A Multiagent Reinforcement Learning Approach for Cooperative Game Playing with Users on a Sugoroku-like Board Game

Kihaya Sugiura and Naoki Fukuta

Shizuoka University, 3-5-1 Johoku , Hamamatsu,
Japan 4328011

{gs16021@s, fukuta@cs}.inf.shizuoka.ac.jp

Abstract

In this paper, we present a reinforcement learning based approach to allow a behavior that is cooperative and also competitive for the human players in our sugoroku-like board game playing environment. We model the game as a multiagent environment and the agents as leaning agents which learn their behaviors by using the seven different reward functions to cooperatively play the game with other players. We show the results of a set of experiments to examine how the behaviors of agents on playing the game with imitated human player agents fit to various skill-levels of users.

Keyword: cooperative game playing, multiagent reinforcement learning, diversity

I. Introduction

In recent years, entertainment in computer games is coming to be more diverse, and the relationship between human and the AI is much more complicated because of the development of the AI that intelligently plays a computer game. Game playing AI is as strong as humans (e.g., AlphaGo[1]) and also capable of playing very complex action games[2] which human-players do. However, to make a game enjoyable for diverse skill-level of users, realizing a cooperative behavior of (computer) players to entertain the human players to give them a feel of competing to other (computer) players

as well as overcoming them at the end of the playing. We argue that, realizing a peaceful playing of a competitive game by multiple players is an important goal to make the platform of gaming viable and long living. It could be viewed as a similar one on discussing a shared issue with multiple users[3][4] without breaking their human relations. Furthermore, even on a bilateral negotiation among just two entities is very complex to be analyzed[5], the complexity of three or more entities for a repeated game could be far difficult to be analyzed theoretically.

In this paper, we show an analysis on our reinforcement-based learning approach to allow a behavior that is cooperative and also competitive for the human players in our sugoroku-like board game playing environment. We model the game as a multiagent environment and the agents as learning agents which learn their behaviors by using the seven different reward functions to cooperatively play the game with other players. In the approach, each player agent is initially constructed with different types of reward functions and formed its learning process on a single-agent reinforcement learning with randomly behaving agents. To evaluate our approach, we prepare different skill-levels of imitated human player agents in the game environment and also we prepared several kinds of the learning agents based on the different reward functions for the reinforcement learning.

At the same time, we examine whether the learning agents can induce the entertainable (cooperative and competitive) situation for the user player when multiple agents are set to play with the imitated human player. The purpose of the work is to realize a cooperative behavior of agents for the user players in our multiplayer simulation game environment. We use some techniques used in the multiagent systems[6] and agent learning techniques[7] to realize cooperative player agents on our simulation game environment to design a user-entertainable and also a competitive-playing place in the game play.

II. Background

A. *Difficulties on Analyzing Repeated Multiplayer Games*

A board game is usually played with multiple players and the profits of those players are sometimes conflicted each other in the result of the decision of players. In a simple prisoner's dilemma model that two players to select the cooperation or non-cooperation, it is known that there is a dominant strategy, which does not depend on the decision of the other player[8]. Therefore, in this simple scenario, the players can make their decisions whether they know the other player's decision or not. N-repeated prisoner's dilemma model is a prisoner's dilemma model that has multiple players and players make decisions over times, the process of learning appropriate behaviors can be seen as an evolutional model[8]. Although the N-repeated prisoner's dilemma model is similar to the real world model, on a real world, players cannot repeat making decisions infinitely and the repeats can be finite times by the reason of time limitation. For above reasons, a gaming on a real life often has a time limitation and a termination condition to finish playing the game. They make the analysis of multiplayers' limited-time repeated games difficult without experimental analysis.

B. *Railway Management Simulation Game*

As a platform to analyze multi player repeated games, we designed a game with a motif of railway management. Figure 1 shows the model of railway management game in this paper. This game can be played on a Web Browser. Players go around the stations by throwing the dice when their turn comes. An important rule in this game is that a player can aggressively attack other players after stepping into a yellow station instead of throwing a dice. On this game, obtaining a profit for a player does not always mean the loss of other players and therefore this game is not zero-sum game. Since the map of this game may include a loop structure with branches, players throw the dice to move and then the players have to decide the directions to move.

