T. Hirashima et al. (Eds.) (2011). Proceedings of the 19th International Conference on Computers in Education. Chiang Mai, Thailand: Asia-Pacific Society for Computers in Education

Ontological Approach to Support Authoring for Game-Based Learning Environments

Takanobu Umetsu[a], Takuya Azuma[a], Tsukasa Hirashima[b], Akira Takeuchi[a]
[a]*Computer Science and Systems Engineering, Kyushu Institute of Technology, Japan*
[b]*Department of Information Engineering, Hiroshima University, Japan*
umetsu@ai.kyutech.ac.jp

Abstract: We previously implemented an authoring system to generate a learning environment based on a card game expressed in a flowchart. We improved the authoring system to automatically generate a learning environment by using words or sentences drawn from the rules of card games. The improvement is based on a card game model that is a structured representation of words and sentences from the rules of card games. In this paper, the card game model and the improved authoring system are introduced. The experimental evaluation of the improved system is also reported.

Keywords: Game-based learning, Authoring, Ontology, Automatic generation

Introduction

We have developed an authoring system to generate a learning environment based on a card game from a flowchart that consists of operations and if-then statements [1]. However, it is difficult to describe the flowchart to create the environment exactly according to an author's intention. Therefore, we improved the authoring system. The improvement is based on a card game model that is a structured representation of words and sentences from the rules of card games. In this paper, the card game model and the improved system are introduced.

In game-based learning, the activity to play is not only attractive as a game but is also useful for learning. Because it is a promising approach to achieve highly motivated learning, it is one of the most important topics in the research of computer-based learning environments. To date, many computer-based learning games have been implemented [2].

However, it is difficult to develop a learning game, because it is necessary to satisfy both the gaming and learning aspects in an activity. Malone suggested that a learning game should be designed such that the skills required to play the game are also as valuable as learning the goals [3]. Klawe analysed the role of navigation and interaction [4]. Maragos emphasized the role of multiple stages [5]. Halff dealt with adventure games, in which the player assumes the role of a character in a fantasy world. He proposed that a learning game should restrict the behaviour of a player to make the player learn in a suitable way [6].

There have been several investigations on the design methods of learning games; however, most of these investigations deal with only a restricted part of the design process. Therefore, developers are required to have sufficient knowledge and abilities in both learning and games development.

We thus investigated concrete methods to embed problem-solving exercises into an existing card game. We call this the EPIC (Embedding Problem-solving exercises Into a Card game) method [7]. In this method, an existing card game is transformed into a learning game by exchanging cards of the game for cards with problem statements. The answer to the problems is used instead of the property of the original cards. In this way, a player uses the

exchanged cards to solve the problems. We also implemented an authoring system to automatically generate computer-based learning games from a flowchart based on EPIC method. The results of the experimental evaluation confirmed that the application generated many useful learning games [1].

However, it is difficult to create the flowchart to generate the environment just as an author intended. In the experimental tests of this authoring system, which will be introduced in this paper, there were many authors who were unable to use the system or took long time to create a simple learning game.

We think this difficulty is caused by the gap between the flowchart and playing the game-based learning environment. Therefore, we developed a model that bridges the gap. The model is a structured representation of the components of the flowchart and the rules of the activity, and they are related by subsumption (is-a) and meronymy (part-of) relations. In other words, the model explains a way to transform the rules into the flowchart.

Based on the model, we improved the authoring system. The new authoring system generates a learning environment from words or sentences that are used to describe a card game rule. The new authoring game can transform the words and sentences into a flowchart with the model and generate a game-based learning application from the flowchart.

This paper introduces the models and the improved authoring system. The experimental evaluation of the system is also reported.

1. Related Work

1.1 EPIC Method

The embedding problem-solving exercises into a card game (EPIC) method is a design method for game-based learning that embeds problem-solving exercises into a card game [7]. Figure 1 shows the framework of the EPIC method.

A card game is played using cards, for example, “Poker”, “Blackjack” and “UNO”. In such games, each card has certain properties, such as “number”, “mark (suit)” and “color”. The values of these properties are utilized while playing the card game. For example, Spit (also known as Speed or Slam) is a card game in which a player can place a card on another card if the numbers of the two cards are sequential.

