Indian Journal of Science and Technology, *Vol 10(1), DOI: 10.17485/ijst/2017/v10i1/109385 January 2017* *ISSN (Print) : 0974-6846* *ISSN (Online) : 0974-5645*

Reinforcement Learning Algorithms: Survey and Classification

N. R. Ravishankar[1] and M. V. Vijayakumar[2]

[1]Dr. Ambedkar Institute of Technology, Bangalore – 560056, Karnataka, India; Indianrravi98@yahoo.com
[2]Department of Computer Science New Horizon college of Engineering, Bangalore – 560103, Karnataka, India

Abstract

Reinforcement Learning (RL) has emerged as a strong approach in the field of Artificial intelligence, specifically, in the field of machine learning, robotic navigation, etc. In this paper we try to do a brief survey on the various RL algorithms, and try to give a perspective on how the trajectory is moving in the research landscape. We are also attempting to classify RL as a 3-D (dimensional) problem, and give a perspective on how the journey of various algorithms in each of these dimensions progressing. We provide a quick recap of basic classifications in RL, and some popular, but old, algorithms. This research paper then discusses some of the recent trends; and also summarizes the entire landscape as can be seen from a bird's eye view. We provide our perspective in saying that Reinforcement learning is a 3D problem and conclude with challenges that remain ahead of us. We have deliberately kept any deep mathematical equations and concepts out of this paper, as the purpose of this paper is to provide an overview and abstractions to a serious onlooker. We hope this paper provides a brief summary and trends in RL for researchers, students and interested scholars.

Keywords: Artificial Intelligence, Cognitive Search, Game Theory, Machine Learning, Reinforcement Learning

1. Introduction

Reinforcement Learning (RL) falls into realms of machine learning, wherein the agent learns how to act by the rewards or punishments that it gets from the environment[1,2]. A reward could be a point scored, as in the game of table-tennis, and a punishment could be negative score or an opponent getting a point over the agent. RL agent learns by experiencing in the environment – what is a good action and what is not. The goal is to perform actions that gives it best long-term and short term rewards and lessen punishments. RL has very good connections with human and animal learning behaviors. This section briefly touches on the different types of reinforcement learning algorithms, classifications, their benefits and limitations. This section gives a recap of hitherto known and popular learning algorithms. However we don't intend to explain every algorithm listed here. Instead, we give pointers to the legacy algorithms and their classification, while do more briefing on the latest algorithms.

1.1 Model-Based and Model-Free RL

Under Reinforcement Learning it is very well known, there are 2 broad classifications as Model-based and Model-free RL [3]. Model-based RLs have the knowledge about the environment in which the agent acts, and about the agent, per se, as well. The state transition-action mapping combined with the reward model is available a-priori. That means the agent knows the environment in which it is acting; it knows the state transitions very well –that is P(s '|s, a). It also has the reward matrix available. The agent's job is to find an optimal policy from a given state to the goal-state – that is expected total reward. Whichever path has the best overall utility that is considered as the optimal policy. Some of the governing equations in model-based RLs are:

Utility Eq.: $U = E\,[\Sigma \gamma t R(St)]$

Bellman's Eq.: $U^{\pi}(s) = R(s) + \gamma\ \Sigma s' P(s'|s, \pi(s))\ U^{\pi}(s')$

Where γ is the discount factor.

While in model-based RLs, the environment could

** Author for correspondence*

be fully observable or partially observable. There can be environments that are Markovian Decision Problems (MDP) or non-Markovian. Thus we have a combination based on "observability of environment" and "behavior of the environment". To be more specific, we can say a model is MDP, or Partially Observable MDP (POMDP), or a non-MDP, etc.

Many algorithms have evolved in the last couple of decades; and some of the most popular, legacy, model based RL algorithms are[1, 4,5]:

- Policy evaluation
- PEGASUS
- RL for POMDPs

Model-free RLs do not have any knowledge about the environment in which the agent acts[3].The agent only acts in any given state, but doesn't know where it leads to. So they rely on the instantaneous reward obtained by taking an action in a state, and then evaluate the utility of that state. Utility is defined as the expected total reward from the current state to the goal-state. Once all the paths are traversed they then evaluate the utilities of each of the states experienced. As always, the action sequence that yields the maximum utility is considered as the optimal policy. So they learn the utilities by trial-and-error method. We can imagine this as a situation giving directions to a blind man: "take a right, a left, and then right, and then you hit the door!!"

