Reinforcement Learning and Routing Algorithms

Alec Situ AASITU@GMAIL.COM
University Hill Secondary School

Editor: Alec Situ

Abstract

This paper discusses many reinforcement learning algorithms and the shortest path algorithm when applied to routing problems. There has been a proliferating desire for fast, intelligent, and effective algorithms in routing problems. Through extensive analysis and comparison of the main algorithms of Q-Routing and Dijkstra, I found many attributes that improve and hinder the algorithm's performance. Q-Routing is an intelligent and adaptive algorithm, while it may sacrifice some performance compared to the Dijkstra algorithm. Ultimately though, the ability to learn is far more valuable and the slight performance difference may be worth the sacrifice.

Keywords: Reinforcement Learning, Q-Routing, Q-Learning, Shortest Path Routing, Dijkstra's Algorithm

1. Introduction

In modern day machine learning, reinforcement learning is applied to a multitude of areas, such as games, automation, networks, etc. In this paper, I discuss the reinforcement learning algorithms for network routing. Furthermore, I compare it to another commonly used algorithm for routing problems, the shortest path algorithm (Dijkstra, 1959).

There are many issues to be addressed in routing algorithms, which include optimizing certain criteria (i.e. packet loss, delivery rate), adapting to changing traffic, hosting multiple servers, etc (Mammeri, 2019). And the need for an algorithm that can quickly and effectively determine the optimal path to send a packet is growing as the world's networks become more intricate and advanced.

2. Related Works

Before machine learning, algorithms were straightforward and generally performed well in identifying the optimal path (Gupta, 2016). However, as technology and machine learning became more refined, reinforcement techniques were implemented to increase algorithm intelligence. Boyan and Littman first proposed a Q-Routing algorithm that is based on Q-Learning and provided researchers with an adaptive algorithm for routing problems (1994). However, there is a general lack of comparison

in the wide variety of algorithms that can be applied to routing problems. While (Tekiner, 2004) compare some reinforcement learning algorithms, there is little comparison with the Dijkstra algorithm on a deep level. In this paper, I analyze the background of many algorithms and clearly articulate the differences in these algorithms.

3. Background

3.1 Reinforcement Learning

Reinforcement learning (RL) is a type of machine learning in which machines or systems utilize past experiences and interactions to decide future actions. Reinforcement learning has been particularly valuable in areas such as game theory, automated systems, and networks. In this paper, I discuss reinforcement learning in routing networks.

Reinforcement learning models usually consists of a set of states, actions, matrix of state probabilities, and a reward function (S, A, P, R). The learner (agent) chooses actions based on the matrix of probabilities and learns the rewards. Often times, however, there is not a matrix probability. An agent (usually a system) conducts certain actions according to a policy (π) and observes the rewards by doing so. Overtime, the model improves its policy. The ultimate goal of any reinforcement learning is to choose the best course of actions to maximize the rewards or the best policy.

3.1.1 Markov Decision Process

The Markov Decision Process (MDP) is defined by a four-tuple (S, A, P, R), like most reinforcement learning models. In MDPs, the following condition must be satisfied:

$$P(S_{t+1}|S_t) = P(S_{t+1}|S_1, S_2, \ldots, S_t).$$

In other words, the probability of the next action is independent of the previous actions.

The policy (π) determines the decision process of a Markov chain. In this case, it is usually the probability matrix of current state s and possible actions. Another important aspect of an MDP is the discount factor (γ), which discounts rewards n steps away by a factor of γ^n, where $0 < \gamma < 1$. The discount simply decreases the rewards due to potential uncertainty about future actions.

With these attributes, Bellman proposed equations to solve the problem and optimize the rewards through the Bellman's Equations. In particular, he created a mathematical formula for the value function, which calculates the potential value of a state. The equation is given in the following page where $R(s, a)$ is the reward of taking an action a at state s, and $T(s, a, s')$ is the transition probability of taking action a at state s to end up in state s'. Applying a simple algorithm with Bellman's

$$V(s) = \max_a R(s,a) + \gamma \sum_{s'} T(s,a,s')V(s'),$$

Bellman's Optimality Equation

$$Q(s,a) = R(s,a) + \gamma \sum_{s'} T(s,a,s')Q(s',a'),$$

Q-Value Function

Optimality Equation will yield the optimal set of actions. First, set all $V(s)$ to 0. Then, following the actions dictated by the policy, iteratively update each $V(s)$ using Bellman's Optimality Equation. Given enough iterations, the estimated set of $V(s)$ will allow us to evaluate the given policy.

