PAPER • **OPEN ACCESS**

Dynamic Routing Algorithm with Q-learning for Internet of things with Delayed Estimator

To cite this article: Fang Wang *et al* 2019 *IOP Conf. Ser.: Earth Environ. Sci.* **234** 012048

View the article online for updates and enhancements.

Dynamic Routing Algorithm with Q-learning for Internet of things with Delayed Estimator

Fang Wang[1], Renjun Feng[1] and Haiyan Chen[1]

[1] State Grid Jiangsu Electric Power Limited Company Suzhou Power Supply Branch, Suzhou Jiangsu 215004, China
grammynow@aliyun.com, frj1989@126.com, 13372175016@189.cn

Abstract. With the popularity of the Internet of things (IoT), tremendous objects are connected to the network, making the network topology complex. Mechanical engineering achieves intelligent identification, positioning, tracking, monitoring and management of engineering machinery through the IoT, where information exchange and communication require novel intelligent routing algorithms as traditional routing algorithms are unfit for current network environment. Q-routing implemented a dynamic adjustment which was based on the network environment by combining the Q-learning algorithm. However, Q-routing is a highly random network environment and leads to a decline in performance because of overestimation of values. To solve the problem, we propose an algorithm called Delayed Q-routing (DQ-routing), which uses two sets of value functions to carry out random delayed updates so as to reduce the overestimation of the value function and improve the rate of convergence. The experiments indicate that DQ-routing algorithm gets well performance in several problems.

1. Introduction

In recent years, with the development of Internet of things (IoT), the overall communication data of the network is keeping increasing. To handle the complex networks, it is particularly important to develop effective routing strategies. Traditional static routing algorithms are difficult to apply to the large-scale network with uncertain load as they usually use fixed rules such as routing table, which cannot be adjusted promptly to the fluctuation of network state [1]. Although dynamic routing algorithms can automatically adjust the routing strategy according to the current network environment, the general dynamic algorithm needs the global information and the complexity is high. As a result, the network load usually tend to increase [2].

IoT is the network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these objects to connect and exchange data [3]. The number of online available devices in the world has reached 8.4 billion in 2017 [4], and experts expect more than 30 billion objects will be connected to IoT and the global market value of IoT will reach 7.1 trillion by 2020 [5]. Each device in IoT can be regarded as a route, and routing algorithm is used when data exchange and transmission between routes.

There are two kinds of routing algorithms: static routing and dynamic routing. The purpose of the routing algorithm is to transfer data from the starting node to the target node as fast as possible. If the network is congested, the routing algorithm must quickly adjust the routing strategy to improve the transmission efficiency. For QoS routing problems, researchers have done a lot of work and put forward many related methods. Liu [6] proposed a novel agent-assisted QoS-based routing algorithm for wireless sensor networks. In this method, the synthetic QoS of WSNs is chosen as the adaptive value of a Particle Swarm Optimization algorithm to improve the overall performance of network.

Dubois [7] proposed a distributed routing algorithm for vertically partially connected regular 2D topologies of different shapes and sizes. This paper proves that independently of the shape and dimensions of the planar topologies and of the number and placement of the TSVs, the proposed routing algorithm using two virtual channels in the plane is deadlock and livelock free. Wang [8] proposed a trust-based QoS routing algorithm (called TQR) from the trade-off between trust degree and link delay. This paper introduces the concept of trust and QoS metric estimation into establishing a trust-based QoS model. Amgoth [9] proposed an energy aware routing algorithm for cluster based WSNs. The algorithm is based on a clever strategy of cluster head (CH) selection, residual energy of the CHs and the intra-cluster distance for cluster formation. Zeng [10] proposed an Improved Harmony Search Based Energy Efficient Routing Algorithm (IHSBEER) for WSNs, which is based on harmony search (HS) algorithm. In recent years, researchers have adopted reinforcement learning to solve routing problems. Farahnakian [11] used Q-learning to alleviate congestion in the network. Shilova [12] proposed an extension of full echo modification of Q-routing algorithm and uses adaptive learning rates to improve exploration behavior.

To solve the problem, we proposes an algorithm called Delayed Q-routing (DQ-routing), which uses two sets of value functions to carry out random delayed updates. The random delayed update of the estimator can reduce the overestimation of the value function and improve the rate of convergence.

