Moves by Taking Account into Other Players in Othello

Hisashi Handa
School of Science and Engineering
Kindai University
Kowakae 3-4-1, Higashi-Osaka 577–8502
Email: handa@info.kindai.ac.jp

***Abstract*—In this paper, we introduce a modified Temporal Difference method which takes account into the moves of other players in populations for 1-ply Othello. That is, by using Temporal Learning method, we introduce Memetic Algorithm, i.e., hybridization of EA with local search method. The proposed modified TD-method can maintain the diversity in populations. Experimental results show the effectiveness of the proposed method.**

I. Introduction

Memetic Algorithms, a combination of Evolutionary Algorithms and Local search methods, have attracted much attention in EC studies. In fact, most EC studies for combinatorial optimization problems constitute Memetic Algorithms to solve their problems. In the case of board game researches with Evolutionary Algorithms, the Temporal Difference Learning proposed by Sutton is used as a local search method for Memetic Algorithms. It is difficult to maintain the diversity in the population in Memetic Algorithms if fitness evaluations are carried out for certain champion algorithms. The reason of this is that various individuals tend to converge into the same strategy which is suitable for the champion algorithms. In order to address such difficulty, this paper proposes a modified TDL which takes account into the moves by other players.

The next section introduces the rule of the Othello. Section 3 introduces Temporal Learning Method for the Othello. Section 4 examines two sorts of evolutionary approaches, i.e., evaluation based on leagues and evaluation against known good players. Section 5 shows the Evolutionary Multi-Objective Optimization, by referring to modifed TD-method.

II. Othello

Othello is one of the most famous board games in the world. Othello is played by two players. The size of board is 8x8. Fig. 1 shows the initial state of the board of the Othello. As depicted in this figure, there are two black pieces and two white pieces. In the Othello, the two players put a their piece in turn: The black and white pieces are for the first-move player and the second-move player, respectively. There is a constraint to put a piece of the Othello: We can put a piece where it bookends at least one opponent piece with the self piece along either or both of portrait, landscape, and/or diagonal orientation. The bookended opponent pieces are changed to self-color. If there is no such bookended position, the turn is changed to the opponent. A player with more pieces wins if there is no position to put pieces for both players.

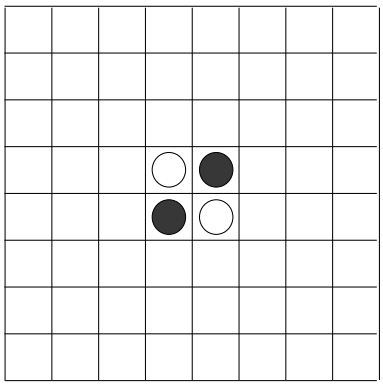

Fig. 1. The initial board configuration of the Othello.

1-ply Othello is one of competitions held in IEEE international conferences such as the CEC and CIG. In the 1-ply Othello, the game tree search, which has been commonly used in conventional board game studies, is not allowed to use. That is, the agents have to decide their moves by using only current board evaluation functions Therefore, we concentrate to develop better evaluation function of boards.

The board evaluation function used in this paper is described as follows: Suppose that x_{ij} represents the state of cell at ith row and jth column ($i = 1, \ldots, 8$, $j = 1, \ldots, 8$).

$$x_{ij} = \begin{cases} 1 & \text{black piece} \\ 0 & \text{no piece} \\ -1 & \text{white piece} \end{cases}$$

We can define a simple board evaluation function $E(\mathbf{x}, \mathbf{w})$ by using the following equation:

$$E(\mathbf{x}, \mathbf{w}) = \sum_i \sum_j w_{ij} x_{ij}, \tag{1}$$

where $\mathbf{w} = \{w_{11}, \ldots, w_{ij}, \ldots, w_{88}\}$ ($i = 1, \ldots, 8$, $j = 1, \ldots, 8$) denotes parameters associated with corresponding cells. The learning or evolution is made by modifying this parameter vector $\mathbf{w}$.

Agents in this paper put a piece by using the following way: Let $m(\mathbf{x})$ be a set of board configurations such that a piece is put against the current board configuration $\mathbf{x}$. We can put

978-1-4673-2743-5/12/$31.00 ©2012 IEEE

a piece by choosing from the succeeding board configuration with the highest evaluation as below:

$$\text{argmax}_{\mathbf{x}' \in m(\mathbf{x})} E(\mathbf{x}', \mathbf{w}) + \xi, \quad (2)$$

where ξ indicates a perturbation for evaluations, a random variable over the normal distribution such that the mean is 0, and the variance is 0.01. In the case of white player, the argmax is changed into argmin.

