Multi-Criteria Comparison of Coevolution and Temporal Difference Learning on Othello

Wojciech Jaśkowski, Marcin Szubert, and Paweł Liskowski

Institute of Computing Science, Poznan University of Technology, Poznań, Poland
{wjaskowski,pliskowski,mszubert}@cs.put.poznan.pl

[Preprint. Published in EvoApplications, EvoGames, Granada 2014]

Abstract. We compare Temporal Difference Learning (TDL) with Coevolutionary Learning (CEL) on Othello. Apart from using three popular single-criteria performance measures: i) generalization performance or expected utility, ii) average results against a hand-crafted heuristic and iii) result in a head to head match, we compare the algorithms using performance profiles. This multi-criteria performance measure characterizes player's performance in the context of opponents of various strength. The multi-criteria analysis reveals that although the generalization performance of players produced by the two algorithms is similar, TDL is much better at playing against the strong opponents, while CEL copes better against the weak ones. We also find out that TDL produces less diverse strategies than CEL. Our results confirm the usefulness of performance profiles as a tool for comparison of learning algorithms for games.
Keywords: reinforcement learning, coevolutionary algorithm, Reversi, Othello, board evaluation function, weighted piece counter, interactive domain

1 Introduction

The board game of Othello constitutes a non-trivial interactive domain, which has become a popular testbed for evaluating and comparing different computational intelligence algorithms [1,2,3]. The most popular algorithms used for learning Othello-playing strategies include *competitive coevolutionary learning* (CEL) [4] and *temporal difference learning* (TDL) [5]. TDL is a well-recognized example of reinforcement learning [6], in which the playing agent aims to find a value function for predicting chances of winning a game from a particular state. CEL, on the other hand, searches the space of strategies directly by maintaining a set of candidate solutions that compete against each other and are randomly tweaked by means of evolutionary operators such as mutation or crossover. The essential difference between TDL and CEL is that TDL guides the learning using the whole course of the game while CEL uses only the final game outcome.

Since both CEL and TDL can be applied to the same problem of learning game-playing strategies, it is not surprising that they have been the subject of comparative investigations. In one of the first such comparisons for Othello, Lucas and Runarsson [1] found out that when learning strategies represented with simple weighted piece counters *"TDL learns much faster than CEL, but properly tuned CEL can learn better playing strategies"*. However, Szubert et al. [7] showed that the difference between TDL and CEL largely depends on the performance measure used: CEL and TDL perform similarly when playing against a random player, but TDL is superior to CEL when compared against a heuristic hand-crafted player. These results were confirmed also for a non-linear, complex n-tuples strategy representation [8], for which CEL is substantially worse than TDL while a hybrid of CEL and TDL works even better [3]. Interestingly, in the context of Backgammon game, Darwen showed that CEL can beat TDL [9].

Therefore, the general conclusions of the results of research comparing CEL and TDL are not clear and they depend, among others, on selected objective performance measure whether this is a fixed hand-crafted opponent, a random player or a round robin tournament [3]. This insight led to devising performance profiles [10], a multi-criteria method for comparison players and algorithms. Performance profiles allow to analyze and present graphically the performance of different players when facing opponents of various strength. Thus performance profile conveys much more information about player's characteristics than the commonly used single-criteria performance measures which are prone to compensation due to aggregation of results: the awards received in interactions with a certain group of tests can cancel out the penalties incurred in interactions with another group of tests.

Our main contribution in this paper is the comparison of CEL and TDL using the multi-criteria performance measure based on performance profiles. In this way we are able to precisely pin-point performance differences between the strategies learned by the two analyzed algorithms and explain the differences in results obtained on single-criteria performance measures. We notice that it is easy to misjudge relative algorithms strength basing only on a single-criteria performance measure.

2 Othello

2.1 Game rules description

Othello is a perfect information, zero-sum, two-player strategy game played on a 8×8 board. There are 64 identical disks which are white on one side and black on the other. The game begins with each player having two disks placed diagonally in the center of the board. Players alternate placing disks on the board, with the black player moving first. A move is legal if the newly placed piece is adjacent to an opponent's piece and causes one or more of the opponent's pieces to become enclosed from both sides of a horizontal, vertical or diagonal line. The enclosed disks are then flipped. The game ends when neither player has a legal move. A player who has more pieces on the board wins. If both players have the same number of pieces, the game ends in a draw.

