ISSN:0975-9646

Sequential Decision Task by Adaptive Reinforcement Learning Method

Ankur Verma
Department of Computer Engg, DYPIET Pimpri
SavitriBai Phule Pune University, India

Prof. Pramod Patil
Department of Computer Engg, DYPIET Pimpri
SavitriBai Phule Pune University, India

Abstract - **There is great interest in building intrinsic motivation for artificial intelligent systems using the reinforcement learning framework. There are many dynamic situations in which sequences of actions come with circumstances which are convenient. When the action is taken these consequences of actions can emerge at a multitude of time, and shall be concern with the strategies for both their short term and long term consequences. An approach is proposed here based on a model which requires constructing the model of state transaction and payoff probabilities. Such kind of tasks can be termed as a dynamical system whose behavior of decision changes over time under the impact of a decision taken for the action. This modeling of the behavior of decision of the system is greatly simplified by the concept of state and policy associates on action with that system states. It proposes methods for estimating optimal policy in the absence of a complete model of the decision tasks which are known as adaptive or decision model. Practical importance of adaptive method is; if this adaptive method can able to make improvement in decision policy sufficiently rapidly may be less.**

Index Terms –Reinforcement Learning, Decision policy, state-action function, Q-Learning, Temporal Difference Learning,

I. INTRODUCTION

There are many unsolved problems that computers could solve if the appropriate software existed. The tasks in which the consequences of an action can emerge at a multitude of times after the action is taken are only considered and such strategies for selecting an action on the basis of both there short-term and long-term consequences are interested. [3] Some of the situations in which system has to make sequence of actions to bring about circumstances favorable for their survival can be formulated in terms of the dynamical system. The behavior of such systems unfolds over time under the influence of decision makers actions.

Computer scientists are interested in developing devices and programs based on the life science by studding its engineering. [8] These studies are based on “synthetic learning” which has produced various methods and mathematical theories for pattern classification, prediction and adaptive control of dynamic systems. The problem still arises in synthetic learning is the nature of correspondence lies between the behavior of a system in classical conditioning experiment and mathematical theories and the computational procedures. Justification to these problems observed that the classical conditioning experiments are far from computationally trivial. These computational methods are also useful for adaptive prediction by making use of “Temporal Difference model of conditioning”.

Earlier dynamic programming was used in behavioral ecology for calculating information pertinent to decision making at each stage on the basis of information previously calculated from that stage to the task’s end but that worked backward. Than the TD procedure is overcoming the dynamic programming by accomplishing the same results by repeated forward passes through a decision tasks instead of back-to-front computation used by dynamic programming.

This TD model is defined under the reinforcement learning which defines that for achieving the goal the interaction with the problem of learning is framed to be straight forward. The “learner” and “decision-maker” are called the agent. The environment is the thing through which agent interacts from the outside. Through these interactions the agent continuously selects actions and environment replying to those actions and providing new behaviors providing the agent. The rewards garneted by the environment also tried to maximize over time by the agent using a special signal. A task is defined by a full specification of an environment, with single instance of the reinforcement learning problem.

Modeling the behavior of such system is greatly simplified by the concept of static. The state of a system at a particular time is a description of the condition of the system at that time that it is sufficient to determine all aspects of the future behavior of the system when combined with knowledge of the systems future input. Describing a decision task in terms of system states permits one to make a relatively simple statement of how action and state sequence determines the total amount of pay off an agent receives.

II. RELATED WORK

Some are the key techniques used over time:

- Supervised Learning: Is the simplest and most studied type of learning which has immediate feedback (labels provided for every input).
- Unsupervised Learning: No feedback (no labels provided).
- Reinforcement Learning: Delayed scalar feedback (a number called reward).

Esther Levin [1] et al. proposed a quantitative model for learning the dialog strategy. Also optimized the problem of dialog design by an objective function reflecting over different dialog dimensions required for a given application. It is shown that by state space, action set and strategy a dialog system can described as a sequential decision process. Additionally it is described that a dialog system can relate with stochastic model known as Markov decision process (MDP). It is also produced that a combination of supervised and reinforcement learning for effective use of training data available. Which is tested by them for learning in an air travel information system (ATIS). The experimental results presented in there paper show's a state space representation, a simple criterion, and a simulated user parameterization in order to learn automatically a complex dialog behavior

Larry D. Pyeatt [2] et al. presented an approach based on decision tree for the approximation of a function in the reinforcement learning. That is useful in scaling reinforcement learning problems with large number of states and actions. Also compared the decision tree providing better learning performance over neural network function approximation and solving large infeasible problems using the lookup table. These comparisons were held on the mountain car, pole balance problems and a simulated automobile race car.