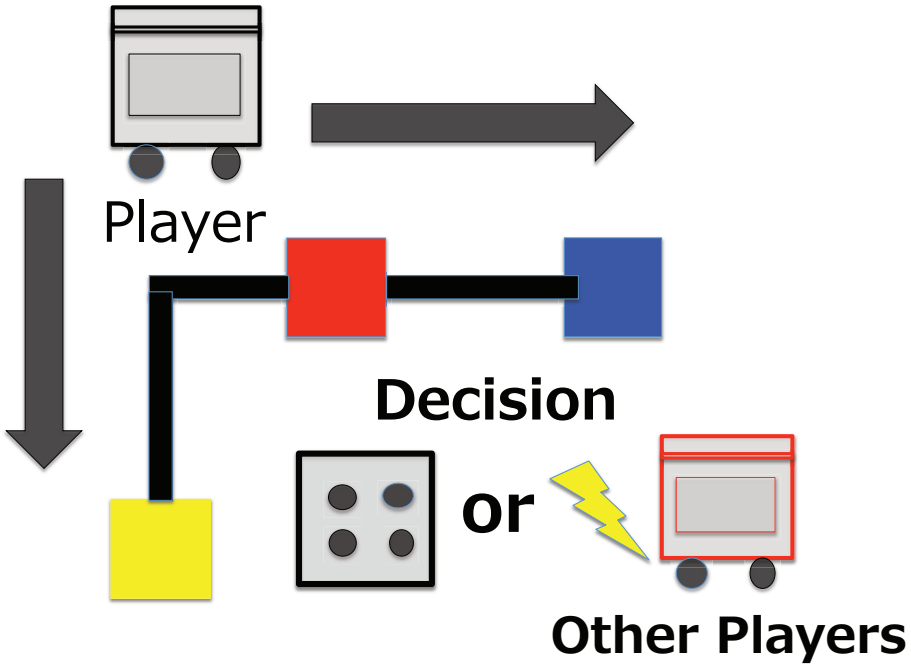


Figure 1. Basic model of the railway management simulation game

Following are the details of our game.

- *Player*

A player is allowed to choose one action from two basic actions: throwing a dice to go around the station, and aggressively attacking to another player when a player is in a yellow station. The aim of each player is to maximize its owned money at the end of the turn of the game.

- *Station*

There are "blue", "red" and "yellow" stations in the game. When a player stopped at a blue station, the player gets the specified amount of money. When a player stopped at a red station, the player loses the specified amount of money. When a player stopped at a yellow station, the player gets a chance to aggressively attack another player. This attack can be done at once in any turn after stopped at a yellow station.

- *Attack*

If a player has a right to attack another player, the player can select to attack a specified player instead of throwing a dice. In such a case, appointed player loses the specified amount of money.

C. *Applying Multiagent Learning*

Wicke[6] suggested that a reinforcement learning-based cooperative behavior could be obtained with a very simple reward function without having complex communications among agents. On an environment that has multiple learning agents, there are two different situations for acquiring their cooperative behaviors. One is the situation that several agents share their purpose or task and they try to communicate each other to share their intensions or the obtained information to achieve the goal. In particular, approaches based on the game theory[9] is one of the possible way. The other situation is that each agent has its own goals with a shared mechanism that naturally enforce the agents to cooperate[6]. Since there are noises on the agent's learning environment because of the agents' exploration behaviors, noise reduction method is often necessary for such multiagent learning environment[7].

III. Implementation

We have implemented our prototype platform to make the game playable on a web browser as well as making some experiments with the learned agents. Figure 2 shows an example of the game platform with a simple map and log view, and Figure 3 shows another example with more complicated maps with a playable user interface. Figure 4 shows the structure of the implemented platform. Here, we used Node.js[10] and WebSocket protocol to implement communications among the Web browsers and our platform. We implemented the WebSocket game server that runs on the Node.js to store and hold the game state object. We also implemented the learning player agents as WebSocket clients to communicate with the game server to play the game on behalf of human players. Because of the implementation on the

WebSocket, our platform does not deeply depend on its running environment. Unlike ordinary WebSocket communications, since the server can intentionally push some data to the clients, it is easy to reflect the game state to all connected clients.