The governing equations remain the same as of Model-based RL algorithms.

Some of themost popular, legacy model-free RL algorithms are[1, 6, 7, 8, 9]:

- Q-learning
- SARSA
- Dyna-Q
- Temporal Difference (TD)

To summarize, model-based RL algorithms look for learning the best policy that leads to the goal. On the other hand, model-free RL algorithms look for the best overall utility that of course, lead to the goal. In fact, Nathaniel D. Daw[10], in his paper puts the difference between Model-free and Model-based as similar to goal directed and habitualinstrumental behaviors.

2. Problems that Plague Mode-based and Model-free RLs discussed thus far

- Memory intensive: Model-based algorithms need a look-up table for storing state-transition maps or probabilistic state transitions P(s, a,s') in case of MDP or POMPD environments.
- While model-free RLs need a look-up table to store state-action pairs Q(s, a).
- Long convergence time: Both the models need long time to converge to an optimized policy or utility.
- Cannot be adjusted in the event of a goal change, reward change or state change[10].
- Another subtle, yet important aspect of these algorithms are they are greedy in nature – they try to find a policy or a path that can maximize the reward. But in doing so they often shun treading some of the paths which tend to have lower rewards initially (short term), but may have better long-term rewards. This is akin to human's taking short-cuts to reach their goal faster – getting rich overnight, lawn chair millionaire!! But it may turn out that the so called optimal policy chosen may be far from reality!

Value Function Approximation: When the state space is very large (infinite), or when it is a continuous space (e.g. helicopter navigation), many of the methods discussed above will not be able to converge to optimal policy within reasonable time. Thus emerged a powerful way of generalizing (or abstracting out) the state space, called as value function approximations. George Konidaris, et al11, state this as "the value function is represented as a weighted aggregation of a set of features, computed from the state variables".

Some of the different types of function approximations could be listed here11-14:

- Linear function approximation
- Neural network
- Fourier based
- Nearest neighbor based.

There have been several good works in this area over the years. But one thing that strikes is all these methods are bit mathematically intensive, and getting a good representation of the value function approximation is a big challenge. Also some of the researchers have opined that learning gets unstable when neural networks is used to represent the Q values[12].

2.1 Relational RL (RRL)

Improvisations to model-free RLs, more specifically Q learning, came in the form of Relational RRL. To overcome the curse of dimensionality, RRL brought a new light, with possibly overcoming the challenges of function approximations. Tadepalli, et al[15], argue that function approximations do not work well in relational domains. RRL brings in an insightful way of exploiting the relation between objects, and thus creating a relation based Q-values (in direct alluding to Q learning). It is also argued that such an RRL can be more effective than function approximations such as neural n/w based or Fourier based etc.

There are several approaches that have been devised in RRL – TG algorithm, Gaussian based algorithm[15], and instance based RRL. Relation b/w states could as well be the distance b/w states or nearest neighbors etc.

Interestingly Tadepalli, et al15, propose, very emphatically, RRL as the panacea for machine learning in AI, in general!However, we will see in the next sections how the algorithms evolved and what challenges still remain.

2.2 Active and Passive RLs

Another classification of RL algorithms is based on the way learning happens. Learning can be achieved either by observing the utilities as the agent interacts with the world and while having fixed policy (Passive RL) OR by acting in the world (Active RL), and learning which actions are best in a given state[1].

In passive learning the agent already knows what action it has to take in any given state. So it has to only find out what is the reward, and hence the utility. While in active learning, the agent doesn't have any fixed action in any state. It has to find out what action it can take, get the reward, and then estimate the utility.

An agent can be any combination of Active/Passive RL and model-based/model-free. This combination, however, does not give it any distinction or significance with respect to convergence or learning challenges discussed earlier.

Some of the well-known Active RL algorithms are Q learning, RMax, ε-greedy, etc. Similarly, Monte-Carlo Direct estimations, Adaptive Dynamic Programming, Temporal Difference learning are some of the Passive RL algorithms.

