Journal of Experimental & Theoretical Artificial Intelligence

ISSN: 0952-813X (Print) 1362-3079 (Online) Journal homepage: http://www.tandfonline.com/loi/teta20

RMRL: improved regret minimisation techniques using learning automata

Safiye Ghasemi, Mohammad Reza Meybodi, Mehdi Dehghan Takht-Fooladi & Amir Masoud Rahmani

To cite this article: Safiye Ghasemi, Mohammad Reza Meybodi, Mehdi Dehghan Takht-Fooladi & Amir Masoud Rahmani (2018): RMRL: improved regret minimisation techniques using learning automata, Journal of Experimental & Theoretical Artificial Intelligence, DOI: 10.1080/0952813X.2018.1554711

To link to this article: https://doi.org/10.1080/0952813X.2018.1554711

Published online: 15 Dec 2018.

JOURNAL OF EXPERIMENTAL & THEORETICAL ARTIFICIAL INTELLIGENCE
https://doi.org/10.1080/0952813X.2018.1554711

Taylor & Francis
Taylor & Francis Group

ARTICLE

RMRL: improved regret minimisation techniques using learning automata

Safiye Ghasemi[a], Mohammad Reza Meybodi[b], Mehdi Dehghan Takht-Fooladi[b] and Amir Masoud Rahmani[c,d]

[a]Department of Engineering, Sepidan Branch, Islamic Azad University, Sepidan, Iran; [b]Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, Iran; [c]Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran; [d]Computer Science, University of Human Development, Sulaimanyah, Iraq

ABSTRACT

Game theory as one of the most progressive areas in AI in last few years originates from the same root as AI. The unawareness of the other players and their decisions in such incomplete-information problems, make it necessary to use some learning techniques to enhance the decision-making process. Reinforcement learning techniques are studied in this research; regret minimisation (RM) and utility maximisation (UM) techniques as reinforcement learning approaches are widely applied to such scenarios to achieve optimum solutions. In spite of UM, RM techniques enable agents to overcome the shortage of information and enhance the performance of their choices based on regrets, instead of utilities. The idea of merging these two techniques are motivated by iteratively applying UM functions to RM techniques. The main contributions are as follows; first, proposing some novel updating methods based on UM of reinforcement learning approaches for RM; the proposed methods refine RM to accelerate the regret reduction, second, devising different procedures, all relying on RM techniques, in a multi-state predator-prey problem. Third, how the approach, called RMRL, enhances different RM techniques in this problem is studied. Estimated results support the validity of RMRL approach comparing with some UM and RM techniques.

ARTICLE HISTORY
Received 8 June 2017
Accepted 28 November 2018

KEYWORDS
Incomplete-information games; predator-prey problem; regret minimisation; reinforcement learning

Introduction

There are deep connection between Game theory and AI; they both originate from similar roots. Game theory as one of the most progressive areas in AI in last few years studies mathematical models of conflict and cooperation between intelligent rational decision-makers (Myerson, 1991); it is extensively applied to a wide range of decision-making problems in humans, animals and computers. There are much different types of games (Shoham & Leyton-Brown, 2008), such as zero-sum, general-sum, simultaneous and sequential games.

Mostly, agents have complete information of previous choices of other agents and their utility functions as well (Fudenberg & Tirole, 1991). But it is not always possible as the agents may have neither the ability nor the patience to provide their information to others. Therefore, decisions must be made without specifying a complete utility function. Finding solution concepts for such games has recently received considerable attention (Zinkevich, Johanson, Bowling, & Piccione,

CONTACT Mohammad Reza Meybodi mmeybodi@aut.ac.ir Computer Engineering and Information Technology, Amirkabir University of Technology, Tehran, Iran

2007). Many researches are focused on solving incomplete-information games of agents, which are competing with each other to reach goals.

Reaching a solution concept in incomplete-information games seems to have many challenges; Bayesian games may be proposed for analysing such situations (Harsanyi, 1967); Bayesian learning specifies a prior model. Specifying such model is a difficult process and sometimes impossible. Therefore, applying other learning approaches may solve the approach more efficiency. Some approaches are introduced as follows. Reinforcement learning as a popular type of machine learning allows agents to automatically determine the optimal behaviour within a specific context based on feedback from the environment to maximise performance. Some simple feedback is required for the agent to learn its behaviour. The reward can be computed to maximise the utility of agents or to minimise their regret which leads to utility maximisation (UM) or regret minimisation (RM) techniques, respectively. In the former, only the chosen alternative counts, while in the latter the agent bilaterally compare the considered alternative with the other available alternatives. Thus, ignoring the alternatives in UM approaches makes RM approaches to be addressed more. UM approaches such as learning automaton and Q-learning (Narendra & Parthasarathy, 1991; Narendra & Thathachar, 1989; Poznyak & Najim, 1997; Sutton & Barto, 2011) are widely applied to partial observable environments. These approaches compute the utility of doing each action and try to increase the utility while RM approaches assess the chosen action by computing the regret of not doing other actions. RM approaches seem to get better results based on not focusing on just the selected action. In addition to reinforcement learning, some heuristic algorithms may be used to find optimal solutions of such situations (Berryman, 1992; Ferreira, Ribeiro, & Da Costa Bianchi, 2014; Hensher, Greene, & Ho, 2016; Watkins & Dayan, 1992). These algorithms mostly find local optimums and have greater complexity in comparison with reinforcement learning approaches.