2.2 Weighted Piece Counter (WPC) strategy representation

We represent strategies using WPC, a simple, linear board state evaluation function, which indicates how desirable a given board state is. WPC assigns weight w_i to board location i and uses scalar product to calculate the value f of a board state $\mathbf{b}$: $f(\mathbf{b}) = \sum_{i=1}^{8\times 8} w_i b_i$, where b_i is 0, $+1$ or -1 for empty location, black piece or white piece, respectively. The players interpret $f(\mathbf{b})$ in a complementary manner: the black player prefers moves leading towards states with a higher value, whereas lower values are favored by the white player.

All algorithms considered in this paper employ WPC as a state evaluator in 1-ply setup: given the current board state, the player generates all legal moves and applies f to the resulting states. The state gauged as the most valuable determines the move to be made. Ties are resolved at random.

3 Coevolutionary Learning

Coevolutionary algorithms [11] are variants of evolutionary computation in which an individual's fitness depends on other individuals. Similarly to the evolutionary one, coevolutionary algorithm use mechanisms such as selection and variation that mimic the natural evolution. The driving force of coevolutionary algorithms is the continuous Darwinian *arms race* taking place between one or two competing populations [12]. The difference between coevolutionary and evolutionary methods lies in the evaluation phase, when the fitness of individuals is assessed. Evolutionary algorithms that solve optimization problems have access to the objective function of a problem, thus individuals' fitness is directly computed. In coevolutionary algorithms, individuals' fitness is typically only estimated by aggregating results of multiple interactions between individuals from the maintained populations.

In this paper we use a one-population variant of competitive coevolution and apply it to learn Othello board evaluation function. The algorithm uses mutation as the only variation operator while fitness of an individual is defined as a sum of two values: i) the average result of games played against other individuals from the population (a round robin tournament) and ii) the average result of games played against a sample of random WPC players. This particular coevolutionary algorithm has been recently found superior to a typical one-population coevolution [10].

4 Temporal Difference Learning

Temporal Difference Learning (TDL) is a reinforcement learning (RL) method which has become a popular approach for elaborating game-playing strategies [13,1,2]. The use of RL techniques for such applications stems from modeling a game as a sequential decision problem, where the task of the learner is to maximize the expected reward in the long run (game outcome).

In this paper we use $TD(\lambda)$ algorithm [5], which solves prediction learning problem that consists in estimating the future behavior of an unknown system from the past experience. Learning occurs whenever system's state changes over time and is based on the error between the temporally successive predictions. Its goal is to make the preceding prediction match more closely the current prediction (taking into account distinct system states observed in the corresponding time steps). Technically, at a certain time step t, prediction P_t can be considered as a function of two arguments: current system state and the vector of weights $\mathbf{w}$. The $TD(\lambda)$ algorithm is expressed by the following weight update rule:

$$\varDelta \mathbf{w}_t = \alpha(P_{t+1} - P_t) \sum_{k=1}^{t} \lambda^{t-k} \nabla_w P_k,$$

where α is the learning rate, and λ is the decay parameter, which influences the magnitude of changes applied to all the preceding predictions within a single learning episode. When applied to the problem of learning game-playing strategy represented as WPC, P_t estimates the chances of winning from the game state $\mathbf{b}_t$, by mapping the outcome of the WPC function f to a closed interval $[-1, 1]$ using hyperbolic tangent, so that $P_t = tanh(f(\mathbf{b}_t))$.

The process of learning consists of applying the above formulas to the WPC vector after each move of a self-play game. During game play, moves are selected on the basis of the most recent evaluation function. Othello is a deterministic game, thus the course of the game between two deterministic players is always the same. This feature reduces the number of possible states a learner can explore, which makes learning ineffective. To remedy this situation, at each turn, a random move is forced with a certain probability ϵ. As a result, players are confronted against a wider spectrum of possible behaviors.

5 Experimental Setup and Parameters Tuning

In order to fairly compare the algorithms, we set them up so that their total computational effort is the same and it is equal to $2,000,000$ training games. To evaluate a given algorithm we measure the performance of its *best-of-run* player. In CEL this is the individual from the last generation with the highest fitness, while in TDL this is simply the only learning player. Since both considered algorithms are stochastic, we compare their average results over 100 runs.

