

# A Pendulum Effect in Co-evolutionary Learning in Games

Dimitris Kalles<sup>1</sup> and Panagiotis Kanellopoulos<sup>2</sup>

<sup>1</sup> Hellenic Open University, Patras, Greece, [kalles@eap.gr](mailto:kalles@eap.gr)

<sup>2</sup> Computer Technology Institute, Patras, Greece, [kanellop@ceid.upatras.gr](mailto:kanellop@ceid.upatras.gr)

**Abstract.** When learning how to play a strategy board game, one can measure the relative effectiveness of the learned policies by assessing how often a player wins and how easily these wins are scored. Experimental evidence also shows that when one of the competing players is trained by a sophisticated tutor, performance benefits also flow to the opponent. We present comprehensive experimental evidence that the level of tutor sophistication is best demonstrated by the improvement of the tutored player's opponent. This performance change is termed the pendulum effect, whose rationale we also trace to game theory.

**Keywords:** board games; co-evolutionary learning; reinforcement learning.

## 1 Introduction

Quite a few machine learning techniques have made significant inroads into problems that demonstrate a human-computer interaction component. Learning how to play games is a prime example of such an application; Shannon [1] and Samuel [2] provided the first stimulating examples; Deep Blue defeated Kasparov at chess in 1997 [3] and, more recently, Schaeffer's team solved checkers completely [4]. Today, advances in machine learning allow us to tackle a deeper issue: how can we instruct a computer to play a game by simply showing it how to play? In other words, we are given a computer equipped with a generic learning mechanism and we must present to it a "syllabus" of experience that will allow it to formulate playing knowledge.

Arguably one of the strongest contributions to that problem was the development of the TD( $\lambda$ ) method for temporal difference reinforcement learning [5, 6], which was successfully demonstrated in TD-Gammon for the game of backgammon [7]; therein using reinforcement learning techniques and self-playing, a performance comparable to that demonstrated by backgammon world champions was achieved.

Implementing a computer's strategy is a key point in strategy games. By the term *strategy* we broadly mean the selection of the computer's next move based on a variety of factors which can include its current situation, the opponent's situation and consequences of that move (or, possible next moves of the opponent). In our research, we use a strategy game to gain insight into how we can *evolve* game playing capabilities, as opposed to how we *program* such capabilities (maybe, using a heuristic). Although the operational goal of achieving improvement (measured in a

variety of ways) is usually achieved in several experimental settings [8, 9, 10], the actual question of which training actions help realize this improvement is central if we attempt to devise an optimized training plan. The term *optimize* reflects the need to expend judiciously the training resources, be it computer power or expert guidance.

Exploiting expert interaction in trainable systems has been recently described according to three major directions [11]. In the first one, inspired by mainstream personalization, one collects raw items of user behavior and attempts to elicit habit, structure, or intention from such data [12]. The second direction is based on how control is exerted to generate actions that minimize some measure of a knowledge gap [13]. A third dimension is whether the expert tutor is a part of the training mechanism at *all* times, maybe with continuous but simple feedback [14].

A popular combination of these directions is to use an expert for focused guidance and, subsequently, to use some mechanism of self-improvement up to a point where, again, the expert will be called into action to provide a correction of direction, if required; this can go *ad infinitum*. So, the line between what constitutes sparseness and what constitutes density of learning examples (when the computer acts as a student) is quite fine [15].

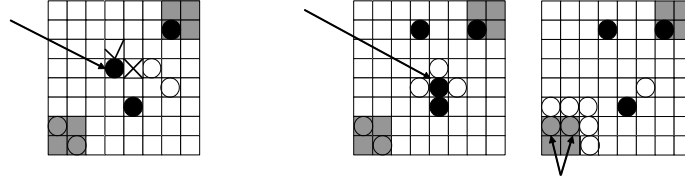
Our workbench on this field is a simple board game, *RLGame*; for legacy reasons, the name draws on reinforcement learning being used as the basic learning mechanism [16]. Recent work evidence suggested that, when one of the competing players is trained by a sophisticated tutor (for example, as implemented by *minimax*), the other player also benefits since it is forced into unexplored ground and manages to learn new tactics that allow it to swiftly improve its performance [17]. To stress the interestingness of such a clearly observable change in performance between two competing player, this was called the *pendulum effect*.