The fundamental idea in current research is accelerating the minimisation of the regret to enhance the agents' learning process. In particular, UM reinforcement learning functions (Narendra & Thathachar, 1989), named learning automata, is applied to RM techniques to enhance its performance. Most of learning methods are concerned with how agents make decisions in a way to maximise their utilities, while RM techniques care about agents' decisions in a way to minimise their regrets. In this research, these two kinds of learning methods are combined to achieve better results; learning automaton uses some potential learning functions to update the probabilities of actions (Narendra & Thathachar, 1989) based on the current state of agents and environment (Narendra & Parthasarathy, 1991). The proposed approach, named as RMRL, is applied to an incomplete-information problem named predator-prey, which is a benchmark for multi-agent multi-objective problems of partially observable environments (Watkins & Dayan, 1992). Using RM in a multi-state environment requires considering an individual regret for each state of the grid map in predator-prey problem, such as what we have in Q-learning algorithms. It is shown that the proposed approach accelerates the minimisation of regret. Some reinforcement learning approaches such as Q-learning algorithms and different RM techniques, and heuristic ones like genetic algorithm-based ones (Antonio & Coello, 2017) and co-evolutionary (Chowdhury, Dulikravich, & Moral, 2009) are applied to the problem. Experimental results show that the proposed approach performs more satisfying than the others.

The rest of this paper is organised as follows. The preliminaries and the problem statement are presented in Section 2, and then in Section 3, proposed RM techniques are introduced. Section 4 discusses the experimental results of the proposed approach on a predator-prey problem. Finally, the paper is ended with some concluding remarks in Section 5.

Review of related works

Since the agents may have neither the ability nor the patience to provide their full strategies' information to each other they make decisions based on their current knowledge (Zinkevich et al., 2007). Learning approaches like reinforcement learning techniques are efficiently applied to such

problems. Firstly, an introduction to reinforcement learning and its techniques, which are used in the current research, are given; then, some researchers of reinforcement learning in game playing environments are reviewed.

Reinforcement learning algorithms

In this section, the preliminary of reinforcement learning algorithms are discussed; in addition, some reinforcement learning algorithms such as Q-learning, learning automata and RM techniques are studied as well.

Reinforcement learning is a major class of learning techniques, which comes from machine learning; in spite of supervised learning, it enables the agents to automatically determine an optimal behaviour within their context in a way to maximise their performance. The decisions of the agents affect the future state of the environment; thereby the agents' opportunities are affected in their future movements. Let $A= \{a_1, a_2, \ldots, a_n\}$ as the set of possible decisions of the agent,
$\Psi_t(A)$ as the set of an n-tuple of probabilities, where $\Psi_t(a_j)$ is the probability of choosing decision a_j at time t; $\Psi_t(A)$ is updated using the data obtained up to time $t-1$.

Simple reward feedback, known as the reinforcement signal, helps the agent to change $\Psi_t(A)$ in a way to improve its performance (Foster & Vohra, 1999). Figure 1 depicts an agent with reinforcement learning (Sutton & Barto, 2011); the agent tries to learn optimal policy from its interactions with the environment. The agent uses its experience to improve its performance over the time and interaction with the environment is essential for such enhance. Decisions are a mapping from the perceived signals to the available set of actions. Reinforcement signal are used to interpret the related actions as good or bad. The total amounts of rewards that agents expect are specified by value function Sutton & Barto, 2011) or action-value function (Poznyak & Najim, 1997).

As every action needs to be tested a number of times for a satisfactory performance, it makes a reinforcement learning algorithm a slow one in comparison with other learning algorithms. Q-learning, learning automata and RM techniques are some machine learning algorithms, which make decisions based on the past experiences. Such algorithms are assumed as reinforcement learning when the environment is stochastic. With a given reinforcement scheme, they recursively update the probability distribution of decisions using received signals based on the learning algorithm. Q-learning and Learning automata enhance the decisions by maximising the utility and RM techniques improve the performance by minimising the loss. More details on these algorithms are as follows:

Q-learning algorithms

Q-learning is one of the most popular methods used for implementing reinforcement learning algorithm. It is a model-free algorithm, which is based on the estimation of value function (Watkins & Dayan, 1992). In Q-learning, there is a set of state, S, a set of possible actions, A, and a reward Function $Q: S \times A \longrightarrow R$ (Jalalimanesh, Haghighi, Ahmadi, Hejazian, & Soltani, 2017). Actually, $Q(s_j, a_k)$ shows the reward that an agent receives after performing action a_k in state s_j. The history of an

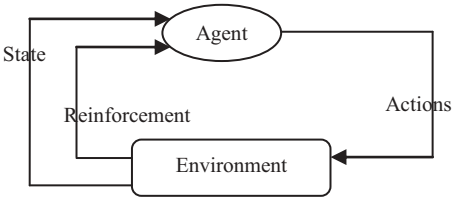


Figure 1. Reinforcement learning model of interaction.

agent can be considered as a sequence of these state-action and rewards. Q-learning uses temporal differences of these values to estimate the value of $Q^*(s,a)$. This algorithm learns an optimal policy. Q values are maintained in a table for being updated after receiving rewards in each step. The following relation is used to estimate the new values of $Q(s,a)$,

$$Q(s,a) = \ Q(s,a) + \alpha[R + \gamma \max Q(s,a'))]$$

Where α is the learning rate and R is the current reward of the agent after performing action a in state s. after updating the value of $Q(s,a)$, the current state is exchanged.

Learning automata

Learning automata learns the optimal actions through repeated interactions with the environment. Actions are chosen based on the probability distribution defined on actions. The environment evaluates the selected action.

Formally, the automaton selects action $\alpha(t)$ using its action probability distribution, $p(t)$ in time t. Then, it applies $\alpha(t)$ to the environment and receives stochastic reinforcement signal $\beta(t)$. Finally, the automaton updates the probability distribution of actions, using the following equations.

$$\begin{aligned} p_i(t+1) &= p_i(t) + a(1 - p_i(t)) \\ p_j(t+1) &= p_j(t) - ap_j(t) \qquad \forall j\,, j \neq i \end{aligned} \tag{1.1}$$

$$\begin{aligned} p_i(t+1) &= (1-b)p_i(t) \\ p_j(t+1) &= \tfrac{b}{n-1} + (1-b).\,p_j(t) \qquad \forall j\,, j \neq i \end{aligned} \tag{1.2}$$

Where a and b denote reward and penalty parameters, respectively; these parameters determine the rate of changes that might be applied to the probabilities of action. Equation (1.1) is used if the selected action is rewarded by the environment and Equation (1.2) is used when it is penalised.

