An Experimental Adaptive Expert System

Xiaoshan Pan

Collaborative Agent Design (CAD) Research Center
Cal Poly, San Luis Obispo, CA, USA

1. Abstract

This paper explores an approach to building an adaptive expert system in an environment of human-computer collaboration. Components of an adaptive system are identified, with an emphasis on the mechanisms that enable adaptive behavior to occur. Knowledge representation in a rule-based, object-orientated expert system is described through the establishment of appropriate relationships utilizing heuristic rules, objects, and agents. The experimental expert system displays low level learning capabilities that show sufficient promise to warrant further research.

This work was undertaken as a Master Degree Thesis in the Architecture Department of the College of Architecture and Environmental Design at California Polytechnic State University (Cal Poly), San Luis Obispo, California.

2. Keywords

expert system; adaptation; learning; knowledge representation; rule-based system; object-oriented design; data; meaning; information; objects; agents; human-computer collaboration.

3. Introduction

“An expert system is a system that employs human knowledge captured in a computer to solve problems that ordinarily require human expertise (Turban, 1992, p. 74).” Well-designed systems emulate the reasoning processes used by experts to solve problems, and are popularly used in medicine, business management, design, and searching for natural resources.

“Even though expert systems aim to mimic human experts, they lack an extremely important capability of human intelligence: the ability to learn from experience (Turban, 1992, p. 666).” First of all, it takes a significant amount of time to build an expert system with many hours of testing and debugging. If a human expert comes to an incorrect conclusion, he may be able to learn from the mistake and avoid making the same or similar mistakes in the future. Once an expert system is found to have an error, the only way to correct that error is to reprogram the expert system. In other words, most current expert systems are lacking an adaptive capability.

Computer based adaptive capabilities are essential in situations where environments change, in situations where standards of expertise are changing, and in situations where there are no historical data and learning occurs as a task that has been performed.

Webster (1991) defines 'adapt' as a capability "to adjust (oneself) to new or changed circumstances (p. 15)". In nature, adaptation is sometimes referred to as 'evolution'. In the context of a system it may be referred to as a 'self-organization capability', and, in the case of human beings, as a 'learning capability'. Therefore, since the intent of expert systems is to emulate human experts, this paper refers to an adaptive expert system as a learning system or, a learner.

A goal of the research described in this paper is to explore strategies that enable an expert system to adapt to, or learn from, interactions with users. It is assumed that a relatively primitive computer-based adaptive capability can be of significant value in a problem-solving environment in which a computer is used as a collaborative decision-support tool.

Because of its capability to support rule-based programming and object-oriented design, the C Language Integrated Production System (CLIPS) Version 6.10 was chosen as an expert system tool for this project (NASA, 1998).

4. Rule-Based Programming and Object-Oriented Design

Rule-based programming is commonly utilized to develop expert systems. Rules are used to represent heuristics, which specify a set of actions to be performed for a given situation. According to Giarratano and Riley (1998), rule-based expert systems are popular for a number of reasons:

- Modular nature. This makes it easy to encapsulate knowledge and expand the expert system by incremental development.
- Explanation facilities. By keeping track of which rules have fired, an explanation facility can present the chain of reasoning that led to a certain conclusion.
- Similarity to the human cognitive process. Rules appear to be a natural way of modeling how humans solve problems. (p. 28)

Object-Oriented design is "a technique for developing a program in which the solution is expressed in terms of objects – self-contained entities composed of data and operations on that data" (Dale et al. 1996, p. 162). Taylor (1998) proposes that object-oriented design is an ideal approach for building adaptive systems because "objects are naturally adaptive elements. If allowed to interact freely, managing their own actions, they can be excellent components for building up adaptive systems "(p. 147).

5. A Learning Environment

The word 'learning' is mainly reserved for human beings. Researchers have long wondered whether computers could also learn. In order to answer this question, a definition of learning is necessary. According to Lacey (1998), learning is “Any relatively permanent change in behavior brought about through experience – that is, through interaction with the environment (p. 25)."

It could be assumed that learning may involve four factors: the learner; the environment; their interaction; and, state. A learner could be defined as a relatively independent system, exhibiting a learning capability and adapting to its environment. In other words, a learner is a system with the capability to change itself as a result of interactions with its environment. The constantly changing environment provides a basic force that drives a leaner to learn.