In this paper, we design and carry out a comprehensive experimentation to investigate the pendulum effect, to associate it with the rational conclusion that a tutor's impact is best assessed by its absence and that one learns better by facing a strong opponent. To arrive at these results we have designed and carried out several millions of games with diverse configurations in a distributed fashion over a grid computing infrastructure.

The rest of this paper is structured in four sections. We first review earlier work on the subject, providing a brief background on the game, to settle the base upon which we build our work. We then present the experiments we designed and carried out, as well as the data post-processing automation that allows us to massively generate the output that we then analyze for spotting instances of the pendulum effect behavior. We then discuss our findings and, finally, we conclude, also presenting the ramifications of our work for learning systems at large.

## 2 A brief background on a strategy game workbench

The game [16] is played by two players on an  $n \times n$  square board. Two  $a \times a$  square bases are on opposite board corners; the white player starts off the lower left base and the black player starts off the upper right one. Initially, each player possesses  $\beta$  pawns. The goal is to move a pawn into the opponent's base.



**Fig. 1.** Examples of game rules application.

The base is considered as a single square, therefore a pawn can move out of the base to any adjacent free square. Players take turns and pawns move one at a time, with the white player moving first. A pawn can move vertically or horizontally to an adjacent free square, provided that the maximum distance from its base is not decreased (so, backward moves are not allowed). The distance from the base is defined as the maximum of the horizontal and the vertical distance from the base. A pawn that cannot move is lost (more than one pawn may be lost in one move). A player also loses by running out of pawns.

The leftmost board in Fig. 1 demonstrates a legal (“tick”) and an illegal (“cross”) move for the pawn pointed to by the arrow, the illegal move being due to the rule that does not allow decreasing the distance from the home (black) base. The rightmost boards demonstrate the loss of pawns, with arrows showing pawn casualties. A “trapped” pawn automatically draws away from the game; so, when there is no free square next to the base, the rest of the pawns of the base are lost.

In *RLGame* the *a priori* knowledge of the system consists of rules only. To judge what the next move should be reinforcement learning is used to learn an optimal policy that will maximize the expected sum of rewards in a specific time, given the current state of the environment. The value function on the game state space is approximated with neural networks [6, 16], where each next possible move and the current board configuration are fed as input and the network outputs a score that represents a belief degree that one will win by making that move, as adopted in Neurogammon [7]. Learning employs an  $\epsilon$ -greedy policy with  $\epsilon=0.1$  (the system chooses the best-valued action with a probability of 0.9 and a random action with a probability of 0.1). All non-final states are assigned the same initial value which, after each move, is updated through TD(0.5), thus halving [5] the influence of credits and penalties for every backward step that is considered.

## 2.1 Reviewing the effects of expert involvement

Initial experiments demonstrated that, when trained with self-playing, both players would converge to having nearly equal chances to win [16] and that self-playing would achieve weaker performance compared to a computer playing against a human player, even with limited human involvement [18].

Further experimentation culminated in the design of a metric to measure the relative effectiveness of two distinct policies [19]. Assuming that one has available a player,  $X$ , with its associated white and black components,  $W_X$  and  $B_X$  (the components being the neural networks), one can compare it to  $Y$  by first pairing  $W_X$

with  $B_Y$  for a  $CC^{1000}$  session ( $CC^{1000}$  stands for 1000 computer-vs-computer games), then pairing  $W_Y$  with  $B_X$  for a further  $CC^{1000}$  session, and subsequently calculating the number of games won and the average number of moves per game won (see Table 1 for an example showing how player  $X$ 's components collectively win  $Y$ 's).

