Advanced Research in Computers and Communications in Education
G. Cumming et al. (Eds.)
IOS Press, 1999

Applying the Evaluation Model of Problem Solving to Agent-based Instructional System

Janie Chang, Maiga Chang and Jia-Sheng Heh
Dept. of Info and Comp Eng., Chung-Yuan Christian Univ., Chungli, 32023, Taiwan
aptjanie@ms29.hinet.net, maiga@ms2.hinet.net, jsheh@cycs01.ice.cycu.edu.tw

This paper proposes an intelligent tutoring architecture by applying the evaluation model of problem solving to intelligent tutor. Such an evaluation model is based on the problem solving graph (PSG). According to the evaluation model, a suitable operation for solving a problem can be hinted by the intelligent tutor (agent). An experiment environment of agent-based instructional system is also constructed with Microsoft Excel 97 based on both evaluation model and the architecture of intelligent tutor for demonstrating purpose.

Keywords: **problem solving, problem solving network, intelligent tutor, cooperative learning**

1. Introduction

Computer Assisted Instruction (CAI) divided knowledge into simple unit for users to learn easily and conveniently mention at 1995 by Tiffin and Rajasingham. As time goes by, Internet plays an important role in this high technology world. By this way, traditional central CAI software is replaced by the distanced education that means teachers and learners can learn on different places. [2]

In distanced education, one kind of representation of thinking could also be performed. In such process, a problem has many characteristics: the givens, the goals and obstacles.[4] The given contains a certain state with certain conditions, objects and pieces of information; whereas, the goal indicates the desired or terminal state of the problem. The procedure of problem solving is to transform the problem from the given to the goal state which is proposed by Johnson at 1972.

In order to evaluate the learning state of learners in such environment, this paper proposes an evaluation model of problem solving to agent-based instructional system. In Section 2, the graph of PSG/PSN in the process of problem solving is formulated. Besides, several evaluation issues are also described in this section. Section 3 presents a modified problem solving path, called CPSN (Coordinate Problem Solving Network), which can be used to evaluate the performance of problem solving paths. After the CPSN evaluation model is presented, the architecture of cooperative interaction of intelligent tutor (agent) is designed by Section 4. An experiment system and its related intelligent tutor are then implemented on Microsoft Excel 97 in Section 5. Section 6 makes a simple summary.

2. Issues in Evaluation Model of Problem Solving

Problem solving is a style of thinking. Simon defined a *problem state* as a description of the elements in the problem and an *operation* is any legal move that changes the problem from one state to another at 1978. Moreover, most researchers define a *problem-solving path* (*PS path*) as a sequence of operations leading from one state to another.[6] For example, a mathematics problem $2X + 3 = 7$, can be saw as an initial problem state. After a validate operation, it can be transited to another problem

state 2X = 7 – 3, and so on. Until the problem can be solved.

A given start conditions are denoted as *initial state* or *source problem* p_s and the final or goal situation is called *goal state* or *destination problem* p_d. The changes of the problem from initial state to goal state through applying possible sequences of *operations* $t_k \in \Pi = \Xi \times \Xi$. [6] These transformations of problems can be illustrated by a directed graph, called *problem solving graph* (*PSG*)

$$PSG(P,T) \in 2^{\Xi} \times 2^{\Pi}, \quad (1)$$

where the node set is a *problem set* $P = \{p_i\} \subseteq \Xi$ and the edge set is an *operation set* $T = \{t_k\} \subseteq P \times P \subseteq \Pi$.

The *immediate problem states* in the problem space Ξ consist of all the states that are generated by applying an operation to a state. An *operation* in Eq.(1) transforms one problem state into another state, that means,

$$p_{i+1} = t_k(p_i) \text{ or} \quad (2)$$

$$t_k(p_i, p_{i+1}) = \overline{p_i p_{i+1}}, \quad (3)$$

which represents an edge incident on p_i and p_{i+1} in PSG.

A PSG is a collection of successful PS paths $\left\{\overline{p_s p_d}^{\,q} = (t_k^q, \ldots, t_l^q)\right\}$ from the source problem p_s to the destination problem p_d can be illustrated by a more sophisticated graph, called *problem solving network* (*PSN*)

$$PSN(P,T,p_s,p_d) = \left\{\overline{p_s p_d}^{\,q} = (t_k^q, \ldots, t_l^q)\right\}, \quad (4)$$

as shown in Figure 1. [7]

An evaluation model should consist of two capabilities, one is indicating the operation choice by learners for solving the current problem state is better or not; the other one is diagnosing the operation made by learner is correct or not. The former capability can measure the performance about the operation which learner taken, and the last one is used as learners going wrong. The PSN described above can only show out the number of solving ways for the particular problem unfortunately, but do not have capability for giving learners a quickly nor correctly information about what next operation of solving problem should be chosen. Besides, the PSN also can not tell which operation learners taken is misuse or just a mistake made by learners. Since both issues of evaluation model can not be achieved by using PSN, a necessary transformation methodology needs to be analyzed. This paper focuses on the evaluation model analysis and design for the former capability.