A state is a collection of characteristics that can be used to define the status of a learner and its environment at a certain time. An environment is partially comprised of the external circumstances of a learner, and influences the way in which a learner behaves. In other words, a learner exists in an environment, which changes over time. The internal impact of this external environment impels the learner, which results in adaptive action on the part of the learner.

An approach to learning, therefore, is through the interaction between the learner and its environment. The interaction operates in terms of three basic elements: percepts; actions; and, goals. Shen (1994) defines these elements as follows:

> “An action is a physical change that occurs inside the learner, such as a muscle contraction inside a human body or an electrical signal sent to some switch in a robot.
>
> A percept is a representation inside the learner about the state information of the environment.
>
> A goal is defined as a set of percepts that the learner wishes to receive from the environment.” (p. 8)

Shen (1994) also explains the learning process as:

> “With actions and percepts as building blocks, the learner is to construct a model of the environment so that it can predict the consequences of its actions and direct its actions towards the goal.” (p. 9)

Accordingly, a learning process is a process in which a learner builds models of its environment.

In addition to the physical world that is external to the learner, it is suggested that there also exists an internal model world within the learner. The learner builds models and stores them in this model world. These models represent the learner’s understanding and knowledge of the external physical world. Minsky (1985) states that, “We all make models of ourselves and use them to predict which sorts of things we’ll later be disposed to do (p. 303)."

According to Shen’s definition of learning, a system should not be called a learning system if it merely contains the capability to repeat exactly what it has been designed to do. A learning system must have the capability to gain knowledge and adapt its behavior accordingly. In terms of the two worlds concept, a process which gains knowledge is equivalent to a process that revises old models and builds new models based on interactions between the two worlds. In this regard, memorization and creation are two possible methods of knowledge acquisition.

When knowledge is available, and a learner possesses and uses a capability to obtain and store this knowledge from its external environment, this capability could be defined as memorization. An example is a child studying arithmetic from a textbook.

When an old model has been evaluated as inappropriate to any given circumstance, and no existing, applicable knowledge is available, creation becomes an important approach to the production of new knowledge. Creation may be considered a higher level of learning.

Memorization and creation represent an initial phase of a learning process. The construction of a model, according to Shen (1994), is “essentially a loop process of generate-and-test: one starts with a model, tests the model by observing its performance, and generates a new model if the current one is not good enough (p. 25)". This might suggest another significant factor in a learning system: evaluation. Evaluation generates feedback after an action has been taken. This feedback then goes back to the percept process, causing the updating of the model. If we regard a system that cannot learn as a dead system, then this evaluation process is an essential difference between a learning system and a dead system.

In this generate-and-test loop, the evaluation adjusts the actions as well as the goals of a learner, and sends stimuli back to the loop for model revision. From the point of view of human beings, a process that discerns between right and wrong, or appropriate and inappropriate, must contain a set of criteria. When a change in the physical world causes a change in the criteria system, a corresponding change in the learning system results. This process would not occur in a dead system, because it is not represented by a loop, but rather by a unidirectional flow, with no provision for feedback. Memorization and creation do not exist in a dead system because the process of evaluating and handling feedback, which affects actions, does not exist.

6. Knowledge Representation

6.1 From Data, Meaning, Information to Knowledge

Computer systems have displayed significant increases in sophistication and capability while maintaining basic data storage and manipulation principles. Data in a computer system, based on a particular computer language, are letters, numbers, strings or special characters. A single piece of data consists of a combination of electrical pulses. It has no meaning to the computer by itself.

Minsky (1985) states, “the secret of what anything means to us depends on how we’ve connected it to all the other things we know… a thing with just one meaning has scarcely any meaning at all (p. 64).” In other words, the more associations a thing has with other things, the more it will mean to us.

The meaning of a single piece of data derives from its association with other data. For example, A Chinese character may have little meaning to English speakers if they do not know anything about the Chinese language. For them to gain understanding, an explanation of this Chinese character in the English language is necessary. During this process, the English speakers establish associations between this character and what they already know, and then meaning appears.

Approaches to building associations between data in a rule-based, object-oriented expert system include writing rules and object methods that manipulate the behaviors of data. Information is data with behavioral characteristics and relationships. Data are basic building blocks of information, and relationships among data endow meaning. Functionally, a piece of information must describe a state, such as, ‘mail is sent’ and ‘it is raining‘.