2.3 Exploration v/s Exploitation

A very important consideration with respect to approaches in policy evaluation, or utility evaluation, etc. is exploration v/s exploitation. This is already hinted undersec 1.1, under "problems plaguing RL". Consider a state-space being very large and the agent has still not explored all possible routes yet. But with whatever exploration it has so far done it has found an optimal route – it has exploited an optimal route within its experienced landscape. However, it is possible that there may exist better routes in the unexplored routes. So to avoid being stuck to the "rut", exploration of new routes is a very important consideration for an agent. Many exploration mechanisms have been evolved in the last as many years[1, 16].

- GLIE
- Policy gradient
- Least Mean squares

However all the above approaches stated so far suffer from a major setback when the state-space dimension is very large – the curse of dimensionality.

3. Some of the Recent Developments in RL

3.1 Topological Q-learning

It uses the topological ordering of the states as and when the agent experiences the environment. This algorithm consists of 2 phases called task learning and exploration optimization. In "task-learning phase the agent builds the topological ordering of the states, and in the exploration-optimization an internal reward function is used to guide the exploration"[17]. So this algorithm tries to achieve a guided exploration based on an intrinsic or internal reward mechanism. Guided-exploration apparently sounds better than any of the randomized exploration strategies – because there is a clear purpose behind exploration. Topology of states is built using Instantaneous Topological Map(ITM) model. State values are updated in a backward fashion, but the update is restricted only to the states that have been visited/traversed by the agent thus far – and not the entire network of states. This looks to be a reasonable principle, as updating non-traversed states' value may result in over or underestimations.

Also, the internal reward function will give a negative reward for paths that lead to no change in policy values. This means when a path does not lead to an increase in value, it instigates the agent to explore non-treaded paths – thus the rationale for exploration is sound.

3.2 Epoch-Incremental RL

The algorithm evaluates and improves an agent's policy by combining conventional TD(0) or TD(ʎ) and the novel Epoch mode. TD(0) or TD(ʎ) will build the environmental model (by directly experiencing the environment), and after reaching the terminal state it gets into the Epoch mode. In this mode, the distances from the explored nodes to the terminal state is computed and the shortest distance from any initial state S0 to the terminal state is adjudged the best policy. So the Epoch mode, somewhat, is akin to a Breadth-First-Search algorithm[18].

3.3 Multi-scale RL

Makes an abstract of the state-space graph by using certain mathematical functions. It creates levels of abstractions as shown in Figure 1. So on the reduced abstraction-map, the action selection is performed using multi-scale Softmax selection[19].

This can be comprehended as akin to simplified mathematical modeling of real-world problems. As with any such simplified models there do exist problems of over-simplification, ignoring nitty-gritty's and it might lead to bad performance in real-world.

However, the advantages of simplified models can never be overlooked nor underestimated. Striking a balance between over-simplification and retention of abstraction seems to be the key trick in such models.

3.4 Hierarchical RL (HRL)

A larger goal is sub-divided into a hierarchy of sub-goals. Each subtask could still be decomposed to sub-, tasks, and the lowest level of tasks, typically, are primitive actions[20].

This approach shifts the focus of RL problem form being a state-to-state or action-to-action oriented towards sub-goals to larger-goal oriented. HRL can be summarized as an approach that abstracts and divides the state space into key landmarks, from start to the final goal. Thus tackling large dimensions of state-space could be easier.

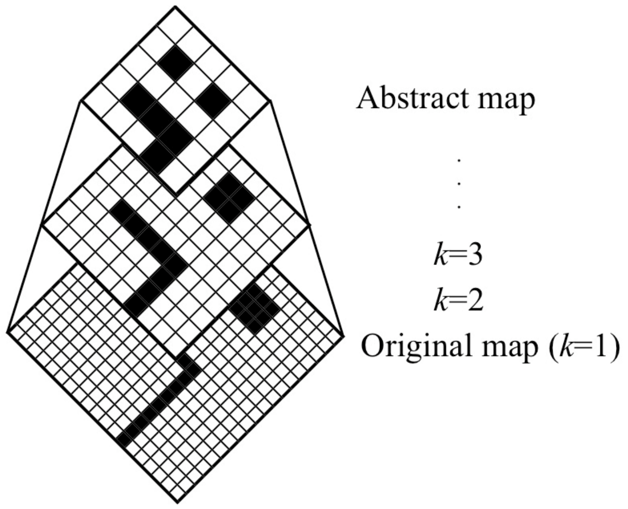

Figure 1. Multi-scale Value Functions. Courtesy[19].

