第6卷第1期 广西科学院学报 Vol.6, No.1
1990年2月 Journal of The Guangxi Academy of Sciences Feb 1990

用图形知识库方法建立智能教育系统

Stewart N.T.Shen

Carlos de Mora

刘连芳

我们研制了图形知识库方法并为其构造了一些工具。这种方法可使结构化的和分布式的知识更方便地表达和检索。结构化的知识表示，可以以层次和网状形式表示教学系统中的知识。知识的分布式表示可以用于在图形背景上描述知识。有时，这种形式对于有效的智能教育系统也是必要的。我们的方法允许相同或不同的类型的知识表示相互嵌套。在多任务太阳工作站的大屏幕站上实现这些设想容易使用是建立智能教育系统的有力工具。我们较详细地描述了图形知识库方法，并用简略的教育系统的例子，加以图解说明，这些系统也是用上述设想设计的。

THE GKB METHODOLOGY AS AN INTELLIGENT TUTORING SYSTEM BUILDER

Stewart N.T.Shen

Carlos de Mora

Lianfang Liu

Old Dominion University, USA

Universidad Nacional de Educacion a Distancia,Spain

Guangxi Academy of Sciences, China

Abstract-- We developed the Graphic Knowledge Base (GKB) methodology and implemented some shells for this methodology. The methodology allows convenient representation and retrieval of structured and distributed knowledge. The structured knowledge representation allows hierarchical and network representations of knowledge that can be used in tutoring systems. The distributed knowledge representation allows the description of knowledge over a pictorial background which again is sometimes necessary for effective tutoring systems. Nesting of the same or different types of knowledge representations is allowed. Implemented on the large screen, multitasking Sun workstation these shells are an easy to use, powerful, and effective intelligent tutoring system builder. We describe the graphic knowledge base methodology in some detail and illustrate it with abbreviated examples of tutoring systems built with the shells that we have developed.

1.INTRODUCTION

With the rapid advances in technology, more and more sophisticated systems have been implemented and placed in the market. The learning curves for the many sophisticated systems are not necessarily very high, but the fact that there are so very many such systems makes it rather difficult for users to keep up-to-date and to be able to take full advantages of these systems.

In recent years, the discipline of Expert Systems have become popular

and such systems can serve in place of human experts to diagnose problems, provide recommendations, and so on[2,8,13,15,16]. There are many advantages of using such exprt systems for tutoring, particularly for people facing the problems of being overwhelmed by numerous unfamiliar equipment and systems that they can or must use.

Most of the expert systems for tutoring or othorwise are conversational. The more traditional tutoring systems often follow some definite paths depending on the expected answers from the users (students). They are not flexible enough to allow the users to select own paths of learning at ease. The users can not look ahead and can not conveniently select just what they want. The more recent intelligent tutoring systems can respond to unexpected answers and even unexpected questions. However, such systems are rather limited in their capabilities in responding to the unexpectedness.

People usually can understand and remember better with a picture. Graphic techniques are getting used more extensively in computer software engineering, database design, and many other disciplines for this reason. When large picture is given, people can also decide from a broad context more comfortab ly which parts are of interest and which other parts are not. Such decisions are often best left with the people, who has natural intelligence, rather than with the computer. There is some knowledge that the popular knowledge representation methods such as logic[1,4,10,11,14], frame/semantic network[7,9,17], and production rules [6,12] can not represent effectively, For example, one can not describe precisely a certain tree on some mountain without showing a picture of it and its surroundings. These ideas are the basis of our methodology and the shells for it, which can be used easily and effectively for building tutoring systems. Our approach in tutoring is some what a mixture of the traditional frame-oriented CAI and the modern information-structure oriented CAI as defined by Carbonell[3].

Our methodology provides a user-friendly approach to represent knowledge graphically on the computer screen. Seeing such graphics a user can learn sophisticated systems more effctively and can choose the desired paths or parts to learn at will. In particular, when implemented on a multitasking workstation and if the object being tutored is a computer software on the same computer, then the user can even experiment the software simultaneously on the same computer as he is being tutored. This will make the tutoring system even more effective.