Information may be defined as data linked by relationships, which can describe the state of an event, a condition, an item, or an object.

“When human beings or computer-based agents draw conclusions (i.e., inferences) from information then these conclusions are knowledge” (Pohl, pers.com.). Knowledge exists when humans attempt to solve problems or make decisions based on the state of a condition, and information provides a description of the state. In a rule-based computer system, the process of taking certain actions based on certain conditions can be represented as rules. Thus, information and rules can define knowledge.

“Rules appear to be a natural way of modeling how humans solve problems (Giarratano and Riley, 1998, p. 28).” When humans solve problems and make decisions, rules represent heuristics, or “rules of thumb”, which specify a set of actions to be performed for a given condition. Heuristic rules provide shortcuts when describing relationships and behaviors. Heuristic rules mostly come from experience. Intermediate steps are skipped because those steps might be functionally unnecessary or unlikely to lead to a quick solution of the problem. This is why an expert usually has been perceived as a person who is capable of "solving problems without thinking".

Heuristic reasoning does not refer to the working of the internals of the system, but rather associates readily definable external observations with a plausible conclusion. Instead of heuristic rules, a deeper causal model also might come to the same conclusions, but would in principle need to examine many other things, and some of which might not be easy to answer without additional measuring facilities. In a rule-based expert system, rules may represent associations among data and reasoning processes used to solve problems and make decisions.

6.2 Objects and Agents

“An object is a software package that contains a collection of related procedures and data (Taylor, 1998, p. 7).” Procedures may be referred to as methods, which are relationships and behaviors among data. Objects are defined as encapsulations of properties (facts) and behavior (relationships among facts) in CLIPS Version 6.10 (NASA, 1998). Objects represent information. A well-defined object describes its states with the data it holds, and manipulates the data through its methods. An object-oriented system processes information, rather than data.

Everything an object ‘knows’ is captured in its variables (data), everything it can do is expressed in its methods. This notion not only keeps simple things simple, but also makes complex things simple as well. No matter how many objects or what types of objects exist inside a system, relationships among those objects are clearly described by their methods. The more objects the system contains, the more complex it can become. When a change occurs in an object, other objects will react correspondingly to adapt to the change; thus the system can reorganize itself automatically.

Objects are ideal building blocks to build adaptive systems. Since methods are encapsulated with data into objects in an object-oriented computer system, when a change is necessary, even if it is possible to change only a single or a few objects then the whole system will adapt to this change automatically through the reactions of other objects. However, in a conventional system (i.e., non-object-based), a change will require significant restructuring, anything less will result in a broken system.

Information in an object-based computer system is not separated into data and procedure, but rather stored as objects. Objects can be exchanged with, or shared by, other systems, enabling adaptive behavior to occur not only inside a system but also among different systems.

During the related processes of problem-solving and decision-making, human beings may be capable of drawing conclusions, finding solutions and taking actions based on available information. A computer-based system that behaves in such a fashion may be referred to as an agent. "Agents are nothing more or less than objects with rules and legs (Taylor, 1998, p. 131)." In a rule-based, object-oriented expert system (like CLIPS), an agent is comprised of one or more rules that are capable of solving problems or making decisions based on information derived from objects. Objects are useful for solving problems and making decisions, but become more useful when they are converted into agents.

Taylor (1998) explains: “It is an unfortunate fact that most objects are rather stupid. They contain methods for executing procedures, and they have variables to hold the inputs and outputs of those procedures. But they lack a good mechanism for capturing the rules that determine when and how these procedures should be carried out. If you ask them to do something, they do it. If you don't, they just sit there” (p. 125).

Agents act at the knowledge level. Data within an object describe ‘what’ the object is, and object methods indicate ‘how’ it will react to changes. A rule, which acts as an agent, will provide some control of ‘when’ and ‘whether’ the object carries out its methods.

6.3 Notions from the Theory of Complex Adaptive System

All scientific studies are in the business of trying to make sense of the world, and research of Complex Adaptive System (CAS) is one of these. A driving force behind CAS research is a desire to understand how complex order arises out of the interaction of relatively simple components, such as: “How did life emerge out of the interaction of inert molecules? How do termites cooperate to build arches and other architectural structures? How do the phenomena of thought and consciousness arise out of the interactions of neurons in the brain?” (Taylor, 1998, p. 137)

Wholes are made of parts but are more than the sum of their parts. An example of a complex adaptive system is the collective behavior of ants. An individual ant may exhibit relatively simple behavior. When a significant number of ants collaborate, interactions among them reach a level of complexity, and then a high-level order from which adaptive behavior arises. This high-level order enables all ants to act as a whole. The whole is likely to achieve a level of intelligence unattainable by any individual ant.