This algorithm can be drawn as parallel to Work Breakdown Structure, which is followed in many of the software project management processes and elsewhere [21].

3.5 Deep RL

This algorithm, from DeepMind labs[22], was evaluated on a game called Atari 2600, and several other computer games. The same, single architecture was used in all these games and this algorithm showed much better results than any other RL methods available!

Deep RL is a combination of Deep-learning (convolutional neural network) and Reinforcement learning (Q-learning).Deep learning enables high level abstraction of data by multi-layer graph processing. By this we can get to know what data is useful for the current objective/Goal. While Q-learning evaluates the best utility.

The algorithm uses 2 key ideas –

- Experience replay mechanism that removes the correlations between consecutive observations.
- Iterative update mechanism that adjusts Q values towards the target value. But this is done at some specified periods only; so that correlation with the target is reduced.

Experience replay mechanism is a biologically inspired mechanism. A stunning aspect of this algorithm is its ability to learn and play 40+ different games, and with better dexterity than humans or any other RL algorithms, so far!!

3.6 RL in the realms of Game theory

Gaming theory finds its applications in several competitive market scenarios such as trading of goods and services[23]. Gaming theory proposes a unique challenge when it comes to autonomous learning aspects. The primary challenge being that it is a non-stationary environment. As many agents interact with the environment the world changes at every instance– complexity increases. Secondly an agent will interact with other agents too (be it cooperative or non-cooperative), so its action selection might be constrained, have to be re-planned, probably at every instance. It can be cited from bowling and veloso [24] that "an agent's optimal behavior depends on the behavior of other agents, and vice versa". A simple example being "Rock Scissor Paper" game played by kids.

Gaming problems can be broadly divided as Stochastic and Matrix games[24]. In both these categories there is a set of actions available to each agent, and then there is a joint action space, as every agent's action has an impact on other agents' actions. However in stochastic games, the agent's policy shall also be stochastic – which means it's a probability distribution of states-to-action. Key aspect of all gaming theories is the "Nash equilibrium (NE)"25 – which is a collection of strategies of all players/agents. Every player/agent strives to its equilibrium – in other words, best possible outcomes for themselves; but that which cannot be achieved independent of other agents! Now this is what we need to remember from the game theory perspective.

Consequently, we can see two orthogonal dimensions in solving gaming problem, one arising out of gaming theory to achieve Nash equilibrium; and the other arising out of Reinforcement learning. Both these aspects have to amalgamate in such a way as to get the best possible outcome for the agents involved in the game.

Bowling and Veloso's paper has examined a number of algorithms that solve multiagent RLs for Stochastic games, and it is noted that most of these algorithms, namely Shapely, PollatschekandAvi-Itzhak, Van der Wal, Fictitious play, etc make use of variations of Temporal difference learning aspect of RL.

Hu and Littman[26] use a multi-agent Q-learning method under stochastic framework to obtain Nash equilibrium. Here each agent maintains two tables of Q values, one for its own Q values while the other table for Q values of the other agent, obtained by observing the other agent's action. The update of Q values thus has two equations:

$$Q^1_{t+1}(s, a^1, a^2) = (1- \alpha_t)\, Q^1_t(s, a^1, a^2) + \alpha_t\, [r_t^{\,1} + \beta\pi^1(s')\, Q^1_t(s')\, \pi^2(s')] \qquad (1)$$

In order to find the other agent's policy this agent must observe his actions, and learn as

$$Q^2_{t+1}(s, a^1, a^2) = (1- \alpha_t)\, Q2_t(s, a^1, a^2) + \alpha_t\, [r_t^{\,2} + \beta\pi^1(s')\, Q^2_t(s')\, \pi^2(s')] \qquad (2)$$

Thus it needs to maintain two table of Q values! Also, it is assumed that the game has a unique equilibrium; which however is not true in many of the stochastic games. There could exist multiple equilibrium states.

