Hierarchical Knowledge Representation Based on Approximations

KENICHI YOSHIDA and HIROSHI MOTODA
Advanced Research Laboratory, Hitachi, Ltd.

ABSTRACT: It is difficult, even for an expert, to fully understand the behavior of a complex system. One can view a system at different levels of abstraction based on the system's functional structure. The way we understand a complex system is mostly hierarchical. This paper presents a method of hierarchical representation in a complex system. This method enables acquisition and simultaneous utilization of knowledge that is expressed in multiple levels with different abstractions based on approximations. A new method is proposed, which is an extension of the existing explanation-based learning method, to support the construction of a consistent hierarchical knowledge base that complies with the proposed representation scheme. Examples are taken from the domain of analog/digital circuits to explain the proposed representation and the method of constructing a consistent hierarchical knowledge base. The use of the hierarchical knowledge base for problem solving is also discussed.

1. INTRODUCTION

It is difficult, even for an expert, to fully understand the behavior of a complex system. One can view a system at different levels based on the system's functional structure. Furthermore, he tries to identify the role of each part in the working mechanism of the system. He then identifies a given set of elements as a meaningful composite and regards it as a single entity. A complex system is most often understood and described as a hierarchy.

Hierarchical knowledge representation is needed to realize the way an expert understands a complex system. Qualitative reasoning can be regarded as a method to express physical systems (de Kleer, 1984; Williams, 1984; Kuiper, 1986). The need for a hierarchical approach in describing a complex system has previously been addressed in an early phase of the related research (Patil et al., 1981; Bylander & Chandrasekaran, 1985). In this work the concept of hierarchy is based on the difference in the degree of approximation in functional descriptions. The work of Doyle (1986), Bennett (1987), Mozetic (1987), and Falkenhainer and Forbus (1988) are the important prior research related to this issue. These studies simplify the described system by using various approximations.

Little effort has been made to construct a consistent hierarchical knowledge base. Almost all of the work related to this issue has relied on user effort to achieve a consis-

International Journal of Expert Systems Pp. 105–119, Volume 5, Number 2. ISSN: 0894-9077

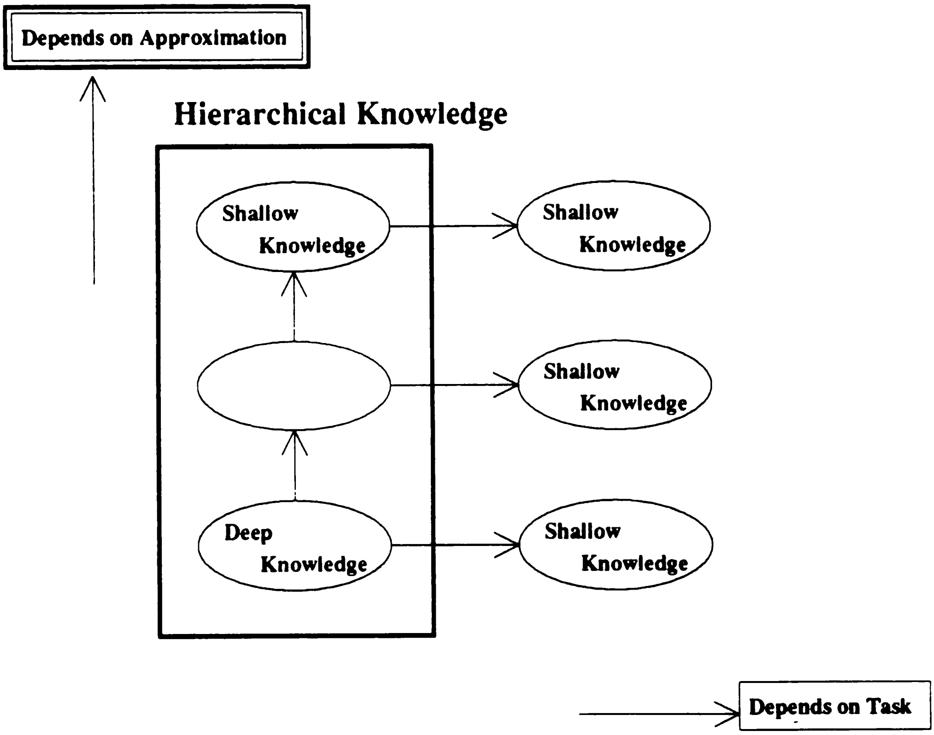

Figure 1. Knowledge structure of a complex system.

tent knowledge base. Component consistency is very important. For example, a knowledge compilation method can be used to derive a diagnostic system from a hierarchical knowledge base (Sembugamoorthy & Chandrasekaran, 1986), but inconsistency between each hierarchical level will result in either a wrong answer or the lack of an answer. Also, a top-down refinement design system (Steinberg, 1987) can be developed using a hierarchical knowledge base. The problem of inconsistency in this case is that the system cannot refine a design plan if the knowledge base does not maintain a consistent relationship between each hierarchy.