Complex order may arise through the interaction of simple components. Objects are natural building blocks of adaptive systems. The research described in this paper is based on the premise that a computer-learning capability can come into existence through the interactions of simple elements (e.g., objects).

7. Human-Computer Collaboration

While engaged in the process of problem-solving and decision-making, human beings typically draw upon a broad range of personal experience and often rely upon intuition. Computers, on the other hand, have a narrow area of expertise, greater computational power, and lack a good mechanism for knowledge acquisition through experience.

Human-computer collaboration proposes an approach in which computers and humans co-operate to solve problems, especially to solve those problems that neither can effectively solve alone. For example, an expert system helps non-expert humans to solve problems and make decisions based on internal expertise. The system might not work appropriately if the problem changes, unless human users provide more knowledge. The system may thus be said to learn through its interactions with humans.

8. Building an Adaptive Expert System

An expert system could be viewed as one heuristic rule, or if-then statement, since all rules in the system work together to define a specific condition, which is achieved by asking a user relevant questions. Thereafter, the system reacts with one or more actions. It is postulated that an expert system becomes adaptive during an interaction with a human user if this system proves to be capable of the formulation of new relevant questions and the adjustment of action(s).

8.1 Knowledge-Node-Network

The concept of a knowledge-node-network is proposed as an approach to knowledge representation in building an adaptive rule-based, object-oriented expert system. In a knowledge-node-network, all nodes are represented as objects, and adaptivity of the system is achieved through node modification and creation. Two types of nodes exist in a knowledge-node-network: question nodes (i.e., representation of relevant questions); and, action nodes (i.e., representation of corresponding actions).

A question node may contain one or more input-connections, each weighted, whose values change based on a successful search or a failed search. Each input-connection corresponds to two output-connections, each of which connects to another question node or action node. A question node carries a relevant question and a corresponding answer-key. During a reasoning process, if a question node is activated by a message, the question will be evaluated, and the return value will lead to the next activation.

An action node may contain one or more input-connections, each weighted, whose values change based on whether user feedback is positive or negative. An action node carries one or more actions, and may be activated by messages sent from question nodes.

Also, such a knowledge-node-network contains an initial body of knowledge (i.e., a group of question nodes and action nodes with appropriate connections). Upon interaction with human users, this initial body of knowledge may prove inadequate within the context of a changing problem. The system adapts to these changes through the modification of existing knowledge nodes and the creation of new knowledge nodes.

In a knowledge-node-network, each node is associated with a value represented by either a relevant question or an action. Each node also contains connections to other nodes. The creation of new nodes and the modification of either the value or the connections of the node are defined as constituting a form of system adaptation in response to a change.

8.2 Concept of Mind Object

The concept of "Mind" object is implemented as an agent-control strategy. Similar to human behavior, no matter how many things we want to do, we can choose to do only one at a particular time. "One part of me wants this, another part wants that. I must get better control of myself (Minsky, 1986, p. 40)".

In the experimental expert system that was developed as part of the research, a "Mind" object acts as an "agent manager" that decides which agent should be activated at any particular time. The "Mind" object is designed with six different "states"; namely “inactive”, “querying”, “feedback”, “modifying”, “creating”, and “reporting”. Each state represents a condition that will trigger a particular agent to be active. The concept of a “Mind” object proposes a useful approach to solving conflicts, especially when both randomness and probability are unlikely good decision-making approaches, because a “Mind” object provides uniqueness. Since the state of the “Mind” object is changed dynamically by the activation of agents and object methods, each agent and object evolves as a part of the decision-maker. In other words, the control mechanism is not hierarchical but parallel.

In the experimental system, a state of the "Mind" object involves one agent only. However, according to Minsky (1993), a state of the human mind involves innumerable agents and subagents, leading to a much higher degree of complexity: “Let's oversimplify matters for the moment and imagine that the mind is composed of many "divisions," each involved with a different activity, like vision, locomotion, language, and so forth. This pattern repeats on smaller scales, so that even the thought of the simplest ordinary object is made up of smaller thoughts in smaller agencies…a mind can have exactly one total state at any moment, but it can be in many partial states at the same time” (p. 85).