Vishnu and Tapas[27] propose a RL algorithm to solve an n-player matrix game, of course a non-cooperative one. The paper describes a new approach to obtain Nash equilibrium of n-player matrix games using a differentiated value-iteration algorithm one for pure strategy and the other for mixed strategy. R-values are selected based on 2 criteria: Greedy action selection for pure strategy Nash Equilibrium[26] and a probabilistic action selection for mixed strategy.

Though there is no mathematical proof for the convergence of their learning algorithm, they tend to give reasonably good results (equilibrium attainment), and rewards for the agents. However, the NEs did differ when they compared their results with that of an openly available software called GAMBIT.

4. The Journey so Far

We would summarize Reinforcement Learning as a 3 dimensional problem, and try to get a bird's eye view of the journey in all 3 dimensions, and represent it as in Figure 2. It would be, at the least, incorrect to say that our subjecting of RL to a 3-D problem is the best way to summarize the RL journey. But rather we attempt to give a perspective as to how we can look at this learning mechanism, and how the journey proceeded over the years in these 3 dimensions. We hope this can show some seeds of thoughts in a serious onlooker:

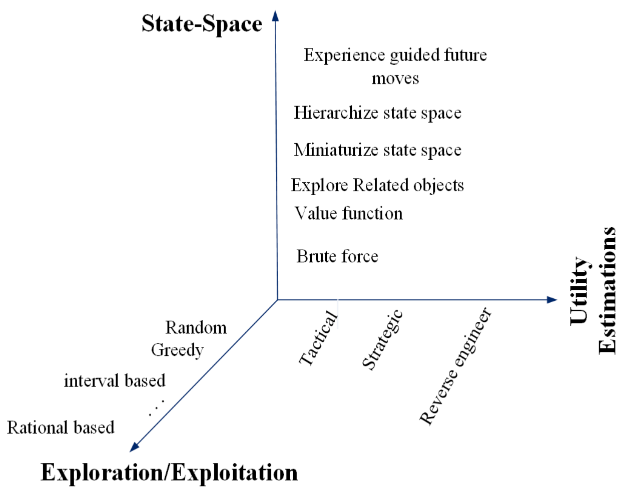

Figure 2. The 3-D view of RL problem.

4.1 State-space Dimensionality

The journey can be seen as a progression from brute force methods(explore all states) to

- Making approximations of the state-space and exploring to
- Choosing only related state-spaces(RRL) to
- Creating thumb-nail version (similar to mathematical models of a natural Phenomenon) of the state space and thus shrinking the whole environment itself, to
- Using past experience to guide the next moves

4.2 Utility Estimation Dimensionality–Direct Estimation, TD Estimation, Epoch Estimations, etc.

In this dimension, the journey can be summarized as one from a tactical to strategic/goal oriented. Direct estimations and TDs are mostly tactical, where they are more worried about the current state rather than the end goal. So short-term changes and deviations seem more probable. When it comes to Epoch learning we see the utility estimations are End-goal oriented. Thus they also factor the long term implications of taking an action – that's what we call Strategic.Another important aspect in the utility dimension, in the wake of absence of any rewards,or very few if any, from the external world a mechanism called "Inverse Reinforcement Learning" has been proposed[28] to find a suitable reward function. So here we have an action policy, and need to determine the rewards for such action policy – reverse engineering!

4.4.3 Exploration/Exploitation dimensionality

To arrive at an optimal policy within the purview of above 2 dimensions (State space and Utility), we need to have a right balance or make right choice of exploration v/s exploitation.

The initial days of GLIE, random exploration, or time-based exploration, were focused on ad-hoc ways to take-time off the usual rut, and do some exploration. It then evolved to greedy-techniques, to interval based techniques[29]. But later techniques like Topological RL and Curiosity factors are more rational in taking the un-treaded path.

So the journey can be summarized as a one from random or ad-hoc to rational based.

5. Conclusions

Applications of RL have been diverse and growing in the recent past. From lab experiments as hover-crafts to game playing to operations research, it has found its place, and seem to be growing.

In commercial space, such as power and energy[30,31], RL has found applications relating to dynamic pricing[23] of electric energy in the micro grid. B-G Kim, et al, have shown by numerical results how RL has helped improve dynamic pricing and energy scheduling in a multi-customer environment(pricing is a factor of electricity cost and the load demand.)

