Automatic Reward Shaping in Reinforcement Learning using Graph Analysis

Maryam Marashi
School of Math and Computer Science
Amirkabir University of Technology
Tehran, Iran
marashi_maryam@aut.ac.ir

Alireza Khalilian
School of Computer Engineering
Iran University of Science and Technology
Tehran, Iran
khalilian@comp.iust.ac.ir

Mohammad Ebrahim Shiri
School of Math and Computer Science
Amirkabir University of Technology
Tehran, Iran
shiri@aut.ac.ir

Abstract—**Reinforcement Learning is a popular context of machine learning that aims at improving the behavior of autonomous agents that learn from interactions with the environment. However, it is often costly, time consuming, and even dangerous. To deal with these problems, reward shaping has been used as a powerful method to accelerate the learning speed of the agent. The principle idea is to incorporate a numerical feedback, other than environment reward, for the learning agent. However, finding an efficient potential function to shape the reward is still an interesting area of research. In this paper, a new algorithm has been proposed that receives the environment graph, performs some new analysis, and provides the extracted information for the learning agent to accelerate the speed of learning. This information includes sub goals, bad states, and sub environments with different exploration, or reward, values. To evaluate this algorithm an experimental study has been conducted on two benchmark environments, Six Rooms and Maze. The obtained results demonstrate the effectiveness of the proposed algorithm.**

Keywords-Reinforcement Learning; Q-Learning;Reward Shaping;Artificial Feedback

I. INTRODUCTION

Reinforcement Learning (RL) is an interesting area of the machine learning that aims at improving the behavior of the intelligent agent based on the reinforcement signals received from the environment [1]. The agent, instead of an explicit plan, is allowed to traverse the environment, and adopts an appropriate policy based on the instructive feedback taken from the environment over each action that results in the maximum achieved reward. Thus, the reward function implicitly defines the optimal behavior of the agent [1].

In some real situations, however, like a football game, the environment reward is given to the agent with some delay which leads the agent to spend large amounts of time to attain the optimal behavior. There are a number of ways proposed in the literature, to overcome this issue [2-3]. Reward shaping is a method employed in the field to deal with this problem [4]. In this method, the designer produces a reward function virtually and provides it for the learning agent. In this method, the agent receives the reward in the first episodes of learning from this virtual reward function, since it does not take any reward from the environment in the early episodes of learning. The agent uses this virtual reward to optimize its behavior. Therefore, the shaped reward could have a significant effect to speed up the learning process.

The principle idea of reward shaping is to provide additional feedback by the designer, other than that of environment, for the agent in order to improve its convergence rate and learning speed [4]. In the next episodes, the agent adjusts its actions based on the knowledge learned from the environment, and the effect of the shaped reward is gradually reduced.

Even though reward shaping has been proved to be a powerful method, determining the values of the shaped rewards for the real big environments is a challenging and sometimes even impossible task. To deal with such problems, one can produce the shaped rewards in some way automatically [5].

In this paper, a new approach based on graph theory has been proposed that provides shaped rewards automatically for the reinforcement learning. In this approach, the agent estimates suitable rewards for different actions based on an analysis on the environment graph. Specifically, in this analysis two kinds of special states are recognized, sub goals and bad states. Sub goals are bottleneck states that have a major role in guiding the agent to the goal. The actions leading to these states are given a higher reward values. On the contrary, bad states are those that might increase the exploration time and have the least importance. Each action that leads to these states is given less reward value.

Moreover, the graph of the environment is divided into some disjoint parts according to the dependency between its nodes. Then, the shaped reward for each action of each state is assigned based on a probability distribution. Each state in a bigger sub environment may have a greater likelihood to guide the agent to the goal, so it must be assigned a higher reward value. The shaped reward in this paper consists of the values gained considering sub goals, bad states, and probability distribution of the states in environment.

In the learning function of the agent, a reducing factor has been considered for the shaped reward. While the shaped reward speeds up the learning process, the reducing factor preserves the effect of the learned knowledge from the environment, hence leading to hold the convergence property of the algorithm.