RM techniques

RM as an online learning concept relates to a family of powerful learning algorithms (Zinkevich et al., 2007); it operates based on measuring the regret of agents after their making a decision. The algorithm is a way to capture the intuition that the agent performs well, no matter what others do (Zinkevich et al., 2007). RM algorithms send recommendations to the agents to enable them to compute their regrets. The regret computation process has the role of the value function in reinforcement learning algorithm; in this process each agent compares its current utility with the utility of the recommended decision of the coordinator. In game theory, players are expected to achieve an action probability distribution called correlated equilibrium, which is known as the most relevant non-cooperative solution concept (Shoham & Leyton-Brown, 2008), as the following definition.

Definition 1. Probability distribution Ψ on A is a correlated equilibrium of G if, for every $a, a' \epsilon A$, it is (Foster & Vohra, 1999; Hart & Mas-Colell, 2000; Jalalimanesh et al., 2017):

$$\sum_{a \in A} \Psi(A)[u(a') - u(a)] \leq 0.$$

Where $\psi(A)$ denotes the probability distribution of available actions; $u(a')$ and $u(a)$ are the utility of the agent for action a' and a, respectively. The above inequality denotes the agent does not obtain a higher utility by choosing a' instead of a.

Every finite game has a nonempty, closed and convex set of correlated equilibrium (Shoham & Leyton-Brown, 2008); a proper adaptive procedure ensures that the distributions of the game will

converge to a set of correlated equilibrium with probability one (Foster & Vohra, 1999). The regret of each agent after performing action a is computed as follows,

$$R(a, a') = [max_{a' \in A} u(a')] - u(a) \tag{2}$$

Generally, the regret is the reward that an agent has lost by performing action a instead of a'. Foster (Foster & Vohra, 1999) showed that the average of the realised regret will reach zero as long as the game iterated. Therefore, when the agents continue playing based on RM techniques, after a while the probability distribution of their decisions reaches the optimal state.

Related works

There are many works, which focused on applications of reinforcement learning techniques in different fields such as games. These reinforcement-learning techniques may solve incomplete-information game scenarios (Zinkevich et al., 2007); decision-makers in these techniques aim to minimise anticipated regret (Hensher et al., 2016). Random RM models are compared with random UM models by Hensher et al. (2016) in a stated choice context of choosing among petrol, diesel and hybrid fuelled vehicles to provide useful features for travel demand analysis. It aims at exploring and increasing the potential of RM as a discrete choice model; the approach anticipates the regret with respect to all other alternatives that performs better than the current one; it allows choosing between different travel's multi-attribute alternatives. A modular heuristic algorithm is presented in Ferreira et al. (2014) for finding the optimal solution of multi-agent reinforcement learning problem. Furthermore, this problem has been solved by genetic algorithm in a co-evolutionary framework, in which a prey represents a decision variables space instance and a predator is the entity that deals with the objective functions (Antonio & Coello, 2017).

A model-free algorithm named Q-learning (QL), known as UM reinforcement learning approach, are able to find the optimal solutions based on the state and the current action that maximises the utilities of the agent (Watkins & Dayan, 1992). In solution of a multi-objective optimisation problem such as predator-prey, the original model is built up analytically from the rules of the model (Oremland & Laubenbacher, 2015). An efficient learning algorithm for MDPs is presented in Banerjee (2012) to work in adversarial repeated games.

The optimisation task in incomplete-information problems, such as predator–prey, is performed by a genetic algorithms based multi-objective optimisation technique as well; the technique is utilised in training process of a feed forward neural network (Pettersson, Chakraborti, & Saxén, 2007). Modified Predator-prey Algorithm (MPP), introduced in Chowdhury et al. (2009), is an algorithm, which uses a two dimensional lattice where prey and predators interact.

Proposed RM techniques based on reinforcement learning (RMRL)

As multi-agent environments are among stochastic one, a finite Markov decision process could not be extracted simply; therefore, a model-free reinforcement algorithm such as RM is appropriate for optimising the stochastic behaviour of agents. In games with incomplete-information, RM algorithms are applied to enable players to acquire some knowledge of the game. Using RM algorithms has some benefits, such as no need to coordinate between players, and no necessity for players to know the utility functions of other players (Fudenberg & Tirole, 1991). At each round of the game, a player may play same as previous period, or it may switch to another action based on probabilities that are commensurate with obtained regrets. Let U be the total utility up to now, and $U(a')$ be the total utility, which is obtained if action a' was played instead of the current action. Then, action a', whose utility is larger than U, would be better exchanged with current action. An essential requirement of

any RM decision maker is to avoid or at least reduce the regret. In this section, the extended RM algorithms are introduced; some novel updating equations are applied to traditional RM techniques introduced by Foster and Vohra (1999). The proposed equations enable RM to achieve no-regret in a shorter time.

Furthermore, this study focuses on different RM techniques, which include *external, internal* and *swap*. In *external* RM, the coordinator recommends the agent an action, which makes the utility as more as possible; therefore, the agent wins the game and the other players would get less utilities. The coordinator in *internal* RM uses a mapping between each two actions; it recommends a particular action instead of each possible action of agent; thus, the agent understands the benefits of choosing a particular action in comparison with another one. Finally, in *swap* RM, the coordinator randomly recommends an action instead of the current action of the agent.