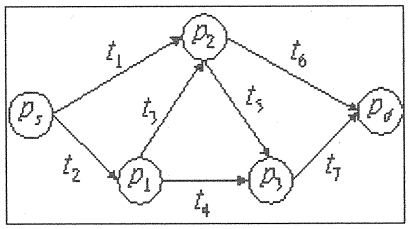


Figure 1. PSN formed by a collection of PS paths

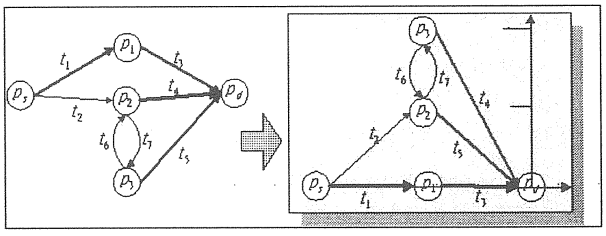


Figure 2. A Example from PSN to CPSN

Even have the evaluation model in the instructional system, there still exists a limited use. Take resource limitation as example, an instructional system is always being distributed nowadays, with the evaluation model centralized will cause slow interaction or even lack of data storage. Therefore, agent-based instructional system is being more significant and efficient in computer aided instrument. These intelligent tutors (agents) could play

different roles, such as learner and instructor, and achieve several cooperative activities during the cooperative problem solving process. Following list three major cooperative activities:

- Cooperative learning: Whenever student is in trouble, he can always consult with the intelligent tutors (agents) and solving a problem together.
- Competitive learning: Everyone is unwilling to be left behind others. On the pretext of mentality for competition, it will be to inspire student for learning.
- Supervised learning: When a student is in trouble or make some mistakes, the agent can be a teacher to provide hints or message to the student.

With the evaluation capability an intelligent tutor (agent) can provide different assistance depends on which role it plays.

After these design issues of evaluation model for problem solving are described, the appropriate transformation methodology should be taken as first consideration in order to translate PSN to more declared and efficient form. The architecture then might be able to design for applying the evaluation model to intelligent tutor in the agent-based instructional system.

3. The Analysis of Coordinate Problem Solving Network

CPSN (*coordinate problem solving network*) is a coordinate graph of PSG/PSN. The objective of CPSN is to rearrange PSN on a coordinate plane, therefore, any operation during problem solving can be evaluated accordingly. In CPSN, two coordinate functions of problems, symbolized as $x(p_i)$ and $y(p_i)$, are defined to give each problem state a unique position in the two-dimensional plane. Following are several requirements which have to satisfied in the design of coordinate functions $x(p_i)$ and $y(p_i)$:

Requirements 1

(A). The $(x(p_i),y(p_i))$-is the coordinate of any problem p_i, and should be unique.

(B). The process of problem solving should be from left to right which means, $x(p_s) < x(p_d)$.

(C). The *destination problem* p_d is placed on the origin ($x(p_d) = 0$) to make a standard point of the evaluation.

The coordinate function $x(p_i)$ above is called *target offset* of problem p_i to the *destination problem* p_d, which is defined as the nearest distance from the problem state p_i to p_d. Formally, the *target offset* can be determined by:

$$offset(p_i) = \min_q \left\{ \text{path_cost}\left(\overline{p_i p_d}^{\,q}\right) : \overline{p_i p_d}^{\,q} \in PSN(P, T, p_s, p_d) \right\}. \qquad (5)$$

In Eq.(5), *offset*(.) places all shortest paths from those possible problem states p_i of solving a specific problem to the *destination problem* p_d. Since it is always possible to find the (non-unique) shortest solving paths from all problem states to the *destination*

problem p_d in a given $PSN(P,T,p_s,p_d)$ through some algorithms, such as Dijkstra algorithm.[7] These shortest paths form an incoming spanning arborescence $ARB(P,T,p_d)$, which includes two components for each problem p_i:

- $pred(p_i),\forall i$, $pred(p_i)$ means the predecessor of problem state p_i. As Figure 1 shown, $pred(p_1)$ is p_s.
- $\min_q \left\{ path_cost\left(\overline{p_i p_d}^{\,q}\right) \middle| \overline{p_i p_d}^{\,q} \in PSN(P,T,p_s,p_d) \right\}$.