For playing the game, operations of the cards are decided based on only three evaluations of the card’s value: assignment, comparison and calculation. Therefore, an existing card game is transformed into a playable new game by exchanging cards of the game for other cards having properties on which the three evaluations can be performed.

In other words, a new game can be developed by exchanging cards of an existing card game for cards with problem statements. The problem statement consists of given information and questions. A question is used instead of the property of the original card. The answer to the question is used instead of the value of the property. Thus, for playing the game created by the exchange, a player must derive answers from the given information, because the answers are used instead of the values of properties. Because the derivation activity is a problem-solving exercise, this game is game-based learning.

As an example, the Spit game, mentioned above, is transformed into a learning game for arithmetic calculations. A Spit game card has only “number” as a property. For example, the value of the “six of hearts” is 6, the value of the “seven of hearts” is 7, and the value the “eight of diamonds” is 8. A player knows the card’s number. In the Spit game, a player places a card on another card if the two cards are sequential in number. For example, a player can place the “six of hearts” or the “eight of diamonds” on the “seven of hearts”. The left side of Figure 2 shows this example, in which numbers are compared in the Spit game.

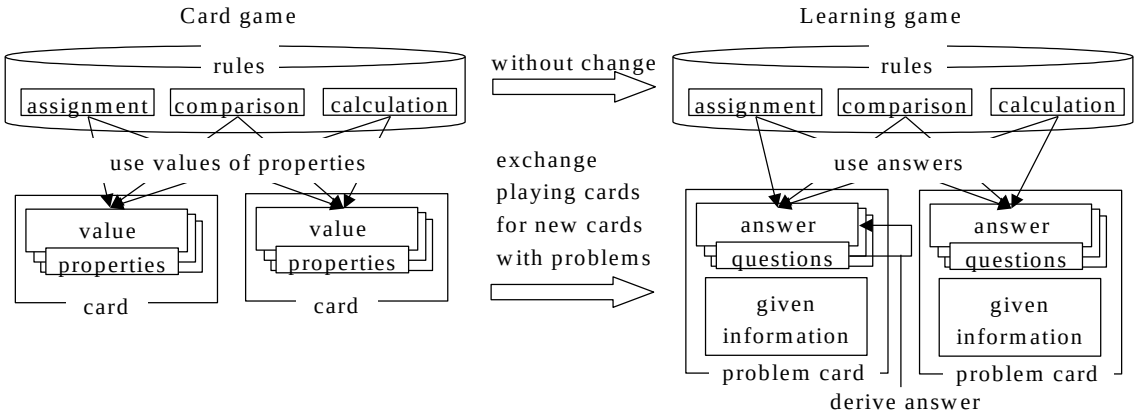

Figure 1. Framework of EPIC method

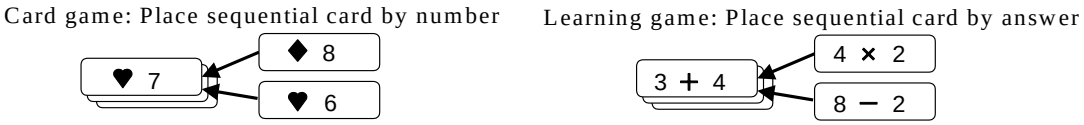

Figure 2. Example of learning game created by the EPIC method

Using the EPIC method, the Spit game is transformed into a learning game using new cards with problem statements instead of playing cards. For example, "Tom had 7 candies. Bill had 3 candies. Then, Bill received 3 candies from Tom" and "How many candies does Tom have?" are written on a new card (card A). "Tom had 3 candies. Bill had 1 candy. Then, Bill received 2 candies from Tom," and "How many candies does Bill have?" are written on another card (card B). The former sentences are given information, while the latter sentences are questions. In this learning game, a player has to solve the problem shown on the card, because the answer to the question is used instead of the value of a number. For example, a player can place the card A on the card B, because the answers are sequential numbers, that is, "4" and "3", respectively. A player must solve the problems to judge whether or not the cards can be placed. Figure 2 shows another simple example of a learning game created using the EPIC method from the Spit game and arithmetic formulas. In the example, questions of the problems are omitted because the questions are obvious.