The players update the probability of actions using their average regrets. The average regret through time T, in state s, for not playing a' when a is played, is computed as

$$R_T^s(a,a') = max\left\{\frac{1}{T}\sum_{t\le T} R_t^s(a,a'), 0\right\} \tag{3}$$

$R_t^s(a,a')$ is the regret of the player in state s for not performing a' instead of a; it is computed as follows.

$$R_t^s(a,a') = u_t^s(a') - u_t^s(a), \tag{4}$$

Where, $u_t^s(a)$ denotes the utility of the player in state s when a is performed in time t; $u_t^s(a')$ is its utility in state s if action a' is performed in time t.

Finally, according to the computed regret via Equation (4), the probabilities of actions are changed as the following.

$$p_{t+1}^s(x) = \begin{cases} p_t^s(x)\left(1 - \frac{R_T^s(x,a')}{R_{max}(a')}\right) & , \forall x = a \\ \frac{R_T^s(x,a')}{R_{max}(a')\times(|A|-1)} + p_t^s(x)\left(1 - \frac{R_T^s(x,a')}{R_{max}(a')}\right) & , \ else \end{cases} \tag{5}$$

Where, $p_t^s(x)$ shows the probability of action x in state s; $R_{max}(a')$ is the maximum regret that the player may experience if it had not played action a'; $|A|$ denotes the number of possible actions. If the set of actions of players is infinite then $\frac{R_T^s(x,a')}{R_{max}(a')\times(|A|-1)}$ will be zero, without loss of generality. The bigger $R_T^s(x,a')$, the smaller probability is assigned to action x and vice versa. Equation (5) is derived from learning functions of learning automata in the case that the reinforcement signal of the environment is a penalty >0 as presented in Equation (1.2). It is to be noted that the penalty parameter of learning automata is replaced with a variable value based on the regret. All the generated probabilities in Equation (5) are between 0 and 1 inclusive. Algorithm1 presents the details of the proposed RM technique (RMRL) as follows.

The input of Algorithm1 is the set of actions and their probabilities and the recommended action, which comes from regret techniques such as *external, internal* or *swap RM*. In line 1, each player of the game selects an action from set A, based on the distribution probabilities, P. After that, each player computes its utility, in Line 2. Then, the regret is computed via Equation (5) based on the recommended action in Line 4 to update the probabilities of actions. The process is repeated until the regret becomes zero or near to zero; this state is known as correlated equilibrium. Finally, the selected actions of the agents are sent as the output.

Algorithm 1 The proposed regret minimisation Reinforcement Learning-based algorithm (RMRL)

The algorithm is run by the learner agent *i*.

Input:
A: List of available actions of the agent
P: probabilities of actions of the agent
RecAct: recommended action for computing

Output:
Sel_A: selected actions of the agent in current state

Do

1 a = Select(*A, P*);
2 *Sel_A*[*i*] = *a*;
3 *UtilityList*[*i*] = Utility(*i, Sel_A*);
4 *RecAct* = Recommend();
5 *Regret*[*i*] = RM(*Sel_A, i, RecAct*);
6 Update(*P*);
7 **If** Avg(*Regret*) **matches** Equilibrium **then**
Break;
End
8 **while** 1

Evaluation of proposed RM techniques

In order to evaluate RMRL, some experiments are designed, which perform the predator-prey problem (Berryman, 1992). The first experiments, with the aim of sensitive analysis, are designed to evaluate the performance of the proposed approach regarding to the time needed to converge to an optimal solution. The second experiments are designed to verify the performance of RMRL in comparison with the following state of the art algorithms, the presented algorithm in Foster and Vohra (1999) named as traditional RM (TRM), random RM (RRM) (Hensher et al., 2016), QL (Watkins & Dayan, 1992), MPP (Chowdhury et al., 2009), CGA (Antonio & Coello, 2017) and HAMRL (Ferreira et al., 2014). Finally, the last experiments compare the performance of different RM techniques including *external, internal*, and *swap*. The experiments are performed on a 2.4Ghz Core(TM) 2Duo Intel(R) CPU.

Given that the applied model in this research is a predator-prey system. The modular heuristic algorithm presented in Ferreira et al. (2014) is for finding the optimal solution of multi-agent reinforcement learning problem. The proposed algorithm is evaluated in a predator-prey problem, which is a benchmark for multi-agent multi-objective problems (Berryman, 1992). In addition to heuristic algorithms, Q-learning (QL) and RM algorithms are able to find the optimal solutions. QL algorithms operate based on the state and the current action that maximises the utilities of the agent (Watkins & Dayan, 1992). Based on the regret theory, a random RM algorithm is proposed in Hensher et al. (2016) to provide useful features in travel demand analysis (Hensher et al., 2016). Additionally, some algorithms based on genetic algorithms and co-evolutionary are proposed for solving the predator-prey problem (Antonio & Coello, 2017; Chowdhury et al., 2009; Oremland & Laubenbacher, 2015).

Introduction of the predator-prey problem

Dynamic relationship between a predator and a prey is a suitable domain for figuring out the effectiveness of problem solution approaches. The predator is trying to catch the prey, and the

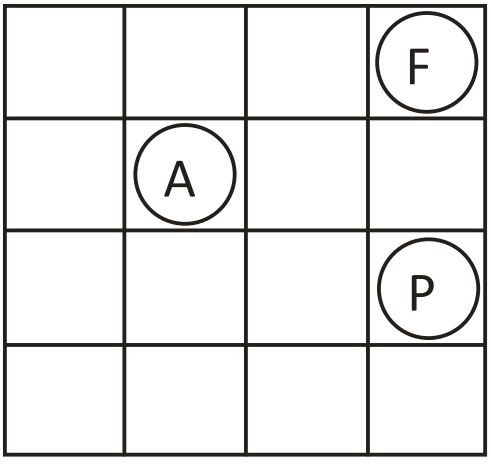


Figure 2. A predator-prey problem sample, 'A' indicates the prey, 'P' indicates the predator and 'F' is the food.

prey is trying to reach the food while avoiding the predator. They are placed in discrete positions of a grid map as depicted in Figure 2.