2. THE GRAPHIC KNOWLEDGE BASE METHODOLOGY

Our methodology includes two components. One component is the representation of knowledge. Another component is the architecture of the system that supports such a representation and the corresponding applications. We describe these two components in this section. We have also implemented some shells to illustrate our methodology. They are described in the next section.

2.1. Knowlebge Representation in the Methodology

A tutoring system requires that the knowledge on the object be represented in some way so that the student can easily absorb the knowledge and become versed in it. Knowledge representation in our methodology currently takes two forms. They are the *structured Graphic Knowledge Base* (*SGKS*) and the *Distributed Graphic Knowledge Base* (*DGKB*) represetations. Both of them communicate with users using graphics represented on what we call a Canvas. The basic constituents of knowledge representation in our methodology include *Elements* which represent objects, Text Files which provide detailed textual descriptions for elements, *Arrows* which form structures in the SGKB representation, and *Pictorial Images* which from the *Background* of the DGKB representation. Different knowledge bases can be interconnected disregarding their forms of representation.

2.1.1. Basic constituents of the knowledge representation

2.1.1.1. Elements, the basic objects

Elements are the basic objects in the graphic knowledge base that may appear in a picture and may be described by some texts. Each element has a name. Each element may have an associated text file that provides the desired description. For convenience, an element can be split into several. An element can also represent another graphic knowledge base and a user can proceed from an element to another graphic knowledge base.

Elements also provide the entitles for search during tutoring. A user can search visually in the graphic representation of the knowledge base or rely on the assistance of the computer when the name of the element is provided to the shell.

The *names of elements need not be unique in any graphic knowledge* base. there may be situations in which it is more appropriate that different elements have the same name. The graphic contexts and/or the associa

-ted textual descriptions can differentiate these elements. This situation corresponds to our knowledge of the real world environment. For example, a certain part may contain four identical screws. There is typically no need to name these screws differently except to show them in different positions of the part.

2.1.1.2. Text files, the detailed descriptions of elements

Natural language is a powerful tool for representing and communicating knowledge. Thus we allow the use of natural language and have the description of elements in natural language stored as text files. Mathematical formulas representable as text strings are of course also allowed. Every element may have an associated text file. With text files, descriptions of arbitrary lengths can be accommodated and can be retrieved and updated conveniently on the computer with the help of a text editor. Weh ave developed some conventions for aesthetic representation and efficient search, but they are somewhat subjective and implementation dependent thus we are not to describe them in here.

2.1.1.3. Arrows, the structure formation ingredient

We use arrows to form structures in the SGKB. In most cases, arrows leading from one element to other elements indicate that these other elements are successors of subcomponents in some sense. However we use horizontal arrows, arrows coming out from an element in the graph horizontally, at least initially, to link to other elements that are split from the source element. As mentioned before, an element can be split into several just for convenience. This is typically to highlight different aspects of the whole element. These split elements map be needed by the user on their own, without needing the other parts of the whole element simultaneously. Arrows are used only in the SGKB, not in the DGKB.

The structured graph of a SGKB is to be displayed to a user to facilitate the user's understanding of the problem domain and in his searching the knowledge base visually for desired knowledge. This is different from many other systems such as the database management systems and the systems using semantic networks. Such systems do utilize structures but the structures are not displayed to the users for their benefit in understanding the systems. The hypertext technology[5] is very powerful and is liked by many users. But it typically does not provide the structural view visually to the user thus is weaker in a way than our methodology.

2.1.1.4. Pictorial images, the background

Pictures are more effective in representing knowledge than any other method in many situation. No one can give a very clear description of most physical objects in practically any method without using a picture. Thus in our methodology we include pictorial images as the backgrounds of the DGKBs.

The pictorial images may be drawn by a teacher, or the creator of the tutoring system, with a mouse or other devices with the help of the graphic packages of a computer system. They may also be the products of some digitizers. Eaxmples of the pictoral images are the sketch of a building and a digitized photo of an automobile.