The remainder of this paper is organized as follows: Section 2 gives a brief overview of the Markov Decision Process and reinforcement learning. The shaping reward is introduced in Section 3. The main contribution if this paper is presented in Section 4. Section 5 describes the experimental studies and the results. Finally, some concluding remarks are given in Section 6.

II. Reinforcement learning

Markov decision process is a formal notation to model reinforcement learning problems that has been widely applied in discrete environments.

A. *Markov Decision Process*

Markov Decision Process which was first introduced by Bellman in 1957 [6], is a model for sequential decisions. It is used when the function of an agent depends on a set of sequential decisions not just the current decision. Markov decision process is shown as a 5-tuple (S, A, T, R, α) in which each part is defined as follows [1]: S, all possible states in the environment; A, all possible actions for an agent in each step of decision; T, indicates the transition probability from state s to state s' if the action a has been selected from the set of possible actions; R(s, a, s'), the received reward by the agent after selecting the action a and transition from state s to state s' in the environment and α is a discount factor.

B. *Reinforcement Learning*

The first aim in the reinforcement learning is to transform the problem to one of the types of the Markov Decision Process. Then, the next step is to find an optimal policy for this problem. Dynamic programming strategy is one of the methods for finding an optimal policy. However, computational complexity of these algorithms increases exponentially by increasing the number of states. Q-Learning algorithms [7] and Sarsa learning [8] are the most famous methods of reinforcement learning. For example, in Q-Learning algorithm in the discrete state, a $Q(s, a)$ is defined for each pair of state-action. This value includes the whole reward received when the agent starts from the state s, performs the action a and follows the algorithm's policy. $Q(s, a)$ function is updated with (1) until it converges to the optimal amount:

$$\begin{aligned} Q_{t+1}(s,a) \leftarrow (1-\alpha)Q_t(s,a) + \\ \alpha * [r(s,a) + \gamma * \max_{a' \in A_s} Q_t(s',a')] \end{aligned} \tag{1}$$

In each step of this algorithm, an action with the highest value among the possible actions will be chosen.

III. Reward shaping in reinforcement learning

The existing reinforcement algorithms attempt to find a policy which results in achieving the highest possible total reward. Thus, the reward function might describe an optimal behavior for the agent. Using an appropriate or optimal reward function can have a significant effect to speed up the learning.

It should be noted that although shaped reward is a powerful tool in accelerating the learning of the agent, but a wrong definition of this function can considerably result in misleading of the agent [9]. Different studies exist in literature on how to define the shaped reward since 1992. These studies can be divided in two categories: (1) quantifying and assignment by the domain expert, and (2) automatic estimation.

A. *Use of the Domain Expert Knowledge*

In 1996, Bishop used the idea of multi-layer neural networks for quantifying the shaped reward [10]. In 2003, Dejong and Laud came up with the idea of valuing the states by changing the reward function [11]. In this approach, the agent just acts in a part of environment which is more likely to meet the goal and find the optimal policy. In 2005, Abeel and Ng [12] introduced a method about learning the reward function of domain expert by the agent. In this method, the agent tries to achieve the optimal solution by observing the behavior shown by the expert, avoiding interaction with the environment and just by his prior knowledge about determining assignments. In 2006, Ng and Bagnell proposed reward abstraction to shape the reward for the agent [13]. Then, in 2003, Wiewiora [14] proved that if agent has the shaped reward as the potential function in (2), it will finally converge to the optimal policy.

$$F(s,s') = \gamma * Q(s') - Q(s) \tag{2}$$

B. *Automatic Calculation of the Shaped Reward*

In 2007, Marthi first introduced the idea of automatic shaping [15]. He applied state abstraction as input instead of the potential function which is totally dependent on the amount of expected reward and shaped the reward based on the abstraction. He proposed an algorithm which receives all feasible states of the environment as input and renews all the states which have the similar action in a task. It also accepts some temporary abstract actions and defines a new potential function and uses it in the structure of shaping the reward. In 2008, Asmuth [16] proved that the algorithm based on R-max model with the defined potential function can maintain the optimal policy if:

$$Q(s) \geq \max_a Q(s,a) \tag{3}$$

In 2010, Kudenko and Grzes, presented a model [17] for learning the potential function in parallel with reinforcement learning based on the R-max algorithm. The presented algorithm defines a dynamic model from the environment for learning the potential function in an on-line manner. However, in all of the existing algorithms, this function is dependent on the knowledge of domain expert or value of the states based on the learned knowledge of the agent from the environment exploration. Thus, none of these algorithms can effectively accelerate the learning especially in the real and large environments.

