Network Routing Based on Reinforcement Learning In Dynamically Changing Networks

SARA KHODAYARI, *Graduate Student of Artificial Intelligence*
S.KHODAYARI@ECE.UT.AC.IR
M. J. YAZDANPANAH, *Associate Professor*
YAZDAN@UT.AC.IR
Control and Intelligent Processing Center of Excellence (CIPCE)
Department of Electrical and Computer Engineering
University of Tehran, P.O. Box: 14395/515, Tehran, Iran

Abstract

In this paper we propose a reinforcement learning (RL) algorithm for packet routing in computer networks with emphasis on different traffic conditions. It is shown that routing with an RL approach, considering the traffic, can result in shorter delivery time and less congestion. A simple, but rational simulation of a computer network has also been tested and the suggested algorithm has been compared with other conventional ones. At the end, it is concluded that the suggested algorithm can perform packet routing efficiently with advantage of considering the dynamics in a real network.

Keywords*: Adaptive Routing, Traffic Control, Reinforcement Learning, Neural Network, Computer Network*

1. INTRODUCTION

With an ever-increasing demand for good communication network services, packet routing has been placed as an important matter in computer network field. Routing packets, controlling network traffic and also load balancing is considered to be some open fields of interest in computer networks. In classical solutions for routing such as Bellman-Ford algorithm, the main effort is to route the packet via the shortest path; traffic control and load balancing is less put into consideration. Even in those which traffic was considered, the average delivery time has arisen highly (Peshkin, Savova, 2002).

Routing has been considered from different points of views. Some papers such as (Smeda, et al. 1999; Pierre, et al. 1998; Ahn, et al. 2001; Mehmet, et al. 1993) have focused on it from a neural network approach. Different methods, such as Hopfield network or gradient descent approaches have been applied in order to find paths for routing. Pasupuleti (2002) has applied a fuzzy approach in order to find the path with the least cost for routing. Tsuchiya (1988) has used a new method called Landmark Routing to solve the problem of routing.

In this paper we focus on a new look from classical to an intelligent, instead of having an unrealistic static network as a base, some dynamics and changes are also concerned. Reinforcement learning has been applied in order to find relevant paths for packets. The Q-Routing algorithm considered here, chooses a policy to minimize the number of hops from source to destination, considering the available traffic. We put the main focus on avoiding congestion and balancing the load on the nodes. The training is done by calculating the previous packet's delivery time. Training is online, continuous and local; therefore is capable of adaptation to dynamic changes in network. In here, routing, is finding the most relevant path with highest performance and is a decision which every router has to make independently. So it can be modeled as a "Multi-agent reinforcement learning" problem.

A simple simulator has been designed to demonstrate the happenings inside a computer network. A network is fed into the simulator as an adjacency matrix. The network which has been considered in this paper is an irregular 6*6 grid network which shows special traffic properties and was also studied by Peshkin and Savova (2002) and Boyan and Littman (1994).

2. ROUTING AS AN RL PROBLEM

Looking at routing as a multi-agent problem, each router is an agent, supposed to pass the packet in hand, to a neighbor node. Therefore, there is no necessity for a global knowledge of network since local information will do.

In reinforcement learning, we are looking for a mapping between states and possible actions. In this problem, states are the present node that the packet is in and the actions are the possible neighbors to be chosen in the next step.

Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05)
1082-3409/05 $20.00 © 2005 **IEEE**

Learning is acquired by the previous rewards received from steps taken to the destination node. The rewards are, in fact, in the form of average delivery time and since it's to be minimized, it is counted as *penalty*. The routing is evaluated every iteration and the Q-table is updated. As mentioned above, the performance of algorithm is evaluated from the time packets spend in the network to get to the destination. Each packet gets to the destination node via middle nodes. In each hop, a penalty is given to the selected situation. The penalty includes time in travel and also the time a packet waits in queue. That is:

$$p = t_q + t_l$$

Which t_q is time waiting in queue and t_l is time in travel between nodes. If a few nodes intend to send their packets via a specific node, the traffic in that node increases and the queue time rises. So if that node is even placed in the shortest path of some other packet, it should be considered as a loaded node and the packet may have to choose some other path.

The Q-table was initialized with zero values for a low number of nodes. In the next steps, in order to increase the speed of convergence, the trained Q was used as the initial value for the Q-table. The Q-function was considered as the following equation:

$$Q_{new}(x,y)_d = (1-\alpha)Q_{old}(x,y)_d + \alpha(penalty + \gamma \min_{z \in neighbors\ of\ y}(Q(y,z)_d))$$

α is the learning rate and shows how important the previous state is; γ is the importance of penalty. Smaller α, results in slow convergence and inertia of keeping the previous state grows. $Q(x,y)_d$ is the value node x collects in order to route the packet to its neighbor node, y, for the destination d. Q-value is the predicted penalty sum the packet receives to get to the destination (Harmon, 2004).

