Adaptive Automata-based Model for Iterated *n*-Player's Prisoner's Dilemma

Sally Almanasra[1], Khaled Suwais[2], and Muhammad Rafie[1]
[1]School of Computer Sciences, Universiti Sains Malaysia, Malaysia
[2]Faculty of Computer Studies, Arab Open University, Saudi Arabia

Abstract: *In this paper, we present a new technique of representing the player's strategies by adaptive automata, which can handle complex strategies in large populations effectively. The representation the player's strategies have a great impact on changing the player's behaviour in rational environments. This model is built on the basis of changing the behaviour of the player's gradually toward the cooperation. The gradualism is achieved by constructing three different adaptive automata at three different levels. The results showed that our model could represent the player's strategies efficiently. The results proofed that the model is able to enhance the cooperation level between the participated player's through few tournaments.*

Keywords: *Adaptive automata, prisoner's dilemma, cooperative behavior, INPPD.*

Received October 3, 2013; accepted June 9, 2014; published online March 8, 2015

1. Introduction

Prisoner's Dilemma (PD) is a symmetric matrix game with a transparent payoff matrix, where both players' can act simultaneously without knowing the other's actions. As a player, your ultimate goal is to achieve the highest score against other opponents. In game theory, this is possible by finding the best path which leads to the highest score [14]. There is one-shot version of the PD when the rational player is faced with playing a single game. Therefore, it is not very interesting to play that since most of the real life applications of one-shot PD are limited. Also, the dominant mutual defection strategy relies on the fact that it is a one-shot game, with no future. While if the player's met and played several games in a row then the game called Iterated PD (IPD).

The key to the IPD is that two player's may play against each other several times. This allows players to generate strategies based on previous interactions. Therefore, a player's move has a considerable effect on future opponent's behavior. The consequences of IPD results in eliminating the single dominant strategy of mutual defection as player's use more complex strategies based on game histories in order to maximize the payoffs they receive.

The Iterated N-Player PD (INPPD) is more realistic game in modeling real-life problems. The importance of INPPD is that it completely displays the problem of social dilemma which is normally happen in a collective action. INPPD has various applications where individual's defections at the expense of others lead to overall less desirable outcomes. Therefore, in this research we will consider the INPPD in modeling the player's behaviors.

One way in modeling the player's behavior is by utilizing the power of adaptive systems. Adaptive systems have successfully obtained the attention in the last few decades due to their suitability in modeling several complex systems. In typical self-learning systems, a self-operating machine is known by an Automaton. The automaton consists of sequence of instructions that are designed to achieve a specific goal. The behavior of a given automaton can be controlled by a pre-determined set of rules or it can adopt new behavior based on the environment in which it operates.

The adaptive behaviors are strongly affected by the learning process. Thus, the adaptive automaton adapts to the responses from the environment and then attempts to learn the best action from a set of possible actions that are offered by the environment. Based on that, we intended to utilize the concept of adaptive behaviors to encourage player's to develop new behavior which tends to be more cooperative.

The rest of the paper is organized as follows: Section 2 discusses INPPD and the existing techniques that are used for representing the player's strategies while section 3 introduces the new adaptive automata-based model. Section 4 discusses the results obtained by the intelligent model while section 5 gives concluding remarks on our work.

2. Background

INPPD is used for finding and building different models which are related to altruism, stable cooperation, co-evolutionary learning, community structure and etc., [2, 3, 10, 11, 17]. INPPD is formulated by the following two statements [15, 16]: Regardless of what the other player's do, each player receives a higher payoff for defecting behavior than for cooperating behavior; all player's receive a lower payoff if all defect than if all cooperate.

As in regular I2PD, each INPPD player should choose either to Cooperate (C) or to Defect (D). The payoff of a player is a function of the number of existing cooperators (i). Let c_i and d_i (for all player's where i= 0, 1, ... , n-1) be the payoff of a given player under consideration if it cooperates and defects, respectively. In other words, for a particular player p who is under the consideration, there should be $c$ooperators and n-i-1 defectors. The player p will receive c_i or d_i when it cooperates or defects, respectively. The player needs not know the number of cooperators and defectors for computing the payoff.

Obviously, INPPD depends on rewarding or punishing the player's based on their actions. Therefore, the payoff matrix which asses the player's actions is crucial in determining the minimum coalition size in INPPD game [9].