**Table 1.** Comparative evaluation of learned policies  $X$  and  $Y$ .

	Games Won		Average # of Moves	
	White	Black	White	Black
$W_X$ vs. $B_Y$	715 $_X$	285 $_Y$	291	397
$W_Y$ vs. $B_X$	530 $_Y$	470 $_X$	445	314

While the above metric allows one to measure how a policy  $X$  fares against a policy  $Y$ , describing a result as clear or not is a more difficult task. For example, a  $(W_X + B_X):(W_Y + B_Y)$  result of 200:1800 unequivocally demonstrates a sizeable difference while a result of 900:1100 significantly less so.

## 2.2 Reviewing the effects of minimax-as-expert involvement

The above metric has helped quantify the quality of learning. It has been also useful subsequently in testing a minimax algorithm for the moves of the white player [17].

Since minimax can be used with a different look-ahead set-up, initial experiments consisted of short sessions for various look-ahead values (note that a look-ahead of  $2n+1$ , denoted by  $MC_{2n+1}$  indicates  $n+1$  moves for the white player interleaved with  $n$  moves for the black player; MC stands for minimax-for-white-vs-computer-for-black). These experiments took place on a 8x8 board, with 2x2 base size and 10 pawns for each player, and for look-ahead values of 1, 3, 5, 7, and 9. Moreover, each look-ahead experiment consisted of 100 MC games (totalling 500 games).

Note that the neural networks for the white and the black player are both continuously updated. Adopting minimax for white means building a (white's) model based on its tutor's advice. That model is also used whenever minimax examines a leaf state (in the minimax tree) that is not also a final game; that state's value is used as a surrogate for the minimax value which, of course, cannot be computed (because it is not within the look-ahead range). Essentially, this means that minimax is implemented in the context of an ( $\epsilon=0$ )-greedy policy.

One examines MC experiments with the expectation that the white player's performance will be improved the longer experimentation is allowed to carry on. There is a simple reason for that: the white player is better situated to observe winning states and then update its learning data structures accordingly, also for those states that lead to a win but have not been yet reached via minimax. This behavior was observed for all initial experiments. Subsequently, a  $CC^{1000}$  session was run on the output of each of the  $MC_1$ ,  $MC_3$ ,  $MC_5$ ,  $MC_7$  and  $MC_9$  sessions (totaling a further 5,000 games) and this is where the pendulum effect was observed for the first time.

Specifically, it was observed that the minimax tutor for the white player was actually training the black one by forcing it to lose; when the tutor was drawn out of

the game, the black player overwhelmed the white one, which then had to adapt itself again due to black pressure as fast as possible [17].

### 3 Detecting the pendulum effect: experimentation and analysis

We now review the experiments that we designed and carried out to investigate the “*pendulum effect*” hypothesis. We have decided to do so because the original experimental session, though indicative, was deemed too short to offer any conclusive evidence as to whether this effect was really in play.

For a fixed combination of board size ( $n$ ), base size ( $a$ ), number of pawns ( $\beta$ ) and exploitation-exploration trade-off ( $\epsilon$ ), we ran a  $CC^{10000}$  session, to serve as the fundamental self-playing benchmark. We also ran three  $MC^{100}$  sessions, each one with a different look-ahead value (1, 3, 5). Subsequently, we ran three further  $CC^{10000}$  sessions, basing each one of them on one of the concluded  $MC^{100}$  sessions. A diagram of the above arrangement is shown in Fig. 2.

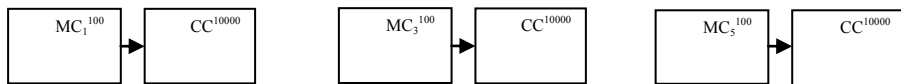


Fig. 2. The experimental workflow.

The follow-up CC sessions are to be compared to the self-playing benchmark which was not based on any prior experience. The hypothesis is, then, that the benchmark session should be, in principle, less complex than its  $MC_x$ - $CC^{10000}$  counterpart, since that counterpart has been initially evolved using minimax and should be better placed to demonstrate the pendulum effect in the absence of the corresponding  $MC_x$  tutor.