Moreover, the farthest problems to the destination p_d, that is *terminal problems* of $ARB(P,T,p_d)$, can be found at the same time:

$$Pterm(ARB(P,T,p_d)) = \{p_i : pred(p_i) = \phi(nullnode) \text{ in } ARB(P,T,p_d)\}. \quad (6)$$

These terminal problems, $Pterm(\cdot)$, can be sorted based on the $offset(\cdot)$ in order to indicate the increasing target offsets of problem solving:

$$order(p_i, ARB(P,T,p_d)) = \begin{bmatrix} p_i \in Pterm(ARB(P,T,p_d)): \\ offset(p_i, ARB(P,T,p_d)) \le offset(p_j, ARB(P,T,p_d)), \\ \forall j > i \text{ and } p_j \in Pterm(ARB(P,T,p_d)) \end{bmatrix}. \quad (7)$$

On the other way, those non-terminal problems must possess more than or equal to one successor, no matter whether terminal or non-terminal problems. Hence, the following definition of Coordinate PSN (CPSN) is made. By the way, those descriptions and proofs of more detailed mathematics formalization of above equations and functions are ignored here for saving space in this paper.

Definition 1 ***CPSN (Coordinate PSN)***

$CPSN(P,T,p_s,p_d,x,y)$ is one kind of $PSN(P,T,p_s,p_d)$, whose (x, y) coordinates of problems are defined as:

$$x(p_i) = -offset(p_i, ARB(P,T,p_d)), \quad (8)$$

$$y(p_i) = \begin{cases} order(p_i,\cdot) - order(p_s,\cdot), & \text{when } p_i = \text{terminal node} \\ \min\{order(succ(p_i),\cdot)\}, & \text{when } p_i = \text{non-terminal node} \end{cases} \quad (9)$$

One CPSN example is shown as Figure 2. And requirements 1(A) to 1(C) in the above analysis will be satisfied.

And finally according to these lemma and theorem analyzed above, an algorithm is then proposed to construct a CPSN from a given series of problem solving paths as follows.

Algorithm 1 ***(CPSN Construction from Problem Solving Paths)***

1) Given the *source problem* p_s and the *destination problem* p_d.
2) Obtain a serial problem solving paths from source p_s to destination p_d, $\left\{\overline{p_s p_d}^{\,q} = (t_k^q, \ldots, t_l^q)\right\}$.
3) Form the corresponding problem solving network $PSN(P,T,p_s,p_d)$.
4) Find the incoming arborescence from destination p_d, $ARB(P,T,p_d)$.

5) Find the terminal problems $Pterm(ARB(P,T,p_d))$ from $ARB(P,T,p_d)$ in Eq. (6).
6) Sort $Pterm(ARB(P,T,p_d))$ by target offsets to find their order in Eq. (7).
7) Find x-coord. or problems from Eq.(8) and then y-coordinates of problems from Eq.(9).

A PSN is those possible problem states (or problems) generated through the process of solving a specific problem. Then the PSN can be simply mapped into a coordinate plane by the above algorithm, is so-called CPSN. Since the shortest path from the *source problem* to the *destination problem* is found out with the analysis made in this section, the better solution operation for the current problem state is possible evaluated or suggested to learners.

4. Designing Architecture for Intelligent Tutors

An architecture should be designed for applying the evaluation model analyzed above to integrated intelligent tutor (agent). Inference model of intelligent tutor (agent) should be designed firstly to archive those cooperative activities described in Section 2. Since CPSN which analyzed previously can be used to provide the evaluation capability for intelligent tutor (agent) inference, therefore, the current problem state should be fed into the intelligent tutor (agent). The intelligent tutor (agent) will be able to solve problems and make appropriate response according to the inference model described as Figure3.

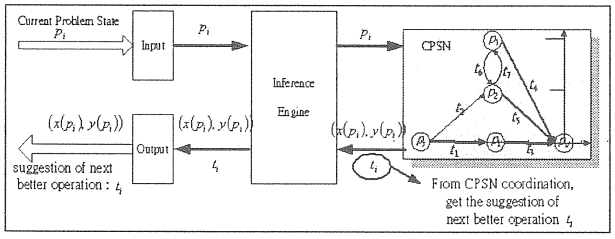


Figure 3. The inference model.

