Knowledge Modelling for an Intelligent Tutoring System (ITS) Domain: Precalculus

Rathiah Hashim Email: rudyhashim@hotmail.com
Abd Manan Ahmad Email: manan@fsksm.utm.my
Mohd Radzi Mohamed Yunus
Fakulti Sains Komputer & System Maklumat
Universiti Teknologi Malaysia

Abstract- **This paper presents a knowledge model (KM) of an ITS with Precalculus as the domain knowledge. The Knowledge Model is represented by Semantic Net, organized as interconnected knowledge entities in the form of nodes. The comprehensiveness of the knowledge domain depends largely on the richness of the interconnectivity of nodes. A working prototype, built using an object oriented paradigm, exists that shows the correctness and feasibility of this model. Any standard Java compatible World Wide Web (WWW) browser will be able to access the knowledge model with a simple Graphical User Interface (GUI) to view the content of its structure.**

1. INTRODUCTION

The first ITS is generally acknowledge to have originated in the work of Carbonell at MIT in the early 70s[1]. By 1988, the book Foundations of Intelligent Tutoring Systems was published, in which Bruns and Capps [10] summarized the 18 years of research in ITS with an anatomy that creates convenient classification of the research and development dimensions. Expert, student modeling and tutor are the three models that have been generally believed to be the key components to the success of ITS. The performance of an intelligent system relies mostly on the knowledge it contains and the way the knowledge is processed. Defining and representing knowledge that a system needs to perform intelligent tasks is a central issue in Artificial Intelligence (AI) research.

Researchers have classified knowledge entities in three cognitive groups:

i) Procedural knowledge entity conveys how a task is performed in the knowledge domain.

ii) Declarative knowledge entity is a fact related to a concept.

iii) Qualitative knowledge entity implies a causal relationship among other knowledge entities.

Broadly speaking, knowledge is useful information about the world and the state of the world. Writing it down and embedding it into machines so that knowledge can be manipulated (processed and reasoned) is known as the representation problem [1]. There are various models for representing knowledge in the ITS community but in practice, only a small number of schemes have been used to represent knowledge [9], eg. rule-based, frame, and semantic net.

2. SEMANTIC NET AS KNOWLEDGE REPRESENTATION SCHEME

Semantic Net is well suited for representing knowledge of a hierarchical nature. The semantic network provides a graphical view of a problem's important objects, properties and relationships. It contains nodes that can represent objects, object properties or property values and arcs that connect the nodes. The arcs represent the relationships between the nodes. There are labels for both the nodes and arcs that clearly describe the objects represented and their natural relationship.

Knowledge representation is usually associated with rules, but they do not necessarily always provide the most appropriate representation [9]. To represent the organizational structure of a subject like Mathematics, semantic net modeling scheme is more relevant and appropriate compared to rule-based and frame. In this modeling scheme, knowledge is organized as an interconnected network of nodes, which may represent various knowledge entities such as concepts, examples, exercises, applications, etc.[4].

The richer the interconnectivity among nodes, the more comprehensive is the representation of the knowledge domain. This view is in accord with object-oriented methodology, is easier to modify for other domains, and is easier to implement and maintain compared to the previous model. Moreover, the graphical portrayal of knowledge can also be somewhat more expressive than other representation schemes [8]. This probably accounts for the popularity and the

diversity of representation models for which they have been employed.

Similar study that uses Semantic Net as knowledge representation (KR) schemes in designing expert modules of tutoring systems can be found in many earlier systems such as Number Systems Tutor – NST, SCHOLAR, BUGGY [17], BIP-II [16], WUSOR-III [14], SCENT-3 [6], CIRCSIM-TUTOR [7] shows the significance of the study's modeling approach.

This research focused on the development of a knowledge model for an ITS using Semantic Net. Each node in the Semantic Net represents a knowledge entity or a class of closely related entities (similar to a section or a chapter in a textbook). The edges represent various types of relationships among nodes. The number and types of relationships depend partly on the nature of the knowledge domain entities and partly on the intended functionality of the system. Semantic Net helps in defining the skeleton of the Knowledge Model.