Instead of selecting arbitrary values of parameters for CEL and TDL, we perform a series of preliminary experiments to optimize them. As the optimization goal we use the performance measure of *generalization performance* [14] (also known as expected utility [10]). Generalization performance of a player is defined as its expected score over all possible opponents. To approximate this measure we compute the average game result of the player against $50,000$ random players. A random player is a random weighted piece counter player that weights are drawn uniformly at random from a fixed interval of $[-1, 1]$. In each game players are rewarded 1 point for a win and 0.5 point for a draw.

5.1 Temporal Difference Learning

The $TD(\lambda)$ algorithm described in Section 4 has two parameters: learning rate α and decay $\lambda \in [0, 1]$. Additionally, since the algorithm learns on the basis of self-play games, there is another important parameter — random move probability ϵ. The results of different combinations of α and ϵ for $\lambda = 0$ are presented in Table 1.

Table 1: The results obtained by TDL players for different α and ϵ. Generalization performance values are presented in percent points.

$\alpha =$	.01	.02	.03	.04	.05	.06	.07	.08	.09	.1
$\epsilon = .0$	81.4	82.3	84.2	84.0	84.3	82.1	79.9	75.4	71.5	72.0
$\epsilon = .05$	84.0	81.1	82.7	86.2	**87.9**	87.7	87.4	87.4	87.0	86.8
$\epsilon = .1$	84.4	81.5	82.4	84.4	87.3	87.7	87.2	86.4	85.4	85.4
$\epsilon = .15$	84.9	82.7	82.2	83.3	86.1	87.8	86.7	85.5	84.6	84.5

On the basis of these results we chose $\alpha = 0.05$ and $\epsilon = 0.05$ as the best parameters. In the second stage, we checked different values of decay λ. However, changing λ did not provide statistically better results than those obtained with $\lambda = 0$. This observation confirms previous results [2].

5.2 Coevolution

CEL has several quantitative and qualitative parameters. The former include population size, random sample size, mutation probability and mutation range, while the latter mutation and selection operators. The weights of the individuals in the initial population were drawn at random from the $[-0.1, 0.1]$ interval.

For each of the parameters we considered several possible values. We chose to test six population sizes and six random sample sizes: 4, 10, 20, 50, 100, 200. We selected four selection strategies:

- *tournament selection,* with tournament size 5,
- *stochastic universal sampling* [15], a variant of a roulette-wheel selection that guarantees that the frequency of selection for each individual is consistent with its expected frequency of selection,
- $(\mu + \lambda)$ evolutionary strategy*,* where $\mu = \frac{1}{2}popsize$ and $\lambda = \frac{1}{2}popsize$, and
- (μ, λ) evolutionary strategy*, where* $\mu = \frac{1}{2}popsize$ and $\lambda = popsize$.

Additionally, we considered *Gaussian mutation* and *uniform mutation.* These mutation operators perturb each weight of the WPC with probability p by adding to it a random value drawn uniformly from the interval $[-r, r]$ in case of uniform mutation, or from $\mathcal{N}(0, \sigma)$ for Gaussian mutation. For parameters p, r, and σ we considered values of .05, .1, .2, .3, .4, .5, .6, .7, .8, .9 and 1.0.

Results

We carried out the tuning of CEL parameters in two stages. In the first stage, we fixed the mutation operator to Gaussian mutation with $p = 0.1$ and $\sigma = 0.25$ and focused on finding the best selection strategy and best values of population size and random sample size. For this purpose, we evaluated all $6 \times 6 \times 4 = 144$ combinations of population size, random sample size and selection operators.

The results of optimization are shown in Fig. 1. On this basis, we decided to use (μ, λ)-ES, population size of 20 and random sample size of 200.

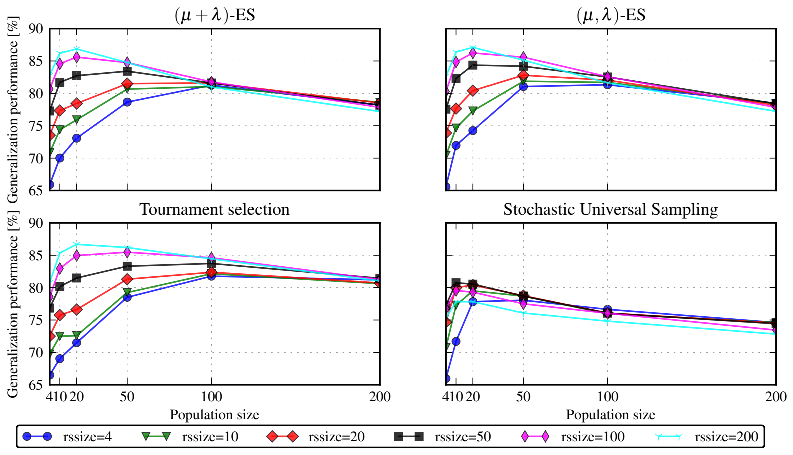


Fig. 1: The impact of population size, random sample size and selection operator on the generalization performance of CEL. The values are presented in percent points.