2. Reinforcement Learning

Reinforcement learning is widely used in machine learning [13]. Reinforcement learning is not limited to a certain method. In practical problems, the agent should interact with the environment so as to achieve its goal [14]. This is a basic idea to be followed in reinforcement learning. The definition of reinforcement learning is defined by describing a learning problem, and any method that can solve this problem can be regarded as a reinforcement learning method. To achieve this goal, the agents we define must be able to interact with the environment, and to a certain degree of awareness of the environment, and then through perception to make an action that can affect the state [15]. With these prerequisites, we must have one or more goals to form an environment for reinforcement learning. Markov Decision Processes (MDPs) are a mathematically idealized form of the reinforcement learning problem for which precise theoretical statements can be made [16].

A MDP can be regarded as a 5-tuples $<S, A, R, P, \gamma>$, where S represents states, A represents actions, P is the state-transition probabilities, which means the probability of the agent making action a at state s and moving to state s', R represents the immediate reward and γ is the discount rate which determines the importance of future rewards [17].

The agent's goal is to maximize the cumulative reward in the long run [18]. Given a sequence of rewards received after time step t is denoted by r_{t+1}, r_{t+2}, r_{t+3}, …, then the agent's goal is to maximize the expected return. The return is the sum of the rewards:

$$G_t = r_{t+1} + r_{t+2} + r_{t+3} + \ldots + r_T \tag{1}$$

where T is a final time step. But we often use the discounting return as the goal:

$$G_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{2}$$

In reinforcement learning, we use the value function to indicate the quality of a state or a state action pair [19]. The rewards the agent can expect to receive in the future depend on the policy. Formally, a policy is a mapping from states to probabilities of selecting each possible action. If the agent is following policy π at time t, then $\pi(a|s)$ is the probability that $a_t=a$ and $s_t=s$. The value of a state s under a policy π, denoted $v_\pi(s)$, is the expected return when starting in s and following π.For MDPs, we can define $v_\pi(s)$ as:

$$v_\pi(s) = \mathrm{E}[G_t \mid S_t = s] = \mathrm{E}[\sum_{k=0}^{\infty} \gamma_k r_{t+k+1} \mid S_t = s] \tag{3}$$

According to the property of MDPs, we can rewrite the equation (3) into a recursive form of Bellman equation for $v_\pi(s)$ [20]:

$$V_\pi(s) = \sum_a \pi(a \mid s) \sum_{s'} \sum_r p(s', r \mid s, a)\,[r + \gamma v_\pi(s')] \tag{4}$$

3. Modelling with Q-learning

The routing algorithm solves the problem of how to choose the nodes to transmit so that the packets can be sent to the destination as fast as possible. Many dynamic routing algorithms need to use global information, which is difficult to obtain accurately in practical applications. The problem can be solved by utilizing reinforcement learning. Reinforcement learning is not dependent on the global information, and the optimal strategy can be learned by calculating the value function for each route in the network, which only has the current state information and the follow-up information after transferring data to next route.

First, determine what state is. In IoT, every node that can send and receive data can be regarded as a state. The goal of routing algorithm is to transmit data as quickly as possible to the target route. The node x in the network can be represented as (x, d), where d denotes the target route. The reason for adding a target route in the state is that the same route may handle packets of different target route. In order to distinguish these packets, we add their target routing to form different states to get more accurate value functions.

Then we consider the action set of the state (x, d). Route set $\{y_1, y_2, y_3\ldots\}$ includes all routes that route x can choose to transmit packets. We use $y(x)$ to denote the route set and the action y that can be chosen under the route x belongs to this set. Next, we're going to determine the reward information. In reinforcement learning, reward information will affect the whole learning process. Different reward information will result in different results and sometimes affect learning efficiency. To ensure that data is transmitted to target routing as soon as possible, we divide the reward into two parts. The first part is the packet waiting time w, which indicates the waiting time of packets in the routing queue. The second part is the packet transmission time t, which represents the time that the packet is transferred from the current route to the next route. These two parts form a complete reward and use only the information of the current route without any other global information.