2.1 SEMANTIC NET FOR REPRESENTATING PRECALCULUS

The hierarchy of the nodes in Semantic Net is arranged such that it starts from the most general class of concepts, which is the name of the subject domain, in this case, Precalculus. This class is represented by root rode. The next layer of nodes contains the first subject classification in the knowledge domain. In this case, they are - (1) Graphs, (2) Functions, (3) Trigonometry, (4) Equations, and (5) Geometry - represented by nodes in the first layer. The subject domain will be subdivided into more specific subtopics as it goes down layer by layer. The partial structure of Semantic Net for Precalculus is depicted in Figure 2. This structure is compiled from a few popular textbooks for the subject.

A driver program will be created to facilitate the building of Semantic Net. To ensure that the model is platform independent, the program will be written in Java and purposely build for a Web-based ITS. An input file will be prepared to build the semantic net node by node, based upon the structure of the knowledge domain identified in phase two of the project. It contains knowledge entity of the domain knowledge which represents the organization of the whole domain. *SemanticNet* class is built to implement the network and it does not contain hard-coded reference to the knowledge base. It only contains symbolic reference to the knowledge entities.

3. KNOWLEDGE MODEL

Logical structure of the KM is shown in Figure 2. KM consists of two main parts, Semantic Net and KM Toolset, a set of modules which perform various operations and inquiries on Semantic Net. These modules are implemented as methods at each node of the Semantic Net.

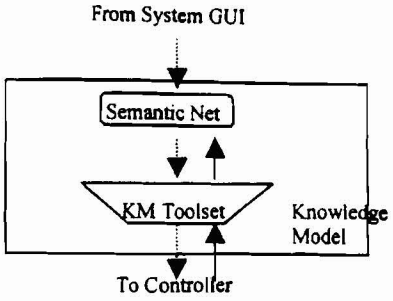

Figure 1: The structure of Knowledge Model

4. NODES IN SEMANTIC NET

The three types of nodes in Semantic Net are Classification, Container and Wrapper. Symbolic references to the domain contents in the Knowledge Base will be kept only in Wrapper nodes. Others are used for classification purposes, as explained below.

a. Classification Nodes

The classification nodes act as a navigational and classification tool where a general subject domain is refined into more specific topics. An analogy for this structure is the Subject-Chapter-Subsection hierarchy in a typical textbook. It cannot be a leaf node and that each is specified by the values of its fields:

- a field to identify the node
- fields to identify the parents of this node
- fields to identify the child nodes of this node.

The parent of a Classification node can be only of Classification type. It is the node in the upper layer of the Semantic Net. The child node is the current node, representing a more specific topic for the subject. All the Classification and Container nodes, except the root node, have one and only one parent. The root node does not have any parent. The current node is a child node for the parent node. It can be either of Classification type or of Container type. A node can have an arbitrary number of child nodes.

b. Container Nodes

Container nodes is the last of the sequence of Classification nodes in Semantic Net with their the parents are of classification type with each having one and only one parent. A Container node plays the role of a structured table-of-content for the learning materials, referred to as tutoring objects, that could be found in the Knowledge Base, corresponding to a particular subject. In Semantic Net, tutoring objects are represented (wrapped) by a class of nodes which collectively call Wrapper nodes, and are further discussed in the following subsection. There are five types of Wrapper