An arrangement like the one shown above eventually delivers seven logs in total; we used *gnuplot* to put all seven logs in one graph (see Fig. 3).

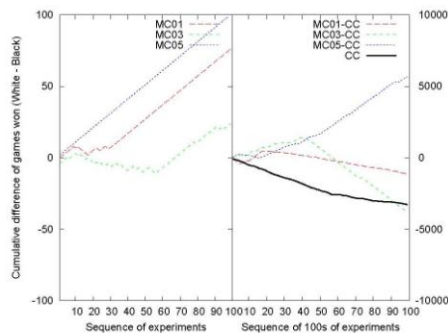


Fig. 3. A pictorial representation of number of games won by each player in each session.

Fig. 3 shows a cumulative graph demonstrating whether it is the white or black player who won a game. The left part has a vertical axis spanning from -100 (all games won by black) to 100 (all games won by white), since any MC session is at most 100 games long. The left part and right parts are aligned with respect to their zero-value in their respective vertical axes. The right part, however, has a vertical axis spanning from -10000 to 10000 to account for the length of the CC sessions. Coloured dashed lines differentiate between MC and MC-CC variants; a solid black line (only available in the right part) serves as the benchmark.

This example graph also shows some fine examples of interesting behaviour:

- $MC_1$  suggests the existence of a (relatively) strong white player, whose CC sequel, however, only delivers for roughly 1/10 of the session; thereafter, the black player assumes leadership and gradually outperforms the white one.
- $MC_3$  suggests the existence of a white player which is not as strong as the  $MC_1$  variant, but whose CC sequel manages to prevail until about midway through the session; the black player manages to take control thereafter and decisively prevails until the end of the session.
- $MC_5$  does not offer confirming evidence for the pendulum effect.
- The CC benchmark generates speculation that it could have been cut shorter.

Such a group of seven sessions totals 300 MC games (3 x 100) and 40,000 CC games (4 x 10,000). To substantiate the existence of the pendulum effect, we experimented with a multitude of parameter combinations, as detailed in Table 2.

**Table 2.** A description of game configurations.

Board size ( $n$ )	5, 6, 7, 8, 9, 10
Base size ( $a$ )	2, 3, 4
Number of pawns ( $\beta$ )	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
Exploitation-exploration trade-off ( $\epsilon$ )	0.1, 0.3, 0.5

The above would account for  $6 \times 3 \times 10 \times 3 = 540$  configurations. However, for any given board size the two bases may neither overlap nor be within one square of each other, so we only allow  $(n,a)$  pairs as follows: (5,2), (6,2), (7,2), (7,3), (8,2), (8,3), (9,2), (9,3), (9,4), (10,2), (10,3), (10,4). These constraints eventually produced a total of 357 distinct<sup>1</sup> configurations (of seven sessions each), which correspond to over 100,000 MC games and over 14,000,000 CC games, which were distributed (at the session level) over a grid computing infrastructure.

We first note that the standalone CC sessions collectively suggest that the two players after extensive self-play converge to having about equal chances to win a game. We examined all standalone CC sessions grouped by exploitation-exploration values. In Table 3 we report the (absolute value of the) difference between games won by the white and the black player, where 100% stands for one player winning all the games of a session, and 0 stands for both players winning the same number of games in a session; we group sessions by reporting smaller differences in a finer resolution.

<sup>1</sup> We did not experiment with  $\beta=1$  in the  $(n = 5, a = 2)$  configuration since it contains a forced win for the first player in 4 moves (or 7 semi-moves).

It is apparent that large differences are a minority, and significantly more so in sessions where exploration is stronger.

**Table 3.** Absolute value of difference in games won by white-vs.-black in standalone CC sessions (119 in each column).

	Number of Sessions		
	$\varepsilon = 0.1$	$\varepsilon = 0.3$	$\varepsilon = 0.5$
0 – 25%	52	49	57
25% – 50%	32	43	42
50% – 100%	35 (~30%)	27 (~23%)	20 (~17%)

These results collectively confirm our earlier fairness claims [16]; Table 4 reviews in detail our current results for the  $\varepsilon = 0.1$  configuration used in [16]. For example, the value “21” in the “21-14” cell in the “50 – 100%” row means that for each of 21 sessions eventually won by the white player, that player won at least 5,000 more games than the corresponding black player for that session.