In the second stage, we evaluated Gaussian mutation and uniform mutation for every combination of mutation probability, mutation operator and their parameters, $7 \times 7 \times 2 = 98$ experiments in total. Surprisingly, we have found no evidence of any statistical differences between either the mutation types or their parameters with the sole exception of the combination of Gaussian mutation with $\sigma = .05$ and $p \in \{.05, .1\}$ that was statistically (t-test, $\alpha = 0.05$) inferior to other combinations. As a result, for the rest of the experiments we use Gaussian mutation with $p = 1.0$ and $\sigma = 1.0$.

6 Comparison of Coevolution and Temporal Difference Learning

6.1 Single-Criteria Comparison

We start the comparison between CEL and TDL by applying three commonly used [16,1,17] single-criteria performance measures:

– *generalization performance* — the average performance against randomly generated WPC players (see Section 5).
– *heuristic performance* — the average performance against a "standard" hand-crafted WPC heuristic player [18,3]. Since this player is deterministic, following earlier work [1], we force players to make random moves with probability $\epsilon = 0.1$, and thus we slightly alter the game definition.
– *head to head* — indicates how well a set of players copes in games against players from another set.

Table 2: Comparison of CEL and TDL using three performance measures: i) generalization performance, ii) heuristic performance, and iii) the result of head to head match. The results are shown in percent points, where 100% means getting all possible points (winning all games). Values of generalization performance and heuristic performance are accompanied by 95% confidence intervals. Note that the results of head to head match sum up to 100%.

	Performance measure [%]		
Algorithm	generalization	heuristic	head to head
Coevolutionary Learning (CEL)	86.97±0.21	32.31±0.62	21.2
Temporal Difference Learning (TDL)	87.26±0.32	46.14±0.96	78.8

Table 2 presents the results for CEL and TDL using the above-described single-criteria performance measures. To compute generalization performance and heuristic performance we played $50,000$ games for each best-of-run player. The results were averaged over 100 best-of-run players for each algorithm.

The comparison using the three single-criteria performance measures is equivocal. The performance measure of generalization performance shows no statistical difference between CEL and TDL (t-test, $\alpha = 0.01$). However, TDL is clearly superior to CEL when playing against a heuristic player and in a head to head match. Can we then claim with a confidence that TDL is "better" than CEL?

6.2 Multi-Criteria Comparison with Performance Profiles

Performance Profiles Single-criteria methods of performance evaluation do not draw a clear picture of the relative performance of analyzed methods. To better understand the characteristics of compared methods we use *performance profiles* [10]. They compare performance of players using sets of opponents of various strength, treating the result of match against opponents of each such set as a separate performance criteria.

To prepare a performance profile, we first generate a number of opponents and group them into bins according to their strength. To this aim, we randomly generated about 1,000,000 players (opponents) by sampling WPC weights uniformly and independently from the $[-1, 1]$ interval. Next, the generalization performance of each opponent was estimated by taking average from the results of games 2,000 against random WPC strategies. The range of possible performance values, i.e., $[0\%, 100\%]$, is then divided into 100 bins of equal 1%-performance width, and each opponent is assigned to one of these bins based on its generalization performance.

However, finding extremely strong or weak strategies in this way is very difficult, if not impossible. To overcome this, the strongest (performance $> 81\%$) and the weakest (performance $< 13\%$) opponents were obtained using multiple independent runs of evolutionary learning with random sampling [10]. In this way, we were able to fill 93 bins ($4\% - 96\%$), each one of containing $1,000$ opponents. Note that building the opponents database is computationally expensive. However, once created, it can be reused[1].

The set of opponents partitioned into bins forms the basis for building performance profiles. The player to be assessed plays games against all the opponents from each bin, and the average game outcome is plotted against the bins. Performance profile is a multi-criteria performance evaluation method since the performance of a given player is measured separately on every bin, each being a different criterion.