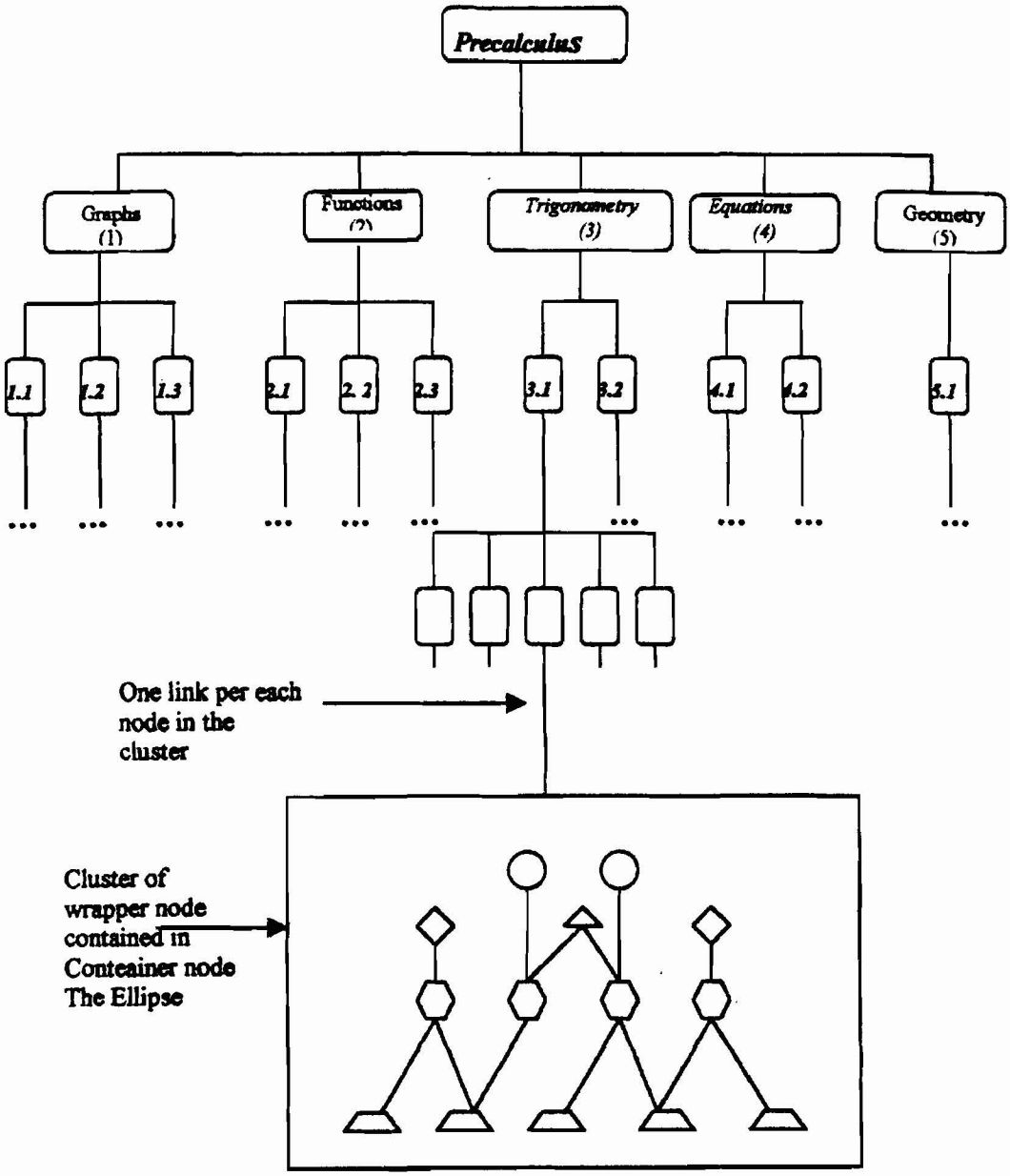

Figure 2: Part of a sample Semantic Net for Precalculus

nodes which correspond to the same type of tutoring objects.

A Container node contains its tutoring objects in the form of links to the corresponding Wrapper nodes in Semantic Net. The links are classified in five groups whose corresponding fields in a Container node are named Introduction, Definitions, Exercises, Examples, and Techniques. In a sense, a Container node is the hanging point for the cluster of Wrapper nodes which are all conceptually related to one specific subject. The internal structure of this node is explained in the figure2.