Interestingly, gaming theories have also adopted RL methods to obtain optimal solutions. It needs a special mention about gaming theory, as this domain has a more complex problem to solve, and the problem representation itself adds another dimension – for e.g. Matrix games, Stochastic games, and the solution lying in finding Nash Equilibrium state. Considering the fact the gaming theory have huge impact in operations research, stock trading, goods trade, etc. RL provides a very huge opportunity for commercial success. RL has seen very good real world success in solving MDPs[32, 1, 33]. Deep RL algorithm [22] seems to be more promising and provokes thoughts as "can it be a panacea?!"

Another interesting and upcoming area of research is the confluence of RL and Cognitive and neurosciences area [34].Recent studies in Psychology and Neurosciences about the dopamine response of the human brain to stimulus, and the brain's action choice has been studied in relation to RL algorithms[10, 35, 36] – more so with respect to model-based and model-free RLs. Though the research in this aspect of cognitive search and planning is still in infancy, there are parallels drawn to attribute the dopaminergic response and search planning of the brain akin to model-based RLs. Thus RL holds a potential research problem and a commercial utility in more than one ways.

Now getting backto the 3-D view of RL problem, as can be seen in the previous section there has been significant evolution in all the dimensions of the RL problem. We are seeing more optimizations and strategy based approaches, moving away from Brute-force or ad-hoc approaches. The rationale are stronger and approaches are more compulsive. However, it apparently seemsstate-space dimensionality has evolved a lot more and a lot faster than compared to the other two dimensions. Make no mistake, despite there being several algorithms in, for e.g. Utility estimation, they all fall into similar categories. As an e.g., there are several algorithms propounded pertaining to the inverse RL. But they all solve the same problem in a different way. So they don't carve out a new category on the evolution scale.

It would be interesting to see if future research can balance all the 3 orthogonal aspects and achieve, what can be fancily called, "Learning Equilibrium".

6. Acknowledgment

I would like to thank my guide Prof Dr. Vijayakumar M V, who guided me all through, and infused lot of confidence and encouraged me all through my work. Equally important are the works from various authors across the globe, who made their works available as conference papers, journals, thesis, books, lecture slides, etc. Without access to these my knowledge would never have got elevated. Thanks to their immense works and sharing their thought process. Last, but not least, the various technological forums, journals, reviewers and conference facilitators who are responsible in ensuring confluence of all the knowledge in this world, at all times.

7. References

1. Russell S and Norvig P. Reinforcement Learning: Artificial Intelligence A Modern Approach. 3rd edition. New Delhi, India: Dorling Kindersley; 2014. ch. 22, sec. 2-5, p. 872–902.
2. Sutton R, Barto A.Reinforcement Learning: An Introduction. Cambridge, U.S: MIT Press; 1998.
3. Quentin JM, Huysa B, Anthony C, Peggy S. Reward-Based Learning, Model-Based and Model-Free. Encyclopedia of Comput. Neurosci. 2014.
4. Jaakkola T, Singh SP, Jordan MI.Reinforcement Learning Algorithm for Partially Observable Markov Decision Problems.Adv. I n Neural Info. Proc. Syst. vol. 7. Cambridge, MA: MIT Press; 1995.
5. Ng AY, Jordan MI.PEGASUS: a policy search method for large MDPs and POMDPs. Proc. of the 16th conf. on Uncertainty in Arti. Intell. San Francisco: 2000.
6. Watkins CJ,Dayan P.Q-learning. Mach. Learn. vol. 8.1992.p. 279–292.
7. Sutton RS.Dyna, an Integrated Architecture for Learning, Planning, and Reacting. ACM SIGART Bulletin.1991; 2(4):160–3.
8. Sutton RS.Learning to predict by the methods of temporal differences. Mach Learn.1998;3(1): 9–44.
9. Tesauro G.Temporal difference learning and TD-Gammon. Commun. ACM. 1995; 38:58–68.
10. Daw ND.Model-based reinforcement learning as cognitive search: Neuro computational Theories.Center for Neural Sci. and Dept of Psychology. New York University; 2012.
11. Konidaris G. Osentoski S, Thomas P.Value Function Approximation in Reinforcement Learning Using the Fourier Basis. Proc. 25th AAAI Conference on Artificial Intelligence. 2011.p. 380–5.
12. J. Tsitsiklis, Roy BV.An analysis of temporal-difference learning with function approximation. IEEE Trans. Automatic Control. 1997; 42(5).
13. S. Mahadevan, Amarel SM.Automating value function approximation using global state space analysis. Proc. of 20th Nat. Conf. on Arti. Intell. and the 17th Innovative Appl. of Arti. Intell. Conf. (AAAI-05). Pittsburgh, U.S., 2005.p. 1000–05.
14. Lagoudakis MG, Parr R.Least-squares policy iteration. Journ. of Mach. Learn. Res., 2003; 4: 1107–49.
15. Tadepalli P, Givan R, Driessens K.Relational Reinforcement Learning An Overview. Proc. of the ICML workshop on Relational Reinforcement Learning.2004; Banff, Canada.
16. R. Munos.Policy gradient in continuous time. Jour. of Mach. Learn.Res., vol. 7, pp. 771–791, 2006.
17. M. B. Hafez and C. K. Loo.Topological Q-learning with internally guided exploration for mobile robot navigation. Neural Computation and Application, 26, 2015.
18. Zajdel R.Epoch-Incremental Reinforcement Learning Algorithms. Int. Journal. Appl. Math. Comp. Sci. 2013; 23(3): 623–635. 2013.
19. Takase N, Kubota N, Baba N.Multi-scale Q-Learning of A Mobile Robot in Dynamic Environments. SCIS-ISIS. Kobe, Japan. Nov. 2012.
20. . Jose JFR. Solway A, Diuk C, McGuire JT, Barto AG, Niv Y.A Neural Signature of Hierarchical Reinforcement Learning.Neuron. 2011 Jul; 71(2): 370–379.
21. Chapman JR. Work Breakdown Structures, ver. 2.01, [Online]. 2004. Available: http://www.hyperthot.com/pm_wbs.htm