Two important issues arise when constructing a consistent hierarchical knowledge base. First, information about approximations used at each level should be stored and retrieved as necessary. Without this information, contradictions introduced by approximations between different levels decreases the functional ability of the knowledge base. Second, complex systems have many aspects, and there are many domain theories to apply to each aspect. Representation methods should be clearly expressed in order to manipulate this information.

The primary objective of this research is to develop a method for hierarchical representation of a complex system which maintains consistency among the representations in each level of the hierarchy. This method is an extension of the existing explanation-based learning method (Mitchell et al., 1986; DeJong & Mooney, 1986), and is applicable to an intractable domain theory.

The proposed hierarchical representation scheme is described in the next section together with the knowledge acquisition method to support the construction of a consistent hierarchical knowledge base. The use of this hierarchical knowledge base for problem solving is also discussed. Some acquired knowledge is explained in section 3 as well as the detailed acquisition process to show how this scheme represents the knowledge. Section 3 also explains how a computer program constructs a consistent hierarchical knowledge base using an EBL-like method.

2. DESCRIBING A SYSTEM BY HIERARCHICAL REPRESENTATION

2.1 General Framework of Hierarchical Knowledge Representation

The knowledge structure of a complex system is shown in Figure 1. Complex systems have many aspects, as well as many applicable domain theories. In this figure, ellipses on the left represent the functional description of a complex system. In the domain of a digital circuit, these ellipses represent the functional description of various aspects. For example, there are three aspects in the case of a NOR circuit (Figure 7, explained below). The lowest ellipse represents the analog behavior of the circuit. It includes equations which describe some aspect of the physical principles that hold in the circuit (e.g., Ohm's law, Kirchhoff's law, and the relationship between voltage and current of the transistor node). The second lowest ellipse represents the behavior of switches. Each transistor can be seen as a switch. The knowledge at this level is the behavior of each switch and its connections. The uppermost ellipse represents the logical behavior of the NOR circuit (which is derived from the switches). Hierarchical knowledge of a complex system consists of these types of knowledge. Here, the concept of deep/shallow is based on the difference in the degree of abstraction which are treated as approximations in the functional description.

Considering any two adjacent levels, the upper level knowledge (shallow knowledge) can be regarded as the specification, and the lower level knowledge (deep knowledge) can be regarded as the implementation. This nature is recursive; the description of some knowledge level can be regarded as a set of specifications of the deeper level knowledge, and also can be regarded as the implementation of the shallower level. For example, a switch (specification) is implemented by a transistor (implementation) and a NOR circuit (specification) is implemented by three switches (implementation).

The ellipses on the right represent the task dependent knowledge. Sembugamoorthy and Chandrasekaran (1986) developed a method to compile task dependent knowledge

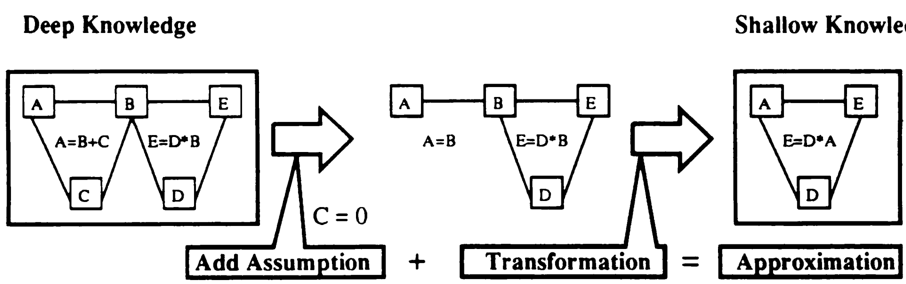

Figure 2. Approximation process.

from a functional description. In conventional usage, shallow knowledge refers to task dependent knowledge, and deep knowledge refers to a functional description. The words "deep" and "shallow" are used to distinguish between functional hierarchies in this paper. In both cases, shallow knowledge can be constructed from deep knowledge using additional information. Conventional shallow knowledge is constructed using the task knowledge as this additional information; however, the information about approximations is used in this study. This paper is limited to the hierarchy of the functional description and the related issues. The approximation process, which concerns the knowledge hierarchy, is shown in Figure 2.

The lower/deep knowledge in Figure 1 assumes less approximation, the upper/shallow knowledge is transformed from the deep knowledge by introducing some new approximations. In this study, approximations are processed in two steps. First, some assumptions are added to the deep knowledge, depending on the function of the whole or a part of the system. Next, a logically correct transformation is performed to make the shallow knowledge.