In Figure 1, a cooperative coalition makes sense only when the payoff is no worse than all defection. In other words, we need at least M cooperators to form a coalition. If M is larger than one half of the total number of player's, a cooperative coalition cannot emerge in the game. If M is smaller than one half of the total number of the player's, a cooperative coalition may emerge with respect to the number of player's in the game and the used payoff function [13].

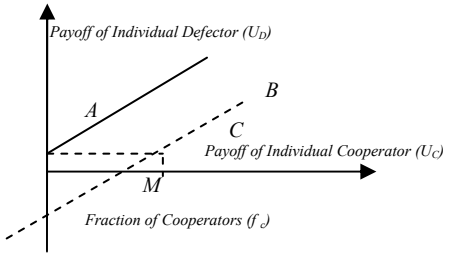

Figure 1. Minimum coalition size M, payoff lines for defectors and cooperators as a function of the fraction of cooperators for INPPD.

The resulting dynamic tends to decrease the number of cooperators within a group. The deficient outcome of PD inheres the fact that the payoff of defectors when there are a minimum number of cooperators (point A) is lower than the payoff of cooperators when there are a maximum number of cooperators (point B) [4]. Thus, even though for a given state of the system an individual benefits more by defection than cooperation, still cooperators in a group of cooperators get more benefit than defectors in a group of defectors.

Generally, the actions of each player are evaluated using the same payoff function and they can choose either to cooperate or defect. The INPPD payoff matrix is presented in Table 1. The columns and rows refer to the number of cooperators and choices that a given player can make, respectively.

Table 1. Payoff matrix of INPPD.

		No. of Cooperators Among the Remaining n-1 Player's				
		0	1	2	…	n-1
Player A	Cooperate	c_0	c_1	c_2		c_{n-1}
	Defect	d_0	d_1	d_2		d_{n-1}

3. Adaptive Automata

An automaton is a self-operating machine. The output of one automaton is a combination between the consequences of the current input and the history of the machine's previous inputs. An automaton is designed to automatically follow a predetermined sequence of operations. Figure 2 illustrates the concept of automaton, where the possible states are presented by S1, S2, S3, S4 and S5.

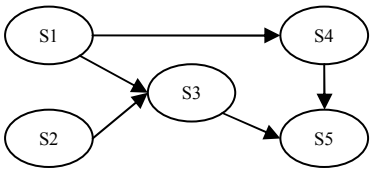

Figure 2. The structure of automata.

An adaptive automaton consists of a set of states, a finite non-empty alphabet, initial state, set of final states and a transition function. The transition function is composed of two transitions levels: Internal and external. The internal transitions are similar to those in finite-state automata. On the other hand, the external transitions are responsible for the calling and returning scheme. In each transition of an adaptive automaton, the current state and the current input symbol of the automaton determine a set of possible transitions to be applied. Mathematically [12] adaptive automata can be defined as a 10-tuple R as follows:

$$R=(S;\ \alpha;\ I_0;\ F;\ \delta;\ U;\ \Gamma;\ H;\ Q;\ \Delta) \quad (1)$$

Where the tuples of R are classified into the following categories:

1. *State Tuples*: this category includes tuples S, I_0 and F, such that:
   - S: Is a set of states.
   - I_0: Is the initial state of the automaton.
   - F: Is the final (acceptance) state.
2. *Input Tuples*: This category includes tuples α and Γ, such that:
   - α: Is a set of input symbols.
   - Γ: Is a set of parameters and variables.
3. *Transition Tuples*: This category includes tuples δ, U, H, Q and Δ, such that:
   - δ: Is the transition relation, where this relation takes two elements: an element from U and a set of mapped parameters from Q.
   - U: Is a set of adaptive function labels.
   - H: Is a set of generators.
   - Q: Is used for mapping parameters, variables and generators to U.
   - Δ: Is a set of adaptive actions {+, -, ?}.

Each adaptive action consists of the *type* and the *transition* of the adaptive action. The action type can

be either a query, remove or insert actions, represented by ?, - and + respectively. Adaptive actions are formulated as calls to adaptive functions with a set of parameters. These actions describe the modifications which should be applied to the adaptive automaton whenever they are called. Technically, simple finite automaton can be turned into an adaptive automaton by allowing its rules to change dynamically.