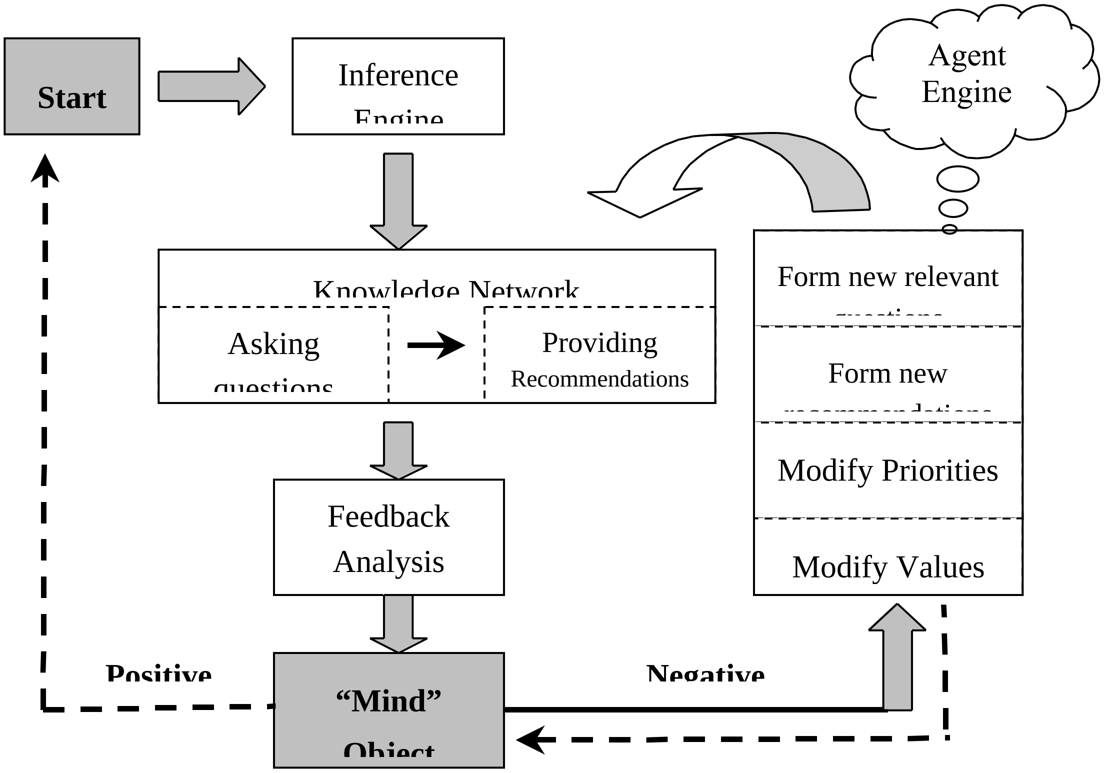

Figure 1: System Architecture

8.3 System Architecture

In terms of inference process, an adaptive expert system describes a generate-and-test loop that endows a system with a leaning capability (Figure 1), while a conventional expert system describes a unidirectional flow with no adaptive behavior. From a system structure perspective, an adaptive expert system differs from a conventional expert system as follows:

A). Unlike a static knowledge base in a conventional expert system, an adaptive expert system contains a dynamic knowledge base. Knowledge is represented by a knowledge network (i.e., objects and their interactions) that can be dynamically modified in real-time based on interactions with the human user.

B). In addition to containing the capabilities of a conventional inference engine, an adaptive expert system is capable of real-time adaptation to feedback and restructuring of the knowledge network.

The architecture of such an adaptive expert system consists of three major components: an inference engine; a knowledge network; and, a node-modification agent engine. The inference engine infers through condition-satisfaction action selection. In the rule-based system, it searches for the rules satisfied by facts or objects, and executes the rule with the highest priority. The knowledge base is composed of knowledge nodes and their connections. The node-modification agent engine consists of a node-maintainer, a node-builder and a node-merger. These agents analyze feedback and reconstruct the knowledge base accordingly.

The goal of the system is the development of corresponding answers to satisfy the needs of clients based on client provided information. The mechanism provided by this system demonstrates a primitive adaptive capability of a computer-based expert system (Figure 1).

