Journal of Theoretical and Applied Information Technology

www.jatit.org

Reinforcement Learning with Application to Adaptive Network Routing

David Kelley
{david.kalley@gmail.com}

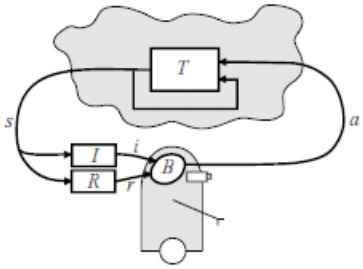
Abstract

Reinforcement learning (RL) is learning from interaction with an environment, from the consequences of action, rather than from explicit teaching. It is the learning performed by an agent by trial and error interactions with a dynamic environment. This paper discusses Reinforcement learning along with application to static routing.

I. Introduction

Reinforcement Learning is a relatively old field. Recently, during the last five to ten years, it has attracted increasing attention in the communities of machine learning and artificial intelligence. The basic concept of reinforcement learning is the use of agents with the mechanism of reward and punishment. There is no stored information for the agent to use. The agent gets the input from the environment, takes a decision and sends the output back to the environment. This changes the state of the environment which is sent back to the agent. The agent may decide further action depending on the resultant input it got from the environment. In those cases where the agent has no supervised learning the above described method can prove to be very efficient. However to achieve this goal certain obstacles have to be solved.

Reinforcement learning is not a protocol or algorithm. Instead it is a unique idea applied to certain problems. We can think of reinforcement learning problems as a class of problems on which we may apply the techniques or strategies of reinforcement learning. These strategies can be roughly classified into two. First is to use genetic algorithms and genetic programming while the second is to perform statistical techniques and dynamic programming methods.



II. Q-Learning

Q-Learning is a technique to solve specific problems by using the reinforcement learning approach. Q-Learning is an incremental version of dynamic programming for solving multistage decision problems. In this case it is the adaptive traffic control problem. This is a model-based approach. The controllers in such an environment should adopt such routing policies so as to minimize the average packet delivery time. The controllers usually have little or no prior knowledge of the environment. Finding the optimal policy in such an environment is very difficult. Moreover the optimal routing policy may change with time as the environment is non-stationary.

An algorithm for implementing Q-Learning is as follows

1. Initialize (with 0's or random values) $Q(s,a)$ for all s S and for all a $A(s)$
2. Repeat (for each episode)
3. Initialize s
4. Repeat (for each step episode):
5. Choose a from s using a policy derived from Q (e.g., -greedy)
6. Take action a, observe resultant state s'

and the reward r.
7. $Q(s,a) \leftarrow Q(s,a) + [r + \max_{a'} Q(s',a') - Q(s,a)]$
8. $s \leftarrow s'$;
until s is terminal

II. Q-Routing

An application of the Q-Learning framework to solve network routing known as Q-Routing was first proposed by Littman and Boyan (1993). Unlike the original Q-Learning algorithm, Q-Routing is distributed as each network node has a separate logical controller, which does not rely on the global information of the network for decision making or selecting of optimal policy.

Exploration Vs Exploitation

In Q-Routing a reinforcement learning module is embedded into each node of a switching network. Only local communication is used or is available. The goal is to learn a routing policy which balances minimizing the number of hops of a packet to reach its destination with the possibility of congestion along some network paths. Experiments on Q-Routing with a discrete event simulator prove the supremacy of this approach over a non-adaptive algorithm based on precomputed shortest paths.

III. Predictive Q-Routing

The problems with the original Q-Routing algorithm are tried to be solved here. The problems are

- Inability to fine tune routing policies under low network load
- Inability to learn new optimal policies under decreasing load conditions.

IV. CQ-Routing

CQ-Routing uses confidence values to improve Q-Routing. Q-values become old when not updated for a long time and thus do not represent the true state of the network. Therefore a confidence value C-value is associated with every Q-value. A C-value close to 1 indicates that the Q-value with which it is associated is reliable and closely represents the state of the network while a C-value of 0 indicates that Q-value is almost random.

When node x sends a packet to its neighbor y, not only it gets back the best estimate of y but also its associated confidence value.