In a predator-prey problem, the actions of the agents are to go (up, down, left, right). In the experiments the prey is the agent who equipped with learning approaches i.e. the learner agent; it must learn to reach the food while escaping the predator. It is assumed that the food has a fixed place; however, the existence of the predator and its (random/rational) movements makes the world dynamic. The prey uses an experimental distribution which is changed while the game proceeds via Equation (5).

Since the food has a fixed place, in a $m \times n$ map, there must be considered $(m \times n)^2$ states. Each state has an individual value which is known as action-value of that state, and it is computed by the value function of the reinforcement learning. Using RM in such multi-state problem requires considering an individual regret for each state of the supposed grid map, such as what we have in QL algorithms. The value of each state is updated after moving from that state by computing the regret. In UM approaches like QL (Watkins & Dayan, 1992) and HAMRL (Ferreira et al., 2014), the utility value of each state, like Q, is updated in each time step; and for regret minimising approaches including TRM (Foster & Vohra, 1999), RRM (Hensher et al., 2016) and RMRL, the regret of each action is updated in each time step according to the current state. As mentioned the modular approach proposed in HAMRL (Ferreira et al., 2014) uses sum of Q values of the modules as the agents' selection action function, which combines the values from different modules.

Required parameters of learning algorithms

The mentioned algorithms are evaluated in the same testing conditions. Discount factor is set as 0.9 and learning rate is 0.2 for UM approaches. The reward is $2n^2$ for finding the food and it is $-2n^2$ for being captured by the predator in an $n \times n$ grid map. The cost of performing each action is -1. Then, the utility of prey for moving in state s, in time step t, is computed as follows,

$$u_t(s) = \begin{cases} -2n^2 & , \ dis(pred, prey) = 0 \\ +2n^2 & , \ dis(prey, food) = 0 \\ dis(pred, prey)/dis(prey, food) & , \quad else \end{cases} \tag{6}$$

Where $dis(A,B)$ indicates the distance between two points A and B; their positions are equal to (X_A,Y_A) and (X_B,Y_B), respectively. $dis(A,B)$ is computed as $\sqrt{\left[(X_A - X_B)^2 + (Y_A - Y_B)^2\right]}$, which is known as Euclidean distance. After the prey and the predator move in time step t, if $dis(pred,$

prey) = 0, which means that the predator has caught the prey, then the utility of the prey for its move is $-2n^2$; if *dis*(*prey,food*) = 0, which means that the prey has found the food, then the utility of the prey for its move is $2n^2$; otherwise the utility is computed by last equation in Equation (6), which is between $-2n^2$ and $2n^2$. Thus, the largest utility which the prey might gain is $2n^2$ and the least is $-2n^2$.

However, in many real applications, the use of direct Euclidean distance is not possible as the existence of obstacles. Therefore, the grid is modelled as a graph. Then, BFS algorithm is used to find the shortest path between nodes *A* and *B*; dis(*A,B*) is the number of nodes which are traversed to reach *B* from *A*.

*In situ*ation than a rational predator exists the utility function of the predator after moving to state *s*, is computed as follows.

$$u_t(s) = \begin{cases} +2n^2 & , \ dis(prey, pred) = 0 \\ 1/dis(prey, pred) & , \ else \end{cases}.$$

The greater distance has less utility and vice versa. It is to be noted that when there is a rational predator in the grid, the prey reaches the goal in a longer time as expected.

Evolutions of the learning rate

In this section, some experiments are designed to evaluate the implications of the level of dynamics of world on learning rate. The considered dynamics level of world are firstly, the predator does not move at all; secondly, it moves partially, i.e. The predator may move with the probability of 0.5 for any movements of the prey; thirdly, the most dynamics of world is occurred when the predator always moves. A 100 × 100 grid map is applied for the agents to play in following experiments.

Figure 3(a) depicts the evolutions of the learning rate while the only movable agent of world is the prey, i.e. the predator does not move. In such world, the prey can reach the food rapidly in less than 2*n* moves. Then, in Figure 3(b), the predator moves partially. The learning rate decreases as Figure 3(b) depicts. Finally, the fully movements of the predator is assumed in Figure3(c), which depicts slower learning rate. As depicted in Figure 3, learning rate in the case with less dynamics of the world is faster than the case that both the prey and the predator moves completely; thus, the less stochastic the world the faster the learning rate can be. Comparing learning rate of a rational random predator with a random one presents larger regrets in all types of worlds.

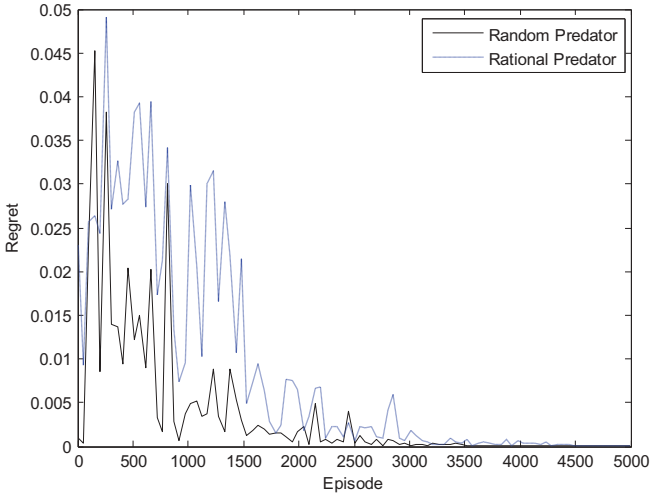
Comparing performance of RMRL with other techniques