In order for adaptive automata to do self-modification, adaptive acts adhered to their state-transition rules are activated whenever the transition is used. Adaptive mechanism can be defined as adaptive actions which change the behavior of adaptive automata by modifying the set of rules defining it. The simple notation for representing adaptive automata should have some features such as, being, at least compact, simple, expressive, unambiguous, readable and easy to learn, understand and maintain.

The work presented in [7] has paid attention to INPPD. An evaluative probabilistic automaton was created for strategy modeling. It has been shown that genetic automata are well-adapted to model adaptive strategies. As a result, we noticed that modeling the player behavior needs some adaptive attributes. The computable models related to genetic automata are good tools to model such adaptive strategy. In [6], the authors used genetic algorithms to generate adaptive behaviors to be applied for modeling an adaptive strategy for the PD. Generally, this model is found efficient in generating efficient strategies in complex environments. The main limitation of the above works is that they support limited number of INPPD player's.

Another modeling scheme was presented in [18]. This scheme has formed the collection of automata in a tree-like structure. The modification of action possibility continued at different levels according to the reward signs provided for all hierarchical levels. This scheme is found not efficient in handling large number of player's.

The work presented in [1] has focused on the models which can be used for simulating INPPD. The work showed how existing models and algorithms, in game theory, can be used with adaptive automata for representing the behaviors of player's. The dynamical and adaptive properties can be described in term of specific operators based on genetic algorithms. In addition, the work showed that genetic operators on probabilistic automata enable the adaptive behavior to be modeled for PD strategies.

However, finite automata are a particular case of adaptive automata. If the automata have no rules associating adaptive functions to transitions, the model can be reduced to finite automata. This characteristic is considered important to use adaptive automata naturally where finite automata are required. On the other hand, adaptive automata have a unique computational power [8]. Thus, strategies presented by adaptive automata may show more complex behaviors than the ones described by finite automata.

In the next section, we introduce our new adaptive automata-based model that is composed of multiple automata distributed over different levels through *m* tournaments.

4. The Adaptive Automata-based Model

In this model we have developed a new component Intelligent Decision System (IDS) which is able to represent complex strategies regardless the number of player's. The IDS is an intelligent component operated by one random stage and three levels of adaptive automata that are responsible for representing the player's strategies. IDS divide the total number of tournaments into four main stages, where the first stage (r_1% of the total number of tournaments) allows the player's to play their actions randomly. The second stage is activated in the second r_2% of the total number of tournaments. Consequently, the third and fourth stages are activated in the third r_3% and fourth r_4% of the tournaments, respectively.

In the second stage, an adaptive automaton is designed to help each player to get some information about the other player's. Technically, the second stage of IDS encourages each player to balance between the cooperation and defection behaviors. In other words, the player's will be cautious to be generously cooperative unless some cooperation patterns among the player's are detected. Figure 3 represents the design of the automaton that will be activated in the second stage of IDS.

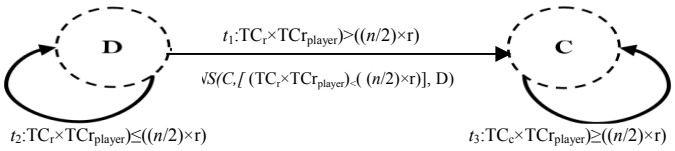

Figure 3. Adaptive automaton activated in stage 2.

Note that, the above automaton is associated with a set of parameters and one adaptive function INS. The function INS is responsible for adding a new transition from state *C* to state *D* when t_1 is invoked. Adding the new transition is responsible for forcing the player to defect if the defection ratio among the other player's (including the player itself) in the population is dominating as shown by the condition of transition $[(TC_r \times TCr_{player}) < ((n/2) \times r)]$.

In term of parameters, the adaptive automaton above consists of a set of parameters that are essential for studying the behavior of the player's. From this point onward, we will refer to the current move, game and generation numbers by m_c, gm_c and gn_c respectively. The current move played at any specific time is presented by the notation (m_c-gm_c-gn_c). In this model, the three stages of automata are controlled by the following set of parameters:

- TC_c: Refers to the total number of cooperated player's (within the same group) in current move

(m_c-gm_c-gn_c). This parameter measures the behavior of each group of player's in term of cooperation actions.