9. Typical Scenario

The following sequences exemplify some adaptive behavior capabilities of the experimental system, simulated by the interactions of the user with the system.

The 1st Interaction:

System : Hello, this is Shawn's-Kitchen-Design-Adviser!
May I have your name, please?
User : Lee, Tony.
(Identifying the user…)
System : Tony, You are my new client, welcome!
For providing the best service, I need to ask you some questions.
If you are ready, answer 'y' to continue...
User : y
(System asks the user relevant questions…)
System : How much do you entertain at home?
1. Several times per week
2. Once a week
3. Several a month
4. Once a month
5. Hardly ever

User : 1.

System : **…**

User : **…**

System : How important is energy conservation in your kitchen?
1. Very important
2. Desirable
3. Average
4. Unimportant
5. Don't know

User : 1

System : Well, Tony, one moment please...

(Searching for design recommendations based on the user's responses…)

System : Tony, the followings are my recommendations for you:
. . .
5). -- Window should face south for collecting maximum natural lighting.
6). -- Adopting passive solar panel as an energy saving approach.
7). -- Using skylights for collecting natural lighting.
...
Are you satisfied with them? y/n

User : n *(negative feedback)*

(System identifies the problem since it receives a negative feedback…)

System : That's fine, Tony, let's work this problem out together...
May I know which one seems not appropriate to you?
Choose the number from above...

User : 6

System : What you've chosen is...
----
6). -- Adopting passive solar panel as an energy saving approach.

Why is it inappropriate in your case?
1. It doesn't make any sense at all.
2. It basically is fine; you just want to refine it.
3. You have a physical constraint against it

User : 3

(All three options listed above cause modifications in the knowledge-network. Option one and two trigger agents to update the value and connection weight of knowledge nodes, and option three triggers agents to build a new knowledge node.)

System : Would you please explain to me what is the constraint?

User : because my wife will not like it.

System : What do you mean 'it' exactly?

User : the idea of passive solar panel.

System : So, it is inappropriate in your case because...
Your wife will not like the idea of passive solar panel. Am I right?

User : y

(During the above interactions, the system analyzes the user's response in order to achieve a certain degree of understanding.)

System : Well, since your wife will not like the idea of passive solar panel,
do you have any suggestions? y/n

User : y

(A negative response here causes a modification of the existing knowledge-network, while a positive response leads to the creation of a new knowledge node.)

System : What would be a good description of it?

User : Building a greenhouse adjacent to the kitchen.

System : your suggestion is...building a greenhouse adjacent to the kitchen. Is it correct?

User : y

System : Well, Tony, the followings are my recommendations for you:
. . .
5). -- Window should face south for collecting maximum natural lighting.
6). -- Building a greenhouse adjacent to the kitchen.
7). -- Using skylights for collecting natural lighting.
. . .
Are you satisfied with them? y/n

User : y *(positive feedback)*

System : Tony, thanks for using Shawn's-Kitchen-Design-Adviser. See you!

(The end of the first interaction.)

The adaptive expert system has acquired knowledge that will be applied in the next sequence of interactions

The 2nd Interaction:

System : Hello, this is Shawn's-Kitchen-Design-Adviser!
May I have your name, please?

User : Tate, Tom

(Identifying the user…)

System : Tom, You are my new client, welcome!
For providing the best service, I need to ask you some questions.
If you are ready, answer 'y' to continue...

User : y

(System asks Tom relevant questions, and Tom's situation happens to be similar to Tony's.)

System : How much do you entertain at home?

1. Several times per week
2. Once a week
3. Several a month
4. Once a month
5. Hardly ever

User : 1.

System : …

User : …

System : How important is energy conservation in your kitchen?

1. Very important
2. Desirable
3. Average
4. Unimportant
5. Don't know

User : 1

(The following interactions indicate the new knowledge that the system obtained through "The 1st Interaction".)

System : Will your wife like the idea of passive solar panel?

1. Yes
2. No
3. It doesn't matter

User : no

(A negative response here leads the system to access the knowledge node created in the "The 1st Interaction".)

System : Well, Tom, one moment please...

(Searching for design recommendations based on the user's responses…)

System : Tom, the followings are my recommendations for you:

. . .

5). -- Window should face south for collecting maximum natural lighting.

6). -- Building a greenhouse adjacent to the kitchen.

7). -- Using skylights for collecting natural lighting.

...