1.2 Generation of Game-based Learning from Flowchart

Based on the EPIC method, we have implemented an authoring system to automatically generate a game-based learning application from a flowchart of an existing game and problems. The flowchart is created by an author from primitive parts. Figure 3 shows an example of a part of the flowchart.

There are ten parts: start game; end game; move card; flip card; choose card; shuffle array of cards; assignment value or answer of calculation into variable; choose value; if-then-else statements; and print value. The parts are related by an arrow coming from one part and ending at another part, which represents that control passes to the symbol in direction of the arrow.

Flowcharts of various games can be described with the these ten parts. The results of the experimental evaluation confirmed that the authoring system generated 120 learning games from 29 card games and nine sets of problem-solving exercises. Most of the generated learning games were playable.

However, it is difficult to create the flowchart to generate the environment exactly according to the author's intention. In the experimental tests of this authoring system, which

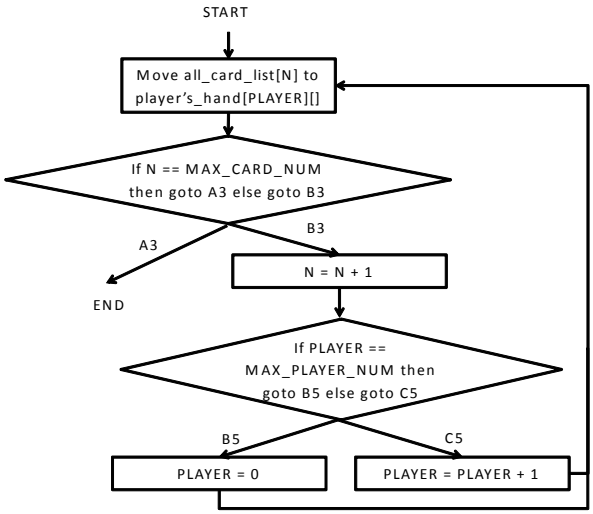

Figure 3. Example of a Part of Flowchart (Translated)

will be introduced in Section 4, there were many authors who were unable to create the flowchart or took a long time to create a simple environment.

The authors may not know which function of the environment the part can perform and how to combine these parts to implement a function of the environment. For example, to implement "the player will take additional two cards in next turn", several parts are needed. One of the parts needed is "assignment value or answer of calculation into variable" as "number of additional cards = number of additional cards + 2". It is difficult to associate "assignment value or answer of calculation into variable" with "the player will take an additional two cards in the next turn." As another example, to implement "deal all cards to the players", "move card" and "if-then-else statements" are combined to implement "move card" many times. Figure 3 shows the detail of this flowchart. However, one author in the experimental tests never thought of the combination.

We think this difficulty is caused by the gap between the ten parts and playing the game-based learning environment. Although the primitive parts can become various flowcharts, the parts are so primitive that there is a huge gap between the parts and flowcharts. Moreover, there is no explanation of which function of the environment the part can perform and how to combine these parts to implement a function of the environment.

Under the circumstances, we developed a model that bridges the gap. We call it the "card game model." We will introduce the card game model in the next section.

2. Card Game Model

Based on an examination of 184 card games, we developed the card game model. We listed typical nouns, verbs and sentences from the rules of 184 card games. The card game model is a structured representation of the nouns, verbs, sentences, and the parts of a flowchart with subsumption (is-a) and meronymy (part-of) relations.

The subsumption relation shows examples of functions of the environment the part can perform. The meronymy relation explains how to combine these parts to implement a function of the environment. In other words, the card game model explains a way to transform the rules into the flowchart.

The card game model is shown in Figure 4. Because the model is so large the detail of the model shown is considerably limited. The model was developed from Japanese rules of the games, so it was meant for Japanese user. The model shown in Figure 4 a direct translation from Japanese, and hence may seem artificial.