- TC_g: Refers to the total number of cooperated player's (within the same group) in move (m_c+1)-(gm_c-1)- (gn_c-1). This parameter is designed to study the player's behaviors that are directly connected to each other (direct neighborhood). The higher is the TC_g, the higher is the cooperation actions being made by the group. However, it is not necessary that all groups within the population have similar TC_g.
- TC_r: Refers to the total number of cooperation actions made by a particular group of directly connected player's in the last r moves. This parameter assists IDS to study the group's behavior in the last r moves. TC_r parameter is associated with each group.
- TCr_{player}: Refers to the total number of cooperation actions made by a particular player. TCr_{player} is responsible for measuring the cooperation behavior achieved by a particular player in the last r moves.
- n: Is total number of player's.
- r: Is length of internal memory of each player.
- TC_{abc}: Refers to the average number of cooperated player's in moves m_c-i-j, where i and j represent the set of all previous moves played before the current game gm_c (for all $i \leq gm_c$) and the current generation gn_c (for all $j \leq gn_c$). Note that IDS computes TC_{abc} with respect to all strategies played by the directly connected neighbors and stored in the knowledge-based. The purpose of this parameter is to study the behavior of the player's, within each group, in the same move number during all previous games and generations.
- TC_{dkb}: Similar to TC_{abc}, IDS computes TC_{dkb} by finding the average number of cooperated player's in moves m_c-i-j, where i and j represent the set of all previous moves played before the current game gm_c (for all $i \leq gm_c$) and the current generation gn_c (for all $j \leq gn_c$) and with the same move number m_c. However, unlike TC_{abc}, TC_{dkb} is computed with respect to the best strategies played by the whole populations. Note that both of TC_{dkb} and TC_{abc} are extracted from the knowledge-based.

In order to, measure and analyze the player's behavior through generations efficiently, the above parameters are designed such that they cover all aspects of communications between the player's (i.e., player, group and population).

Back to stage 2 shown in Figure 3, the automaton is composed of three transitions (t_1, t_2 and t_3). The transition t_1 indicates that the cooperation ratio between the players's within the same group is considered high with respect to the total number of player's in the population. This leads the player to change its strategy from being defective to cooperative. For instance, given a game of 30 player's (n=30) distributed on a lattice space as shown in Figure 4, where each group consists of five adjacent neighbors (player's) with an internal memory of size (r=4).

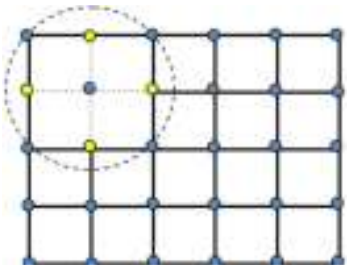

Figure 4. The distribution of 30 player's over the game space.

Assume that a particular player p is currently in the defection state, the strategy of p will only change to C if the group is showing some cooperation behavior as stated by the condition ($TC_r \times TCr_{player} > (15 \times 4)$). The purpose of this transition is to allow p to change its strategy if its group is showing cooperative behaviors with respect to the whole population.

When transition t_1 is triggered, the adaptive function INS attached to t_1 will be activated and a new transition t_2 will be inserted from state C to D, as shown by Figure 5. The purpose of the newly inserted transition is to encourage the player to show more cooperative behavior even if the majority of the player's are not cooperating. But, the cooperative behavior will be limited until the threshold ($TC_r \times TCr_{player} < (15 \times 4)$) in t_2 is reached. In that case, the player has to start defecting since the group is aggressively adopting the defection strategy.

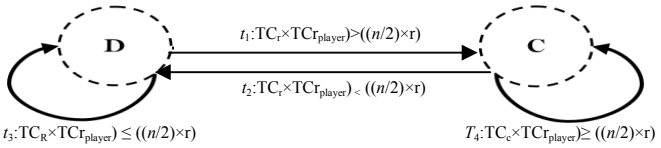

Figure 5. Executing the adaptive function in stage 2.

Regardless the adaptive function, the transitions t_3 and t_4 shown in Figure 5 are also existed in the initial automaton of stage 2 as shown in Figure 3. The main purpose of these transitions is to allow the player to maintain its own strategies based on other player's behaviors. The transition t_3 is responsible for maintaining the defective behavior as long as the group is showing low level of cooperation. While, t_4 is responsible for maintaining the cooperative behavior as long as the group is showing a considerable level of cooperation.

