for collecting teaching material to organize lessons and teaching strategies in a knowledge base is presented in this paper. The goal of having a knowledge base to contain the teaching material is to make ITSs independent from problem domain. A prototype system has been built for realizing this idea, and an example of teaching the usage of prepositions has been presented. Our future goal is to work on the knowledge refinement module for checking consistency and learning from experiences.

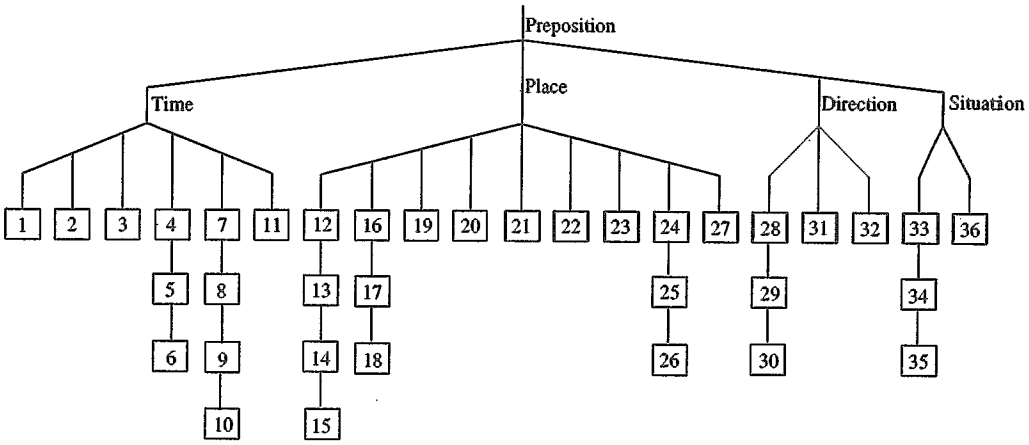

Figure 13: Tree for prepositions

References

[1] K. Yoon and C. Wang, "Authoring system for the development of ITS," in *Proceedings of 1994 IEEE Region 10's Ninth Annual International Conference. Theme: Frontiers of Computer Technology*, pp. 97–101, 1994.

[2] A. Zekl and I. Morschel, "Embedding authoring support in an ITS for the learning of object-oriented programming," in *Proceedings of 1994 IEEE First International Conference on Multi-Media Engineering Education*, pp. 59–64, 1994.

[3] C. Bloom, P. Bullemer, R. Chu, and M. Villano, "A task-driven approach to knowledge acquisition, analysis and representation for intelligent training systems," in *Proceedings of 1992 IEEE International Conference on Systems, Man and Cybernetics*, pp. 509–514, 1992.

[4] H. Motoda, R. Mizoguchi, J. Boose, and B. Ganines, "Knowledge acquisition for knowledge-based systems," *IEEE Expert*, vol. 6, pp. 53–63, Aug. 1991.

[5] B. R. Gaines and M. L. G. Shaw, "Eliciting knowledge and transferring it effectively to a knowledge-based system," *IEEE Transactions on Knowledge and Data Engineering*, vol. 5, pp. 4–14, Feb. 1993.

[6] A. Liu and J. J. Tsai, "A Method for requirements analysis and knowledge elicitation," *International Journal of Artificial Intelligence Tools*, vol. 5, nos. 1&2, pp. 167–183, 1995.