**Table 4.** Numbers of standalone CC sessions won by white-vs.-black ( $\varepsilon = 0.1$ ).

	Number of Sessions
0 – 25%	30 – 22
25% – 50%	11 – 21
50% – 100%	21 - 14

### 3.1 Analysing the results

The pendulum effect is a manifestation of a power shift; whereas a player had a clear advantage over a series of games, that advantage is at some point eliminated by its opponent, which then scores successive wins, until the trend is reversed again, maybe *ad nauseam*. Detecting the pendulum effect implies that we also have a reference collection of results that can be clearly shown not to be an instance of that event; moreover, it also implies that we have access to enough samples to strengthen our claim. The standalone CC session in Fig. 3 (solid black line) is a good example of a result that belongs to that reference collection; it is a pattern that has been overwhelmingly observed across all standalone CC sessions (referred to as *tabula rasa* elsewhere in this paper).

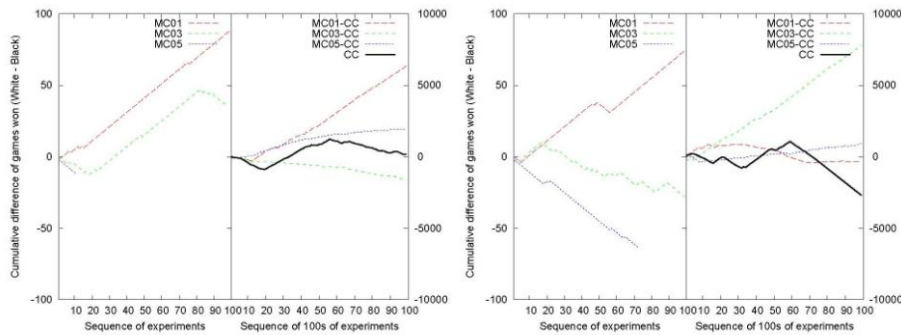
While it may be straightforward to agree on what constitutes a relatively straight line, describing what constitutes instances of the pendulum effect at the standalone CC level is subtler. In Fig. 4 we show the basic alternatives; therein the upper part demonstrates how the pendulum effect brings about a balance of power, whereas the lower part<sup>2</sup> demonstrates an example where, for the length of the recorded session,

<sup>2</sup> Incidentally, large boards ( $n=9$  or  $n=10$ ) and small bases ( $a=2$  or  $a=3$ ) are tough challenges for MC players which demonstrate very bad performance if they are unlucky enough to start exploring a trail that does not lead to a win.

one of the players eventually gains a sustainable edge (observe the rightmost end of the solid black lines with respect to the  $y$ -axis 0 value). Table 5 reviews these results; while CC sessions are overwhelmingly likely to demonstrate a smooth behaviour, increased occurrences of the pendulum effect are associated with larger exploitation values. Similar findings recur for the pendulum effect within MC sessions.