As we move toward new tournaments, the model activates stage 3 to increase the level of cooperation between the players's. Figure 6 presents the constructional design of the automata which operates at stage 3. The automaton is associated with a set of parameters and one adaptive function REM. The function REM is responsible for removing the second part (TC_{dkb}>0.5) of transition t_1 when t_2 is invoked. Removing this part releases the restriction on changing the behavior from being defective to be cooperative.

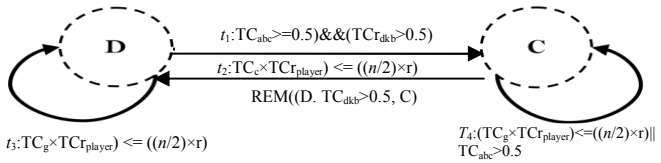

Figure 6. Adaptive automaton activated in stage 3.

With respect to Figure 6, if the player is in the defect state (*D*) and the cooperation ratio of the direct neighbors (TC_{abc}) was at least 50% of the total number of actions, the player can adopt the cooperation strategy, accordingly. One can notice that the third stage of IDS has intentionally added extra rooms for a player to cooperate. For instance, the player can still maintain its cooperative strategy if the group are cooperating in the previous games and generation regardless the current behavior of the group and the player itself, as stated by t_4. Figure 7 represents the structure of the automaton in stage 3 after executing the REM adaptive function.

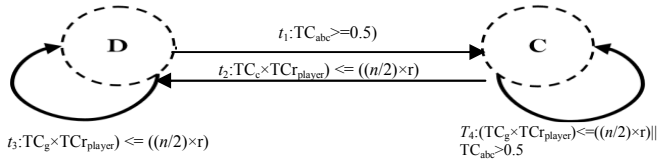

Figure 7. Adaptive automaton activated in stage 3.

At the final stage (stage 4), IDS extends the opportunities for the player's to cooperate as shown by Figure 8. In this stage, the automaton increases the threshold for cooperation and decreasing it for defection. For instance, the automaton in stage 4 has restricted the player to check the value of TC_{abc} and TC_{dkb} in order to maintain the cooperation strategy. If the cooperation ratio within the group (represented by TC_{abc}) and the cooperation ratio within whole population (represented by TC_{dkb}) are forming at least 40% of the actions being adopted, then the player should maintain its cooperation strategy. However, the adaptive automaton after executing the adaptive function of stage 4 is shown in Figure 9.

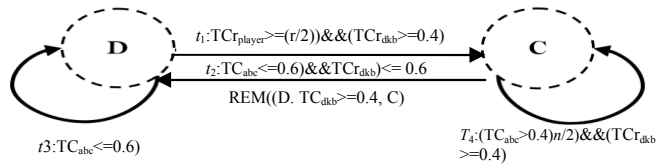

Figure 8. Adaptive automaton activated in stage 4.

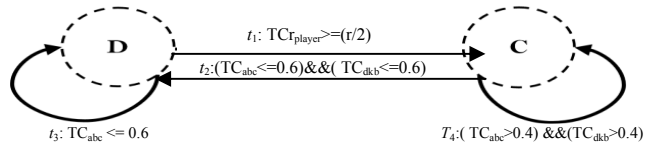

Figure 9. Executing the adaptive function in stage 4.

Based on the state of art of PD, none of the existing models have focused on the importance of communication between the three different layers (player's, group and population layers). In this model we established a set of communication channels between the three layers to provide the player's with a possibility to learn from other player's who have better behaviors in other groups.

5. Results and Discussions

The experiments conducted in this study focused on the ability of our model in evolving the cooperation behaviour between the participated players's. The higher the cooperation ratio, the better is the player's performance.

To standardize the evaluation process, we have initialized the INPPD game as shown in Table 2. These parameters are tuned to measure the performance of the model from different angles. In term of the implementation, the model is coded in Java language under 32bit Windows 7® operating system. The machine is operating on Intel Pentium (R) Dual Core processor of 2.94GH and a RAM of 2.00GB.

Table 2. INPPD game parameters and their initializations.