III. Temporal Difference Learning Method

Temporal Difference Learning method (TD Learning) is proposed by Sutton. Tesauro applied this TD Learning to constitute neural-network agents for Back-Gammon. After this study, the TD Learning has been broadly used in board-games by many researchers. The update scheme of the TD Learning for the ith parameter w_i can be described as follows:

$$\begin{aligned} \mathbf{w}_{t+1} &\leftarrow \mathbf{w}_t + \alpha(v(\mathbf{x}_{t+1}, \mathbf{w}_t) - v(\mathbf{x}_t, \mathbf{w}_t)) \\ &\quad \cdot \sum_{k=1}^{t} \lambda^{t-k} \nabla_{\mathbf{w}} v(\mathbf{x}_k, \mathbf{w}_k), \end{aligned} \quad (3)$$

where α and λ denote the learning rate and the discount rate, respectively. The function v is used to bound the output of evaluation function in equation (1) into $[-1 : 1]$. Sigmoid function is occasionally used for this purpose. This paper adopts the following function as in Simon's paper [4]:

$$v(\mathbf{x}, \mathbf{w}) = \tanh(E(\mathbf{x}, \mathbf{w})) = \frac{2}{1 + \exp(-2E(\mathbf{x}, \mathbf{w}))} - 1 \quad (4)$$

The third term in equation 3 means eligibility trace. $\nabla_{\mathbf{w}} v(\mathbf{x}, \mathbf{w})$ can be described as follows if the evaluation function $E(\mathbf{x}, \mathbf{w})$ is of a linear formula as in equation (1):

$$\nabla_{\mathbf{w}} v(\mathbf{x}, \mathbf{w}) = (1 - v(\mathbf{x}, \mathbf{w})^2)\mathbf{x} \quad (5)$$

Note that the weighted sum of vectors in equation (3) can be calculated by using the weighted sum of vectors at the preceding time.

$$\begin{aligned} &\sum_{k=1}^{t} \lambda^{t-k} \nabla_{\mathbf{w}} v(\mathbf{x}_k, \mathbf{w}_k) = \\ &\quad \nabla_{\mathbf{w}} v(\mathbf{x}_t, \mathbf{w}_t) + \lambda \sum_{k=1}^{t-1} \lambda^{(t-1)-k} \nabla_{\mathbf{w}} v(\mathbf{x}_k, \mathbf{w}_k) \end{aligned}$$

At the end of game, $v(\mathbf{x}_{t+1}, \mathbf{w}_t)$ in equation (3) is replaced with reward R (1 for the first-move player's win; -1 for the second-move player's win).

IV. Moves Taking Account into Other Players

The proposed modification of the TDL is to investigate moves by other individuals in the population. The learning rate $alpha$ is enlarged if the move by current individual is unique.

$$\begin{aligned} \Delta\mathbf{w} &= \alpha(1 - c_t)(v(\mathbf{x}_{t+1}, \mathbf{w}_t) - v(\mathbf{x}_t, \mathbf{w}_t)) \\ &\quad \cdot \sum_{k=1}^{t} \lambda^{t-k} \nabla_{\mathbf{w}} v(\mathbf{x}_k, \mathbf{w}_k), \end{aligned}$$

1.00	-0.25	0.10	0.05	0.05	0.10	-0.25	1.00
-0.25	-0.25	0.01	0.01	0.01	0.01	-0.25	-0.25
0.10	0.01	0.05	0.02	0.02	0.05	0.01	0.10
0.05	0.01	0.02	0.01	0.01	0.02	0.01	0.05
0.05	0.01	0.02	0.01	0.01	0.02	0.01	0.05
0.10	0.01	0.05	0.02	0.02	0.05	0.01	0.10
-0.25	-0.25	0.01	0.01	0.01	0.01	-0.25	-0.25
1.00	-0.25	0.10	0.05	0.05	0.10	-0.25	1.00

Fig. 2. The weight vector of the heuristic player: each number indicate the value of corresponding cell

where c_t means the rate of the same moves among population. This method would have good effect for over-learning if there is a few alternatives and/or if the move is trivial. Because, in such case, the term c_t is closed to 1 so that $\Delta\mathbf{w}$ becomes to 0.

V. Experiments

A. Overview

We can regard the weight vector in equation (1) as an individual in Evolutionary Algorithms. That is, this problem based upon equation (1) is a sort of function optimization problems with 64-dimension real vectors.