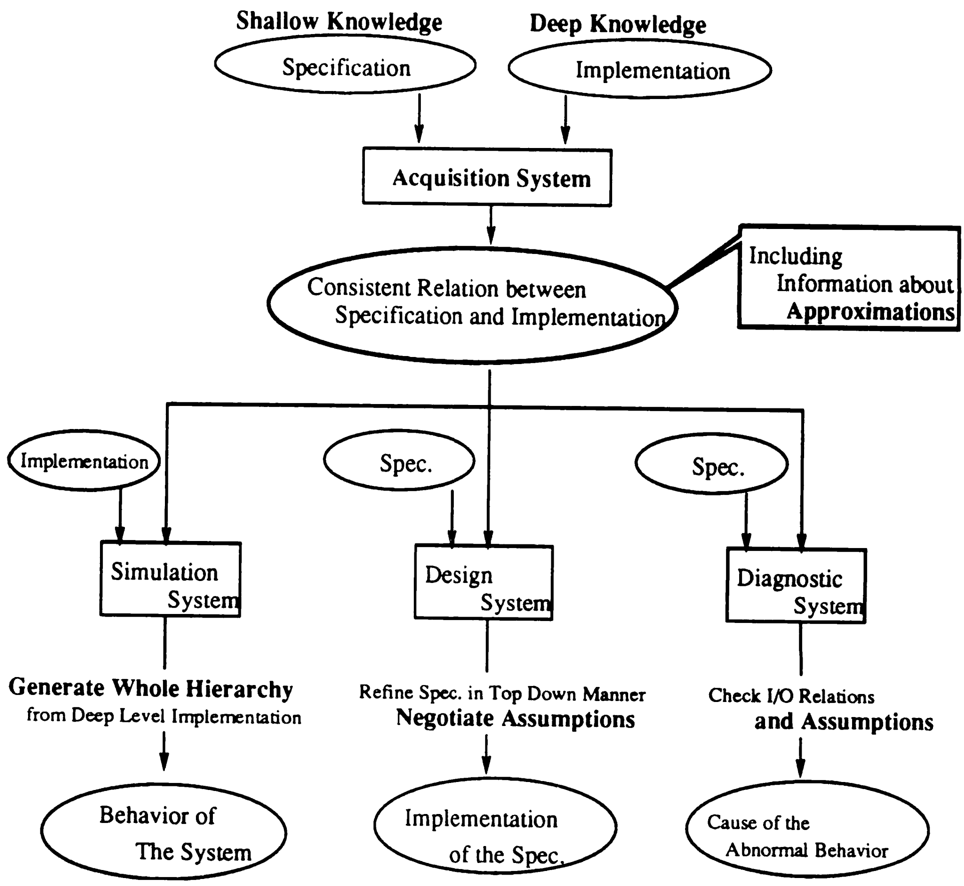

Figure 3. Hierarchical knowledge base in problem solving.

The shallow knowledge generated through this process is easy to use compared to the corresponding deep knowledge. The tasks which can be performed with this shallow knowledge will use it efficiently. The need for deep knowledge still remains, as it is a source of more detailed information. Also, information about the assumptions and the transformation is needed to check the correspondence between deep and shallow knowledge.

2.2 Hierarchical Knowledge Representation and Problem Solving

Problem solving schemes using a hierarchical knowledge base are explained in Figure 3. The output of the acquisition system is regarded as a set of consistent relationships between the specification (shallow knowledge) and the implementation (deep knowledge) at multiple levels. The entire hierarchy can be constructed from the deepest level implementation using these relations.

In a simulation system, the system can choose the appropriate knowledge level, depending on its purpose. The shallow level simulation uses fewer computing resources compared to the corresponding deep level simulation. In simulation whose purpose is to predict a summary of the complex component's behavior, the simulation system uses the shallow level knowledge for efficiency. However, when the purpose is to obtain detailed behavior, it must use deeper level knowledge. Simulation that simultaneously uses multiple knowledge levels is also possible. The consistency of the knowledge base is extremely important. Here, consistency refers to the explicit information about the assumptions. With this information, the simulation system can eliminate meaningless results which are based on contradictory assumptions.

The same information about assumptions is also required in the design system. Without this information, the top-down refinement design system cannot eliminate illegal design plans that include contractions. This aspect of the design system. suggests a new research issue: In order to achieve a top-down refinement design system with a hierarchical domain model that involves assumptions in lower level knowledge, an efficient method should be developed to negotiate between contractions that come from various assumptions. Most conventional design systems that use top-down refinement methods propagate constraints which specify the function of the lower level parts. These constraints are kept in the following lower level design process. Steinberg (1987) studied this constraint propagation. When the hierarchical domain model involves assumptions in a lower level, the design system sometimes comes across a totally new requirement in the lower level design process. The constraint propagation of this new requirement seems to degrade system performance, so a new method for the propagation should be developed. However, this is beyond the scope of the current research. Also, to ensure the functional ability of the design system, the implementation of each specification should be stored in the knowledge base. Thus, in this case, the requirement for consistency is even more important.

A simple diagnostic system checks the relations between input and output of the components. Additionally, it checks information about the environment necessary for the components to function. The assumptions recorded in the consistent knowledge base correspond to this environment. Thus, a diagnostic system with this type of consistent knowledge base clarifies (1) the cause of the malfunction and (2) the incorrect usage of the component by checking the environment. A hierarchical knowledge base enables top-down diagnosis and pinpoints the cause of the malfunction efficiently. The malfunctioning component is investigated further to find the exact cause of the malfunction. However, a knowledge compilation method with a hierarchical knowledge base for the diagnostic system should be reconsidered in order to utilize the hierarchical/approximated nature of the component's knowledge. This issue is also beyond the scope of the current research.