Title:	*Title of this Container node.*
Parent:	*link to the parent of this node.*
Introductions:	*list of links to Introduction nodes.*
Definition:	*list of links to Definition nodes*
Examples:	*list of links to Example nodes.*
Techniques:	*list of links to Technique nodes.*
Exercises:	*list of links to Exercise nodes.*

Figure 2. Internal structure of a Container node in Semantic Net

c. Wrapper Nodes

The tutoring objects are represented by Wrapper nodes. Each Wrapper node encompasses a reference to a tutoring object and contains information about the relations of this tutoring object with other tutoring objects. The nature of this reference is symbolic in the sense that it is up to the Knowledge Base to provide the corresponding actual tutoring objects. Wrappers provide an object-oriented categorization of the knowledge domain entities. The five classifications of Wrapper nodes, Introduction, Definition, Examples, Techniques and Exercise reflect the pedagogical role each type plays in a tutoring process. The internal structure of each Wrapper depends on its type. A Wrapper may have more than one parent. This is because a specific tutoring object may correspond to more than one subject, with various degrees of strength. Besides the parent links, each Wrapper may have links to some other Wrapper node

4.1 RELATIONS

The relation between two nodes can be of various types. Each type serves to establish a certain dependency relationship. These dependencies are used for such goals as classification, navigation or accessing a certain entity in the Semantic Net. Each relation type has an inverse type that is usually different from the relation type. The relation between a parent and a child is of one type, and the relation between that child and the parent is of the inverse type. The types of relations are contains, means-end, prerequisite, similar-to, explanation-for, example-of, exercise-for, and necessary.

5.0 IMPLEMENTATION

Semantic Net is implemented as a class named SemanticNet. It includes the root node, and also modules for building the Net from some input data. To build the SemanticNet class, a driver program named KM.java is used. The Semantic Net can then be modified by adding or removing nodes or edges.

5.1 Node Class

The base class for the Semantic Net is the NodeModel where all nodes share a common set of properties and features. Rather than being a NodeModel type, every node directly inherits from.it. The NodeModel class contains the following fields (attributes and methods):

```
class NodeModel
(       //attribtes
String title;
Vector parents;
String type;
int difficulty;
double learnt;
int level;
String group;
boolean visited;
double limit;
//methods go here.
addParent();
removeParent();
.....
}
```

5.2 Edge Class

This class contains the necessary information about the relations between nodes which also serves as a pointer. The edge class is declared as:

```
class Edge
{ //attributes
NodeModel target;
String target_type;
double relevance;
String type;
// methods go here.
.....
}
```

5.3 Classification Class

Classification nodes are instances of Classification class. A classification node functions as a branching point along a route from the root to a Container node. The formal specification of the Classification class looks like the following:

```
Class Classificaiton extends NodeModel
{
Vector child;
// Methods go here.
.....
}
```

As we see, Classification class has a vector field, child, in addition to the fields which it inherits from NodeModel class.

5.4 Container Class

Container nodes contain tutoring objects. A Container node is an instance of Container class. This class extends the NodeModel class and adds some new fields to it:

```
Class Container extends NodeModel
{
Vector definitions;
Vector examples;
Vector introduces;
Vector methods;
Vector exercises;
// Methods go here.
.....
}
```

Each vector field in the Container class consists of a set of Edge objects which refer to a set of similar Wrapper nodes. Thus:

Vector definitions contains links to a set of Wrapper nodes of type Definition.
Vector examples contains links to a set of Wrapper nodes of type Example.
Vector introduces contains links to a set of Wrapper nodes of type Introduce.
Vector methods contains links to a set of Wrapper nodes of type Method.
Vector exercises contains links to a set of Wrapper nodes of type Exercise.

5.5 Wrapper Classes

A Wrapper node is a node which has a reference to domain contents is specified by the type of the Wrapper node. The mapping of the references in the Wrapper nodes to the corresponding domain contents in the Knowledge Base is done by the KB Interface. In the implementation, the title field of a node is used for referencing. Each Wrapper node is an instance of its corresponding Wrapper class. We have identified five types of Wrapper classes corresponding to five types of Wrapper nodes; Introduction, Definition, Technique, Example, and Exercise. All Wrapper nodes, except Example and Exercise nodes, contain a set of references to its Exercise nodes. Since Wrapper classes inherit from NodeModel class, the only extra attributes needed are the corresponding Example references (or Exercise references in case of Example class). This achieved by a vector field, examples. So, for instance, the definition of the Definition class is :

```
Class Definition extends NodeModel
{
Vector examples;
// Methods go here.
.....
}
```

The definition of other Wrapper Classes follow similarly.