**Table 5.** Relative frequency of the pendulum effect in *tabula rasa* CC sessions (out of 357).

$\epsilon = 0.1$	$\epsilon = 0.3$	$\epsilon = 0.5$
11 %	6 %	4 %



**Fig. 4.** Examples of the pendulum effect in *tabula rasa* experiments.

Of course, the real test for the pendulum effect is to investigate what happens when a tutor is absent. Essentially, we are interested in whether any MC-CC session in some configuration demonstrates more complex behaviour (pendulum effect) than the respective CC baseline, also taking into account the MC session that preceded it. With reference to Fig. 3, it is obvious that the sequence of the MC<sub>5</sub> and MC<sub>5</sub>-CC sessions does not demonstrate a pendulum effect compared to its CC reference; however, seeing MC<sub>1</sub>-CC and its respective predecessor, MC<sub>1</sub>, it is obvious that they demonstrate a more “unstable” behaviour compared to the CC baseline (the group of MC<sub>3</sub> and MC<sub>3</sub>-CC demonstrates a similar behaviour too). However, in this context, exploitation does not drive learners as strongly as the minimax tutor does, which actively maintains a frontier of promising alternatives during game play. Table 6 reviews these results; now, while all sessions are quite likely to demonstrate the pendulum effect, a smaller exploitation drive magnifies the importance of the minimax tutor and makes the pendulum effect more pronounced. Viewing these results vis-à-vis Table 5 makes it apparent that the MC<sup>100</sup> sessions are associated with the subsequent emergence of the pendulum effect.

**Table 6.** Relative frequency of the pendulum effect in combined MC and MC-CC sessions (out of  $357 \times 3 = 1071$ ).

$\epsilon = 0.1$	$\epsilon = 0.3$	$\epsilon = 0.5$
76 %	88 %	87 %

#### 4 On the validity and the implication of the results

Millions of individual games have been carried out, across diverse game size configurations and exploitation-exploration policies. The results amply demonstrate the pendulum effect, mainly by the relative inspection of CC and MC-CC sessions (summarized in Table 7); what initially looks like an interesting minority in standalone CC sessions eventually turns up as the overwhelming majority when there is a previous tutoring (MC) stage.

**Table 7.** Relative frequency of pendulum effect (synthesis of Table 5 and Table 6).

	$\epsilon = 0.1$	$\epsilon = 0.3$	$\epsilon = 0.5$
<i>CC</i>	11 %	6 %	4 %
<i>MC-CC</i>	76 %	88 %	87 %

It is rational to expect that the pendulum effect should arise. When two players interact in a vacuum devoid of other interactions (against other players), none of them has a clue as to whether the other player is inherently good or bad, apart from somehow measuring wins vs. losses. If both players are honest and none of them attempts to trick the opponent by not playing what it thinks it is best for itself, then each player gets an as accurate as possible picture of what its opponents thinks about the state of the game (exploitation is synonymous to honesty; exploration is subtler though). This suggests that the weaker player will be guided through value function updates which will somehow decrease its performance gap compared to the stronger player; this should then bring about, at some point, a stability (or performance plateau). The pendulum effect arises when one of the players has access to a winning policy, since this creates a strong drive away from the stability point; that drive is subsequently compensated by a comparably strong drive towards the stability point again. And, obviously, a tutor (like minimax, for example) does create a winning policy.

How much of a tutor we need is yet another problem. A couple of games or a couple of dozens of games may be only a snapshot. Enlarging that snapshot can be accomplished by observing tutors over even more games or by attempting to generalize from the given instances (as is the case in extensive automatic self-playing in CC sessions), or by combining the two approaches. Combining tutors is a subtler issue where initial results suggest that accumulating experience might not be possible in a distributed fashion [20].

An alternative way of viewing the pendulum effect is to consider learning and un-learning (or, forgetting). Learning *and* forgetting what was learned co-exist in a

reinforcement learning context; good paths need occasional reinforcement so that learned (useful) value functions do not degrade in approximation quality when alternative paths are explored. Again, however, this seems to be linked to the relative density of high value advice (such is, for example, a concrete playing example by an expert), which underlines the conceptual proximity of reinforcement learning to the scaffolding concept in the situated learning approach [21]. A similar problem exists when a tutor unavoidably displays a behaviour that is prone to small errors or deviations, maybe due to physical or mental peculiarities (such is, for example, the case with piloting a helicopter; therein, a best trajectory from a departure point to a destination can be realized as a selection of partially similar, real trajectories as followed by a human pilot). In such a context, the learning mechanism has to be designed with a view to subsequently factor out the deviations [22].

We believe that the pendulum effect can also be well supported by theoretical considerations stemming from game-theoretic considerations. Essentially, the two players play a zero-sum stochastic game (a generalization of Markov decision processes and repeated games). Nash equilibria (i.e., strategies –or suggested actions – for each state of the game, so that no player has an incentive to unilaterally deviate) are known to exist for zero-sum stochastic games [23], where all Nash equilibria have the same value, and even for general-sum stochastic games [24]. Reinforcement learning has been widely studied for the case of stochastic games and Littmann [25] presented the minimax-Q algorithm for finding optimal policies in two-player zero-sum stochastic games.