2.3. A Knowledge Acquisition Method for a Consistent Knowledge Base

The configuration of a knowledge acquisition system[1] that supports the construction of a consistent hierarchical knowledge base is shown in Figure 4. It includes a simulation subsystem to check the behavior of the specification and the implementation of the component. The output of this system is a set of consistent relations between deep and shallow knowledge. Other elements are input to the system. The system acquires consistent relations in the following manner:

1. First, the system selects the deepest level as the implementation level, and the second deepest level as the specification level.
2. The system receives the implementation and the specification of the component.
3. The system symbolically simulates the behavior of the component both in the deep/implementation level and the shallow/specification level.

   In the knowledge representation scheme, information about the relationship between data is stored in the implementation/specification level knowledge. For example, each equation which represents some aspect of a physical principle describes only the name of the relationship between physical data. Interpretation rules describe the actual usage of relationships at each level, such as the qualitative or the logical simulation rules. With these interpretation rules, the simulation system can calculate the value of the data.

   Initial conditions of the simulation are exhaustively generated for each datum, if not specified.[2] The set of values is supplied as a part of the interpretation rules.
4. Deep level simulation sometimes fails due to ambiguity arising from a lack of necessary information. For example, ambiguity of qualitative simulation is widely

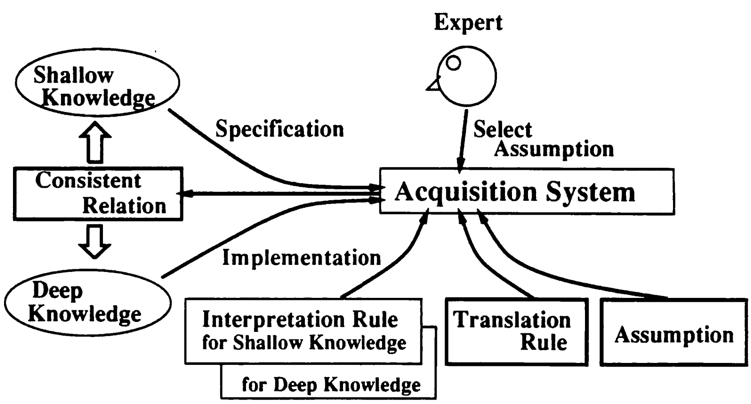

Figure 4. Configuration of knowledge acquisition system.

known. If this happens, the system will generate an assumption to reduce ambiguity, and ask a user for confirmation. User confirmation is needed to reduce the search space of the assumptions.[3]

5. After the system finds the correspondence between behavior of both levels, the EBL-like process generates a generalized relationship. Translation rules are used in two ways. First, they are used to find the correspondence between behavior of both levels. As there is sometimes a difference between the vocabulary of each level, the translation rule converts the vocabulary to find a correspondence. After the system gets user confirmation, translation rules are also used to perform logically correct transformations.
6. If there is another knowledge level in the component, the acquisition process is continued by returning to step 2 in the next level of the deep to shallow sequence.

After the system acquires enough relations between the deep and shallow knowledge, the deepest level knowledge is sufficient to generate the entire hierarchical and consistent knowledge base of a component. The EBL-like process used in the above steps has the following distinguishing characteristics:

1. It is easy for a domain expert to distinguish between important parts of the component. By acquiring the relations between the specification and its implementation from the basic to the complex parts, the expensive chunks which cause degradation are naturally avoided. The knowledge of a complex component is described using knowledge about the simpler components, each of which is recursively constituted from further simpler components. Thus, the resulting chunk is simpler than that made directly from knowledge about the simplest components.

   Because of the characteristics of the pattern match that the system must use, a long chunk excessively degrades the system performance. For example, receiving the deep level equation $A = B$, and the generalized equation in the chunk $X = Y$, the pattern matcher must check two correspondences: $X = A$, $Y = B$ and $X = B$, $Y = A$. This requirement affects the performance far beyond the linear increase. Thus, a number of short chunks are better than a corresponding single long chunk.
2. The acquisition system can automatically check the necessary environment for the chunk to be functional, using the information about the approximations. The assumptions which are used in the approximation process correspond to this environment.
3. The explanation of the implementation level behavior is extended to suppress over-generalization that is introduced by the approximations. In the conventional EBL process, the chunk is made using information about the dependency of the resulting data on the input data. In our system, in addition to the conventional information, the dependency on data which are related to the assumptions is included in the chunk.

3. APPLICATION TO CIRCUIT DESCRIPTION

This section uses the knowledge structure of analog and digital circuits and the acquisition process as examples to show how the proposed idea works.