PSO is used as an Evolutionary Algorithm in this paper. We employed Clerc's constriction factor method as PSO algorithms ([6], [7]). The procedure of this PSO is described as follows:

1) Initialize individuals and their velocities.
2) Each individual is evaluated.
3) Set $pbest$ for each individual and $gbest$.
4) Change the velocity and individual according to following equations:

$$\begin{aligned} v &\leftarrow K * [v + c_1 * rand() * (pbest - x) + \\ &\qquad c_2 * rand() * (gbest - x)] \\ x &\leftarrow x + v \end{aligned}$$

5) Go back to step 2. until a criterion is met.

The learning parameters c_1 and c_2 are set to be 2.05.

$$K = \frac{2}{|2 - \phi - \sqrt{\phi^2 - 4\phi}|} = 0.729, \quad (6)$$

where $\phi = c_1 + c_2$.

In the fitness evaluation in the Step 2., the TDL is also carried out. The fitness function in this section is based on the winning ratio against the heuristic player in Fig. 2 and the best individual in the paper [4]. That is, for each individuals, 50 games with the heuristic player, and 50 games with the best player in the paper [4] are examined. Total points for 100 matches are used as fitness value of corresponding individual.

Table I describes the wining ratio against a random player, the heuristic player and the best player in [4], after evolution. Note that the last two players are used as the fitness function. The winning ratio against the random player is examined for confirming the generalization capability of the acquired agents. "PSO" in this table means that the weight vector in equation

TABLE I
EXPERIMENTAL RESULTS OF PSO WITH FITNESS BASED ON WINNING RATIO FOR THE BEST PLAYER IN [4] AND THE HEURISTIC PLAYER

	Random	Heuristic	best in [4]
PSO	0.56	0.22	0.15
PSO+TD : $\alpha = 0.001$	0.82	0.53	0.38
PSO+TD : $\alpha = 0.01$	0.81	0.44	0.35
PSO+TD : $\alpha = 0.1$	0.70	0.20	0.15
PSO+TD+en: $\alpha = 0.001$	0.75	0.61	0.57
PSO+TD+en: $\alpha = 0.01$	0.76	0.54	0.47
PSO+TD+en: $\alpha = 0.1$	0.80	0.40	0.29

(1) is evolved by the conventional algorithm described in the previous subsection. "PSO + TD" is a Memetic Algorithm with the conventional TDL. "PSO + TD + en" stands for the proposed method. For PSO + TD and PSO + TD +en, three values of α are examined. As described in this table, the resultant fitness of the proposed method is larger than the one of the Memetic Algorithm, i.e., PSO + TD. However, the result for the random player of PSO + TD is better than the proposed method. It may imply that the proposed method over-fitted to the two players. The result of conventional PSO is worst among these algorithms so that the effectiveness of the hybridization is confirmed.

VI. APPROACH BY EMO

A. Overview

Fitness function in the last previous subsection is sum of the winning ratio of two sorts of opponents. In this section, each of the winning ratio of these opponents is regarded as a separate objective function. Hence, we apply Evolutionary Multi-Objective Optimization (EMO) to these objective functions. By using EMO, we can expect evolutionary search with diversity. In order to enhance richer diversity, we devise two kinds of modified TD learning method, which are described in the next subsection.

Mostaghim's MOPSOsigma is used as an EMO algorithm [8]. This is one of natural extension of PSO to solve Multi-Objective Optimization problems. This MOPSO is an extension of PSO for Multi-Objective Optimization such that it employs the notion of Archive and introduces σ value for associating the nearest global best in the archive for each individual: The σ value is defined as a ratio of objective function values: $O_1(x)/O_2(x)$. For each particle, the global best is selected from the Pareto particles, which have the closet σ value for the one of the corresponding particle. Except for deciding the global bests, the calculation procedure of the MOPSO is the same as the PSO.