IV. The proposed approach

In this paper, a new algorithm has been proposed for estimation of shaped reward function which is neither dependent upon the domain expert knowledge nor exploration of the environment by the agent. The proposed algorithm works based on the graph data structure. In addition, the new shaped reward function is defined based on the values gained from the graph analysis, finding the independent sub-graphs, determining bottleneck nodes (middle or suboptimal goals), and also nonsignificant nodes (bad states). Hence, in the next subsection mapping the environment to graph is described. Then, the proposed algorithm is given.

A. *Mapping the Environment to the Graph and Its Anlysis*

In order to construct a nondirectional and weightless graph $G(V, E)$ based on Markov Decision Process, each state $s \rightarrow s'$ is considered as a node in graph and each transition $v \rightarrow v'$ in the transition function is considered as an edge. Then, by automatic analysis of each graph, approximate value of exploration for each state is estimated which would give us the shaped reward function. This method has two advantages: (1) Valuing the shaped reward is not dependent upon the knowledge of the domain expert which might be erroneous. (2) It is also independent of environment exploration by the agent. So the agent does not need to learn and estimate the shaped reward.

The first step of this analysis is to determine the *middle* or *sub goals* as special states in the environment. Reaching to these states would be useful for the agent in that it might cause to guide the agent to the goal faster. There are different methods to determine these sub goals [18]. For the case of our algorithm, the method based on computing the betweenness centrality [18] of the graph has been utilized. This criterion is defined as follows:

$$BC(v) = \frac{1}{n-1} \sum_{u \neq s \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}} \qquad (4)$$

In the (4), the denominator of the fraction is the number shortest paths between the nodes s and t, and nominator shows the number of shortest paths between the nodes s and t that crosses over the node v. The node with the highest $BC(v)$ value would be considered as a sub goal. Upon determining these nodes and defining more virtual reward for them, the next step is to determine *bad states* based on the adjacency matrix of the graph. Bad states are those with the least number of adjacent states that are not sub goal. For these states, less virtual reward would be assigned.

The amount of the virtual rewards for bad states and sub goals are determined based on a proportion of the reward value for the goal state. In the experimental studies Section, we point to the exact values considered for them.

The last step is decomposition of the graph to the independent sub graphs based on the middle goals to estimate a different value for each simple state (nonsignificant or bad states, and sub goals). This value is calculated based on the intuition that the goal has a great likelihood to be located in the bigger sub graph. The reward value of a simple state is proportionate to the number of nodes in the sub graph enclosing that state to the number of total states in the graph. This proportion is then multiplied by the reward value of the bad state.

Finally, by determining the virtual value of each state/action, it is used to form the shaped reward matrix based on the proposed methods. Even though the usage of this reward in Q-Learning equation has a significant effect in accelerating the learning of the agent, but the results of experiments have shown that combination of this reward with the potential function makes an increase of at least 20 percent in convergence rate and learning speed.

B. *The Algorithm*

By analysis of the resulted graph of environment, the virtual value of each state is calculated automatically. This virtual reward is defined as *ManualReward(s, a)*. In the proposed algorithm shown in Fig. 1, by using a discount factor in Q-Learning equation, the agent learns the environment over time and this factor makes the *ManualReward* to lose its effect upon advancing the simulation. This is meant to maintain the algorithm's convergence if there would be any error in automatic valuing of rewards. It prevents from misleading the agent.