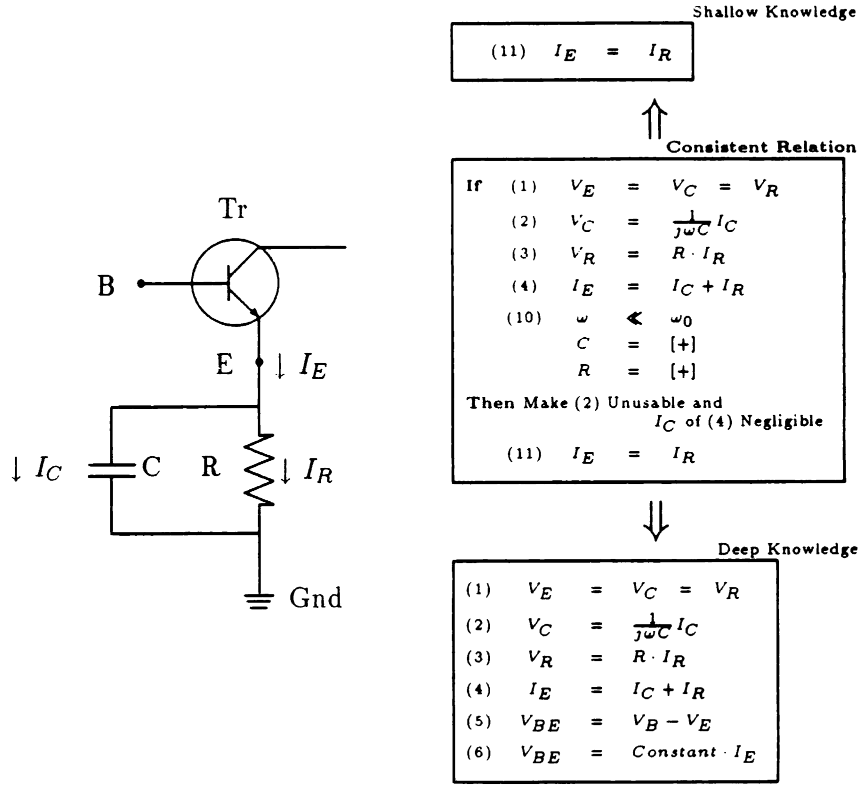

Figure 5. Knowledge structure of amplifier.

3.1. Resistor with a Bypass Capacitor (Analog Circuit)

The circuit shown in Figure 5, which is part of a radio circuit,[4] is used as a bias stabilizer for temperature compensation. The deep knowledge equations describe the underlying physical principle of this circuit.[5]

The shallow knowledge describes the representative behavior of a resistor with a bypass capacitor and has no ambiguity. The qualitative simulation using these deep level equations does have ambiguity. The acquisition system finds the relation between both knowledge types by checking the correspondence of behavior between the two levels.

After starting the behavior simulation in the deep level induced by an increase in I_E, it instantly gets stuck because it does not know how to propagate the change in I_E in equation (4). To suppress ambiguity, the assumption "I_C is negligibly small" is needed. With this assumption, the translation rule, shown in Figure 6, can make a new shallow level equation, $I_E = I_R$, so no ambiguity remains there.

The qualitative simulation using both levels of knowledge has no ambiguity, and negative feedback is detected as shown by

$$I_E \uparrow \Rightarrow I_R \uparrow \Rightarrow V_R \uparrow \Rightarrow V_E \uparrow \Rightarrow V_C \uparrow \Rightarrow I_C \uparrow \Rightarrow I_R \text{ ... negative feedback}$$

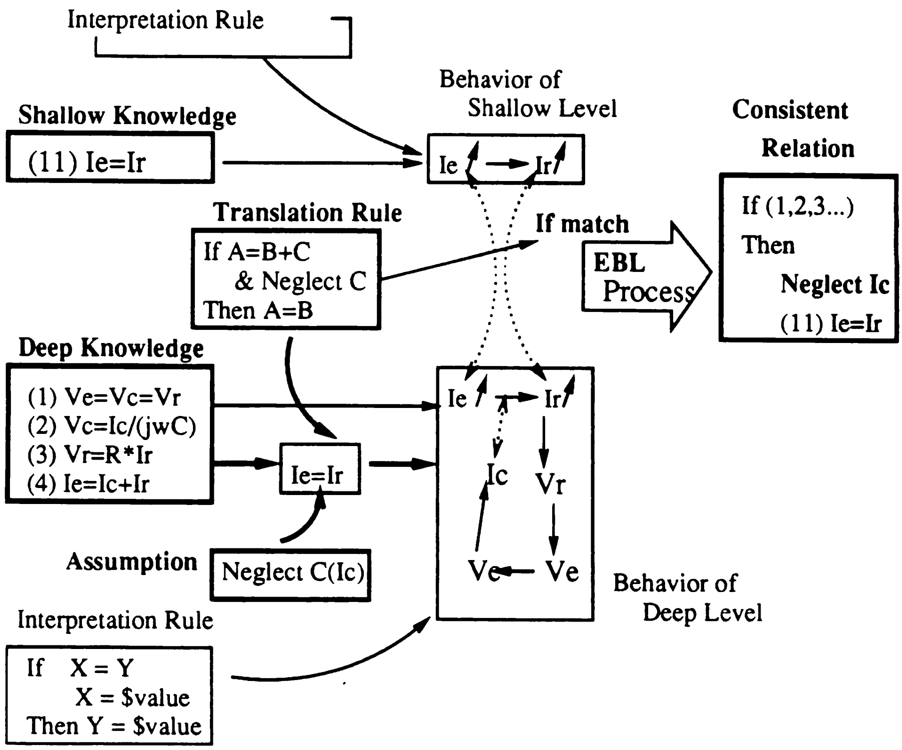

Figure 6. Knowledge acquisition process.