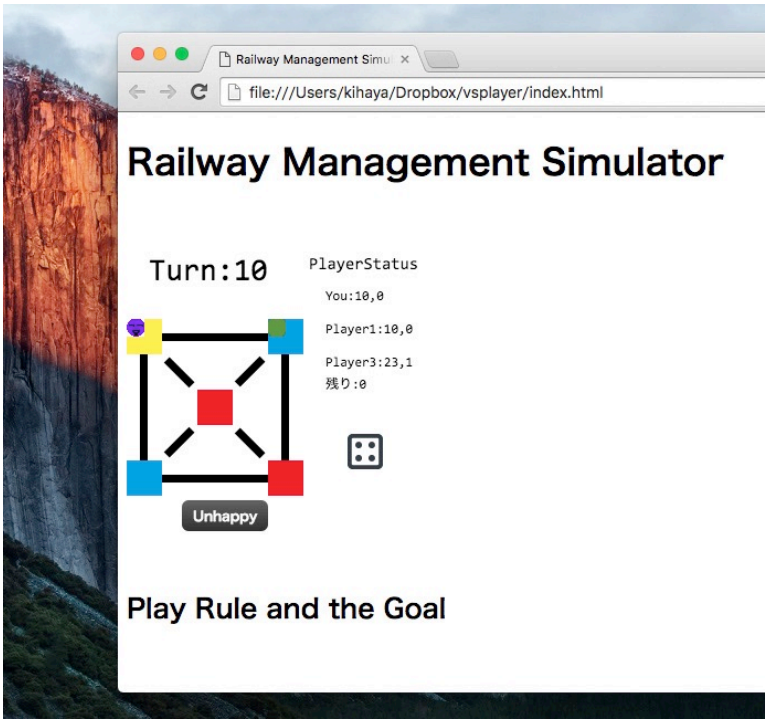


Figure 2. The railway management game (five stations)

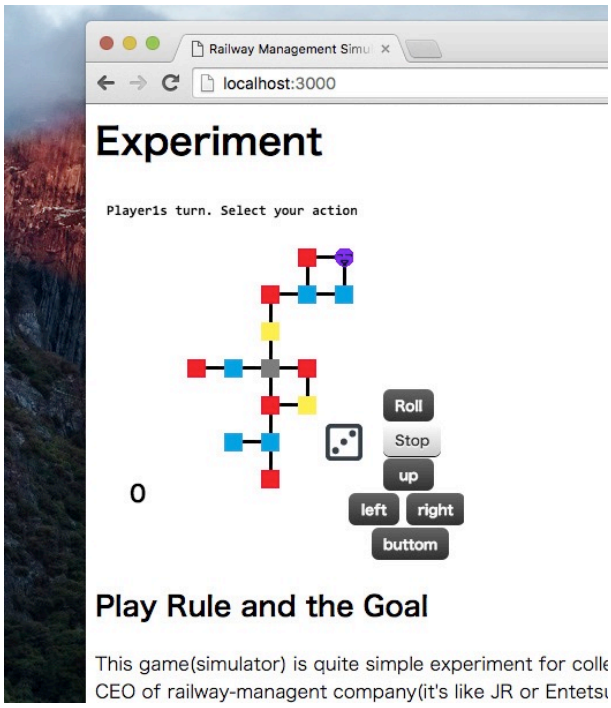


Figure 3. The railway management game (with complex map and playable interface)

A player agent in this paper is implemented as a client program on the client-server model. The player agents connect to the game server and send their decisions to the server when their own turn comes. Player agents have attributes that express the states of the game. For example, a player agent can select to throw the dice or to attack other players when its attack flag is on. If the player agent selected to throw the dice, the agent also calculates where the agent are able to go depending on the points of the dice. After calculating where they can go, the agent makes a decision by using ε-Greedy selection.