Pictorial images are used to serve as the backgrounds and elements distributed over a background form a DGKB. We allow appropriate backgrounds to be superimposed to form another background. As an example, a background of the outline of a house may be superimposed with another background of the windows of the house to form a new background.

2.1.2. The Structured Graphic Knowledge Bases

The first type of knowledge representation in our metnodology is the SGKB. Our philosophy in here corresponds to a cognitive model of the communication process which explains that reading is taking the linear stream of text comprehending it is by structuring the concepts hierarchically, and absorbing it into long-term memory is in the form of a network. Thus the representation of knowledge in the structured form is very easily understood by humen and is easy for them to remember. It is very suitable for tutoring systems.

The basic building blocks of the graphic knowledge bases are elements. Every element is graphically represented by a box in this type of knowledge representation. Inside the box is the element's name. The detailed description of an element is given in its associated text file. An element may be expanded to one or several other graphic knowledge bases, either structured or distributed.

The structure of a SGKB is formed by using arrows to link the boxes. The structure takes the form of networks. The interpretation of a network is left to the tutoring system creator and the user. However, in our methodology we recommend to represent successors, subcomponents, special cases, and so on as the descendants of an element and typically draw

them a little lower than the parent in the graph.

If a SGKB is represented as we recommend then the knowledge associated with a parent and its split twins are generally considered the generic knowledge and the knowledge associated with the descendants are specific knowledge. In case of a conflict, the specific knowledge supersedes the gene-ric knowledge. We give an example of such representation in Figure 1.

The example in Figure 1 illustrates part of a SGKB which is part of a tutoring system teaching users about how to prepare a technical paper using the troff, table, picture, etc. facilities in Unix. For example, in it we illustrate that a paper has four types of components. Each section of a paper again contains four types of components. In support of each element, there is an associated text file that describes the knowledge, in this case, the instructions about the corresponding element Spacing, Size, and so on are highlighted and they are generally applicable to all the components of the element Paper.

For aesthetic reasons or for overcoming system limitations an element can have its "descendant subgraph" constructed physically in another part of the canvas, or on another canvas as a different graphic knowledge base. It Figure 1, the element Paper** has two trailing asterisks in the name. It illustrates that this element is the expansion of another element named Paper*, same name with a single asterisk, somewhere else on the same canvas. The element Picture* has a single trailing asterisk. It illustrates that this element is expanded into somewhere else at Picture**. We discuss in more detail the interconnection among graphic knowledge bases in a separate section below.

2.1.3. The Distributed Graphic Knowledge Bases

A DGKB is meaningful only when it is matched with a compatible background. We mentioned earlier that appropriate backgrounds may be superimposed together. Multiple compatible DGKBs can be matched with a single background simultaneously.

2.1.3.1. A Distributed Graphic Knowledge Base and its compatible backgrounds

A DGKB consists of elements that are built over a given background and we call this background the *Source Background* of the DGKB so created. The elements are to identify the objects in the source background.

Each element has a name and may be associated with a text file which provides the detailed description of the element. On the canvas, an ele

ment is represented by any of the available icons provided by the shell. The icons are typically rather small and the name of an element is displayed only when a user presses a button when the cursor of the mouse is pointed at the icon. Not to display the element names continuously is to enhance the clarity of the background.

A DGKB is said to be *compatible* with its source background. A DGKB is also said to be compatible with any background that is created by superimposing its source background and other appropriate backgrounds. We allow that a DGKB be matched with different compatible backgrounds to enhance the power of the system.

2.1.3.2. Superimposed Distributed Graphic Knowledge Bases

We also allow multiple DGKBs to be superimposed on one background, as long as the background is compatible with all these DGKBs.

When multiple DGKBs are superimposed on one background, only one of them is *active* at a time. Only the active one can be used by the user as usual. The inactive ones serve as a pseudo background which may assist in the visual understanding by the user of the problem environment. For example, we may have a background of a naval ship. Several DGKBs such as one for fire power systems, one for ammunition storage systems, and one for fuel storage systems may all be superimposed on the same background. An officer may see the distribution of other systems over the same background while being tutored on any one of them.