Finally, the value function used by the algorithm is defined. The Q-value function reflects how good or how bad the state is and the updated value will be taken as the true value of the state to participate in subsequent updates. In order to facilitate the presentation, the data is transmitted to the same target routing d. Define the value function as $Q_d(x, y)$, where d indicates the target routing, x indicates the current state, and y indicates the action that the current route x select and the route transmit the packet to. The backup equation can be obtained through the above definition:

$$Q_d(x,y) = Q_d(x,y) + \alpha(w + t + \gamma \min_z Q_d(y,z) - Q_d(x,y)) \tag{5}$$

The Q-learning algorithm selects the optimal action calculation error in the next state when calculating the error. The less time the data takes is, the better the action is, so the action of the minimum value function is selected in the equation (5). The simplest Q-routing algorithm is shown in Algorithm 1.

```
Algorithm 1. Q-routing
Initialization: $\forall x \in \mathcal{S}, y \in \mathcal{A}, Q(x,y)$
Output: learned action policy
1:  Repeat (for each episode):
2:      Initialize $x$
3:      Repeat (for each route):
4:          Using policy from $Q$ to choose action $y$ from $x$
5:          Take action $y$, observe $t, w$
6:          $\delta = t + w + \gamma max_z Q(y,z) - Q(x,y)$
7:          $Q(x,y) = Q(x,y) + \alpha\delta$
```

```
8:              $x = y$
9:          Until $x$ is terminal
10:     Return action policy
```

4. Algorithm

Compared with the previous routing algorithm, although the Q-routing algorithm has the advantage of performance, there is an overestimation of the value function in high random environments as well as the Q-learning algorithm. In practice, the network structure is complex and the network environment is highly random. The traditional Q-routing algorithm is difficult to adjust in different scenes to meet different needs and its overestimation in high random networks will affect the learning rate.

In order to solve the overestimation of value function, this paper proposes a Q-routing algorithm using the delayed estimator, called delayed Q-routing (DQ-routing). The algorithm uses the value of the delayed estimator as an estimate of the target value when updating. This mechanism can avoid overestimation of the current estimator and speed up the learning rate. The delayed Q-routing is shown in Algorithm 2.

Algorithm 2. Delayed Q-routing (DQ-routing)
Initial: $\forall x \in \mathcal{S}, y \in \mathcal{A}$ **,** $Q(x,y), P(x,y)$
Output: action policy after learning

```
1:  Repeat (for each episode):
2:      Initialize $x$
3:      Repeat (for each route):
4:          from $Q$ and $P$ to choose action $y$ from $x$
5:          Take action $y$, observe $t$, $w$
6:          If (switch estimator):
7:              swap($Q$, $P$)
8:          $z = \arg\max_z Q(y,z)$
9:          $\delta = t + w + \gamma P(y,z) - Q(x,y)$
10:         $Q(x,y) = Q(x,y) + \alpha\delta$
11:         $x = y$
12:     Until $x$ is terminal
13: Return action policy
```

From Algorithm 2, we can see that two estimators are used, but in each time step only one estimator is updated. The reason for Q-learning's overestimation is that it always selects the optimal action of the next state as an estimate when updating. Using the optimal value of the next state in Q-learning results in an optimal value function, but in a highly random environment, the value function is inaccurate, and blindly using the optimal value of the next state leads to overestimate and slow down the learning rate. By using the delayed estimator, we can avoid blindly using the optimal value, so that agent can learn the optimal strategy faster and improve the learning rate.

5. Experiment

5.1 Routing Selection

Figure 1 shows the network structure of routing selection problem， where node 1 is the starting route and node 4 is the target route. There are two routes that node 1 can choose, namely node 2 and node 3, and the rewards of these two nodes are 0. Nodes 2 to node 4 have 10 lines and all networks have random fluctuations and conform to a normal distribution. Thus, the expected return for any trajectory starting with node 2 is 0.1 and selecting node 2 in node 1 is always a mistake.

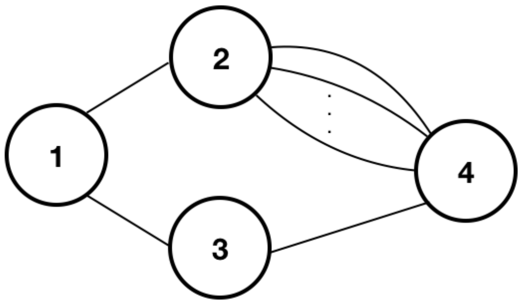

Figure 1. Diagram of routing selection problem.

Table 1 shows the probability of two algorithms to select the wrong nodes. Compared with the Q-routing algorithm, DQ-routing is more likely to choose the best action at the very beginning. This indicates that the delayed estimator can avoid overestimation and find the optimal strategy quickly.

Table 1. Probability of incorrectly selecting nodes by Q-routing and DQ-routing.