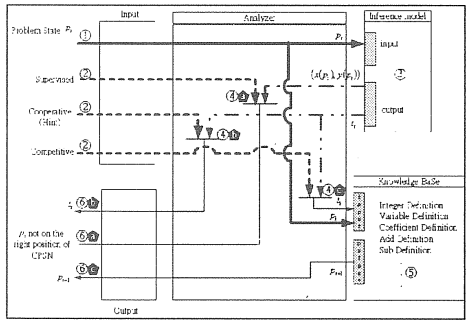


Figure 4. Architecture of intelligent tutor with evaluation capability

In Figure 3, the current problem state p_i is fed into the inference engine, which get the position of p_i with coordinate $(x(p_i),y(p_i))$ mapped on the CPSN. Then the next better operation t_i is got from the CPSN coordinate, and output to be the next better operation suggestion or output coordinate $(x(p_i),y(p_i))$ if not on the right position of CPSN.

Secondly those three cooperative activities that have been introduced in Section2 should be reminded to complete the architecture of intelligent tutor (agent) design. Figure 4 below indicates the architecture of the intelligent tutor with evaluation capability in order to accomplish cooperative activities.

In Figure 4, the architecture of intelligent tutor (agent) with evaluation capability is represented by the flow control Petri Net. [5]

- Step 1: A current problem state p_i is fed currently into the evaluation model.

- Step 2: Cooperative activity type is also fed into the evaluation model currently.
- Step 3: Inference model analyzed before by Figure 3 get the input problem state p_i and then output the next better operation t_i or coordinate $(x(p_i),y(p_i))$.
- Step 4: Considering three major cooperative activities.
  - Step 4a: If the cooperative activity type is supervised and inference model output is coordinate $(x(p_i),y(p_i))$, the output of evaluation model is Step 6a.
  - Step 4b: If the cooperative activity type is cooperative (hint) with inference model output t_i and then the evaluation model will go to Step 6b.
  - Step 4c: If the cooperative activity type is competitive with inference model output t_i as Step 4c, in this time, intelligent tutor (agent) would complete one operation to archive next problem state p_{i+1} by t_i and p_i.
- Step 5: p_i and t_i are input for knowledge base and output p_{i+1}.
- Step 6: Get the output of evaluation model.
  - Step 6a: Applying to intelligent tutor (agent) to remind the learner who goes the wrong problem state with wrong operation.
  - Step 6b: Providing t_i to agent try to give the learner the next better operation.
  - Step 6c: Intelligent tutor (agent) gets the next better problem state p_{i+1}.

When several problem solving paths are generated, the specific CPSN can be modeled. To apply the evaluation model analyzed previously, an intelligent tutor/agent monitors operations made by learner and use the CPSN to determine which operation is better or not.

5. Experiment System

An experiment system of the agent-base instructional system for mathematics is designed based on the architecture of IVC (Internet Virtual Classroom)[1][3] in this paper. Figure 5 shows the architecture for such instructional system following.

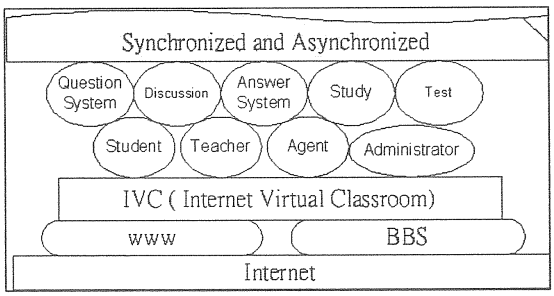


Figure 5. Architecture of the agent-based instructional system

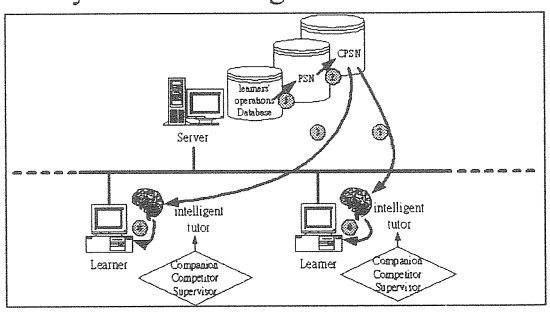


Figure 6. Architecture of experiment instructional system

In Figure 5 shown below, there are several members and roles, such as learner, instructor, intelligent tutor (agent) or even software component. As Figure 6 shown above, each member in this experiment instructional system learns cooperatively when intelligent tutor (agent) plays different roles in three major cooperative activities introduced in Section 2. Intelligent tutor could play as a companion, competitor and supervisor for each learner.