The limitation of having only one active DGKB at a time is for efficiency and effectiveness. The icons of the active one cover those of the inactive ones in case of overlapping. We allow the activation of any superimposed DGKB at the clicking of a button. Thus this limitation only provides more convenience.

2.1.4. The Interconnection Within and Among Graphic Knowledge Bases

Elements within a knowledge base may be interconnected in different ways. In the DGKB, elements may be connected symbolically, not physically, only. The text file associated with an element may specify how to connect to other elements either in the English (natural) language or use a shortened notation which is simply to follow the names of the other elements by single asterisks enclosod in pairs of parentheses. With the given names, a user can request the system to search for the connected elements. The section on the Elements List Frame given below further clari-

fies the aspect of searching. For the DGKBs, we generally use oneway linkage even though linking backwards is also possible.

The SGKB uses the same method above and also the physical arrows shown in the graph for this purpose. Additionally, an element name may be followed by a single asterisk to indicate linking (expanding) to some other place or may be followed by double asterisks to indicate linking (expanded) from some other place as illustrated in Figure 1 earlier.

A Graphic Knowledge Base needs not to end on one canvas. An element in any Graphic Knowledge Base can be expanded into another one also through the description in the associated text file. In other words, the text file of any element can provide the necessary information to allow the loading of another Graphic Knowledge Base, possibly with a new background. This interconnection can be done from a Graphic Knowledge Base of any type to another in any, the same or a different, type.

Thus an element in the SGKB may be extended to or described by another SGKB or DGKB. The same is also true for an element in the DGKB. This interconnection can be done to any level. Hence knowledge representation in our methodology has full flexibility. As an example, we can find an element from the graph of an airplane, then get into the structured description of how to inspect this section of the airplane, and further get into the graphic description of the details of certain part of that section, and this process can continue on as necessary.

2.2. The Architecture of the Graphic Knowledge Base System

The overall architecture of the Graphic Knowledge Base System contains three parts. One part is the user interface which contains the facilities for the system to communicate with the users. Another part is the shells used to carry out the processes of building, updating and utilizing the knowledge bases. The third part is the database that contains the files for the knowledge bases and the backgrounds.

2.2.1. Shells for the system

We propose that separate shells should be used to implement this methodology. One reason for doing so is that each of the shells is powerful enough for a broad field of applications. The different shells can be used for purposes other than tutoring.

We propose to have two shells for the SGKBs and two other shells for the DGKBs. For each type of the knowledge bases, We use one shell for

creating or updating knowledge bases and another shell for the users to consult the knowledge bases when being tutored. Even though the shells for building and updating knowledge bases can be very simple and user-friendly, the shells for ordinary users used during tutoring sessions can be so extremely simple and easy to use that truly no training taking more than a few minutes can enable any user to be on his own.

2.2.2. User interface

Even though we recommend using separate shells for the system, we propose that the user interface for the different shells should have common basic facilities so that switching from one shell to another should not cause any problem for the users. The commands in different shells should be similar or identical whenever possible. In fact in our plans, the next phase of the system would provide an additional command in a DGKB shell to call a SGKB shell and vice versa.

The user interface is proposed to contain three basic types of facilities, the *Graphic Knowledge Base Frame*, the *Element Description Frame*, and the *Elements List Frame*.

2.2.2.1. The Graphic Knowledge Base Frame

The Graphic Knowledge Base Frame contains three parts. The major part is the *Graphic Knowledge Base Canvas Window*. This window is used to disptay the grephic knowledge base and the background, in case of the DGKB. It is scrollable in both horizontal and vertical directions. Thus the canvas on which a Graphic Knowledge Base is displayed is not limited by the size of the screen.

Another part is the *Commands Panel*. This panel contains the bullets for the commands to be used for working with this frame, and the commands to create the other two types of frames. A command can be executed simply by pressing the left button of the mouse when the cursor is at the bullet. Help information for any command is available when the right button is pressed instead. For some commands, a pull-down menu may come up allowing the selection of a subcommand.

The last but not least is the *Messages Panel*. This panel provides the relevant information such as the current directory, the names of the knowledge base and the background. Warning messages and status information are provided through this panel too.