Two criterions measure the fitness of RMRL: the total number of times the prey finds the food in a fixed time period (WINS) (Oremland & Laubenbacher, 2015), and the squared difference (*SQD*) of consecutive episode values of the movements of the prey in each time step. For approaches which use UM, such as HAMRL and QL, *SQD(t)* is computed as $(Q_t(s,a) - Q_{t-1}(s,a))^2$ (Ferreira et al., 2014); for approaches which perform based on minimising the regret, such as TRM, RRM and RMRL, by inspiring (Ferreira et al., 2014) it is,

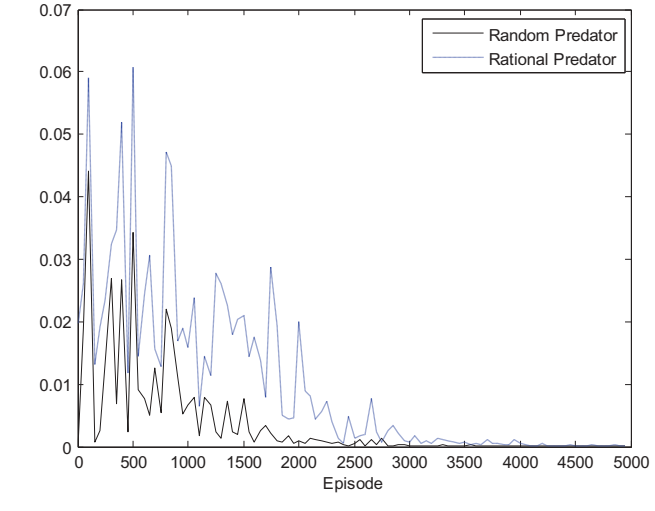
$$SQD(t) = (R_t(a,a') - R_{t-1}(a,a'))^2 \tag{7}$$

where the difference of obtained regret of the prey in two consecutive movements is computed, i.e. *t* and *t*-1. As we are going to compare RMRL with both UM and RM approaches, values of *SQD* must be normalised. The values of *SQD* are mapped to range [0,1] by using Equation (8) as follows:

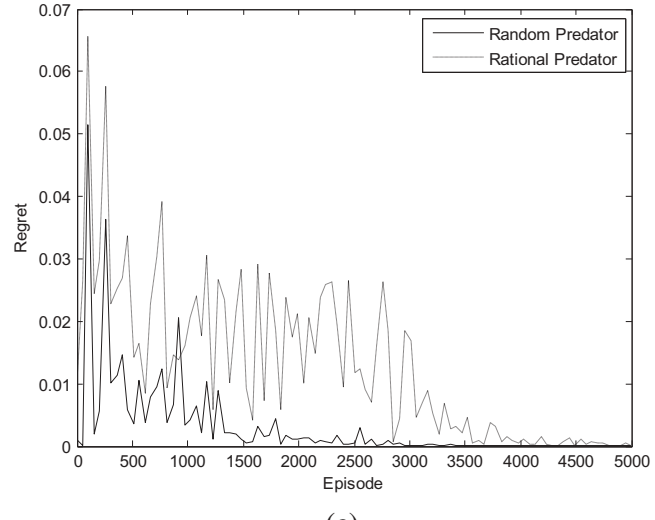
$$N(SQD) = \frac{(SQD - A)}{B - A} \tag{8}$$



(a)



(b)



(c)

Figure 3. Learning rate **(a)** deterministic world **(b)** semi-stochastic world **(c)** stochastic world.

A and B are the minimum and the maximum values that *SQD* may take, respectively. Based on Equations (6) and (7), the values of A and B are supposed zero, and $4n^2$, respectively.

Considering regrets based on states is a novel proposition. States are changed based on the selected actions of agents.

SQD makes it possible to compare the evolutions of considered learning approaches. The decrease of this value shows the stabilisation of the selected action values, i.e. the probability of the chosen action will increase when better rewards are obtained. Figure 4 depicts the values of normalised *SQD* (via Equation (8)) for reinforcement learning techniques in comparison with the proposed one. Also, Figure 5 presents the values of normalised *SQD* for RM-based techniques in comparison with the proposed one. Algorithms that use RM techniques, such as TRM, RRM and RMRL have faster decreases of *SQD* values; i.e. their learning rate is faster. Besides, the values of these algorithms are more close to RMRL in comparison with reinforcement learning based ones.

Next experiment is performed to verify the impact of the learning algorithms on the number of times that the agent succeeds (Figure 6). Initially, the number of times the prey found the food is very close to each other in all algorithms, except QL algorithm, but around the 500th episode, RMRL starts to outperform the others.

Comparison of performance of RMRL in different RM techniques

As discussed in Section 3, RM techniques have different types, such as *external, internal*, and *swap*. However, these techniques compute regret based on the introduced equations previously they are different in the way they choose action a' while computing regret of the agent for not playing a' instead of action a. Generally, a recommendation algorithm recommends action a' for computing the regret.

In this section, the implications of the choice of action a' on the learning rate are discussed and the performance of different RM techniques including *external, internal* and *swap* is studied. For *external* RM, the action 'up' is recommended in all states. Thus in many states no regret is obtained,