This process can be simulated using the equations and the interpretation rules. This negative feedback is necessary behavior for this analog circuit to have bias stability with temperature change. Thus, equation (11) in Figure 5 can be regarded as a specification of this circuit.

After the system finds the correspondence between behavior in two adjacent levels, the EBL-like process generates a relation between both levels using the equations which were used in the simulation processes. With this new relation, the system can recognize negative feedback of the behavior of circuits with the same structure. One important point here is that this relation includes information about the simulation of *I's* value, that is, equations (1, 2, 3, 10). This information is included in the chunk that is the conditional part of the relation shown in Figure 5, because I_C is related to the assumption "I_C is negligibly small." Using this information, the system can distinguish this type of circuit from a similar circuit, such as an imaginary "resistor with a bypass coil."

3.2 NOR Circuit (Digital Circuit)

Figure 7 shows a simple digital circuit which is made up of three transistors (one pull-up and two pull-down). The bottom level is a world of analog circuits and its behavior is described by twenty-two qualitative equations. The second level is a world of

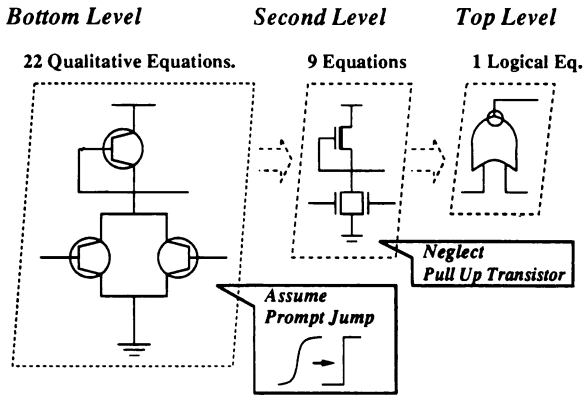

Figure 7. Knowledge structure of NOR circuit.

switches and the behavior of this level is described by nine equations. The top level is a world of logic. In the bottom level world, change needs a certain amount of time. In the second and top level worlds, time is represented by a sequence of events. This example has the following two important aspects:

1. The vocabulary and inferences of each level are completely different. Thus, the system needs appropriate interpretation rules to perform the task in each level. Translation rules are also required to find the correspondence of events which are described using different vocabularies.
2. A step-by-step acquisition scheme is required. The top level logical behavior require a lot of computing resources, if they are to be computed using the bottom level qualitative relations. However, this system first acquires the relationships between the bottom level knowledge and the second level knowledge, and then the relationships between the second level and the top level so that fewer computing resources are required.

The acquisition process for the relationship between the bottom and the second level knowledge, "A transistor acts as a switch," requires the knowledge about the qualitative simulation that was used in the previous example in section 3.1. The following second level knowledge and interpretation rules describe the world of the switch (pull-down transistor).

Equation	switch(base,in,out)		
I. Rule	**If**		switch(base,in,out)
		and	base = [true] at time *a*
		and	in = $value at time *a*
	Then		out = $value at time *b* (next time of *a*)
	If		No driving input for data
		and	data = $value at time *a*
	Then		data = $value at time *b* (next time of *a*)

The following event sequences resulted from the simulation:

at time t0	at time t1
base = [true]	base = [true]
in = [true]	in = [true]
out = [false]	out = [true]
base = [false]	base = [false]
in = [true]	in = [true]
out = [false]	out = [false]
...	...

The bottom level simulation can be performed using a qualitative reasoning technique. Thus, the interpretation rules are different from those for the second level.

Equation MOS Tr.1(Vb,Ve,Ie)
MOS Tr.2(Ve,Ie)
I. Rule Next Ve = Ve + Ve
If MOS Tr.2(V,I)
and I = $value at time *a*
Then V = $value at time *a*

The results for the bottom level, as pointed out before, require a certain amount of time for the change:

at time t0	at time t1′	at time t2′
base = [high]	base = [high]	base = [high]
in = [high]	in = [high]	in = [high]
out = [low]	out = [middle]	out = [high]
...	...	

Assumptions that neglect the state during the change are introduced in the bottom level to find the correspondence between the bottom and second levels. Also, translation rules are used to check the correspondence. For example, the following rule was used to find the correspondence between the voltage and the true/false value.