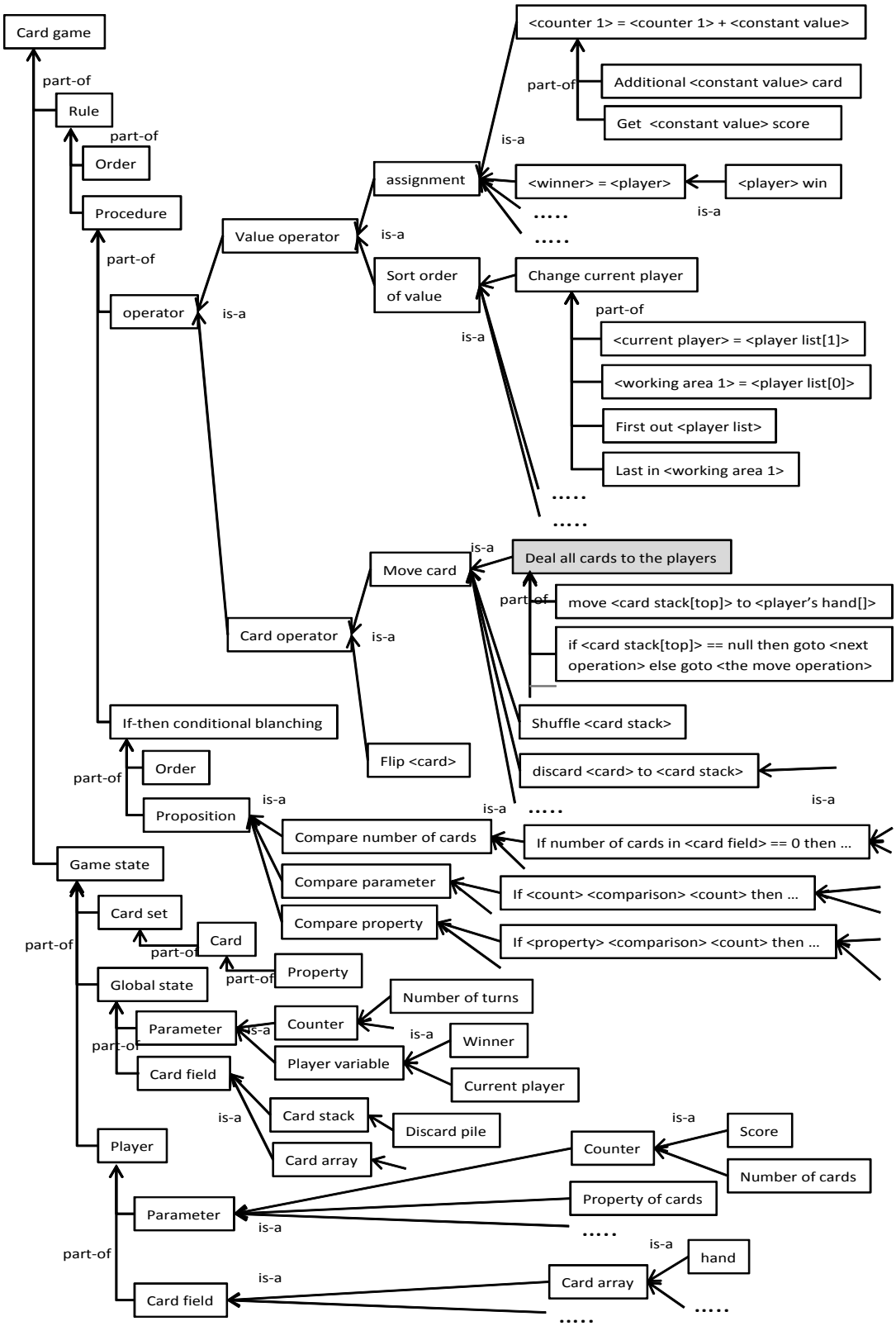

Figure 4. Card Game Model (Translated)

3. Improvement of the Authoring System

We improved the authoring system based on the card game model. As mentioned in Section 2, it is difficult to create the flowchart to generate the learning game exactly according to the

author's intention, because of the gap between the parts and rules of learning games. Therefore, we improved the authoring system to generate a learning environment by using words or sentences within the rules of card games. If to implement a learning game based on Memory game with the old authoring system, the author has to create the flowchart in the left side of Figure 5. On the other hand, if to implement the same learning game with the improved system, the author simply has to enter the sentences in the right side of Figure 5 by clicking the sentences from the card game model. The improvement is based on a card game model that is a structured representation of words and sentences drawn from the rules of card games.

Three new functions of the authoring system were implemented: (1) show instances of parts by the subsumption relation; (2) assemble automatically by the meronymy relation; and (3) operate on structured data. We will explain these functions, but the examples of these functions seemed artificial because the card game model was developed from Japanese rules of the games as mentioned in Section 2.