B. Experimental Results

Below three algorithms are examined. All the algorithms are executed 200 generations with 100 individuals:

- **MOPSO** MOPSO sigma
- **MOPSO+TD** MOPSO with TD learning
- **MOPSO+TD+en** MOPSO+TD with the second modification in the previous subsection

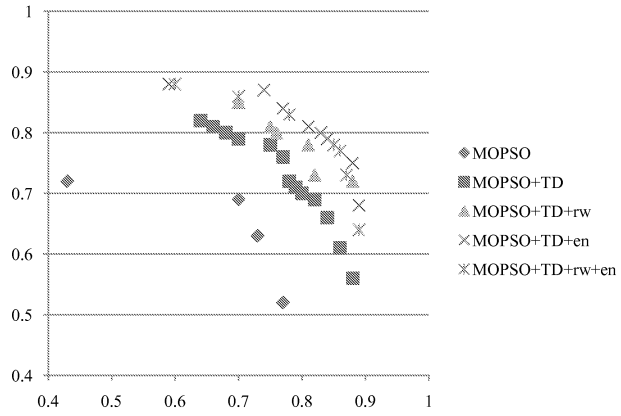

Fig. 3. Reconstructed Pareto set for examined MOPSO ($\alpha = 0.001$)

Table II summarizes hypervolume value, the size of Pareto set, and the averaged winning ratio against the random player and the heuristic player by the individuals in the Pareto set. These are result at the final generation and are averaged over 20 runs. As described in this table, bf MOPSO can improve the results in the previous section. Especially, **MOPSO+TD+en** can outperform both of the random player and the heuristic player.

Fig. 3 reconstruct the Pareto set from 20 runs. X and Y axes denote the winning ratio against the heuristic player and the best player in [4], respectively. We can find that **MOPSO+TD+en** outperforms other algorithms.

VII. CONCLUSION

This paper introduced Memetic Algorithms with TD learning method for 1-ply Othello games. This paper also proposed two modified TD learning algorithms for MOPSOsigma. The proposed method show the better performance.

Future works are summarized as follows: We can replace the opponent in the objective function, i.e., known best players, from evolution results sustainably. We can refer the recent advances in Co-evolutionary Algorithm studies.

ACKNOWLEDGMENT

This work was partially supported by the Grant-in-Aid for Scientific Research (B) and the Grant-in-Aid for Young Scientists (B) of MEXT, Japan (21360191, 21700254, 23700267).

REFERENCES

[1] Tesauro, G.: TD-Gammon, A Self-Teaching Backgammon Program, Achieves Master-Level Play *Neural Computation*, Vol.6, No.2 (1994)
[2] Beal, D.F., Smith, M.C.: Temporal difference learning applied to game playing and the results of application to shogi *Theoretical Computer Science*, Vol.252, pp. 105–119 (2001)
[3] Ghory, I.: Reinforcement Learning in Board Games, *Technical Report of Department of Computer Science, University of Bristol*, CSTR–04–004 (2004)
[4] Lucas, S.M. Runarsson, T.P.: Temporal Difference Learning Versus Co-Evolution for Acquiring Othello Position Evaluation, *Proc. 2006 IEEE Symposium on Computational Intelligence and Games*, pp.52–59 (2006)
[5] Katayama, K., and Ishibuchi, H.!' Memetic Algorithm (in Japanese), Journal of the Society of Instrument and Control Engineers, Vol. 47, No. 6, pp. 478–492 (2008)

TABLE II
THE RESULT BY MOPSO

	hypervolume	Size of Archive	Random	Heuristic
MOPSO	0.11	7.4	0.61	0.50
MOPSO+TD: $\alpha = 0.001$	0.23	7.1	0.81	0.66
MOPSO+TD: $\alpha = 0.01$	0.34	6.7	0.76	0.70
MOPSO+TD: $\alpha = 0.1$	0.15	6.7	0.82	0.55
MOPSO+TD+en: $\alpha = 0.001$	0.29	6.9	0.82	0.68
MOPSO+TD+en: $\alpha = 0.01$	0.39	6.6	0.76	0.75
MOPSO+TD+en: $\alpha = 0.1$	0.35	7.0	0.80	0.79

[6] Eberhart, R.C, and Shi, Y.: Particle swarm optimization: developments, applications and resources, *Proceedings of the 2001 IEEE Congress on Evolutionary Computation*, Vol.1, pp.81–86 (2001)
[7] Clerc, M.: The swarm and the queen: towards a deterministic and adaptive particle swarm optimization, *Proceedings of the 1999 IEEE Congress on Evolutionary Computation*, Vol.3, pp.1951–1957 (1999)
[8] Mostaghim, S., and Teich, J.: Strategies for Finding Good Local Guides in Multi-Objective Particle Swarm Optimization (MOPSO), Proceedings of the 2003 IEEE Swarm Intelligence Symposium, pp.26–33 (2003)
[9] Ishibuchi, H., and Narukawa, K.: Some Issues on the Implementation of Local Search in Evolutionary Multiobjective Optimization, *Proc. of 2004 Genetic and Evolutionary Computation Conference*, part I, pp. 1246–1258 (2004)