If A = [high]
Then A = [true]

After the system finds the relationship between the transistor and the switch, it generates the description of other second level switches from the bottom level description. Ambiguity still arises at the second level behavior concerning the direction of the change in "Out" for "Base = [high]." In this case, both the pull-up transistor and pull-down transistor are closed switches. A new assumption which specifies that the effect of

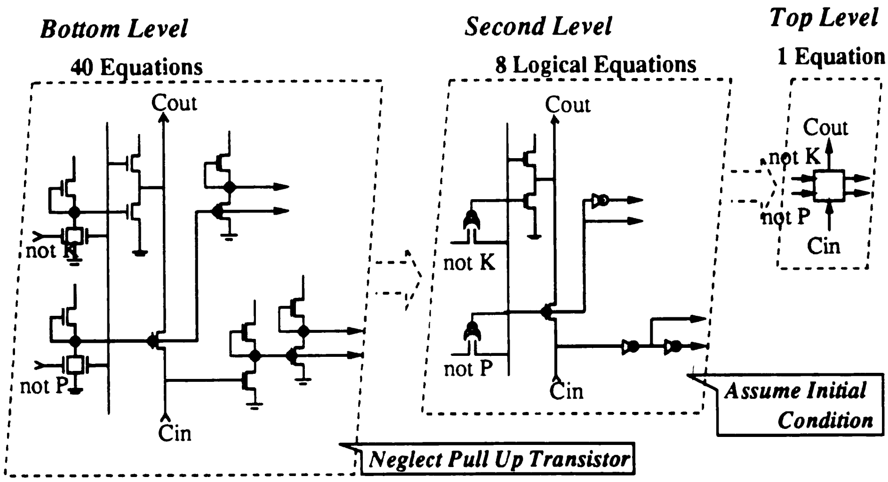

Figure 8. Knowledge structure of carry circuit.

the pull-up transistor is negligible is needed to find the correspondence between the second level and the top level behaviors.

3.3 Carry Circuit (Digital Circuit)

Figure 8 shows part of an adder, which is a slightly more complicated digital circuit than the NOR gate shown above. It receives three inputs, $\neg K$, $\neg P$, and C_{in} (Carry-in), and produces one output, C_{out} (Carry-out), where K and P are nor and xor of the two one-bit inputs to the adder (Mead & Conway, 1980). The level of the approximation at the bottom level where the behavior is described by forty equations is equal to that at the second level of the NOR circuit. Finding that the effect of the pull-up transistors can be neglected at this level, the system generates eight equations at the second level. The second and the top level simulations can be started after assuming the initial values of the above four variables. When the initial conditions are "$K = P =$ [true], $C_{in} =$ [true]/[false] and $C_{out} =$ [false]," the result of the top level simulation gives "$C_{out} =$ [true]," and that of the second level simulation gives "$C_{out} =$ [false]." From this result, the system concludes that it is necessary to precharge C_{out} to ensure the top level specification $C_{out} = \neg K \wedge ((P \wedge C_{in}) \vee \neg P)$. In this case, the initial value of C_{out} is the assumption that is needed to make this carry circuit functional.

In the Figure 8 example, the acquired relation between the top level and second level knowledge includes eight logical equations in the chunk When this relation is directly learned from the transistor level knowledge (i.e., the bottom level of Figure 8), the chunk includes 112 qualitative equations. As discussed in section 2.3, this long chunk excessively degrades the system performance. By acquiring the relations between the specification and its implementation, from the basic parts to the complex parts, the proposed knowledge acquisition scheme avoids this expensive chunk. This is a clear example that shows the advantage of this method.

4. RELATED WORK

Prior to this work, the use of approximations was widely studies for intractable domains. The work presented in this paper builds on this line of research.

Keller (1987) and Ellman (1988) improved efficiency of planners and learners by ignoring some search paths in game trees. Their work is closely related to our study. However, these studies did not use an explanation structure to check the correctness of learned heuristics for approximations. The learned heuristics must be examined in an empirical manner. Tadepalli (1989), Mostow and Prieditis (1989), and Unruh and Rosenbloom (1989) also aimed at improving the efficiency of planners and learners in game domains by approximations. However, their work cannot be used directly in the construction of a hierarchical knowledge base.

Bennett (1987) used approximations to simplify mathematical domain theory. His approach used numerical information to ensure the correctness of the approximate solution. The present study generalizes approximation type from the numerical one to a more general form. Doyle (1986) and Mozetic (1987) also used approximations to simplify domain theory. Their domains are intractable, but have relatively simple structures. Their prespecified inference engine was enough to handle the domain theory, without the additional information which is required as interpretation rules in our study.

Falkenhainer and Forbus (1988) investigated multigrain, multislice models to represent a large system. In their work, the user described hierarchy of the system to dramatically reduce the complexity of the qualitative reasoning task. But the user had to carefully design the whole hierarchical structure to achieve consistency of the whole hierarchy. The proposed method can be used to construct this type of knowledge more easily.

5. CONCLUSIONS AND FUTURE WORK

A hierarchical knowledge representation was proposed. It enables acquisition and simultaneous utilization of knowledge that is expressed in multiple levels with different approximations. The characteristics of the proposed representation scheme can be summarized as follows:

1. Information about approximations which are used in each hierarchy are memorized in an hierarchical knowledge base. Without this information, contradictions which are introduced by the approximations between each level decrease the functionality of the knowledge base.
2. The multiple domain theory and the relation between descriptions of different levels are used to express many aspects of the complex system.
3. A new method which is an extension of the existing explanation-based learning method is proposed to support construction of a consistent hierarchical knowledge base complying with the proposed representation scheme.