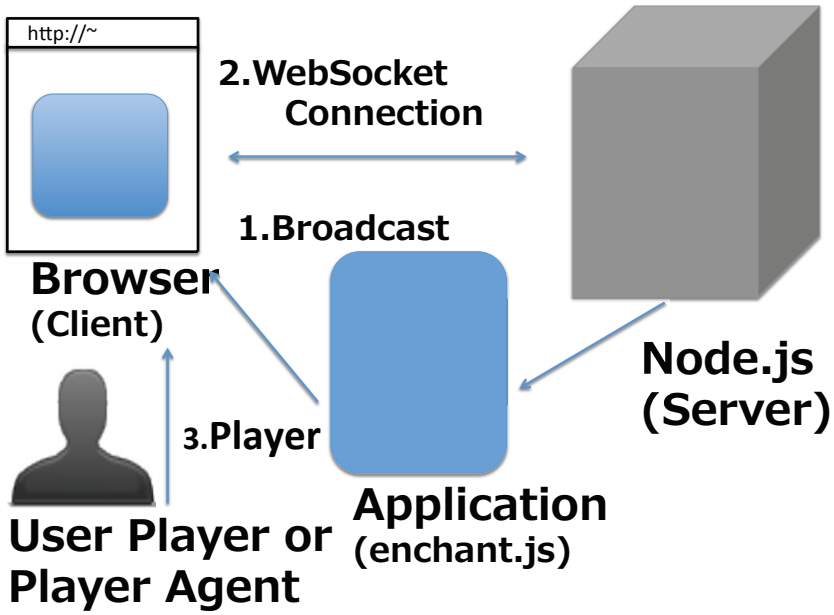


Figure 4. The architecture of implemented platform

Following are some important keywords of the game.

- *Position*

Position of the agent denotes the current station the player agent stopped at.

- *Money*

Stopping at a blue station and getting more money is the most promised way to win the game. At the final turn, comparing the mount of money that each agent has and the player that has largest amount of money is the winner of this game.

- *Attack Flag*

When a player stopped at a yellow station, the attack flag is on. When the attack flag of the agent is on, the player can attack to other players and this makes decrease of the money of the attacked player.

IV. Design of Reward Functions

A. Learning with Reward Functions

In this section, we focused on the acquisition of right behaviors on a Q-Learning agent[11] on the experimental game environment. A leaning agent receives the "reward" from the game environment on the learning steps. We design some different rewards for learning agents to acquire their behaviors.

Following are the details of the prepared reward functions.

- *functionA*

This reward function refers to the actions that the learning agent takes. Going to a blue station or a yellow station and then aggressively attacking to other players will be positive value rewards. Going to a red station will be a negative value reward.

- *functionB*

This reward function refers to the game rank after the player agent's action. This gives its max reward for being at the rank 1 and gives its minimum reward for being at rank 3 when there are three players.

- *functionC*

This reward function refers to the particular player's money after each action of the learning agent. If another player's money is equal to the learning agent's money, it will be a positive reward, otherwise it will be a negative reward.

- *functionD*

This reward function gives a negative reward for the learning agent's aggressive attack after stopping at yellow station. Otherwise this will give positive value reward.

- *functionE*

This reward function refers to the difference of the money between agents. If difference of the money between the learning agent and the closest agent and the value is less than or equals to 2, this will give a positive value reward. Otherwise it will give a negative value reward.

- *functionF*

This reward function refers to the change of the rank in the game. If the rank of a learning agent moves to upward after the learning agent's action, it will give a positive reward. If the rank of a learning agent has been down after the learning agent's action, it will give a negative reward.

- *functionG*

This reward function gives a negative reward when selecting an aggressive attack to other players. If the learning agent selects an action of attacking to other players, it will give a negative value reward. Otherwise it will give a positive reward.

D. Learning Conditions

To make the experiment simple, we set the number of players to three. In this phase, we prepared an agent as a Q-Learning agent and others are random agents that select their action at random. We put five stations in the experimental game environment. We set the maximum turns in one game to 99 turns and conducted 7000 games. Table 1 shows the experimental conditions to acquire the basic behaviors of learning agents.

Table 1. Experimental game condition

num of blue stations	num of red stations	num of yellow stations	num of players
2	2	1	3

As we mentioned above, when a player stopped at a blue or a red station, we increase or decrease the player's money with 1, respectively. If a player decides to aggressively attack other players, we decrease the selected player's money with 2. When a learning agent's turn comes, if it is possible to select an action from throwing the dice or aggressively attacking to other players, the action will be chosen by using ε-Greedy (ε=0.1). If the learning agent decides to throw the dice, the learning agent lists up where the agent is able to go depending on the points of dice throw and selects the direction to go. If the learning agent decides to aggressively attack other players, the learning agent has to select the player to attack. Table 2 shows the performance of learning agents with the shown seven reward functions when playing with two random players.