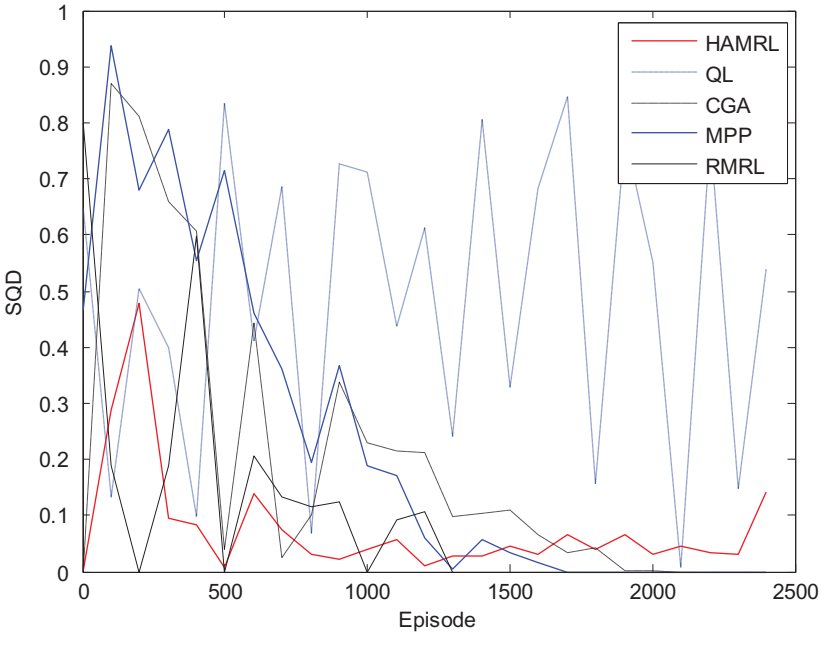


Figure 4. Mean of sum of square difference (*SQD*) of action values for reinforcement learning-based algorithms.

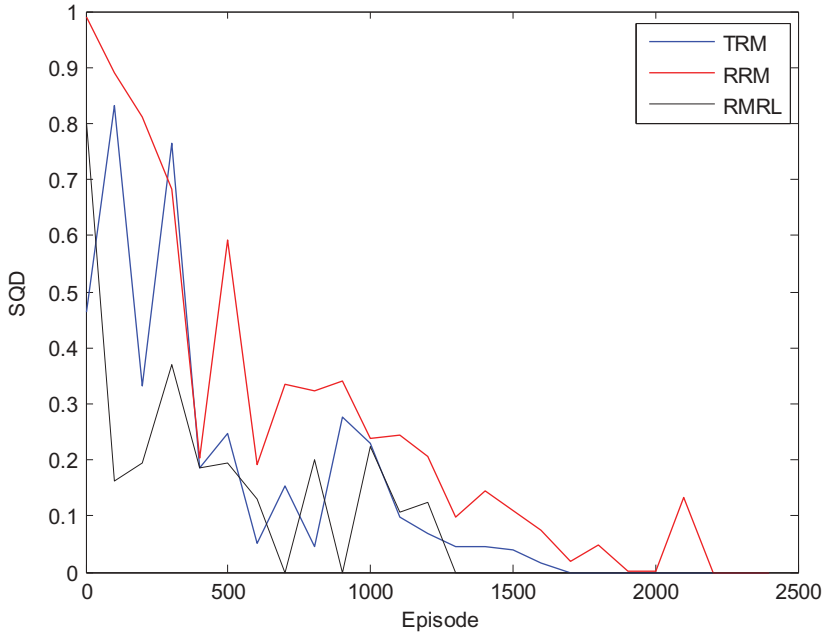


Figure 5. Mean of sum of square difference (*SQD*) of action values for RM-based algorithms.

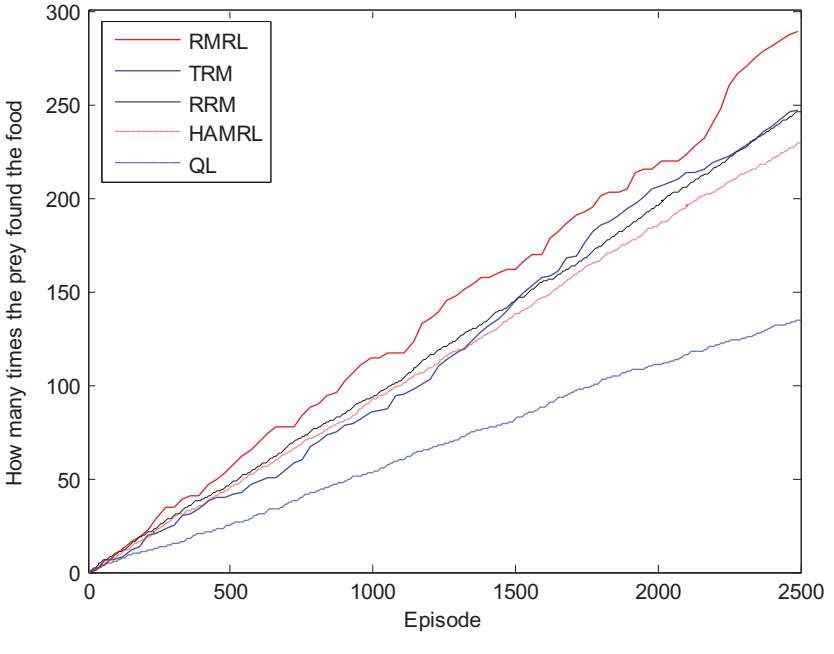


Figure 6. Number of times the prey finds the food.

because the recommended action has not a better utility than the current one. In *internal* RM, for each action a predefined action is specified; action 'up' is mapped to 'down', and for 'left' action 'right' is recommended and vice versa. Finally, in *swap* RM, an action is randomly recommended. Figure 7 depicts the number of times that the prey reaches the food. It can be concluded that the