2.2.2.2.The Element Description Frame

One or more Element Description Frames may be requested by a user. This frame includes a Commands Panel for the bullets of available commands for this frame, and it also has a scrollable text window. The text window allows the display and update of the text file of the associated element and it should accommodate a powerful text editor.

A frame is created at the clicking of a button. The last created Element Description Frame is the *active* frame until another is created or some other one is activated by the user. In other words, only one of them may be active at a time. The inactive frames keep historical data while the active one is associated with whichever element that is selected as the current element by the user. The user can select a current element at the clicking of a button and can delete or activate any of the frames in existence as desired in the same way.

Allowing more than one Element Description Frames to coexist provides the opportunity for the user to examine different elements simultaneously.

2.2.2.3.The Elements List Frame

The user can optionally create the Elements List Frame by clicking a button. The frame also contains a Commands Panel and a scrollable text window. The text window provides the list of names of all be elements in the active knowledge base in lexicographic order. After clicking the left button at any selected name by a user, the Canvas Window in the Craphic Knowledge Base Frame will automatically be scrolled so that the selected element will be approximately centered in the window and the element will blink for attention. The element so selected also becomes the current element and its associated text file is displayed in the active Element Description Frame. This facility allows prompt search when the name of the desired element is known exactly, approximately, or vaguely.

We allow different elements in the same knowledge base to have the same name as explained earlier. In utilizing the Elements List Frame, if a user continues to select the same name for which more than one element exists then these elements will be selected in a circular order. This facility is very useful, particularly in DGKBs when the user knows the name but does not know the location of the elements over the background.

2.2.3. The Database

A database is to be organized to store all the files for the system implemented in our methodology. The structure of the database is implementation dependent thus we do not give any further description of it.

3. EXAMPLE GRAPHIC KNOWLEDGE BASE SHELLS

A few shells have been implemented to illustrate our methodology. They are all implemented on the Sun workstation using Sunview facilities under the Unix operating system.

The choice of the environment for our implementation is based essentially on the sophistication of the environment and the large screen on the Sun workstations. With the many tools available from the Sunview, we gained a lot of power with a moderate amount of programming efforts.

Our shells are fully integrated with the Sunview facilities thus the user can move frames, resize them, move texts from window, to window, and so on with only the clickings of buttons. The shells are very easy to lean, and our experience tells us that most people can be taught in a few minutes.

3.1. The SGKBSA and the SGKBSB

The shells for the SGKBs are called SGKBSA, the Sunctured Graphic Knowledge Base Shell A, and SGKBSB, the Structured Graphic Knowledge Base Shell B, SGKBSA has the full capabilities and is good for knowledge engineers or the creators of the tutoring systems. SGKBSB is basically a subset of SGKBSA and is used by ordinary users for consulting during tutoring only.

In both shells, exactly the Graphic Knowledge Base Frame, the Element Description Frames, and the Element List Frame as described earlier auare used for user interface. Figure 2 illustrates the frames in SGKBSA witha SGKB shown. As mentioned before, the example illustrates part of a tutoring system that teaches users to use the troff, picture, table, ete,. facilities in the Unix operating system to prepare a technical paper.

The major part in the figure is the Graphic knowledge Base Frame. The lower left corner of the figure shows an Element Description Frame, and above it is the Elements List Frame. In the figure, the current element is "title" thus the Element Description Frame shows the text file of that element. Note that many terms in the text file are followed by aste-

risks enclosed in parentheses. They indicate to the user that to find more information on them the user just needs to look into those elements. The user only needs to select the appropriate element name in the Elements List Frame by clicking the mouse and the Canvas Window in the Graphic Knowledge Base Frame will automatically be scrolled so that the selected element is situated approximately in the center of the window and the corresponding box blinks a few times for attention, and the element will become the current element. Both text windows in the figure are scrollable. All the frames can be moved and resized.