Time step	Q-routing	DQ-routing
50	0.86112195	0.32707317
100	0.76940659	0.21558242
150	0.6352766	0.16791489
200	0.52681675	0.14512042
250	0.44927801	0.13029046
300	0.39364261	0.12035052

Figure 2 shows the performance of the Q-routing algorithm and the DQ-routing algorithm in the above domain. Q-routing initially learns to select node 2 much more often than the node 3, and always takes it significantly more often than the 5% minimum probability enforced by ε-greedy action selection with $\varepsilon = 0.1$. In contrast, Delayed Q-routing is essentially unaffected by maximization bias. These data are averaged over 5,000 runs. The initial action-value estimates were zero. Any ties in ε-greedy action selection were broken randomly.

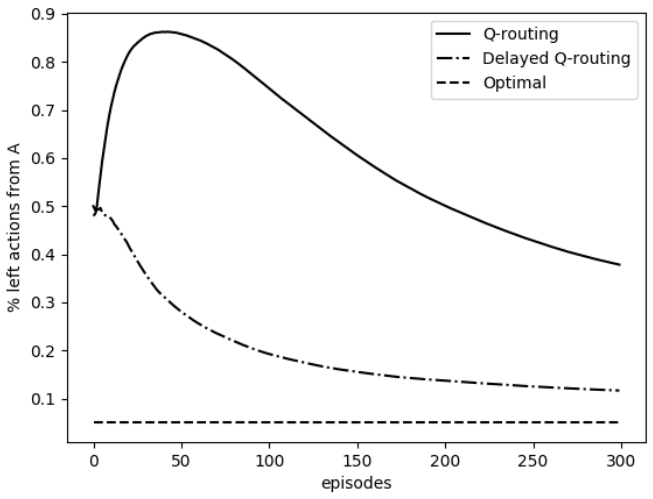

Figure 2. Performance comparison of Q-routing and DQ-routing.

5.2 Packets Transmission

Figure 3 shows the network structure of packets transmission problem, where node 1 is the starting route and node 9 is the target route. The network between the nodes in the graph is unstable. The probability that the network has 50% is connected with reward 10, and the probability of 50% is disconnected with reward -12. When the target route is reached, agent can get a reward of 5.

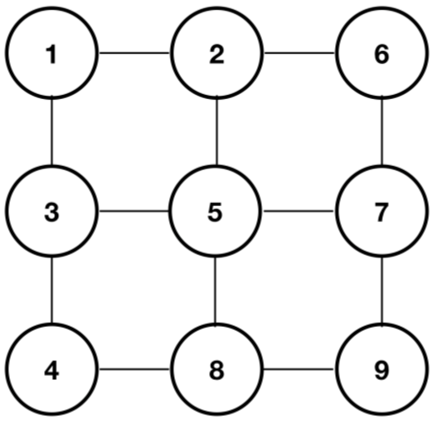

Figure 3. Diagram of packets transmission problem.

Figure 4 shows the average reward per step of DQ-routing and Q-routing. The average reward of DQ-routing is much higher than that of Q-routing, which indicates that DQ-routing has found the best strategy.

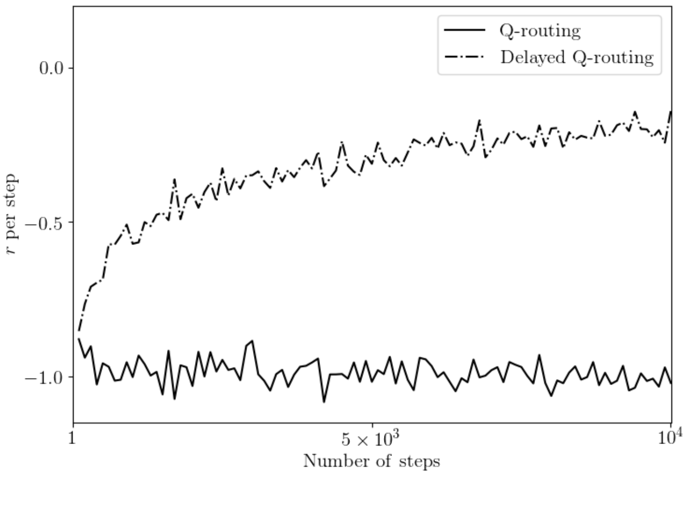

Figure 4. Average reward in packets transmission problem.

Through the comparison of two groups of experiments, we can see that the performance of DQ-routing in high random environment is better than that of Q-routing. Moreover, the performance of DQ-routing algorithm is obviously superior to Q-routing algorithm in the condition of the network environment unstable, and the difference distance will increase with the time increase. The DQ-routing algorithm can find the optimal policy in both scenarios.