According to the new defined equation for updating $Q(s, a)$, two effective parameters in accelerating the learning are simultaneously applied: (1) The potential function, that is updated in parallel with learning the environment. Therefore, it is not required to wait for the agent that reaches the goal and receives the feedback from the environment. (2) Necessary actions to learn the environment and reach the goal would be decreased as a result of receiving the shaped reward obtained from analysis of the graph. This would consequently lead to speed up the learning process.

$$\begin{aligned} Q(s,a) = Q(s,a) + \alpha[r + \max_{a'} Q(s',a') - Q(s,a) + \\ \beta * manual\mathrm{Re}ward(s,a) + \\ \gamma * \max_{a'} Q(s',a') - Q(s,a)] \end{aligned} \qquad (5)$$

$$\beta = e^{\frac{-t^2}{\sigma^2}}$$

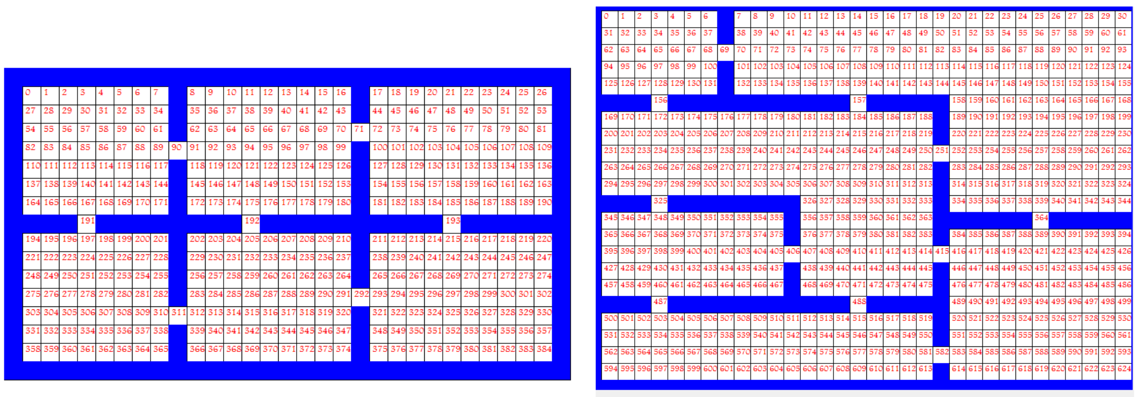
According to (5), using discount coefficient for shaped reward causes the agent to act randomly in first episodes. But over time and by finding sufficient knowledge of environment, it eagerly selects the actions based on the most receivable reward.

V. EXPERIMENTAL STUDIES

The proposed algorithm has been evaluated empirically on the Six Rooms and Maze benchmark environments. The environments have been shown in part (a) and (b) of Fig. 2 respectively. In the experiments, following values were used: $\alpha = 0.5$, $\gamma = 0.9$, $\sigma = 0.9$. In addition, we have considered the values for sub goals and bad states, one percent of the reward value for the goal state. Note that the magnitude of reward value considered for sub goal was positive and for bad state was negative. In all experiments, the random exploration was

```
input: The matrices $Q$, $reward$, and $next$ for a given environment
output: The final $r$ value for each episode of the simulation
declare:
        Beta(episode): A function to reduce the effect of the domain expert reward values, it is defined as follows:
              exp[- (episode * episode) / (sigma * sigma)] in which sigma is a constant value between 0 and 1.
        Modify(manualReward): A function that finds sub goals and bad states automatically, and then modifies
              the manualReward matrix according the these special states.
algorithm QLearningWithShapingAndManualReward begin
    Initialize manualReward(s, a) with reward(s, a)
    Modify(manualReward)
    for each episode do begin
        Initialize s, e.g. randomly
        $r := 0$
        repeat (for each step, action, of episode)
            Choose $a$ from $s$ using policy derived from $Q$ (e.g., ε-greedy)
            $s' := next(s, a)$
            $r := r + reward(s, a)$
            $Q(s, a) := Q(s, a) + \alpha * [r + \max_{a'} Q(s', a') - Q(s, a) + \text{Beta}(episode) * manualReward(s, a) +$
                        $\gamma * \max_{a'} Q(s', a') - Q(s, a)]$
            $s := s'$
        until $s$ is terminal or the number of actions exceeds a threshold
        Store the number of current episode and the corresponding final $r$ value.
    endfor
end QLearningWithShapingAndManualReward
```

Figure 1. The proposed algorithm



(a) (b)

Figure 2. (a) The Six Room environment, (b) The Complete Maze environment

used for the first 50 episodes and then ε-greedy exploration [1] strategy was used for the remaining episodes with *ε* set to 0.1. Moreover, each experiment has been run for 2000 episodes. The results have been shown in Fig. 3 and Fig. 4 for the Six Rooms and Maze environments respectively. In each figure, the part (a) compares the average reward per episode averaged per 20 episodes for Q-Learning and the proposed algorithm, and the part (b) compares the average cumulative reward for the mentioned algorithms and the two benchmark environments. Since the reward values achieves convergence at about the episode number of 500 and 1000 for the Q-Learning and the proposed algorithm respectively, the remaining part of each diagram has been cut.