3.1.2 Q-Value iteration and Q-Learning

Similar to the Value Function created, Bellman also created the Q-Value (aka Quality Values), which allows for an agent to find the optimal policy, not just simply evaluate a policy. The Q-Value, usually denoted as $Q(s,a)$, is the potential value of taking an action a at state s (equation given above). Simply using the Q-Value Function with a simple algorithm similar to the previous one Value Function will grant the optimal policy. First, initialize all the Q-Values to 0. Then, iteratively update the Q-Values by taking actions until the Q-Values are optimized. As Q-Value represent the potential value of future actions, choosing the course of actions with the highest Q-Value every step should be the policy.

Q-Learning is a more realistic application of the Q-Value Iteration, in which the transition probabilities and rewards are not known to the agent. This was first proposed by Watkins (1989). Thus, the agent plays the game randomly and records the estimated Q-Value. Once the Q-Value is accurate enough, the system can then determine the optimal policy.

3.1.3 Q-Routing

Q-Routing is an application of Q-Learning in routing networks proposed by Boyan and Littman. It has a very similar algorithm to Q-Learning, with the main difference being that the goal is to minimize the cost of sending packets, rather than maximizing the benefits.

3.2 Shortest Path Algorithm

The shortest path algorithm, commonly known as Dijkstra's algorithm, is one of the most common algorithms used in routing networks. The "distance" is usually the cost of sending packets (i.e. packet loss, delivery rate, etc.) rather than physical distance. It is relatively straightforward and time efficient to implement.

The algorithm begins by setting the first node to 0 and the rest to infinity. From the starting node, go to adjacent nodes and update their distance based on the shortest distance value. If the previously assigned distance of the node is less than the new distance, do not update it. Otherwise, repeat this process until the end node.

4. Experimentation and Evaluation

I began by implementing the Q-Value Iteration and Q-Learning algorithms. By testing both algorithms on a simple Markov chain, I created the following graphs:

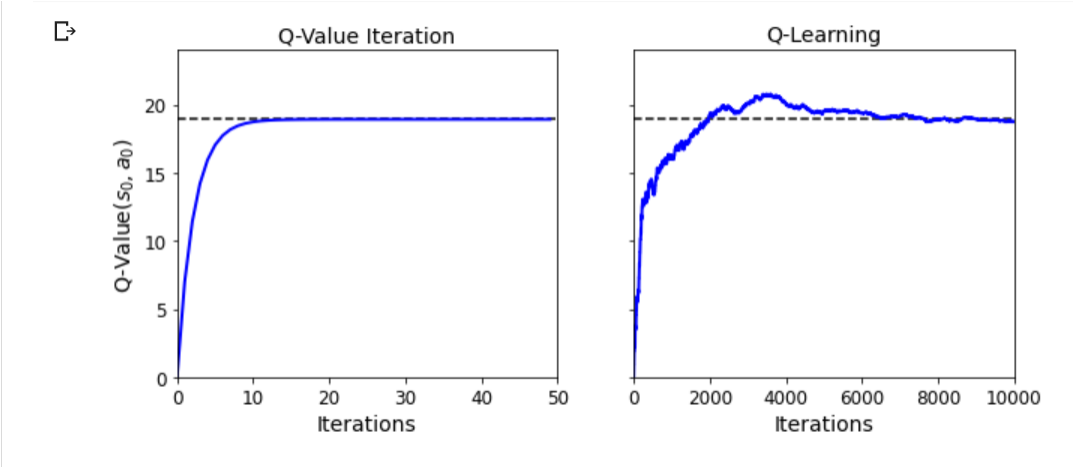

Figure 1: Q-Value Iteration and Q-Learning.