6. Conclusion

Traditional routing algorithms cannot dynamically adjust routing strategies based on network fluctuations and cannot be applied to the increasingly complex network environment. Although the Q-routing algorithm can dynamically adjust the strategy, the performance in a high-random network will be affected by the overestimation of value function.

In this paper, we propose a delayed Q-routing algorithm that avoids overestimation of the value function by using a delayed estimator. Through experiments, it has been proved that DQ-routing not only avoids the overestimation of the value function but also improves the learning rate.

References

[1] Pu C L, Yang J, Pei W J, et al.: Robustness analysis of static routing on networks. Physica A Statistical Mechanics & Its Applications, 392(15):3293--3300 (2013)

[2] Wang, C., Shen, G., & Bose, S. K.: Distance adaptive dynamic routing and spectrum allocation in elastic optical networks with shared backup path protection. Journal of Lightwave Technology, 33(14), 2955--2964 (2015)

[3] Lee, G. M., Crespi, N., Choi, J. K., et al.: Internet of things. In Evolution of Telecommunication Services, pp. 257--282, Springer, Berlin, Heidelberg (2013)

[4] Nordrum, A.: Popular internet of things forecast of 50 billion devices by 2020 is outdated. IEEE Spectrum, 18 (2016)

[5] Hsu, C. L., Lin, J. C.: An empirical examination of consumer adoption of Internet of Things services: Network externalities and concern for information privacy perspectives. Computers in Human Behavior, 62, 516--527 (2016)

[6] Liu, M., Xu, S., Sun, S.: An agent-assisted QoS-based routing algorithm for wireless sensor networks. Journal of Network and Computer Applications, 35(1), 29--36 (2012)

[7] Dubois, F., Sheibanyrad, A., Petrot, F., et al.: Elevator-first: A deadlock-free distributed routing algorithm for vertically partially connected 3d-nocs. IEEE Transactions on Computers, 62(3), 609--615 (2013)

[8] Wang, B., Chen, X., Chang, W: A light-weight trust-based QoS routing algorithm for ad hoc networks. Pervasive and Mobile Computing, 13, 164--180 (2014)

[9] Amgoth, T., & Jana, P. K.: Energy-aware routing algorithm for wireless sensor networks. Computers & Electrical Engineering, 41, 357--367 (2015)

[10] Zeng, B., Dong, Y.: An improved harmony search based energy-efficient routing algorithm for wireless sensor networks. Applied Soft Computing, 41, 135--147 (2016)

[11] Farahnakian, F., Ebrahimi, M., Daneshtalab, M., et al.: Q-learning based congestion-aware routing algorithm for on-chip network. In Networked Embedded Systems for Enterprise Applications (NESEA), 2011 IEEE 2nd International Conference on IEEE, pp. 1--7, IEEE press (2011)

[12] Shilova, Y., Kavalerov, M., Bezukladnikov, I.: Full Echo Q-routing with adaptive learning rates: a reinforcement learning approach to network routing. In Young Researchers in Electrical and Electronic Engineering Conference (EIConRusNW), 2016 IEEE NW Russia, pp. 341--344, IEEE press (2016).

[13] Sutton, Richard, S., Barto, Andrew, G.: Introduction to reinforcement learning. Machine Learning, 16(1), 285--286 (2005).

[14] Lewis F L, Liu D.: Reinforcement Learning and Approximate Dynamic Programming for Feedback Control. IEEE Circuits & Systems Magazine, 9(3), 32--50 2015

[15] Van Hasselt, H.: Reinforcement learning in continuous state and action spaces. In Reinforcement learning, pp. 207--251, Springer, Berlin, Heidelberg (2012)

[16] Puterman, M. L.: Markov decision processes: discrete stochastic dynamic programming. pp. 100--135, John Wiley & Sons, (2014)

[17] Strehl, A. L., Littman, M. L.: An analysis of model-based interval estimation for Markov decision processes. Journal of Computer and System Sciences, 74(8), 1309--1331 (2008)

[18] Taylor, M. E., Stone, P.: Transfer learning for reinforcement learning domains: A survey. Journal of Machine Learning Research, 10(7), 1633--1685 (2009)
[19] Szepesvári, C.: Algorithms for reinforcement learning. Synthesis lectures on artificial intelligence and machine learning, 4(1), 1--103 (2009)
[20] Neu, G., Antos, A., György, A.et al.: Online Markov decision processes under bandit feedback. In Advances in Neural Information Processing Systems, pp. 1804--1812, NIPS press (2010)