Parameter	Value
Maximum Number of Tournaments	2000
Number of Games	50
Number of Moves	50
Size of Internal Memory	4
Number of Player's	30
Number of Simulations	20

Note that, each tournament is composed of 50 games and 50 moves. In other words, each complete strategy resulted from a single tournament consists of 2500 actions (*C* or *D*).

In the first test, we measure the performance of our model by evaluating the cooperation level between INPPD players. The results presented in Figure 10 shows that the player's could achieve high cooperation ratio after 1000 tournaments. As the number of player's decrease, the efficiency of the model gets higher. Obviously, after less than 2000 tournaments, the player's could almost reach 95% of cooperation.

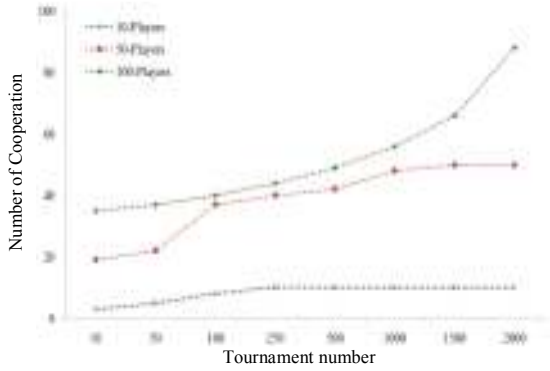

Figure 10. The effect of the model on INPPD player's.

The results also show that the two parameters (TC_{abc} and TC_{dkb}) have a significant impact on the number of cooperation actions played by all player's. Figure 11 illustrates the impact of these parameters, which are extracted from the Knowledge Based (KB), in increasing the number of cooperation actions played through *m* tournaments. Note that increasing the

number of cooperation actions indicates that the player's are changing their behaviors to be more cooperative.

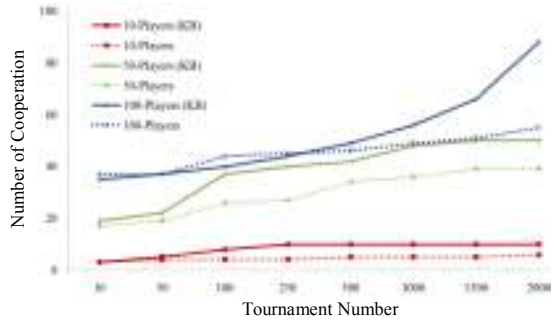

Figure 11. The impact of TC_{dkb} and TC_{dkb} on the cooperative behavior between INPPD player's.

From the other perspective, we found that the length of the internal memory of each player has an effect on the player's performance. We have tested the performance of the model with various internal memory sizes, including (r=1), (r=4), (r=8) and (r=10). The results as presented in Figure 12 shows that the model with 30 players could achieve better results with a memory of sizes 4, 8 and 10. However, we found that increasing the size of the internal memory results in increasing the computational complexity of the model. We also found that larger memory size makes optimizing INPPD infeasible. Thus, the best size of player's internal memory is found to be 4.

In [5] the experiments showed that the occurrence of the following two conditions during a specific tournament indicates that a cooperative behaviour is approaching within the population.

- *Condition* 1: The total number of cooperation actions made by a particular player with the largest payoff is ten times greater than the population size.
- *Condition* 2: The average total number of cooperative actions made by the whole population is ten times greater than the population sizes.

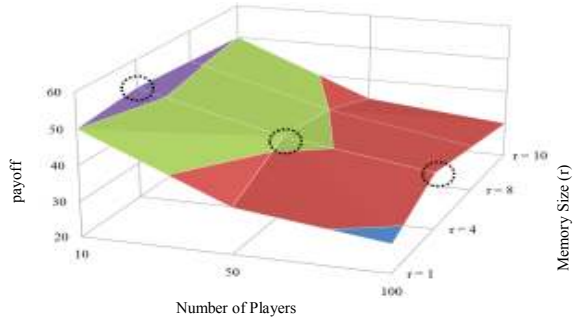

Figure 12. The impact of memory size on player's performance.