As can be seen in part (a) of Fig. 3 and Fig. 4, the new algorithm shows significant improvement over the Q-Learning method for the two benchmark environments with respect to accelerating the learning speed and the convergence rate.

To determine whether the achieved improvement of the proposed algorithm over the Q-Learning method is statistically significant, we have used a statistical approach, called *hypothesis test for the difference of the two means*

[19]. To achieve this, the z value has been computed using (6). This value is then referenced in a table of critical values. The z value was computed to 5.36 and 12.08 for the Six Rooms and Maze environments respectively. We used as reference a table of critical values presented in [19]. For the computed z values, it turns out that we can reject the null hypothesis with the confidence over than 99.99 percent. Rejecting the null hypothesis implies that the difference in the mean values of the obtained rewards achieved by the new algorithm and Q-Learning algorithm is statistically significant.

$$\frac{\overline{x_1} - \overline{x_2} - \delta}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}} \tag{6}$$

VI. CONCLUSIONS

In this paper, a new algorithm has been proposed using the environment graph analysis that achieves a new reward matrix as the shaped reward. The benefit of this algorithm is that it is independent of the domain expert's knowledge, which might be erroneous. Instead, it spends some time to learn the shaped reward by the agent and still preserves the convergence of the algorithm. The experimental results showed that the new algorithm is effective both in increasing the learning speed and convergence rate. Since constructing the environment graph could be implemented in polynomial time, the proposed algorithm can be applied for every real environment, no matter how large it is.

REFERENCES

[1] S.Sutton & A.G.Barto, Reinforcement Learning : An Introduction, 1998

[2] M. J. Mataric. Reward functions for accelerated learning. In Proceedings of the 11th International Conference on Machine Learning(ICML), pages 181-189, 1994.

[3] A.Epshteyn and G.Dejong, Qualitative Reinforcement Learning, Appearing in Proceedings of the 23 rd International Conference on Machine Learning, Pittsburgh, PA, 2006.

[4] Andrew Y.Ng, Shaping and police search in reinforcement learning, PhD Thesis, University of California, Berkeley, 2003

[5] A. LAUD, Theory And Application Of Reward Shaping In Reinforcement Learning, PhD Thesis, University of Illinois at Urbana-Champaign, 2004

[6] Putternman, Markov Decision Process, Wiley-Interscience;1 edition (March 3,2005), 2005

[7] L.P. Kaelbling, et al., Reinforcement Learning : A Survey ,Journal Of Artificial Intelligence Research, vol.4, pp.237-285, 1996

[8] G. A. Rummery and M. Niranjan, "On-Line Q-Learning Using Connectionist Systems," 1994.

[9] Randløv and p.Alstrom, Learning to drive a bicycle using reinforcement learning and shaping, J. (Ed.), In Proceedings of the 15th international conference on machine learning, pages 463-471,Morgan Kaufmann,CA., 1998

[10] Bishop, C. M. ,Neural networks for pattern recognition. Oxford University Press., 1996

[11] Wiewiora, E., Potential-based shaping and Q-value initialization are equivalent. Journal of Artificial Intelligence Research., page 205-208, 2003

[12] Abbeel, P., & Ng, A. Y., Exploration and apprenticeship learning in reinforcement learning. ICML., 2005

[13] Bagnell, J., and Ng, A., On local reward and scaling distributed reinforcement learning, Neural Information Processing System. MIT Press., 2006

[14] J.Asmuth, M. L.Littman, & R.Zinkov, Potential-based shaping in modelbased reinforcement learning. In Proceedings of AAAI conference on Artificial Intelligence. , 2008