The examples which were given in the domain of analog/digital circuits show the ability of the proposed representation and of the knowledge base construction method to express the behavior of physical devices.

Despite the progress in the representation scheme and the construction method of the consistent hierarchical knowledge representation, a number of issues remain to be addressed. The knowledge compilation method for diagnostic systems with this type of hierarchical knowledge base should be reconsidered in order to utilize the hierarchica/approximated nature of the system's knowledge. An efficient method of negotiating contradictions should be developed in order to achieve a top-down design system with approximated domain model.

Acknowledgments: The authors are deeply indebted to Dr. Masami Yamasaki and Dr. Masaki Suwa of ARL, Hitachi, Ltd., for their valuable discussion on qualitative reasoning and suggestions for the explanation-based learning method. They also extend their appreciation to Dr. Eiichi Maruyama, formerly of ARL, Hitachi, Ltd., for his encouragement throughout this work.

NOTES

1. The functionality of this acquisition system was checked using the examples in section 3.
2. To keep the number of initial conditions low, numerical information is treated using qualitative representation.
3. Initial values for data can also be regarded as assumptions. The simulation system that is currently used requires the initial value, so the current acquisition system generates assumptions in two steps. This distinction between assumptions in steps 3 and 4 is not important.
4. Application of the proposed scheme to another part of the radio circuit is explained in Yoshida and Mooda (1989).
5. Examples of the rules and the equations are reformed to increase readability. Deep/shallow knowledge in Figures 5 and 6 is represented in terms of actual relationship among data.

REFERENCES

Bennett, S.W. (1987). Approximation in mathematical domains. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)* (pp. 239–241).

Bylander, T. & Chandrasekaran, B. (1985). Understanding behavior using consolidation. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)* (pp. 450–454).

de Kleer, J. (1984). How circuits work. *Artificial Intelligence, 24*, 205–280.

DeJong, G. & Mooney, R. (1986). Explanation-based learning: An alternative view. *Machine Learning Journal, 1*, 145–176.

Doyle, R.J. (1986). Constructing and refining causal explanations from an inconsistent domain theory. In *Proceedings AAAI-86* (pp. 538–544).

Ellman, T. (1988). Approximate theory formation: An explanation-based approach. In *Proceedings AAAI-88* (pp. 570–574).

Falkenhainer, B. & Forbus, K. (1988). Setting up large-scale qualitative models. In *Proceedings AAAI-88* (pp. 301–306).

Keller, R.M. (1987). Concept learning in context. *Machine Learning Journal, 2*, 91–102.

Kuipers, B. (1986). Qualitative simulation. *Artificial Intelligence, 29*, 289–338.

Mead, C. & Conway, L. (1980). *Introduction to VLSI Systems.* New York: Addison-Wesley.

Mitchell, T.M., Keller, R.M., & Kedar-Cabelli, S.T. (1986). Explanation-based generalization: A unifying view. *Machine Learning Journal, 1*, 47–80.

Mostow, J. & Prieditis, A.E. (1989). Discovering admissible heuristics by abstracting and optimizing: A transformational approach. In *Proceedings AAAI-89* (pp. 701–707).

Mozetic, I. (1987). The role of abstractions in learning qualitative models. In *Proceedings of Fourth International Workshop on Machine Learning* (pp. 272–255).

Patil, R.S., Szolovitz, P., & Schwartz, W.B. (1981). Causal understanding of patient illness in medical diagnosis. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)* (pp. 893–899).

Sembugamoorthy, V. & Chandrasekaran, B. (1986). Functional representation of devices and compilation of diagnostic problem-solving systems. In J. Kolodner & L. Erlbaum (Eds.), *Experience, Memory and Reasoning* (pp. 47–73). Hillsdale, NJ: Lawrence Erlbaum Associates.

Steinberg, L.I. (1987). Design as refinement plus constraint satisfaction: The VEXED experience. In *Proceedings AAAI-87* (pp. 830–835).

Tadepalli, P. (1989). Lazy explanation-based learning: A solution to the intractable theory problem. In *Proceedings AAAI-89* (pp. 694–700).

Unruh, A. & Rosenbloom, P.S. (1989). Abstraction in problem solving and learning. In *Proceedings AAAI-89* (pp. 681–687).

Williams, B.C. (1984). Qualitative analysis of MOS circuits. *Artificial Intelligence*, *24*, 281–346.

Yoshida, K. & Motoda, H. (1989). An approach to hierarchical qualitative reasoning. *Japanese Society for Artificial Intelligence*, *4*(4) (in Japanese).

Address all correspondence to Kenichi Yoshida, Advanced Research Laboratory, Hitachi, Ltd., Hatoyama, Saitama 350-03, Japan. E-mail: yoshida, motoda@harl.hitachi.co.jp