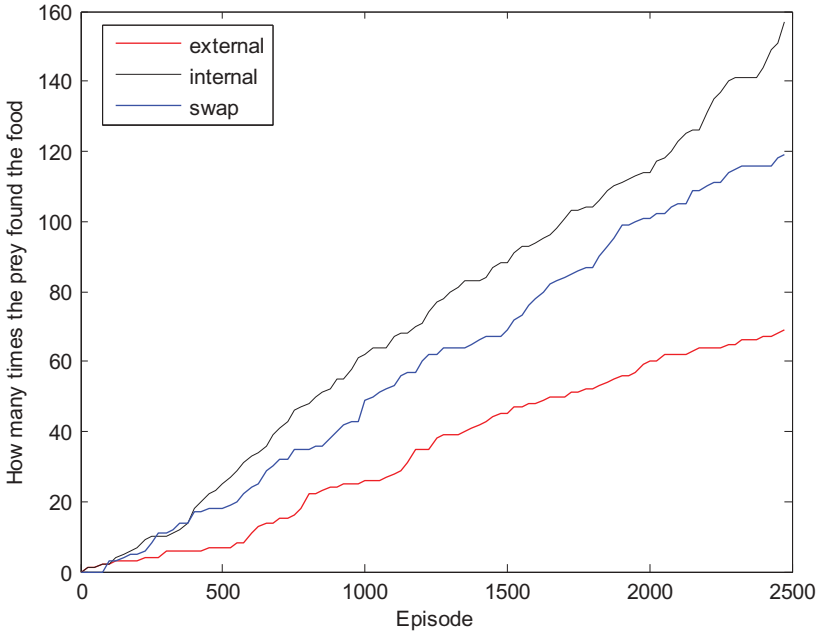


Figure 7. Number of times the prey finds the food.

performance of RMRL, in the case that *external* RM is applied, is the worst and it acts the best when *internal* RM is applied. The reason can be illustrated as the fact that, when there is low number of actions the internal RM has more chance to recommend appropriate actions which cause such outcomes.

Conclusion

Reinforcement learning is a powerful tool for finding the solution of problems with not completely observable environments. When the agents are not aware of others' actions (incomplete-information games) these approaches benefit the agents to find the solutions. A novel RM algorithm is presented that uses reinforcement learning (RMRL) as its learning function. The agents use learning functions to update the probabilities of their available actions based on the computed regret. The proposed RMRL algorithm updates the probability distribution of actions by learning functions of learning automata algorithms. To evaluate the performance of RMRL, it has been applied to the predator-prey problem and the results are compared with QL and HAMRL algorithms as UM techniques and TRM and RRM as regret minimising techniques; the results illustrate that RMRL performs better than others. Some experiments were dedicated to different RM techniques evaluations; different RM algorithms including *external, internal* and *swap* are applied to the problem to evaluate their performances.

Disclosure statement

No potential conflict of interest was reported by the authors.

References

Antonio, L. M., & Coello, C. A. C. (2017). Coevolutionary multi-objective evolutionary algorithms: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 1.

Banerjee, B., & Peng, J. (2012). Strategic best-response learning in multiagent systems. *Journal of Experimental & Theoretical Artificial Intelligence*, *24*(2), 139–160.

Berryman, A. A. (1992). The orgins and evolution of predator-prey theory. *Ecology*, *73*(5), 1530–1535.

Chowdhury, S., Dulikravich, G. S., & Moral, R. J. (2009). Modified predator-prey algorithm for constrained and unconstrained multi-objective optimisation. *International Journal of Mathematical Modelling and Numerical Optimisation*, *1*(1–2), 1–38.

Ferreira, L. A., Ribeiro, C. H. C., & Da Costa Bianchi, R. A. (2014). Heuristically accelerated reinforcement learning modularization for multi-agent multi-objective problems. *Applied Intelligence*, *41*(2), 551–562.

Foster, D. P., & Vohra, R. (1999). Regret in the on-line decision problem. *Games and Economic Behavior*, *29*(1–2), 7–35.

Fudenberg, D., & Tirole, J. (1991). Game theory, 1991. *Cambridge, Massachusetts*, *393*(12), 80.

Harsanyi, J. C. (1967). Games with incomplete information played by "bayesian" players, i–iii part i. the basic model. *The Basic Model. Management Science*, *14*(3), 159-182. doi:10.1287/mnsc.14.3.159

Hart, S., & Mas-Colell, A. (2000). A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, *68*(5), 1127–1150.

Hensher, D. A., Greene, W. H., & Ho, C. Q. (2016). Random regret minimization and random utility maximization in the presence of preference heterogeneity: An empirical contrast. *Journal of Transportation Engineering*, *142*(4), 4016009.

Jalalimanesh, A., Haghighi, H. S., Ahmadi, A., Hejazian, H., & Soltani, M. (2017). Multi-objective optimization of radiotherapy: Distributed Q-learning and agent-based simulation. *Journal of Experimental & Theoretical Artificial Intelligence*, *29*(5), 1071–1086.

Myerson, R. (1991). Game Theory: Analysis of Conflict. Cambridge: Harvard University Press.

Narendra, K., & Thathachar, M. A. L. (1989). *Learning automata: an introduction*. NJ:Prentice-Hall, Englewood Cli s.

Narendra, K. S., & Parthasarathy, K. (1991). Learning automata approach to hierarchical multiobjective analysis. *IEEE Transactions on Systems, Man, and Cybernetics*, *21*(1), 263–272.

Oremland, M., & Laubenbacher, R. (2015). Optimal harvesting for a predator-prey agent-based model using difference equations. *Bulletin of Mathematical Biology*, *77*(3), 434–459.

Pettersson, F., Chakraborti, N., & Saxén, H. (2007). A genetic algorithms based multi-objective neural net applied to noisy blast furnace data. *Applied Soft Computing*, *7*(1), 387–397.

Poznyak, A. S., & Najim, K. (1997). Learning automata and stochastic optimization. Lecture Notes in Control and Information Sciences (225), Springer-Verlag, London.

Shoham, Y., & Leyton-Brown, K. (2008). *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press.

Sutton, R. S., & Barto, A. G. (2011). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.

Watkins, C. J., & Dayan, P. (1992). Q-learning. *Machine Learning*, *8*(3–4), 279–292.

Zinkevich, M., Johanson, M., Bowling, M., & Piccione, C. (2007). Regret minimization in games with incomplete information. In Advances in neural information processing systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems (pp. 1729-1736).