Increasing the population size played a primary role in decreasing the number of cooperative actions made by the best player. We have tested the model on three different populations of sizes 10, 50 and 100. For the population of size n=10, the results showed that the best player could satisfy the first condition in the first few generations. When the population size is increased (n=100), the model could satisfy 70% of the first condition where the best player could make a number of cooperative actions which is approximately 7 times greater than the population size. Figure 13 illustrates the impact of population size on the number of cooperative actions made by the best player of the corresponding populations.

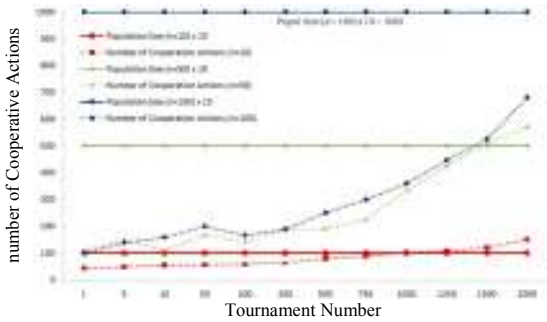

Figure 13. Measuring the behaviour of the populations (n=10, n=50, n=100) in term of the number of cooperative actions made by the best player.

Satisfying condition 2 indicates that the whole population tends to be cooperative. However, we have tested our model against this condition and the results showed that the model has satisfied this condition efficiently with populations of sizes (n=10) and (n=50). With larger populations, the model shows less efficiency in satisfying this condition. A population with 100 players could achieve an average number of cooperative actions that is about 6 times the population size as shown in Figure 14.

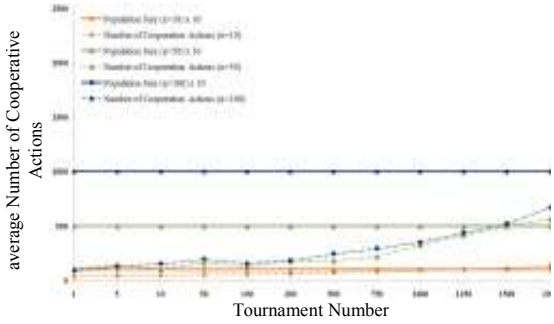

Figure 14. Measuring the behaviour of the populations (n=10, n= 50, n=100) in term of the average number of cooperative actions made by the population.

The other way of proofing that the population favours cooperation rather than defection is by determining the longest sequence of cooperative actions being played by the player's before it get interrupted by a defective action. We tested the model by tracking the cooperative sequences generated by its player's. We found that a population with 10 and 50 players could generate longer cooperative sequences. Figures 15 and 16 show the graphical representation of the behaviour of the best player in INPPD game with populations of sizes 10 and 50, respectively.

Figure 15. The graphical representation of the actions taken by the best player in a population of size (n=30) where dark squares denote the cooperative actions.

Figure 16. The graphical representation of the actions taken by the best player in a population of size (n=50) where dark squares denote the cooperative actions.

6. Conclusions

This paper presents an alternative model for optimizing INPPD. The model focuses on enhancing the cooperation level between rational players in complex systems.

The design of the model is based on incorporating adaptive automata over different layers. These automata are responsible for representing the player's strategies as well as providing the player's with a platform for supporting their decisions.

The experiment results show that the model could evolve the cooperation behaviour among INPPD player's over few tournaments. These results are obtained by tuning different parameters used in the model. Tuning these parameters results in obtaining the optimal values, which lead the player's to achieve significant outcomes.

References

[1] Bertelle C., Flouret M., Jay V., Olivier D., Ponty J., and du Havre L., "Adaptive Behaviour for Prisoner Dilemma Strategies based on Automata with Multiplicities," available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.10.3740&rep=rep1&type=pdf, last visited 2002.

[2] Chiong R. and Kirley M., "Co-evolutionary Learning in the N-Player Iterated Prisoner's Dilemma with A Structured Environment," *in Proceedings of the 4th Australian Conference*, Australia, pp. 32-42, 2009.

[3] Darwen P. and Yao X., "Co-evolution in Iterated Prisoner's Dilemma with Intermediate Levels of Cooperation: Application to Missile Defense," *International Journal of Computational Intelligence and Applications*, vol. 2, no. 1, pp. 83-108, 2002.

[4] Fletcher J. and Zwick M., "N-Player Prisoner's Dilemma in Multiple Groups: A Model of Multilevel Selection," *in Proceedings of the Artificial Life VII Workshops*, Portland, pp. 1-4, 2000.