Mircea Preda [3] et al. proposed a new method constructing decision trees of using reinforcement learning. This method is much efficient that it creates more and more decision trees because it learn this from the training set which constraint is to be tested first in order to classify a subset of examples. Through this the new method is suitable for solving problems where training set changes frequently and classification rules also changes over time. This method is also helpful where various constraints have various testing costs. Performance results and the summary of the features of the implemented algorithm are also concluded.

Lucian Busoniu [4] et al. proposed an approximate, model based Q-iteration algorithm relying on a fuzzy partition state space and discretization of action space. Using their assumptions on continuity of the dynamics and reward function, an consistent algorithm is shown, i.e., that as the approximation accuracy increases the optimal solution is obtained asymptotically. There experimental study indicated that "a continuous reward function is also important" for a predictable improvement for performance when there is increase in approximation accuracy.

Ioannis Partalas [5] et al. studied the pruning problem of an ensemble classifier using reinforcement learning. This contributed a new approach of pruning which takes the use of Q-learning algorithm for approximating an optimal policy for including and excluding all classifiers from ensemble. Comparisons made between the approaches of state-of-the-art pruning and the combination methods shows good results. Also an extension providing improvement over time is presented for certain performance critical-domains.

Mill´an-Giraldo [6] et al. made decisions regarding processing of incomplete and unlabeled incoming objects and also guessing the missing attributes value. These decisions are solved by including them in the training set and by asking regarding the class label at the relevant time. This was not possible in earlier for the complete set of attributes due to the dependency of the attributes. This made possible by the empirical results produced by the better framework of isolatedly. This framework uses the most of human effort and computer effort together by involving them in a behavioral manner.

Prof. Pramod Patil [8] et al. implemented the reinforcement learning algorithm on data streams of diabetes. There algorithm works for the data streams and differentiate the values for blood glucose level for insulin dose and takes the decision for next insulin dose. Depending on taken state and action the payoff is assign to the decision. That helped in classifying the data for doses of diabetes and also helped in making decisions at a particular time for giving specific quantity of dose. In comparison with other methods their proposed algorithm was faster. Also tests methodology are proposed on diabetes data set.

Stefanos Doltsinis Pyeatt [9] et al. through this paper approached ramp-up as a sequential adjustment and tuning process to a desirable performance manufacturing system in the fastest possible time. Focusing on development of a Markov decision process (MDP) model for ramp-up of production stations and enabling its analysis. Their aim is also to capture the cause and effect of the station's response to improve the effectiveness of the process and operator adaptation or adjustment of a station. The investigations are made for the Q-batch learning algorithm's application combined with an MDP model. There approach was applied to a station of highly automated production where sundry ramp-up processes are carried out. A policy is also learned that are applied and compared against previous ramp-up cases.

Jihye Bae [10] et al. introduced a temporal difference algorithm for estimating a value function in reinforcement learning. Correntropy is a robust cost function used by a kernel adaptive system. To find a proper policy the integration of there algorithm with Q-learning is done. Test methods are proposed with synthetic problems and quantified its robustness. Applications are also performed of the same algorithm over a monkey's neural states in reinforcement learning brain machine interface. The results from this observation were potentially beneficial.

III. REINFORCEMENT LEARNING

Reinforcement learning (RL) has become most active area of research now days. The objective of reinforcement learning is to getting maximizing rewards by mapping the state with the actions. In RL an autonomous agent follows a trial and error process in order to reach its goal by learning optimal action to perform in each state. The flow diagram of the RL is shown.

Methodology:

Step 1. The agent chooses an random action on each state.
Step 2. This action may take the agent in new state.
Step 3. In new state the agent will get reward/ penalty.
Step 4. Repeat the steps (1, 2, 3).

Agent eventually learns the action to obtain maximum reward.