First, we will explain (1) show instances of parts by the subsumption relation. The subsumption relation shows examples of functions of the learning game that the parts can perform. The examples help the author choose the appropriate part to implement or automatically generate the function exactly according to the author's intention. For example, to implement "the player will take an additional two cards in the next turn", the author searches for similar words from the card game model in the screen of the authoring system. The author can find that "additional <constant value> cards" is a subclass of "<count 1> = <count 1> + <constant value>" and "<count 1> = <count 1> + <constant value>" is a subclass of "assignment <value> into <variable>". Also the author can find that "number of cards" is subclass of "count". Then, the author clicks the "additional <constant value> cards", chooses 'number of cards" from the dropdown-box of <count> and enters the name of the variables of "number of cards" and the constant value of the sentence. If the author enters "next_card_num" as the name of the variables and "two" as the constant value, the authoring system automatically generates "next_card_num = next_card_num + 2" based on the card game model.

Second, we will explain (2) automatic assembly by the meronymy relation. This helps the author when the author does not know how to combine the parts to implement a function of the learning game. The authoring system can automatically generate a function from words that are used for a description of rules based on the meronymy relation in the card game model. For example, to implement "deal all cards to the players" with the old authoring system, the author had to create the flowchart as shown in Figure 3. With improved authoring system, the author only clicks "deal all cards to the players," which is the gray box in Figure 4, from the card game model in the screen of the authoring system. In

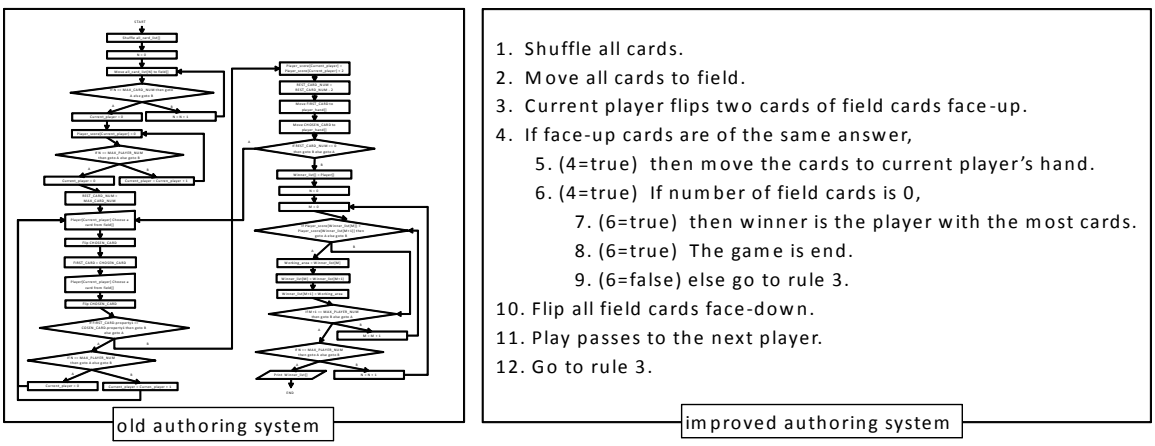

Figure 5. Difference between the Old and Improved System

the card game model, the "deal all cards to the players" consists of "move <card stack[top]> to <player's hand[]>," "if <card stack[top]> == null then goto <next operation> else goto <the move operation>" among others. The authoring system automatically generates the flowchart of the function based on the meronymy relation.

Third, we will explain (3) operation on structured data. In the card game model, data of the game state, which are card array, card stack, player, player hand, player score, among others, are structured. Therefore, we implemented not only operations on one data but also operations on structured data. For example, to implement "a player who has the biggest-value card gets a score of 3 points" with the old authoring system, the author had to create a flowchart that compared two cards many times with working area that contained a bigger-value to know the biggest-value card. On the other hand, with the improved authoring system, the author only clicks "a player who has <characteristics of a card> gets score <constant value> points" and enter the bigger-value card as the <characteristics of a card> and 3 as the <constant value>. Furthermore, because players, cards and score are not related in the old authoring system, the author had to create a flowchart including the relations in a complicated way. On the other hand, with the improved authoring system, it is easy to create the functions in the above way, because they are structured.