Are you satisfied with them? y/n

(Option 6 is from a new knowledge node that was built through the "The 1st Interaction".)

User : y *(positive feedback)*

(The end of the 2nd interaction.)

10. Conclusions

The research conducted by the author suggests that memorization (i.e., storing experience) is one of the fundamental capabilities of an adaptive system. The demonstrated expert system has achieved some features of memorization. During an interaction with a human user:

- The system adjusts priorities of its actions based on the feedback received from the human user in real-time. In this way, useful actions can be enhanced, and inappropriate actions can be combed out.
- The system automatically corrects its inappropriate actions based on evaluations received from the human user.
- The system can define new problems and search for new actions (i.e., creating new relevant questions and actions), according to the feedback received from the human user.

Besides the memorization capability, the system is capable of self-reorganizing its knowledge base (i.e., mergence of knowledge nodes).

Within the context of human-computer collaboration, one of the future research directions may be to explore approaches that would enable computers to ‘understand’ more about humans (i.e., to improve the communication level of the interactions). Specifically, this approach could aim to objectify human language into software objects, since most communication among humans is through language (e.g., English), and objects in computers may represent information. This approach may require building a mechanism that enables a computer-based system to learn new words and syntax during its interaction with human users.

If extended beyond the human-computer collaboration context, the exterior environment of an adaptive system may involve objects other than those that are related to humans directly. Without interactions with humans, it is necessary for an adaptive system to have its own evaluation system and goals. A future study of an evaluation system in an adaptive system may significantly benefit the research study presented in this paper.

11. References

Belth, M., (1977). The Process of Thinking. New York: David McKay Company. (pp. 28-53).

Boden, M. A. (1996). Artificial Intelligence. San Diego: Academic Press.

Brett, J. (1977). The Kitchen: 100 Solutions to Design Problems. New York: Billboard Publications.

Chauchard, P., (1956). Language and Thought. New York: Walker and Company. (pp. 137-150).

Chien, S. (1993). A Explanation Facility for Rule-Based Expert Systems. San Luis Obispo: MS Thesis in Architecture, Cal Poly, San Luis Obispo, CA.

Cotton, J., (1989). KB-STAR: a Prototype Expert System to Assist in Structural System Selection during the Conceptual Stage of Building Design. San Luis Obispo: MS Thesis in Architecture, Cal Poly, San Luis Obispo, CA.

Dale, N. Weems, C., and Headington, M., (1996). Programming and Problem Solving with C++. Boston: Jones and Bartlett.

Dieterich, H., Malinowski, U., Kuhme, T., Schneider-Hufschmidt, M., (1993). State of the Art in Adaptive User Interfaces. In Schneider-Hufschmidt, Kuhme, Malinowski (Ed.), Adaptive User Interfaces: Principles and Practice. Amsterdam: North-Holland. (pp. 13-48).

Duerk, D. (1993). Architectural Programming. New York: Van Nostrand Reinhold.

Ekholm, A. and Fridqvist, S. (1998, October 15). Research project [online]. A Dynamic Information System for Design Applied to the Construction Context.

Available: http://www.caad.lth.se/research/bas.caad/

Feigenbaum, E. (1977). “The Art of Artificial Intelligence: Themes and Case Studies of Knowledge Engineering”. IJCA15, 1977.

Findler, N., (1979). A Heuristic Information Retrieval System Based on Associative Networks. In Findler (Ed.), Associative Networks: Representation and Use of Knowledge by Computers. New York: Academic Press.

Forsyth, R., and Rada, R. (1986). Machine Learning: Applications in Expert Systems and Information Retrieval. New York: John Wiley and Sons. (pp. 14-103).

Frey, B. (1998). Graphical Models for Machine Learning and Digital Communication. Cambridge: MIT Press. (pp. 1-88).

Giarratano, J., and Riley, G. (1998). Expert Systems: Principles and Programming. Boston: PWS.

Hart, A. (1989). Machine Induction as a Form of Knowledge Acquisition in Knowledge Engineering. In Forsyth (Ed.) Machine Learning Principles and Techniques. London: Chapman and Hall. (pp. 23-38).

Hirsh, H. (1990). Incremental Version-Space Merging: A General Framework for Concept Learning. Boston: Kluwer Academic. (pp. 69-82).

Johnson-Laired, P., (1993). Human and Machine Thinking. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Kalay, Y. (1992). Principles of Computer-Aided Design: Evaluating and Predicting Design Performance. London: Butterworths.