22. Mnih V, Kavukcuoglu K, Silver D, et al.Human-level control through deep reinforcement learning. Nature. 2015 Feb; 518: 529–33.
23. Kim BG, Zhang Y, Schaar MV, Lee JW.Dynamic Pricing and Energy Consumption Scheduling With Reinforcement Learning. IEEE Trans. On Smart Grid. 2016 Sep; 07(05).
24. Bowling M,Veloso MM.An analysis of stochastic game theory for multiagent reinforcement learning. Comp. Sc. Dep., Carnegie Mellon Univ., Tech. Rep. CMU-CS-00-165, 2000.
25. Nash J.Non-cooperative games. The Annals of Mathematics.1951; 54(2):286–95.
26. Hu J,Wellman MP. Multiagent reinforcement learning: Theoretical framework and an algorithm. Proc. 15th Intl. Conf. on Mach. Learn. San Francisco: 1998. p. 242–50.
27. Nanduri V, Das TK.A Reinforcement Learning Algorithm for obtaining Nash Equilibrium of Multi-player Matrix Games. IIE Trans.2009; 41(2):
28. Zhifei S, Joo EM.A survey of inverse reinforcement learning techniques. Int. Jour. of Intelligent Computing and Cybernetics. 2012; 5(3): 293–311.
29. Kaelbling LP, Littman ML, Moore AW. Reinforcement learning: A survey J. Artif. Intell. Res. 1996 May; 4:237–85.
30. Kara EC, Berges M, Krogh B, Kar S. Using smart devices for system-level management and control in the smart grid: A reinforcement learning framework. Proc. IEEE Smart Grid Commun, Tainan City, Taiwan: 2012.p. 85–90.
31. Pan GY, Jou JY, Lai BC. Scalable Power Management Using Multilevel Reinforcement Learning for Multiprocessors. ACM Trans. on Des. Auto. of Electronic Sys. (TODAES).2014 Sep;19(4):
32. Gosavi A.Reinforcement Learning: A Tutorial Survey and Recent Advances. INFORMS Journal on Computing.2009 Spring; 21(2): 178–192.
33. Doshi F, Pineau J, Roy N.Reinforcement learning with limited reinforcement: Using Bayes risk for active learning in POMDPs. Proc. of 25th Inter. Conf. on Mach. Learn. (ACM). 2008. p. 256–63.
34. Vijayakumar MV.A Society of Mind Approach to Cognition and Metacognition in a Cognitive Architecture. Ph.D. thesis, Comp. Sci. and Eng. Univ. of Hull, U.K. April 2008.
35. Waelti P, Dickinson A, Schultz W.Dopamine responses comply with basic assumptions of formal learning theory. Nature.2001; 412(6842): 43–48.
36. Wunderlich K, Smittenaar P, Dolan RJ.Dopamine enhances model-based over model-free choice behavior.Neuron. 2012; 75(3): 418–24.
37. Braga APS, Araujo AFR.A topological reinforcement learning agent for navigation. Neural Comput. and Appl.2003; 12:220–36.
38. Silver D. Value Function Approximation. [Online]. Available: http: //www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/FA.pdf
39. Irodova M,Sloan RH. Reinforcement Learning and Function Approximation. Proceedings of 25th National Conference on Artificial Intelligence. (AAAI), 2005.