Results We apply this multi-criteria method to inspect the best-of-run individuals of the two algorithms considered in this paper. The resulting performance profiles are presented in Fig. 2. Since we have 100 runs per algorithm, we average the profiles over 100 best-of-run players. A point of coordinates (x, y) in a plot means that the best-of-run individuals have on average performance y when playing against opponents of performance x. For example, the performance of CEL is nearly 95% for opponents with performance of 20%. The whiskers in the plots mark 95% confidence intervals.

The decreasing trend in each data series confirms the intuition that it is harder to win against the stronger opponents than against the weaker ones.

The most important observation from the plots is that TDL players are significantly better when facing the strong opponents. Moreover, the stronger the opponents the wider the performance gap between CEL and TDL players. On the other hand, CEL players are better than TDL players against the weakest opponents. For example, CEL players win nearly 98% games against the weakest opponents in our database of performance of 4%, while TDL players win only 94% games against them.

6.3 Discussion

In Section 6.1 we showed that there is no statistical difference in generalization performance between CEL and TDL, but TDL is better than CEL in a

[1] The data and Java code for creating performance profiles for Othello are available at `http://www.cs.put.poznan.pl/wjaskowski/projects/performance-profiles`.

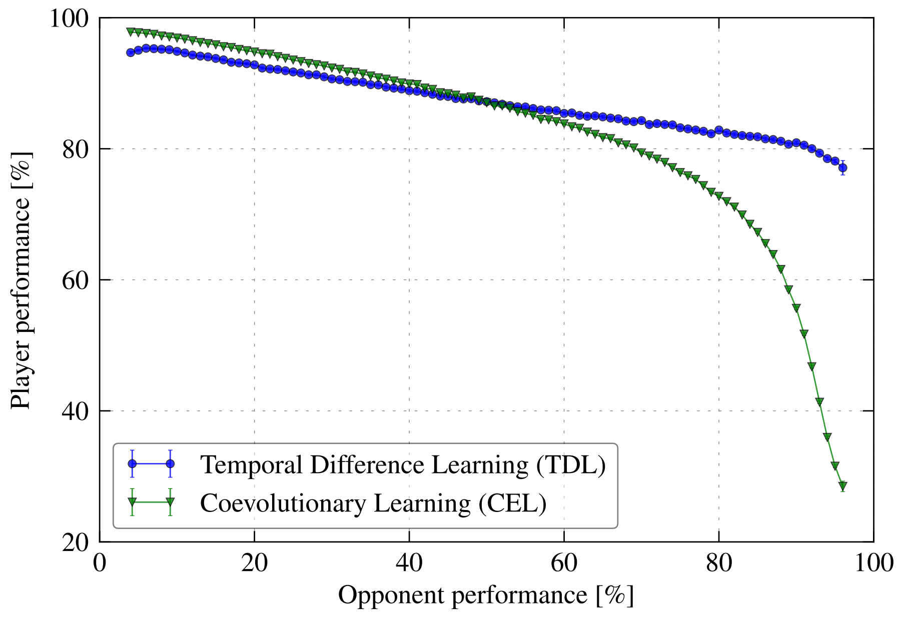


Fig. 2: Performance profiles of coevolutionary learning (CEL) and temporal difference learning (TDL). Each point (x, y) means player performance y against opponents of performance x. Confidence intervals for each point are less than 1%. Right side of the plot indicates that TDL copes much better than CEL against the stronger opponents.

game against a hand-crafted heuristic player and in a head to head tournament. Performance profiles could explain this discrepancy.

First, we can see that in Fig. 2 CEL and TDL curves cross at about 50%, both obtaining performance about 87% at this point. This value precisely matches the generalization performance results obtained by CEL and TDL (cf. Table 2), because the 50%-bin contains average players of performance equal to a random WPC player.

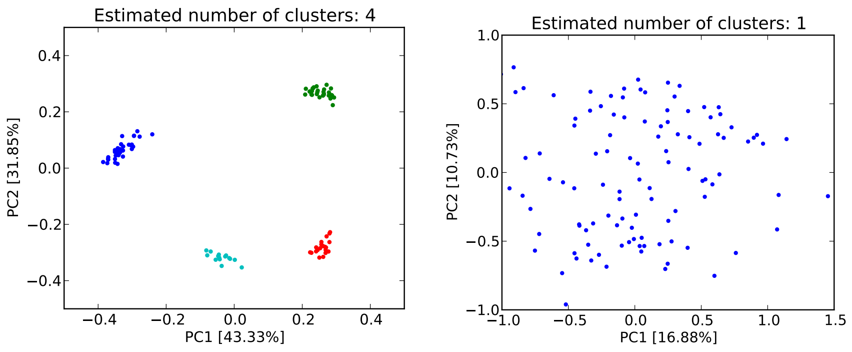
Second, we should realize that the heuristic performance and the result in head to head match determine how a player copes against strong opponents, rather than average ones. Performance profile analysis confirms that TDL fares much better than CEL against strong opponents (cf. Fig 2).