3.2.The DGKBSA and the DGKBSB

Distributed Graphic Knowledge Base Shell A, DGKBSA, and Distributed Graphic Knowledge Base Shell B, DGKBSB, are the two shells for the DGKBs. Again, DGKBSB is basically a subset of DGKBSA and is particularly easy to use. They again use the three types of frames for user interface described ni the methodology section earlier. DGKBSA also uses an additional frame, called the Element Name Frame, just to allow the knowledge engineer to specify names of the elements to the shell.

Figure 3 illustrates the DGKBSA. Only two frames bearing most important information about our example are shown. The background shows a photo of the planet Saturn and its six major satellites. The triangular icons are placed next to the elements to be described. In the example, the current element is Saturn and its associated text file is displayed in the Element Description Frame.

4.CONCLUSION

We have been using our methodology to create tutoring systems to teach people to use the troff in the Unix system, the Sunview facilities, the Emacs text editor, and other software packages. We consider our tutoring systems to be also intelligent tutoring systems due to their fiexibility in responding to user needs. We also have used it to store the knowledge of how to use miscellaneous simple systems. These systems allow and require the users to utilize their own intelligence and common sense while being tutored. We believe this approach is more challenging to the users and helps them remember better what they have learned. In addition to using it as a tutoring system builder, we have used it to create systems to keep track of research and project activities. We also used it to build systems to tell ourselves what to do in cases of different expected activities.

Our experiences with this methodology have been very pleasant.

Our methodology is a good intelligent tutoring systems builder. Yet the potential applications of our methodology are not limited to building tutoring systems but are extremely broad. Together the SGKBs and the DGKBs, can be used to store effectively existing expert knowledge, describe fluently knowledge on new systems or phenomena, and instruct lucidly most subject matters, We can develop Graphic Knowledge Bases for diagnosing and repairing equipment, instructing the operation of sophisticated systems or facilities, describing the organizations or compositions of systems, helping people find locations on a map, and so on so forth.

References

1. Belnap, N.D., "A useful four-valued logic" in *Modern Uses of Multiple Valued Logic*, ed.G.Epstein, Reidel Publishing, Holland, 1977.
2. Carbonell, J.R., "Mixed-Initiative Man-Computer Instructional Dialogues," *Doctoral Dissertation*, MIT, Cambridge MA, 1970.
3. Carbonell, J.R., "AI in CAI: An Artaificial Intelligence Approach to Computer-Assisted Instruction," *IEEE Tran. on Man-Machine Systems*, vol, 11, no. 4, pp. 190-202, 1970.
4. Cohen, A.G., "Mochanising a particularly expressive many sorted logic," *Ph.D.thesis*, Essex Univ, Colchester, 1983.
5. Conklin, J., "Hypertxt: an introduction and survey," *IEEE Computer*, vol.20, no.9, pp.17-42, 1987.
6. Davis, R.and J.King, "An overview of production systems," in *Machine Intelligence* 8, ed D.Michie, Wiley and Sons, N.Y, 1977.
7. Deliyanni, A. and R.A. Kowalski, "Logic and semantic networks," *Communications of the ACM*, vol.22, no.3, pp. 184-192, 1979.
8. Grignetti, M, C. L. Hausman, and L.Gould, "An Intelligent Online Assistant and Tuler," *Proc. Nat. Computer Conf*, pp.775-781, 1975.
9. Hayes, P.J., "The frame problem and related problems in artificial intelligence," *in Readings in Artificial Intelligence*, ed. N. J. Nilsson, Toiga Publishing, Califomia. 1981.
10. Kowalski, R.A., *Logic for Problem Solving*, Elsevier, North Holland, N. Y, 1979.
11. Lloyd, I.W., *Foundations of Logic Programming*. Springer-Verlag. Berlin and N.Y, 1984.
12. Newell, A.and H. A. Simmon, *Human Problem Solving*, Prentice Hall, N. J, 1972.

13. Reiser, B. J, J. R. Anderson, and R. C. Farrell, "Dynamic Student Modelling in an Intelligent Tutor for LISP Programming," *Proc. 9th IJCAI, L.A., California*, pp.8-14, 1985.

14. Robinson, J. A., *Logic : Form and Function*, North Holland, N.Y. 1979.