Kearns, M., and Vazirani, U. (1994). An Introduction to Computational Learning Theory. Cambridge: MIT Press. (pp. 1-28).

Knoblock, C. (1993). Generating Abstraction Hierarchies: An Automated Approach to Reducing Search in Planning. Boston: Kluwer. (pp. 1-51).

Kumlin, R. (1995). Architectural Programming: Creative Techniques for Design Professionals. New York: McGraw-Hill.

Kurzweil, R. (1992). The Age of Intelligent Machines. Cambridge, MA: MIT Press. (pp. 283-303).

Kurzweil, R. (1999). The Age of Spiritual Machines. New York: Penguin Putnam Inc.

Langley, P. (1998). Elements of Machine Learning. San Francisco: Morgan Kaufmann.

Minsky, M. (1985). The Society of Mind. New York: Simon and Schuster.

Minsky, M. (1985, Nov/Dec). Why People Think Computers Can't. In Donnelly (Ed.), The Computer Culture, Cranberry NJ: Associated University Presses.

Mitchell, T., (1997). Machine Learning. New York: The McGraw-Hill. (pp. 52-126).

NASA, (1998). CLIPS Reference Manual, Version 6.10. Houston, Texas: NASA Johnson Space Center.

Natarajan, B. (1991). Machine Learning: A Theoretical Approach. San Mateo: Morgan Kaufmann. (pp. 167-195).

Nicolis, G., and Prigogine, I. (1977). Self-Organization in Nonequilibrium Systems: From Dissipative Structures to Order through Fluctuations. New York: John Wiley and Sons. (pp. 429-447).

Pipes, A. (1986). Computer-Aided Architectural Design Futures. Cambridge: Butterworths.

Pohl, J. (1995, August). The Representation Problem in CAD Systems: Solution Approaches, 8th International Conference on Systems Research, Informatics and Cybernetics, Baden-Baden, Germany.

Pohl, J., Myers, L., Cotton, J., Chapman, A., Snyder, J., Chauvet, H., Pohl, K. J., and Porta, J. L. (1992). A Computer-Based Design Environment: Implemented and Planned Extensions of the ICADS Model. CAD Research Center, Cal Poly, San Luis Obispo, CA, Technical Report: CADRU-06-92.

Pohl, J., Chapman, Cotton, J., A., Pohl, K. J., Primrose, J., and Wozniak, A. (1997).Decision-Support Systems: Notions, Prototypes, and In-Use Applications. CAD Research Center, Cal Poly, San Luis Obispo, CA, Technical Report: CADRU-11-97.

Shen, W. (1994). Autonomous Learning from the Environment. New York: W. H. Freeman and Company.

Smith, S. M., Ward, T. B., and Finke, R. A. (1995). The Creative Cognition Approach. Cambridge, MA: MIT Press.

Somayaji, M., (1993). Kitchen Design Satisfaction Among a Selected Sample of the Mobility Impaired, and Recommended Design Modifications. San Luis Obispo: MS. Thesis in Computer Science, Cal Poly, San Luis Obispo, CA.

Staddon, J., (1983). Adaptive Behavior and Learning. Cambridge: Cambridge University Press.

Tarpy, R., (1975). Basic Principles of Learning. Glenview, Illinois: Scott, Foresman and

Company.

Taylor, D. (1998). Object Technology: A Manager's Guide. Massachusetts: Addison Wesley Longman.

Tsypkin, Y. (1973). Foundations of the Theory of Learning Systems. New York: Academic Press.

Thomas, L. (1974). The Lives of A Cell. New York: The Viking Press, Inc.

Towle, A. (1993). Modern Biology. Orlando, Florida: Holt Rinehart Winston.

Turban, E. (1992). Expert System and Applied Artificial Intelligence. New York: Macmillan Publishing Company.

Webster's New World Dictionary. (1991). New York: Simon and Schuster.

Wesner, D. G. (1994). Representation of Three-Dimensional Objects in a Semantic Modeling System. San Luis Obispo: MS. Thesis in Architecture, Cal Poly, San Luis Obispo.

Wuilleumier, M. (1992). ICEMS: A High-Level 'Strategy' Approach to Knowledge Acquisition. San Luis Obispo: MS. Thesis in Computer Science, Cal Poly, San Luis Obispo, CA.