6. SUMMARY AND CONCLUSION

KM is the component that is concerned with the modeling and representation of knowledge in the system. The main focus of the paper is the building of a KM using Semantic Net as the representation scheme and it is also shown how a generic knowledge domain is represented and modeled. This model is applicable to domains that are hierarchical in nature, best suited for Mathematics.

REFERENCES

[1] Wu, Binghui H., "Artificial Teaching Assistant: A framework for Intelligent Tutoring Systems With Abstract Knowledge", PhD Thesis, Lehig University, 1998.

[2] Ritter S., Koedinger K.R., "Towards Lightweig Tutoring Agents", ITS'96 Workshop, Montreal, June 10th 1996.

[3] Zhou, Gang, Wang, Jason T.L., and Ng, Peter A., "A Knowledge-Based Tutoring System for SQL Programming", IEEE journal, pp. 352-358, 1994.

[4] Nezami, Ahmad Reza, "DiscMath: An Intelligent Tutoring System for Discrete Mathematics", Thesis, University of New Brunswick., 1996.

[5] Durkin, John, "Expert Systems : Design and Development", Prentice-Hall, Inc, Englewood Cliffs, N.J., 1994.

[6] McCalla, G., "The central importance of student modeling to intelligent tutoring", In the New Direction of Intelligent Tutoring Systems, Springer-Verlag, 1992.

[7] Khuwaja, R.A., Evens, M.W., Rovick, A.A., and Michael, J.A., "Knowledge representation for an intelligent tutoring systems based on a multilevel acusal model", In Intelligent Tutoring Systems, New York, Springer-Verlag, pp.125-224, 1992.

[8] Patterson, Dan W., "Introduction to Artificial Intelligence and Expert System", Prentice-Hall, Inc, Englewood Cliff, N.J., 1990.

[9] Marshall, Garry, "Advanced Students' Guide to Expert System", Oxford : Heinemann Newnes, 1990.

[10] Burns, H.L., and Capps, C.G., "Foundations of intelligent tutoring system", In Foundations of Intelligent Tutoring Systems, M.C. Polson and J.J. Richardson, Eds. Lawrence Erlbaum Assoc., pp. 1-19, 1988.

[11] Wenger, Etienne, "Artificial Intelligence and Tutoring System: Computational and Cognitive Approaches to the Communication of Knowledge", Morgan Kaufmann Publishers, Inc, Los Altos, California, 1987.

[12] Brachman, R.J., and Levesque, H.J,. "Introduction. In Readings in Knowledge Representation", R.J. Brachman and H.J. Levesque, Eds. Morgan Kaufmann, Los Altos, California, 1985.

[13] Buchanan, B. G., and Shortliffe, E. H., "Rule-based Expert System : The MYCIN Experiments of the Stanfort Heuristic Programming Project", Addison-Wesley, Reading, MA, 1984.

[14] Goldstein, I., "The genetic graph: A representation for the evolution of procedural knowledge", ITS collection in ITS, 51-77, 1982.

[15] Sleeman, D.H. and Brown, J.S. (Eds), "Intelligent Tutoring Systems", Academic Press, London, 1982a.

[16] Wescourt, K. Beard, M., and Gould, L., "Knowledge representation for an intelligent tutoring systems based on multilevel acusal model", In Proceedings of the National ACM conference, New York, ACM, pp. 234-240, 1977.

[17] Brown, J. and Burton, R, "Diagnostic models for procedural bugs in basic mathematical skills", ICAI 10, Bolt, Beranek and Newman", 1977.

[18] Carbonell, J.R., " AI in CAI: An Artificial Intelligence Approach to Computer-Aided Instruction", IEEE transactions on Man-Machine System 4, pp. 190-192, 1970.