However, whether Nash equilibria are the most suitable notion for convergence has been questioned, and it has been suggested that convergence to correlated equilibria [26] is more relevant in this setting. Although, it has been proven that Q-learning converges to correlated equilibria against an opponent playing stationary policies, there is no theoretical guarantee that Q-learning converges to correlated equilibria against an opponent that also uses Q-learning [27].

In RLGame, for almost all interesting game configurations, experimental evidence suggests that no player has a forced win; hence, we believe that the outcome of the game (assuming it is played by rational and omniscient players) is a draw. This would strengthen the assumption that the second player, which has been forced to best-respond to a somewhat stronger (minimax) opponent, is able to outperform the first player that was trained by playing against a somewhat weaker opponent, when the minimax tutor is no longer present.

## 5 Conclusions and future directions

This paper focused on the presentation of carefully designed experiments, at a very large scale, to support the claim that expert tutoring can measurably improve the performance of computer players in a board game and that this is mostly evident when the tutor resigns and the impact of its absence is examined.

We assigned such a tutor role to minimax, we elaborated on our experimental setup and we presented results to support our claim, using a key statistic: number of games won by each player throughout the tutoring session and throughout a subsequent non-

tutoring session. While calculating this statistic is trivial, associating it with the actual depth of the tutoring expertise to measure the effectiveness of learning is less so. This is mostly due to the fact that, since both competing players attempt to fine tune their performance vis-à-vis their opponent, differences between them sometimes smooth out. Our experimental evidence and intuition both seem to suggest that the larger the difference, the stronger the smoothing trend. This is the pendulum effect whose emergence out of our experiments is quite robust.

The pendulum effect promotes interactive evolutionary learning as a promising research direction. Therein, one would ideally switch from focused tutor-based training (MC or human-tutor sessions) to autonomous crawling (subsequent CC sessions); essentially, we first generate an initial advantage for one of the players and then explore the state space with the aim to catch up on behalf of the other.

But, the pendulum effect also raises a very fundamental question: when should we stop learning according to some policy and try something else? If the pendulum effect can be computationally traced (and not just visually, as we do now), then we will be able to also detect performance plateaus where players learn nothing and should divert their playing behaviour to explore learning opportunities. In artificial agent societies, where the concepts of competition, co-evolution and co-learning are inherent, a central theme is to select a good partner with which to co-learn for some time, before switching partners to learn something else. Viewing an opponent as a partner is key in co-evolutionary learning and being able to estimate whether such a partnership is profitable will be invaluable.

## References

1. C. Shannon (1950). "Programming a computer for playing chess", *Philosophical Magazine* 41(4): 265-275.
2. A. Samuel (1959). "Some Studies in Machine Learning Using the Game of Checkers", *IBM Journal of Research and Development* 3: 210-229.
3. F-H. Hsu (2002). *Behind Deep Blue: Building the Computer that Defeated the World Chess Champion*. Princeton University Press.
4. J. Schaeffer, Y. Bjoernsson, N. Burch, A. Kishimoto, M. Mueller, R. Lake, P. Lu, and S. Sutphen (2005). "Solving Checkers", *Proceedings of the International Joint Conference on Artificial Intelligence*.
5. R. Sutton (1988). "Learning to Predict by the Methods of Temporal Differences", *Machine Learning* 3(1): 9-44.
6. R. Sutton, A. Barto (1998). *Reinforcement Learning - An Introduction*, MIT Press, Cambridge, MA.
7. G. Tesauro (1995). "Temporal Difference Learning and TD-Gammon", *Communications of the ACM* 38(3): 58-68.
8. I. Ghory (2004). "Reinforcement Learning in Board Games", Technical report CSTR-04-004, Department of Computer Science, University of Bristol.
9. D. Osman, J. Mańdziuk (2005). "TD-GAC: Machine Learning Experiment with Give-Away Checkers", *Issues in Intelligent Systems. Models and Techniques*, M. Damiński et al. (eds.), pp. 131-145.
10. K. Wałędzik, J. Mańdziuk (2010). "The Layered Learning Method and Its Application to Generation of Evaluation Functions for the Game of Checkers", *Proceedings of 11<sup>th</sup>*

*International Conference on Parallel Problem Solving from Nature*, Kraków (Poland), 543-552.