Third, what the three single-criteria performance measure miss is that CEL is better than TDL for weaker opponents.

The three single-criteria measures are like three points sampled from a signal; we can hypothesize about its shape, but they are not enough to fully understand it. Performance profiles allow us to not only understand the single-criteria results but also to see the trade-offs between TDL and CEL.

7 Strategies Comparison

The average performance against particular type of opponents allows us to draw conclusions about the superiority of some approaches over others, but it says nothing about the weights of elaborated WPC strategies. For this reason, we investigate the distribution of final strategies learned by TDL and CEL.



(a) Temporal Difference Learning (TDL) (b) Coevolutionary Learning (CEL)

Fig. 3: Learned strategies represented as points in the PCA-reduced space. Different colors indicate clusters identified by the mean-shift algorithm.

To analyze the WPC strategies, we treat them as points in a 64-dimensional space. First, we linearly scale all their weights to $[0, 1]$ interval. Finally, we clustered them using the mean-shift algorithm [19]. Figure 3 illustrates the results of clustering the strategies produced by TDL and CEL in a two dimensional space, which was obtained by applying PCA (Principle Component Analysis). For TDL, we can clearly see four clusters, while CEL strategies are randomly spread in the space. It appears that the players produced by CEL are much more diversified than those produced by CEL.

Selected WPCs are presented graphically in weight-proportional gray-scale in Figures 4 and 5 for TDL and CEL, respectively. In the figures darker squares denote larger weights, which correspond to more desirable board locations.

Interestingly, TDL strategies exhibit some symmetries. In particular, the corners are the most desirable, while their immediate neighbors have very low weights. The only difference between these four strategies is the weight in one of the board corners — it is significantly lower than in the other three corners. In contrast, CEL strategies are less symmetrical and, apart from the typically black corners, they do not exhibit any regularities nor symmetries.

8 Conclusions

This study presents an evidence that while temporal difference learning (TDL) and coevolutionary learning (CEL) obtain similar results against average opponents, TDL copes much better against stronger ones. This was observed using single-criteria performance measures, but the full picture was only revealed using multi-criteria performance profiles. The characteristics of the strategies learned by TDL and CEL differ significantly and this is reflected in strategy weights.

Despite their usefulness, performance profiles have some limitations. Their computation requires numerous opponents of specific performances to be prepared, what is computationally expensive. Therefore, profiling game-playing strategies seems to be only possible for simple and linear representations (like WPC) and for games which can be quickly played (1-ply only). Nevertheless, such settings are perfectly acceptable when the emphasis of the research is put not on the absolute performance of the players, but on the learning algorithms.

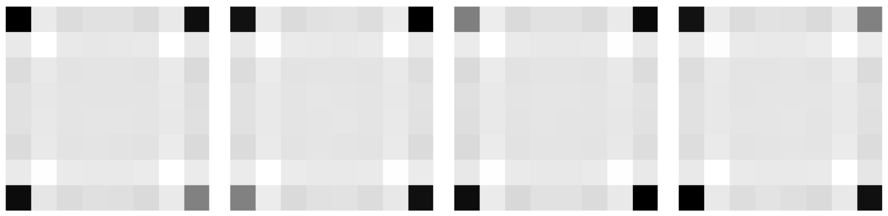


Fig. 4: WPC strategies produced by TDL corresponding to centers of the clusters identified by the mean-shift algorithm (cf. Fig. 3) illustrated as Othello boards with locations shaded accordingly to corresponding weights.