Table2. Percentages of the actions on each reword functions

function	blue station	red station	yellow station	attack	average rank
A	48.2%	7.25%	43.8%	0.69%	1.1
B	49.3%	3.9%	24.0%	22.6%	1.0
C	50.5%	43.0%	6.1%	0.4%	1.9
D	24.8%	46.5%	27.9 %	0.6%	2.6
E	39.9%	42.1%	14.0 %	3.8%	1.9
F	44.84%	22.60%	20.89%	11.67%	1.29
G	25.68%	47.11%	26.56%	0.65%	2.68

V. Evaluating Cooperative Behaviors

A. Experimental Conditions

In this section, we discuss what kind of player agents can cooperate with human player. Based on the experiment on the acquisition of basic behaviors, the agents that are using evaluation function A or evaluation function B performed with better average ranks on playing the game. Here, we prepared the basis to implement an imitated human player by using the agent's stochastic action selection methods for evaluating cooperative behaviors of learning agents

with actual human players who may have different skills to play the game. We have chosen that as the learning agent learned by the evaluation functionB (reward refers to the rank) and the agent chooses actions based on the learned Q table. This agent is appeared to be able to act like keeping high game rank in the game.

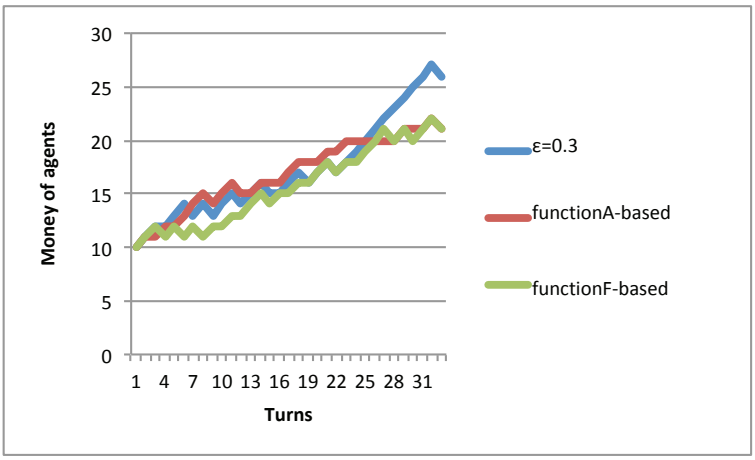
At the time that the imitated human players play the game, it has a mechanism to imitate human player's un-optimal behaviors in the game environment. To realize the behavior, we applied a mechanism that adjusting the ε value in that condition. This shows that, the presented approach could not sufficiently work when the human player is too strong for those learned agents. This is one of the limitations of the presented work. By adjusting the agent's ε values used in the process of choosing probabilistic actions, the agent could imitate different levels of the human players. The imitated human player does not always make decisions that will make benefits for the agent and such random behaviors have been chosen in probability of ε. We conducted the game playing experiment using three player agents. Instead of using a human player, we prepare the imitated human player agent as we mentioned above.

For the ε value for the variety of imitated human player agents, we set 0.1, 0.3 and 0.5 for ε. For the opponent players of imitated user player, we use the best agents, learned with functionA, functionB and functionF, which have been performed in the preliminary experiments with the seven different evaluation functions that we mentioned above. We examined the possible combinations of those agents in the experiments.

B. Results

Figure 5, 6, and 7 show the results of the experiments on game playing that were conducted by three players. In those experimental playing records, one imitated human player and two other learning agents have been used. Here, an important observation is that, in most cases (excluding ε=0.1), having different kind of player agents with a imitated human player can be seen that they cooperatively acting to entertain the user player. In other cases, the human

player has been the leader of the game on their early stages or a computer player leaded the game but the imitated human player had little chance to win. In all cases, the player agents did not have any pre-learning for such cooperative behaviors. This suggests that having a good combination of different kind of players will make the agents cooperative in some conditions. However, at this moment, there is still a question whether this could be viewed as a factor of collective intelligence which can be seen on a human-group[12]. The work could be extended to have an ability to find a better combination of computer players for a specific user to entertain on the game playing.



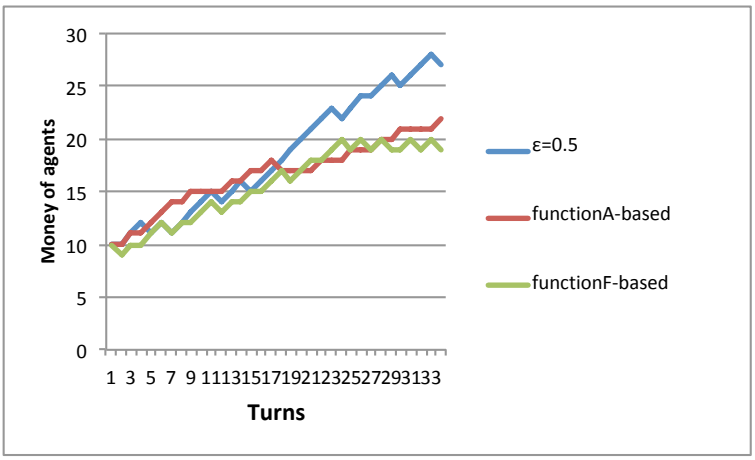


Figure 5. Each agent's transition of money in each turns with imitated human player (ε=0.3:upper,ε=0.5:lower), functionA and functionF (one game)