11. A.L. Thomaz and C. Breazeal (2008). "Teachable Robots: Understanding Human Teaching Behavior to Build more Effective Robot Learners", *Artificial Intelligence* 172: 716-737.
12. T.M. Mitchell, S. Wang and Y. Huang (2006). "Extracting Knowledge about Users' Activities from Raw Workstation Contents", *Proceedings of the National Conference on Artificial Intelligence*, Vol. 21, Part 1, 181-186.
13. D. Cohn, Z. Ghahramani and M. Jordan (1996). "Active learning with statistical models", *Journal of Artificial Intelligence Research* 4: 129-145.
14. F. Kaplan, P.-Y. Oudeyer, E. Kubinyi and A. Miklosi (2002). "Robotic Clicker Training", *Robotics and Autonomous Systems* 38(3-4): 197-206.
15. H.J. van den Herik, H.H.L.M. Donkers, P.H.M. Spronck (2005). "Opponent Modelling and Commercial Games", *Proceedings of IEEE 2005 Symposium on Computational Intelligence and Games*, Essex University, Colchester, UK. pp 15-25.
16. D. Kalles, P. Kanellopoulos (2001). "On Verifying Game Design and Playing Strategies using Reinforcement Learning", *Proceedings of ACM Symposium on Applied Computing, special track on Artificial Intelligence and Computation Logic*, Las Vegas.
17. D. Kalles, D. Kanellopoulos (2008). "A Minimax Tutor for Learning to Play a Board Game," *Proceedings of the AI in Games Workshop, 18<sup>th</sup> European Conference on Artificial Intelligence*, Patras, Greece, pp. 10-14.
18. D. Kalles, E. Ntoutsis (2002). "Interactive Verification of Game Design and Playing Strategies", *Proceedings of IEEE International Conference on Tools with Artificial Intelligence*, Washington D.C.
19. D. Kalles (2008). "Player co-modelling in a strategy board game: discovering how to play fast", *Cybernetics and Systems* 39(1), 1-18.
20. D. Kalles, I. Fykouras (2010). "Time does not always Buy Quality in Co-evolutionary Learning," *Proceedings of the 6<sup>th</sup> Panhellenic Conference on Artificial Intelligence (SETN-10)*, Athens, Greece, 143-152.
21. C. Breazeal, A. Brooks, J. Gray, G. Hoffman, J. Lieberman, H. Lee, A. Lockerd, and D. Mulanda (2004), "Tutelage and collaboration for humanoid robots", *International Journal of Humanoid Robotics* 1(2): 315-348.
22. A. Coates, P. Abbeel and A.Y. Ng (2009). "Apprenticeship Learning for Helicopter Control", *Communications of the ACM* 52(7): 97-105.
23. L. S. Shapley (1953). "Stochastic games", *Proceedings of the National Academy of Science*, 39(10): 1095-1100.
24. J. Filar and K. Vrieze (1997). *Competitive Markov Decision Processes*, Springer-Verlag.
25. M.L. Littman (1994). "Markov Games as a Framework for Multi-Agent Reinforcement Learning", *Proceedings of 11<sup>th</sup> International Conference on Machine Learning*, San Francisco, pp 157-163.
26. R. J. Aumann (1974). "Subjectivity and correlation in randomized strategies", *Journal of Mathematical Economics*, 1(1): 67-96.
27. Y. Shoham and K. Leyton-Brown (2009). *Multiagent Systems: Algorithmic, Game-Theoretic and Logical Foundations*, Cambridge University Press.