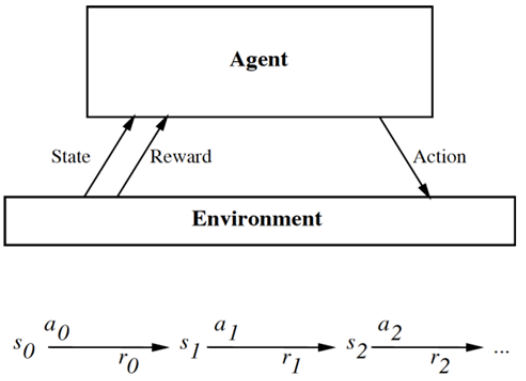

Agent
State
Reward
Action
Environment
$s_0 \xrightarrow[r_0]{a_0} s_1 \xrightarrow[r_1]{a_1} s_2 \xrightarrow[r_2]{a_2} \dots$

Figure 1. Reinforcement Learning Process

Nomenclature Section

No.	Symbol	Meaning
1	s’	State of agent at time t.
2	a’	Action taken by agent at time t.
3	π	Policy taken by agent
4	Q	Action value function
5	ε	Exploration chance
6	α	Learning Rate
7	γ	Discount factor

Elements of Reinforcement learning:

A. ***Policy:*** A policy is the core of reinforcement agent in the way that it is alone to determine the behavior of the agent. The policy only decides the action which is to be taken by the agent by the previous states.

B. ***Reward Function:*** Reward is defining the goal to the agent. Reward maps the action of previous state with a single number. The main goal of the agent to maximize the total reward over long run. The reward only decides that weather the policy is good or bad in order to maximize reward.

C. ***Value Function:*** Value function describes which policy is good for long run. Value of a state defines the amount of total reward an agent can look for to assemble over the future, initial from the state.

D. ***Model of the environment:*** This defines the behavior of the environment. That is for a given state the agent may predict the next state and the reward for the next state.

IV. Temporal Difference Learning

TD learning is the combination of Monte Carlo and dynamic programming. TD methods are capable of learning directly from the raw experience neither requiring the environment dynamics model. [4],[5] TD estimation algorithm used in reinforcement learning is capable of predicting a measure of total amount of expected reward as well as the other quantities over the future.

Simple every - visit **Monte Carlo method**:

$$V(s_t) \leftarrow V(s_t) + \alpha\, [\underset{\downarrow}{R_t} - V(s_t)]$$

Target: the actual return after time *t*

The simplest **TD method**, TD(0) :

$$V(s_t) \leftarrow V(s_t) + \alpha\, [\underset{\downarrow}{\underline{r_{t+1} + \gamma V(s_{t+1})}} - V(s_t)]$$

Target: an estimate of the return .

***ON** and **OFF** policy of Temporal Difference:*

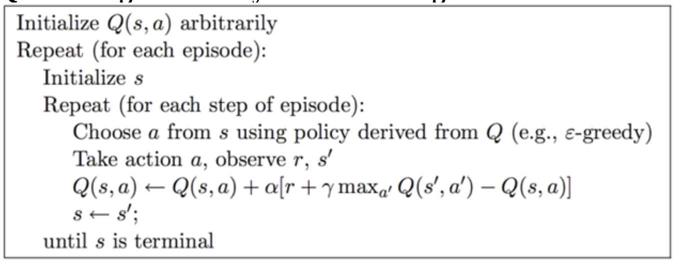
A. Sarsa: On-Policy TD Control:

Sarsa concentrates in the limiting to the greedy policy. Also concentrates on the probability of an optimal policy and action-value function until all state-action pairs are traversed infinitely as shown in algorithm.

B. Q-Learning: Off-Policy TD Control:

Q-learning is a model which is free from reinforcement learning technique. It uses an action-value function for learning that ultimately gives the look for an optimal policy for actions followed by a given state-action and also that given action.

Q-Learning: Off-Policy TD Control algorithm



```
Initialize $Q(s,a)$ arbitrarily
Repeat (for each episode):
   Initialize $s$
   Repeat (for each step of episode):
      Choose $a$ from $s$ using policy derived from $Q$ (e.g., $\varepsilon$-greedy)
      Take action $a$, observe $r$, $s'$
      $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$
      $s \leftarrow s'$;
   until $s$ is terminal
```

Exploration / Exploitation

The basic of exploration and exploitation is that exploration maximizes the reward and exploitation maximizes for being well for long time. It is very important that the agent does not simply follow the current policy. When learning Q (off-policy learning).The reason is that you may get stuck in a suboptimal solution. I.e. there may be other solutions out there that you have never seen. Hence it is good to try new things.

One can trade-off memory and computation by cashing (s,s’,r) for observed transitions. After a while, as Q(s’,a’) has changed, you can “replay” the update. One can actively search for state-action pairs for which Q(s,a) is expected to change a lot (prioritized sweeping). One can do updates along the sampled path much further back than just one step (learning).