In the sample Markov chain, the optimal Q-Value, as shown by both graphs, is 20. However, it is important to note that the Q-Value Iteration only took about 10 iterations to reach a stable asymptote, while the Q-Learning process requires around 8000 iterations to achieve the stable optimal value. Such a vast difference is attributed to the nature of the algorithms. While the Q-Value Iteration algorithm is provided with the transition probabilities and rewards, the Q-Learning agent blindly explores the Markov chain given only states and actions. Thus, it requires more epochs for the agent to accurately estimate the Q-Values for each node.

I also compare the two algorithms designed for routing networks, namely the Q-Routing and Dijkstra algorithm. Both of these algorithms are ubiquitous in routing problems as both produce excellent results. However, both have inherent properties that affect its efficacy in solving routing problems.

Q-Routing is an adaptive learning algorithm (similar to the Q-Learning graph shown above) that can learn quickly and intelligently. While Q-Routing is sometimes unable to generate the optimal path for sending a packet, it is still capable of finding a close substitute in a relatively short amount of time, no matter the traffic. However, it can only find one path, which may or may not be ideal. On the other hand, the Dijkstra algorithm will guarantee an ideal path of a packet, theoretically. However,

it will ignore traffic conditions of the network, as it is not an adaptive algorithm like the Q-Routing.

5. Future Works

In the future, it will be imperative to test the Q-Routing and Dijkstra algorithms on different networks under different traffic conditions to better compare and analyze the precise differences. Furthermore, it will provide statistical proof of some of the theoretical ideas in this paper. In order to do so, I could use the node incidence matrix, which can transform a network of nodes into computer-friendly matrices that can easily be tested in the algorithms.

6. Conclusion

In our study, I were able to analyze and compare the many reinforcement learning algorithms applied. Furthermore, I applied the reinforcement learning algorithms to routing networks and compared them based on modern routing problems. More specifically, we identified the differences between the Q-Value Iteration and Q-Learning. The Q-Learning algorithm is generally a superior machine learning technique because of its nature to learn. In real world applications, the precise transition probabilities and rewards are usually unknown, which makes Q-Learning far more practical. Q-Routing is adapted from Q-Learning and the difference is that Q-Routing minimizes loss rather than maximizes reward. When used in routing networks, Q-Routing learns but has many inherent flaws, such as the inability to find the optimal path consistently. In addition, the Dijkstra algorithm, while capable of finding the optimal path, is unintelligent and ignores traffic conditions. Ergo, most of the time, Q-Routing is a superior algorithm as it learns.

Acknowledgments

Special acknowledgement for Professor Suleyman Uludag at the University of Michigan for guiding and helping me throughout this research.

References

1. C. J. Watkins, "Learning from delayed rewards," Ph.D. dissertation, King's College, London, U.K., 1989.

2. Dijkstra, E.W., "A Note on Two Problems in Connexion with Graphs", Numerische Amthematik 1: 267-271, 1959.

3. Geron, Aurelien. Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. Second edition, O'Reilly Media, Inc, 2019

4. Gupta, Nitin. 'Applying Dijkstra's Algorithm in Routing Process'. International Journal of New Technology and Research, vol. 2, no. 5, May 2016, pp. 122–24.

5. J. A. Boyan and M. L. Littman, "Packet routing in dynamically changing networks: A reinforcement learning approach," in Proc. Neural Inf. Process. Syst. Conf. (NIPS), 1994, pp. 671–678.

6. L. Peshkin and V. Savova, "Reinforcement learning for adaptive routing," in Proc. Int. Joint Conf. Neural Netw., 2002, pp. 1825–1830.

7. Mammeri, Zoubir. 'Reinforcement Learning Based Routing in Networks: Review and Classification of Approaches'. IEEE Access, vol. 7, 2019, pp. 55916–50. DOI.org (Crossref), https://doi.org/10.1109/ACCESS.2019.2913776.

8. M. Wang, Y. Cui, X. Wang, S. Xiao, and J. Jiang, "Machine learning for networking: Workflow, advances and opportunities," IEEE Netw., vol. 32, no. 2, pp. 92–99, Mar./Apr. 2018.

9. S. Chettibi and S. Chikhi, "A survey of reinforcement learning based routing protocols for mobile ad-hoc networks," in Recent Trends in Wireless and Mobile Networks. Ankara, Turkey: CoNeCo, 2011.

10. Tekiner, Firat et al. "Comparison of the Q-routing and shortest path routing algorithm." (2004).