- Step 1: Under the process of solving problem, every problem solving operation is recorded in the learners' operations database. And a PSN can be generated by all possible problem solving paths from learners' operations database.
- Step 2: A CPSN then is translated by PSN is modeled.
- Step 3: The CPSN provides an evaluation model for intelligent tutor.
- Step 4: Intelligent tutor (agent) plays different roles for each learner in the cooperative instructional system.

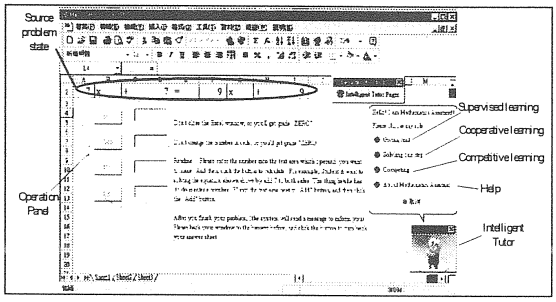


Figure 7. Snapshot of experiment system

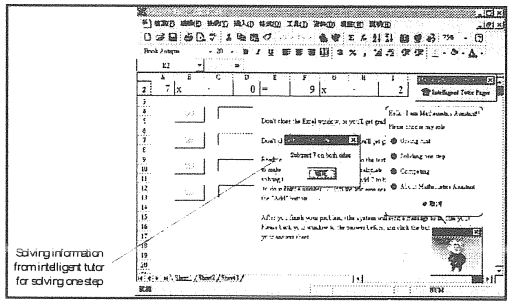


Figure 8.Intelligent tutor solve one step for learner

Considering the benefits of component-oriented analysis, the ActiveX technology of Microsoft is chosen to implement such a system described above. It is easy to divide the whole system into components with abstract models of abilities and behaviors. Besides the ActiveX technology, the VBA (Visual Basic for Application) in the Office 97 is also used to simplify the operation of learners. The snapshot of the experiment instructional system is shown as Figure 7.

In Figure 7, there is a *source problem state* indicates the specific mathematics problem: $7X + 7 = 9X + 9$. Learners could have their operation from operation panel on the left side, and also have their choices about which role intelligent tutor (agent) would play to learn cooperatively. As Figure 8 shown, intelligent tutor plays as a cooperative learner, which solves one step for learner.

6. Conclusion

An evaluation model for providing several cooperatively learning activities of intelligent tutor (agent) in the instructional system, such as hints given, one step tackled and problem solved is proposed and analyzed in this paper. These operations to solve a specific problem compose a problem solving path. By simulating all possible paths, a PSN can be generated. Through the algorithm designed in Section 3, the PSN can be translated to coordinate PSN (CPSN).

Based on the analysis of lemma and theorem of CPSN, the appropriate operation can be simply taken for the better solution of the problem. This paper also designs the architecture of cooperative interaction for intelligent tutor in the instructional system. With the architecture intelligent tutor can accomplish all cooperative activities, and finally an experiment system of mathematics is also implemented on Microsoft Excel 97. An intelligent tutor/agent can give learners hints from the architecture with evaluation capability

based on the CPSN.

Although from the architecture of intelligent tutor/(agent) with evaluation capability proposed in this paper could evaluate the better operation or indicate inaccuracy of problem solving, this could not be known which operations make the incorrect answer by learners. Try to find the impact of operations to problem state and induce several kinds of operation errors from learners would be the future work based on CPSN designed in this paper.

References

[1] Yi-yi Lin, Jyh-Cheng Chang and Jia-Sheng Heh, "IVC: Internet Virtual Classroom," Global Chinese Conference on Computers in Education '99, Macao, 1999

[2] D. Keegan, *Fundations of Distance Education*, Routledge, 1991

[3] J.-L. Lu, K.-Y. Chu, T.-W. Chan, K.-Y. Yang and J.-S. Heh, "Social Learning Systems for Mathematical Problem Solving," *13th Science Education Workshop,* Taipei, 1997

[4] R. E. Richard, *Thinking, Problem Solving, Cognition*, 2nd ed., W. H. Freeman and Company, 1992

[5] James L. Peterson, "Petri Net Theory and the Modeling of Systems", Prentice-Hall, Inc., Englewood Cliffs, 1981

[6] S Russell and P. Norvig, *Artificial Intelligence – A Modern Approach*, Prentice-Hall, 1995

[7] K. Thulasiraman and M.N.S. Swamy, *Graphs: Theory and Algorithms*, John Wiley & Sons, Inc., New York, 1992