V. Dynamic Programming

“Dynamic Programming" (DP) refers to a grouping of algorithms that can be used to compute optimal policies given for a perfect model of the environment in a Markov decision process (MDP). [9] The key idea of DP is to use the value functions for organizing and structuring the search for good policies. Classical DP algorithms are having limited utility in the reinforcement learning because of their assumption of a perfect model and their great computational expense, but these algorithms still are very important in the perspective of theory. DP makes available an essential foundation for understanding the methods. Although, all of these methods can be viewed as attempts to achieve much the same effect as DP, only with the less computation and also without assuming a perfect model of the environment.

VI. Proposed work

A. *Objectives:*

- To develop adaptive prediction method using temporal difference prediction.
- Proposing methods for estimating optimal policy in the absence of a complete model of the decision task.
- Making application easier and more straight forward for the ease of working.
- To develop off-policy (Temporal Difference control) using Q-learning.

B. *Work flow of Proposed system:*

Step 1: An agent begins exploring its environment, trying new actions as it goes interacting with a system at time step t=0 and continues for infinite number of steps.

Step 2: Using discount factor γ, the measure of the total amount of payoff the agent will receive over this infinite time period is

$$\{R_t = r_1 + \gamma r_2 + \gamma^2 r_3 + \ldots + \gamma^n r_{n+1} + \ldots\} \qquad (1)$$

Step 3: When $0 < \gamma < 1$, the power of γ weighting the payoff is decreasing sequence so later payoffs are weighted less than earlier ones.

Step 4: If $\gamma=1$ sum of all payoffs is received.

Step 5: Discount factor adjusts the degree to which the long term consequences of actions must be accounted for decision tasks.

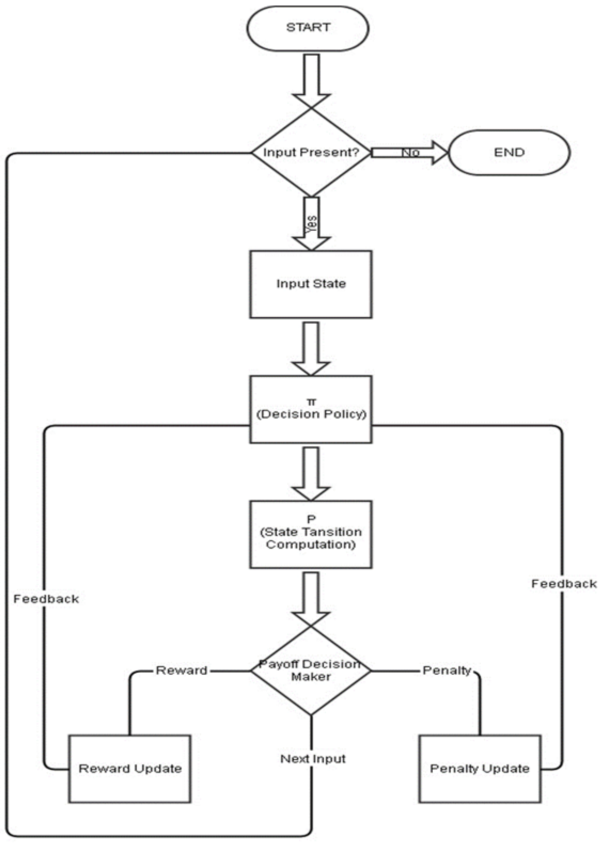


Figure 2. Work Flow of Proposed System

In Figure 2. Workflow of the system is shown and the steps for the system includes.

C. *Algorithm introduced:*

- $Q(s, a) = r(s, a) + \gamma max_{a'}(Q(s', a'))$
- $r(s, a)$ = Immediate reward
- γ = relative value of delayed vs. immediate rewards that is (0 to 1)
- s' = the new state after action a
- a, a' : actions in states s and s' , respectively
- Selected action:

  $\pi(s)=arg\ max_a Q(s, a)$

D. *Experimental Result:*

As the agent when start interacting with the system it will get the reward (1) and the penalty (0) through the output of the system. To deal with environment, we need to maximize expected future discounted reward γ.

The reward R(x,a) the robot gets is simply the number of pixel on the screen it has moved (positive values for movement to the right and negative values for movement to the left). The goal is that to move the robot to the right the faster possible.