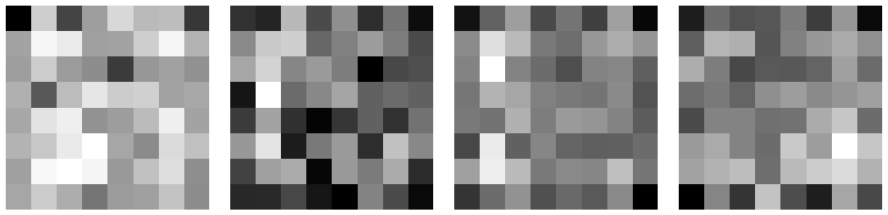


Fig. 5: Selected WPC strategies found by CEL and illustrated as Othello boards with locations shaded accordingly to corresponding weights. The first two strategies from the left are the most distant ones in the 64-dimensional space, the third one is the centroid of the set of all CEL strategies while the right-most one is the best strategy with respect to the generalization performance.

Acknowledgment This work has been supported by the Polish National Science Centre grant no. DEC-2013/09/D/ST6/03932. M. Szubert has been supported by Polish Ministry of Science and Education, grant no. 2012/05/N/ST6/03152.

References

1. Lucas, S.M., Runarsson, T.P.: Temporal difference learning versus co-evolution for acquiring othello position evaluation. In: IEEE Symposium on Computational Intelligence and Games, IEEE (2006) 52–59
2. van den Dries, S., Wiering, M.A.: Neural-Fitted TD-Leaf Learning for Playing Othello With Structured Neural Networks. IEEE Transactions on Neural Networks and Learning Systems **23**(11) (November 2012) 1701–1713
3. Szubert, M., Jaśkowski, W., Krawiec, K.: On scalability, generalization, and hybridization of coevolutionary learning: a case study for othello. IEEE Transactions on Computational Intelligence and AI in Games **5**(3) (2013) 214–226
4. Axelrod, R.: The evolution of strategies in the iterated prisoner's dilemma. In Davis, L., ed.: Genetic Algorithms in Simulated Annealing, London (1987) 32–41
5. Sutton, R.S.: Learning to predict by the methods of temporal differences. Machine learning **3**(1) (1988) 9–44
6. Sutton, R., Barto, A.: Reinforcement learning. Volume 9. MIT Press (1998)
7. Szubert, M., Jaśkowski, W., Krawiec, K.: Learning board evaluation function for othello by hybridizing coevolution with temporal difference learning. Control and Cybernetics **40**(3) (2011) 805–831
8. Lucas, S.M.: Learning to play Othello with N-tuple systems. Australian Journal of Intelligent Information Processing Systems, Special Issue on Game Technology **9**(4) (2007) 01–20
9. Darwen, P.J.: Why co-evolution beats temporal difference learning at backgammon for a linear architecture, but not a non-linear architecture. In: Proceedings of the 2001 Congress on Evolutionary Computation. Volume 2., IEEE (2001) 1003–1010
10. Jaśkowski, W., Liskowski, P., Szubert, M., Krawiec, K.: Improving coevolution by random sampling. In Blum, C., ed.: GECCO'13: Proceedings of the 15th annual conference on Genetic and Evolutionary Computation, Amsterdam, The Netherlands, ACM (July 2013) 1141–1148
11. Popovici, E., Bucci, A., Wiegand, R.P., de Jong, E.D.: Coevolutionary Principles. In: Handbook of Natural Computing. Springer-Verlag (2011)
12. Nolfi, S., Floreano, D.: Coevolving Predator and Prey Robots: Do "Arms Races" Arise in Artificial Evolution? Artificial Life **4**(4) (1998) 311–335
13. Tesauro, G.: Temporal difference learning and td-gammon. Communications of the ACM **38**(3) (1995) 58–68
14. Chong, S.Y., Tino, P., Yao, X.: Relationship between generalization and diversity in coevolutionary learning. Computational Intelligence and AI in Games, IEEE Transactions on **1**(3) (2009) 214 –232
15. Baker, J.E.: Reducing bias and inefficiency in the selection algorithms. (1985)
16. Chong, S.Y., Tino, P., Ku, D.C., Xin, Y.: Improving Generalization Performance in Co-Evolutionary Learning. IEEE Transactions on Evolutionary Computation **16**(1) (2012) 70–85
17. Szubert, M., Jaśkowski, W., Krawiec, K.: Coevolutionary temporal difference learning for othello. In: IEEE Symposium on Computational Intelligence and Games, Milano, Italy (2009) 104–111
18. Samothrakis, S., Lucas, S., Runarsson, T., Robles, D.: Coevolving Game-Playing Agents: Measuring Performance and Intransitivities. IEEE Transactions on Evolutionary Computation (99) (2012) 1–15
19. Comaniciu, D., Meer, P., Member, S.: Mean shift: A robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence **24** (2002) 603–619