[5] Franken N. and Engelbrecht A., "Particle Swarm Optimization Approaches to Coevolve Strategies for the Iterated Prisoner's Dilemma," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 6, pp. 562-579, 2005.

[6] Ghneamat R., "Genetic Algorithms and Application to Adaptive Automata for Game Theory," *Ms Thesis*, Al-balqa University, 2005.

[7] Ghnemat R., Oqeili S., Bertelle C., and Duchamp G., *Automata-based Adaptive Behavior for Economic Modelling using Game Theory*, *Emergent Properties in Natural and Artificial Dynamical Systems*, Springer, 2006.

[8] Guerberoff I., Queiroz D., and Sichman S., "On the Effect of the Expressiveness of Strategy Representation Languages in Multiagent based Simulations: An Experiment with An Evolutionary Model of the Iterated N-Player's Prisoner's Dilemma," available at: http://www.inaciobo.com/files/ria2011.pdf, last visited 2010.

[9] Kretz T., "A Round-Robin Tournament of the Iterated Prisoner's Dilemma with Complete Memory-Size-Three Strategies," *Complex Systems*, vol. 19, no. 4, pp. 363-389, 2011.

[10] O'Riordan C. and Sorensen H., "Stable Cooperation in the N-Player Prisoner's Dilemma: The Importance Of Community Structure," *in proceedings of the 5th, 6th, and 7th European Symposium on Adaptive and Learning Agents and Multi-Agent Systems.LNCS 4865*, Springer, pp. 157-168, 2008.

[11] O'Riordan C., Curran D., and Sorensen H., "Spatial N-player Dilemmas in Changing Environments," *Research and Development in Intelligent Systems XXI*V, pp. 381-386, 2008.

[12] Pistori H., Martins S., and Amaury A., "Adaptive Finite State Automata and Genetic Algorithms: Merging Individual Adaptation and Population Evolution," *in Proceedings of the International Conference in Coimbra*, Portugal, pp. 333-336, 2005.

[13] Seo Y., Cho S., and Yao X., "The Impact of Payoff Function and Local Interaction on the N-player Iterated Prisoner's Dilemma," *Knowledge and Information Systems: An International Journal*, vol. 2, no. 4, pp. 461-478, 2000.

[14] Shamshirband S., "A Distributed Approach for Coordination between Traffic Lights based on

Game Theory," *the International Arab Journal of Information Technology*, *vol.* 9, no. 2, pp. 148-153, 2012.
[15] Suzuki S. and Akiyama E., "Reputation and the Evolution of Cooperation in Sizable Groups," *in Proceedings of the Royal Society B*, pp. 1373-1377, 2005.
[16] Szilagyi M., "An Investigation of N-person Prisoner's Dilemmas," *Complex Systems*, vol. 14, no. 2, pp. 155-174, 2003.
[17] Takezawa M. and Price M., "The Evolution Of Reciprocity in Sizable Groups: Continuous Reciprocity in the Repeated N-Person Prisoner's Dilemma," *Journal of Theoretical Biology*, vol. 264, no. 2, pp. 188-196, 2010.
[18] Zhang J., "Adaptive Learning via Selectionism and Bayesianism. Part I: Connection between the Two," *Neural Networks*, vol. 22, no. 3, pp. 220-228, 2009.

Sally Almanasra is a PhD student at the school of Computer Sciences at Universiti Sains Malaysia. In 2007, she obtained her Master degree in computer science from AL-Balqa Applied University. Currently, she is working in the field of game theory and evolving systems.

Khaled Suwais received his BSc degree in computer science from Al-Bayt University, Jordan in 2004, MSc and PhD degrees in computer science from University Sains Malaysia, Malaysia in 2005 and 2009, respectively. Currently, he is an assistant professor at the faculty of computer studies at Arab Open University, Riyadh. His research interest includes: cryptography, information security, parallel computing and game theory.

Muhammad Rafie received BA in business studies degree from Macalester College, St Paul, Minnesota, USA in 1985 and MBA in management information system from University of Dallas, Texas, USA in 1987. Currently, he is an associate professor at the School of Computer Sciences, Universiti Sains Malaysia, Penang. His research interest includes e-learning, mobile learning, computer games, virtual reality, and RFID.