40. Daw ND, Doya K. The Computational Neurobiology of Learning and Reward. Current Opinion in Neurobiology. Available from: http://www.sciencedirect.com
41. Borkar VS.Reinforcement learning in Markovian evolutionary games. Advances in Complex Systems.2002; 5(1):55–72.
42. Littman ML.Markov games as a framework for multi-agent reinforcement learning.11th Intl. Conf. on Mach. Learn, San Francisco. CA., U.S. 1994.p. 151–63.
43. Shapley LS.Stochastic games.PNAS.1953; 39:1095–100.
44. Maia T. V.Two-factor theory, the actor-critic model, and conditioned avoidance. Learn. Behav. 2010;38(1):50–67.
45. Braga APS . Araújo AFR.Influence Zones: a strategy to enhance reinforcement learning. Neurocomputing.2006;70(1–3):21–34.
46. Busoniu L, Babuska R, Schutter BD,Ernst D.Reinforcement learning and dynamic programming using function approximators.New York, USA: C.R.C. Press; 2010.
47. Millán JDR, Posenato D, Dedieu E.Continuous-action Q-learning. Mach. Learn.2002; 49(2–3):247–65.
48. Dietterich TG.Hierarchical reinforcement learning with the MAXQ value function decomposition. Jour.Arti. Intel. Res 2000;13: 227–303.
49. Frampton M,Lemon O.Recent research advances in Reinforcement Learning in Spoken Dialogue Systems. The Knowledge Engineering Review. Cambridge Univ. Press.2009 Dec;24(4):
50. Uc-Cetina V.A Novel Reinforcement Learning Architecture for Continuous State and Action Spaces. Advances in Arti. Intell. 2013;2013:
51. Vien NA, Ngo H, Lee S, et al. Approximate Planning for Bayesian Hierarchical Reinforcement Learning. Applied Intelligence. 2014 Oct; 41(3):
52. Li H, Liao X,Carin L. Multi-task Reinforcement Learning in Partially Observable Stochastic Environments. J. of Mach. Learn. Res.2009;10:1131–86.
53. Kovacs T,Egginton R.On The Analysis and Design of Software for Reinforcement Learning, with a Survey of Existing Systems. Mach. Learn 2011 Feb; 84:7–49.
54. Moore A,Atkeson C. Prioritized sweeping: reinforcement learning with less data and less real time.Mach. Learn.1993; 13:103–30.
55. Riedmiller M, Gabel T, Hafner R, Lange S. Reinforcement learning for robot soccer.Auto. Robots.2009; 27: 55–73.
56. Bakker B.The State of Mind: Reinforcement Learning with Recurrent Neural Networks. Ph.D. Thesis, Unit of Cogn. Psych., Leiden University, 2004.
57. Bianchi RAC, Martins MF, Ribeiro CH, et al. Heuristically-Accelerated Multiagent Reinforcement Learning. IEEE Trans. On Cyber.2014; 44(2):
58. Ormoneit D, Sen S.Kernel-Based Reinforcement Learning. Mach. Learn. 2002;49(2–3):161–178.
59. Nedic A, Bertsekas DP.Least-squares policy evaluation algorithms with linear function approximation. Discrete Event Dynamic Syst.2003;13:79–110.
60. Lagoudakis MG, Parr R.Reinforcement learning as classification: Leveraging modern classifiers. In Proc. 20th Inter. Conf. on Mach. Learn. (ICML-03), Washington, U.S.; 2003.p. 424–31.