As the expected return $E\{R_t\}$ over time t through reward and penalty when gets maximum (that is when compared with the previous payoff), then the expected result also gets max.

The expected return of the agent is shown in the Figure 3.

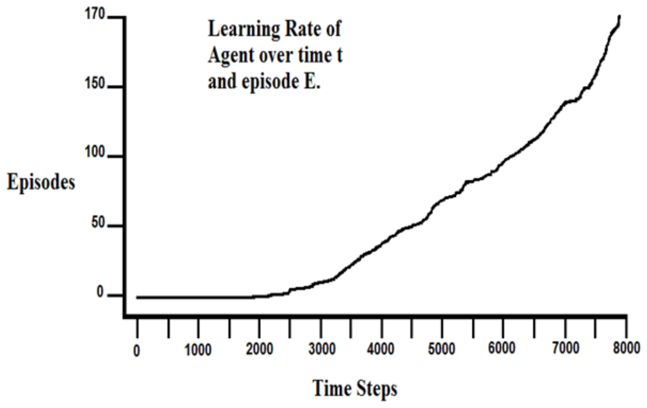


Figure 3. Learning Rate of the Agent

VII. Conclusion

Reinforcement learning addresses a very broad and relevant question: How can we learn to survive in our environment?

This paper presents the development of an adaptive prediction method through the temporal difference prediction method. Also some of the methods are introduced for estimating optimal policy in the absence of a complete model of the decision task. We have looked at Q-learning, which simply learns from experience. So the applications can make easier and straight forward for of use. In this project it is tried to develop an off-policy TD learning approach by using the Q-learning algorithm. There are many extensions to speed up learning. And also evaluation of the learning rate of the agent is done by the result.

ACKNOWLEDGMENT

I would like to take this opportunity to acknowledge the contribution of certain people without which it would not have been possible to complete this paper work. I would like to express my special appreciation and thanks to my advisor Professor Pramod Patil, you have been a overwhelming mentor for me.

REFERENCES

1. Esther Levin, Roberto Pieraccini and Wieland Eckert "A stochastic model of human-machine interaction for learning dialog strategies" *IEEE Transactions on speech and audio processing*, VOL. 8, NO. 1, JANUARY 2000
2. Larry D. Pyeatt and Adele E. Howe "Decision tree function approximation in reinforcement learning" *CiteSeer* , 07/2001
3. Mircea Preda "Adaptive building of decision trees by reinforcement learning" *Stevens Point, Wisconsin, USA* ©2007 , ISBN: 978-960-6766-01-5
4. Lucian Busoniu, Damien Ernst, Bart De Schutter, and Robert Babuska "Consistency of fuzzy model-based reinforcement learning" 2008 *IEEE International Conference on Fuzzy Systems.*
5. Ioannis Partalas, Grigorios Tsoumakas and Ioannis Vlahavas "Pruning an ensemble of classifiers via reinforcement learning" *Neurocomputing - IJON* , vol. 72, no. 7-9, pp. 1900-1909, 2009
6. M. Mill´an-Giraldo, V. Javier Traver, and J. Salvador S´anchez "On-line classification of data streams with missing values based on reinforcement learning" *Springer-Verlag Berlin, Heidelberg* ©2011 , ISBN: 978-3-642-21256-7
7. Sander Adam, Lucian Busoniu, and Robert Babuska "Experience replay for real-time reinforcement learning control" *IEEE Transactions on Systems, man and Cybernetics — PART C: APPLICATIONS AND REVIEWS,* VOL. 42, NO. 2, MARCH 2012
8. Prof. Pramod Patil, Dr. Parag Kulkarni, and Ms. Raczhana Shirsath "Learning and sequential decision making for medical data streams using rl algorithm" *International Journal of Research in Computer and Communication Technology,* Vol. 2, Issue 7, July-2013
9. Stefanos Doltsinis, Pedro Ferreira, and Niels Lohse "An MDP model-based reinforcement learning approach for production station ramp-up optimization: q-learning analysis" *IEEE Transactions on Systems, man and Cybernetics: Systems,* VOL. 44, NO. 9, SEPTEMBER 2014
10. Jihye Bae, Luis G. Sanchez Giraldo, Jose C. Principe, Joseph T. Francis "Correntropy kernel temporal differences for reinforcement learning brain machine interfaces" 2014 *International Joint Conference on Neural Networks (IJCNN)* July 6-11, 2014, Beijing, China