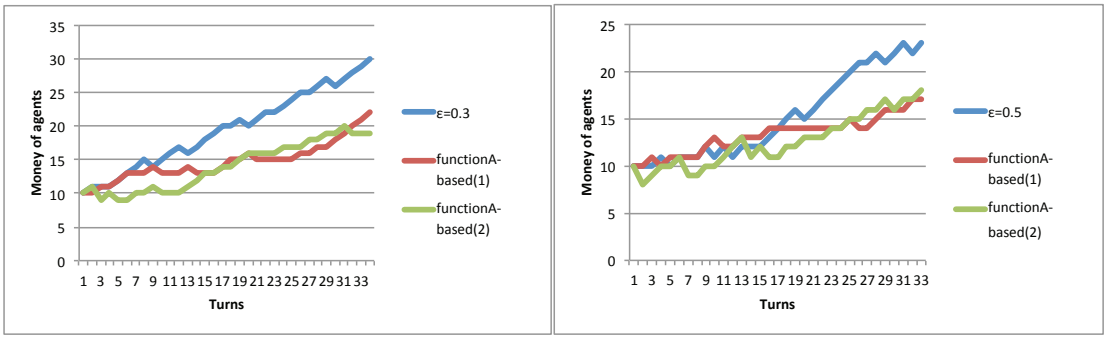


Figure 6. Each agent's transition of money in each turns with imitated human player (ε=0.3:left,ε=0.5:right), functionA and functionA (one game)

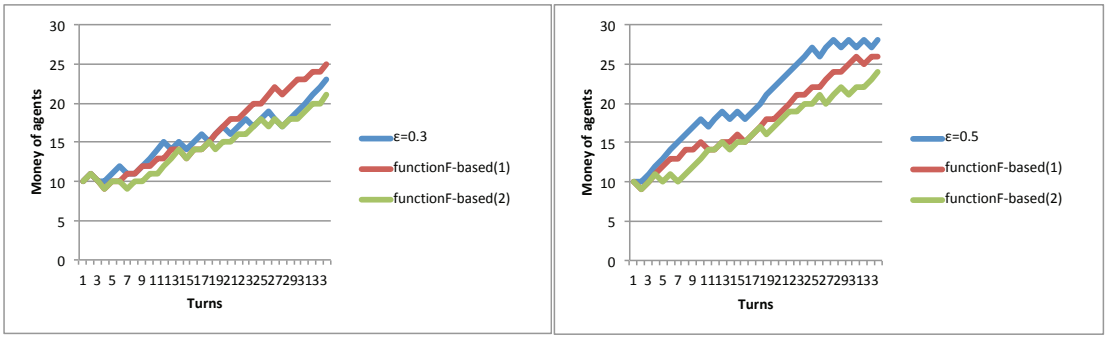


Figure 7. Each agent's transition of money in each turns with imitated human player(ε=0.3:left,ε=0.5:right), functionF and functionF (one game)

VI. Discussion

In the case that the imitated human player with ε=0.1 played with the combination of the agents that is learned by two different evaluation functions, the imitated human player begins to dominate and get the advantage of money among the three players at the turn six. In the case the imitated human player with ε=0.3 or ε=0.5, player's moneys are not so different among three players from the beginning of the game and it continues until the middle to later turns of the game. Finally, the imitated user player wins in that condition. This shows that, the presented approach could not sufficiently work when the human player is too strong for those learned agents. This is one of the limitations of the presented work.