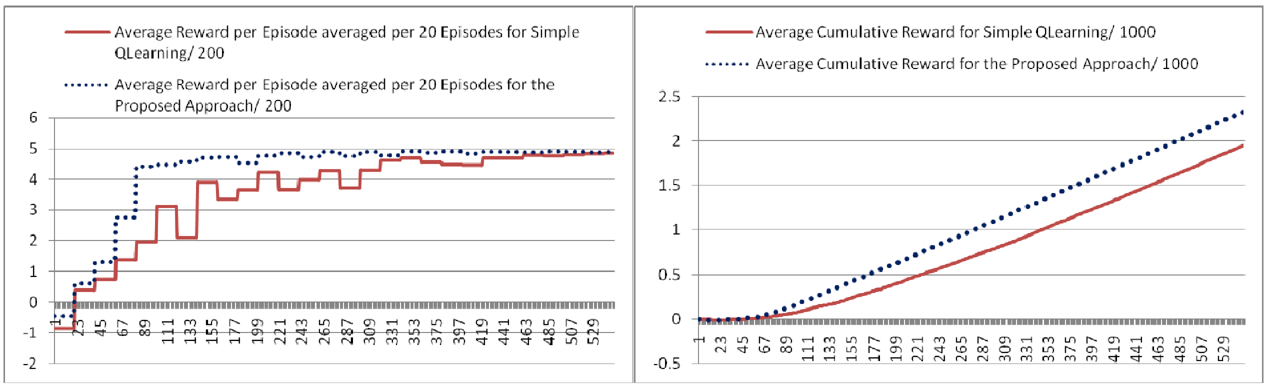
[15] B.Marthi, S.Russell, Automatice shaping and Decomposition of Reward function., In Proceedings of the 24th International Conference on Machine Learning(ICML), pages 601-608, 2007.

[16] J.Asmuth, M. L.Littman & R.Zinkov, Potential-based shaping in modelbased reinforcement learning. In Proceedings of AAAI conference on artificial intelligence. , 2008

[17] Marek Grze, Daniel Kudenko., Online learning of shaping rewards in reinforcement learning., Department of Computer Science, University of York, York, YO10 5DD, UK., pp. 541-550, 2010

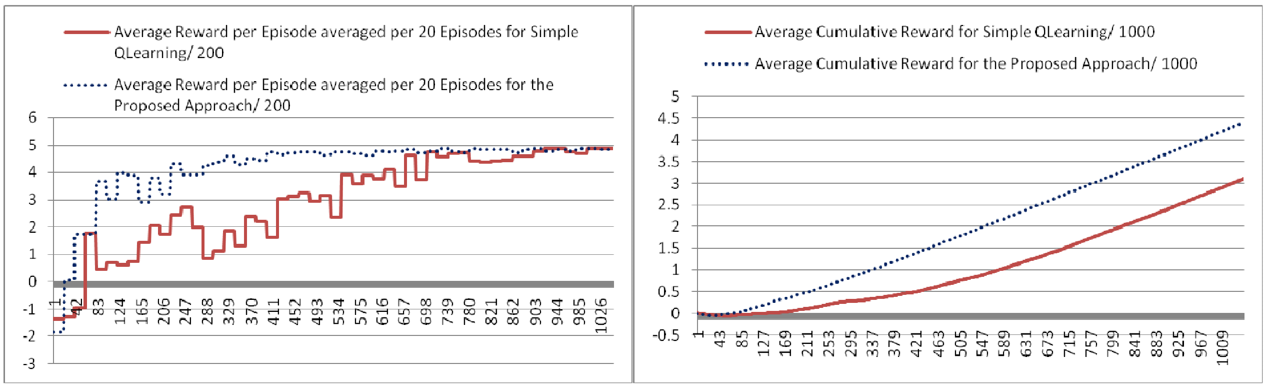
[18] P.Moradi, PhD Thesis, University Iran, 2011

[19] J. E. Freund, *Mathematical statistics*, 5th ed., Prentice-Hall, 1992



(a) (b)

Figure 3. The results of experiment on the Six Rooms environement. (a) The average reward per episode, (b) The average cumulative reward



(a) (b)

Figure 4. The results of experiment on the Complete Maze environement. (a) The average reward per episode, (b) The average cumulative reward