4. Evaluations

We conducted two experimental evaluations. One was an experiment for confirming that the improved authoring system could generate various learning games. The other was to confirm that it was easier to create a learning game using the improved method than the old one.

In the first evaluation, two university students of computer science and systems engineering created 52 learning games from 10 card games and 7 problem-solving exercises using the improved authoring system. They played 52 games and confirmed that they were playable. As additional information, experimental tests on 14 of 52 games were already conducted [1][7]. We confirmed that 14 games were useful for learning and they were as enjoyable as the original games.

In the second evaluation, eight subjects created a learning game using both improved and old authoring systems. The subjects were engineering graduates of a university. The learning game they created was based on the game Memory and arithmetic formulas. The learning game was one of the confirmed 14 games. All subjects were familiar with the Memory game and arithmetic formulas. Also, the subjects played the learning game based on the Memory game and arithmetic formulas that had been developed by us before the experiment.

Four of the eight subjects first created the learning game with the old authoring system, and then created the same game with the improved authoring system. The other four of the eight subjects first created the learning game with the improved authoring system, and then created the same game with the old authoring system. We explained how to use the authoring systems for 20 minutes before the experiment.

In the experiment, the authoring systems recorded the total time for the game creation. If there was a bug when the subjects tested playing the learning game, the subjects had to debug the learning game. The creation time included the debugging time. The number of debugs was also recorded. If the subjects could not create the learning game irrespective how hard they tried, they could give up after 90 minutes.

Table 1 shows the results of the experiment. All eight authors indicated by letters A to H began with the improved authoring system were able to create the learning game. There were four authors who gave up while using the old system. Creating the learning game with

Table 1. Results of the Experiment

Author		Old system		New system	
		Creation time [min]	Number of debugs	Creation time [min]	Number of debugs
Old system -> New system	A	Give up (90)	0	141	8
	B	Give up (117)	0	173	4
	C	566	11	152	5
	D	312	5	126	2
New system -> Old system	E	510	35	210	22
	F	Give up (161)	3	203	4
	G	Give up (110)	0	133	4
	H	491	10	191	3

the old authoring system always took a longer time and had more bugs than the new one. The results suggested that that it was easier to create a learning game than the old one

5. Conclusions

We have implemented an authoring system to generate a learning environment based on a card game from a flowchart. We improved the authoring system to generate a learning environment from words or sentences drawn the rules of the card games. The improvement is based on a card game model that is a structured representation of words and sentences within the rules of card games. Through experimental evaluation, we confirmed that it was easier to create a learning game using the improved system than using the old system.

In future work, we hope to examine the differences between the effectiveness of the learning game created by the improved authoring system compared to the old one. We also plan to confirm that the improved authoring system can generate more varied learning games. The most important future work is the improvement of the card game model.

Acknowledgements

This work was supported by KAKENIHI (22700816).

References

[1] U. Takanobu, H. Tsukasa, T. Akira (2006). Property Exchange Method for Automatic Generation of Computer-Based Learning Games. *Proceedings of ICCE2006* (pp. 283-490).
[2] Marc Prensky (2001). Digital game-based learning. McGraw-Hill.
[3] T. W. Malone (1981). Toward a Theory of Intrinsically motivating instruction. *Cognitive Science, Vol. 5* (pp. 130-145).
[4] M.M. Klawe (1998). When Does The Use Of Computer Games And Other Interactive Multimedia Software Help Students Learn Mathematics? *http://www.cs.ubc.ca/nest/egems/reports/NCTM.doc*
[5] H. M. Halff (2005). Adventure Games for Science Education: Generative Methods in Exploratory Environments. *Proc. of AIED05 WORKSHOP5: Educational Games as Intelligent Learning Environments* (pp.12-20).
[6] K. Maragos, M. Grigoriadou (2005). Towards the design of Intelligent Educational Gaming Systems. *Proc. of AIED05 WORKSHOP5: Educational Games as Intelligent Learning Environments* (pp.35-38).
[7] U. Takanobu, K. Yoshinori, H. Tsukasa, T. Akira (2007). Automatic Generation of Computer-Based Learning Games for Problem Solving. *Supplementary Proceedings: Poster of ICCE2007* (pp. 65-66)