VII. Conclusion

In this paper, we presented an analysis on a reinforcement-based learning approach to allow a behavior that is cooperative and also competitive for the human players in our sugoroku-like board game playing environment. We prepared a simplified sugoroku-like board game design which refers to the competitive railway management simulation game that is a multiplayer board game played on a web browser. In the approach, each player agent was initially constructed with different types of reward functions and formed its learning process on a single-agent reinforcement learning with randomly behaving agents. We prepared different skill-levels of imitated human player agents in the game environment and also we prepared several kinds of the learning agents based on the different reward functions for the reinforcement learning. We examined whether the heterogeneity of learning agents can induce the entertainable (cooperative and competitive) situation for the user player when multiple agents are set to play with the user. As a result, when the human player is on the middle to lower skills at the playing of the game, those heterogeneously constructed team of agents well entertained the playing of the game for the imitated human players. Further discussions could be done on how this kind of good mixture of diverse types[12] of players could be done for the case that has two or more human players.

Acknowledgment

This work is partly supported by Grants-in-Aid for Challenging Exploratory Research 26540162. An initial version of this work has been presented at ICCSE2016[13].

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, “Human-level

control through deep reinforcement learning," Nature, vol. 518, no. 7540, pp. 529–533, 02 2015. [Online]. Available: http://dx.doi.org/10.1038/nature14236

[2] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," in NIPS Deep Learning Workshop, 2013.

[3] T. Ito, Y. Imi, M. Sato, T. Ito, and E. Hideshima, "Incentive mechanism for managing large-scale internet-based discussions on collagree," Collective Intelligence 2016, 2016.

[4] A. Sengoku, T. Ito, K. Takahashi, S. Shiramatsu, T. Ito, E. Hideshima, and K. Fujita, "Discussion tree for managing large-scale internet-based discussions," Collective Intelligence 2016, 2016.

[5] T. Baarslag, A. Reyhan, K. V. Hindriks, K. Fuijita, T. Ito, and M. J. Catholijn, "Automated negotiating agents competition 2010-2015," AI Magazine, 2016.

[6] D. Wicke, D. Freelan, and S. Luke, "Bounty hunters and multiagent task allocation," in Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems,(AAMAS'15),pp.387–394, 2015. [Online]. Available: http://dl.acm.org/citation.cfm?id=2772879.2772930

[7] M. K. Colby, S. Kharaghani, C. HolmesParker, and K. Tumer,"Counterfactual exploration for improving multiagent learning," in Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems (AAMAS '15), pp. 171–179, 2015. [Online]. Available: http://dl.acm.org/citation.cfm?id=2772879.2772904

[8] R. Suzuki and T. Arita, "Interactions between learning and evolution:: The outstanding strategy generated by the baldwin effect," Biosystems, vol. 77, no. 13, pp. 57 – 71, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0303264704000565

[9] Y. Hu, Y. Gao, and B. An, "Learning in multi-agent systems with sparse interactions by knowledge transfer and game abstraction," in Proceedings of the 2015 International

Conference on Autonomous Agents and Multiagent Systems, (AAMAS '15), pp. 753–761. [Online]. Available: http://dl.acm.org/citation.cfm?id=2772879.2773250

[10] "Node.js," https://nodejs.org/en/.

[11] R. S. Sutton and A. G. Barto, Introduction to Reinforcement Learning, 1st ed. Cambridge, MA, USA: MIT Press, 1998.

[12] A. Woolley, C. Chabris, S. Pentland, N. Hashmi, and T. W. Malone, "Evidence for a collective intelligence factor in the performance of human groups," Science, October 2010.

[13] K. Sugiura, and N. Fukuta, "Toward a Multiagent Reinforcemnet Learning Approach for Cooperative Game Playing with Users on a Sugoroku-like Board Game," Proc. of ACE International Conference on Crowd Science and Engineering(ICCSE2016),2016.

Kihaya Sugiura is graduate student of integrated science and technology at Shizuoka University. He is interested in AI and collective behaviors on human and entertainment computing.

Naoki Fukuta is associate professor of informatics at Shizuoka University. He received B.E. and M.E. from Nagoya Institute of Technology in 1997 and 1999 respectively. He received Doctor of Engineering from Nagoya Institute of Technology in 2002. His main research interests include Multi-agetn Systems, Mobile Agents, E-commerce, Applications on Auction Mechanisms, SemanticWeb, Konwledge-based Software Engineering, Logic Programming, and WWW-based Intelligent Agent Systems. He is a member of ACM (Association for Computing Machinery), IEEE-CS(IEEE Computer Society), JSAI (Japanese Society for Artificial Intelligence), IPSJ (Information Processing Society of Japan), IEICE (Institute of Electronics, Information, and Communication Engineers), JSSST (Japan Society of Software Science and Technology), and ISSJ(Information Systems Society of Japan).