IV. DRQ-Routing

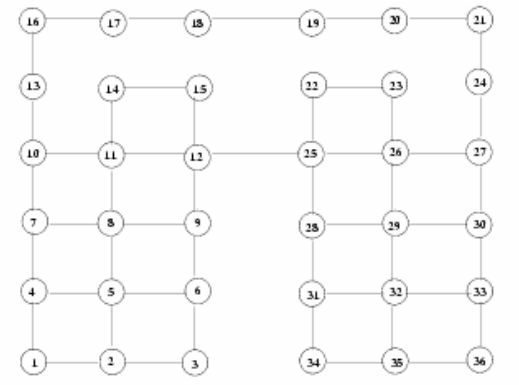
In DRQ-routing Q-routing is improved by updating two Q-values per packet hop. This idea is also known as backward exploration. When a node x sends a packet to node y it may send its own Q-values to y (backward exploration) while node y responds by sending its own information (forward exploration). This technique results in significant increase in performance. Overhead is almost minimal.

IV. CDRQ-Routing

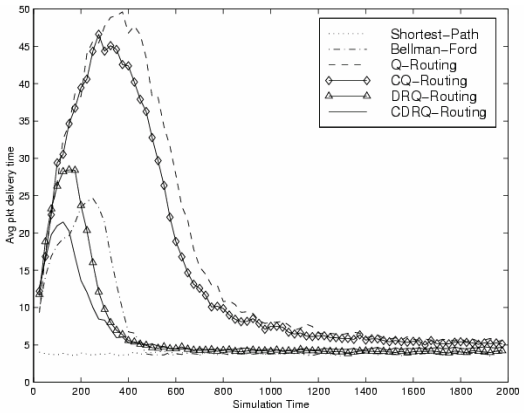
CDRQ-Routing combines the concepts of CQ-Routing with those of DRQ-Routing. This algorithm provides fastest performance as was shown by the experiments. Not only the dual Q-values are exchanged in one packet hop but there confidence values also.

V. Experiments

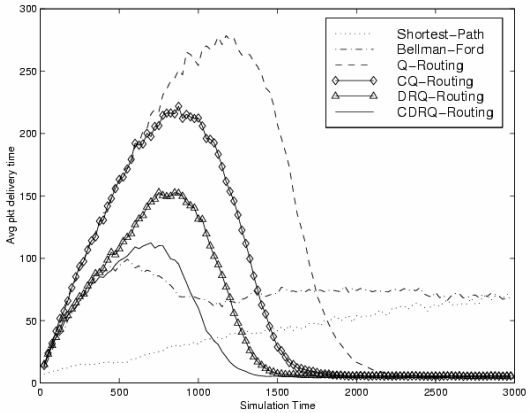
Experiments were performed in a simulated network. Packets were introduced on random nodes and taken off from network as soon as they reach their destination. A 6 * 6 irregular grid was used.



Above shown 6*6 irregular grid was used. There are only two links to connect 18 nodes of one part of network to those of the other. Different experiments were performed by using shortest path non-adaptive algorithm, distributed Bellman Ford, Q-Routing, CQ-Routing, DRQ-Routing and CDRQ-Routing.



As clearly evident from the graph CDRQ-Routing is the most efficient when compared to the other algorithms.



The main deficiency of Q-Routing was converging to optimal policy when network load decreases from high to low. As shown from the above figure CDRQ-Routing almost overcomes this problem.

References

[1] R. Bellman. On a routing problem. Quarterly of Applied Mathematics, 16(1):87--90, 1958.

[2] A note on two problems in connection with graphs, Numer. Math., 1:269 - 271

[3] L. R. Ford, Jr. Flows in Networks. Princeton University Press, 1962.

[4] A. Tanenbaum. Computer Networks. PrenticeHall, second edition edition, 1989.

[5] G. Tesauro. Practical issues in temporal difference learning. Machine Learning, 8(3/4), May 1992.

[6] C. Watkins. Learning from Delayed Rewards. PhD thesis, King's College, Cambridge, 1989.

[7] D Choi, S.P.M, and Yeung,D.-Y. Predictive Q-Routing: A memory-based reinforcement learning approach to adaptive traffic control. 1996