A real important and valuable characteristic of reinforcement learning is its power of exploration. In Bellman-Ford algorithm, each node tries to find the best neighbor to route the packet to. The policy taken into consideration here for action selection is based on ε-greedy. In Q-Learning, two methods of action selection can be chosen: one in which the best action with minimum penalty would be selected[1]; and the other in which the action selected would be random by and ε factor. In this method, the packets *usually* look for the best selection, but every *once in awhile* select the not-necessarily-best link and in a heavily loaded network with high traffic, such random path selection results in faster packet routing, although they might not select the shortest path. In the simulations, ε was set to 0.02. Changes in ε explicitly made great differences in path finding and average delay time.

3. ALGORITHMIC DETAILS

The algorithm was applied to different networks. The main network which the evaluations are based on is an irregular one which includes complex traffic situations (Fig 1).

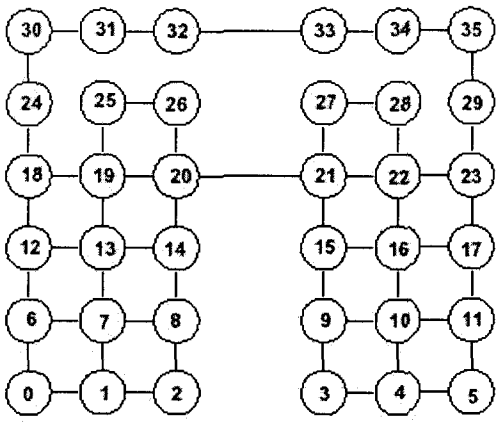

Figure 1: The irregular 6*6 network

This network has two bottlenecks in links 20-21 and 32-33 which may cause critical situations in traffic. If the algorithm would handle these special cases, the general ones would not cause difficulties and can be considered solved.

The tests consist of a few features which will be discussed in the following:

The three algorithms compared in this analysis are the conventional Bellman-Ford algorithm for shortest path finding, the Q-routing without considering the traffic or the normal Q-routing and the load balanced Q-routing. The load balanced Q-routing is actually the enhanced version of the normal Q-routing; only that it directly affects the Q-values in the table according to the changes in traffic. The load balanced Q-routing concluded in more rational and acceptable results from the other two. The advantage of this algorithm over the other ones is in two key features: one issue is in average delay time in delivering the packet from source to destination, which was the main evaluation criterion in this paper and also in papers by Peshkin and Savova (2002) and by Boyan and Littman (1994). The average delay time was evaluated over several loads in the three algorithms and is judged in Figure 2.

[1] Exploitation

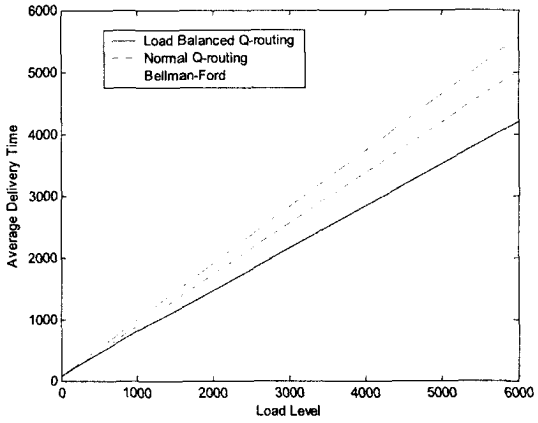

Figure 2: Average delay time in different loads

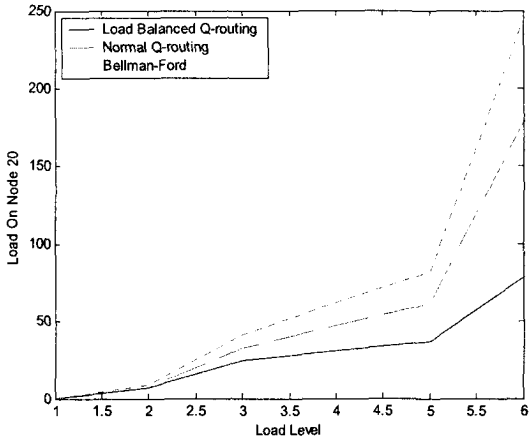

Figure 3: Comparing of load over node 20 in the three algorithms

As it can be seen, the average delay time in load balanced Q-routing is noticeably different with the other two as the load increases. In addition to testing this special network, the same test was applied to other networks which were randomly generated. In all tests, the results of the load balanced Q-routing were quite better than the other ones. The more the load was increased, the more the difference became noticeable.

It is necessary to mention that the simulations guaranteed the delivery of all packets and no packet was discarded. Peshkin (2002) and Boyan (1994) have accounted a time-to-live for each packet and if that time is passed, the average delay time of delivery goes to infinity.

The other issue is the matter of the loads on each node. As mentioned before, the suggested network has a special traffic. The unusual topology of network makes node 20 a critical node, since the loads from right to left, or vice versa, would select this node in their path and since it is placed almost in the middle, it would be selected in most of the shortest paths. The algorithm which is able to distribute the load on node 20 to other nodes in traffic conditions, seems to be a more powerful and successful one. In one of the tests, the source and destination nodes were selected randomly. The load over node 20 in different load conditions is compared in Figure 3.

In the Bellman-Ford algorithm, the load on node 20 rises hastily as the load increases. The normal Q-routing acts the same way, but in a slower fashion. The suggested load balanced Q-routing seems to be distinctly slowing the increase; especially in heavy loads. The tests demonstrated that with the increase of traffic, the Q-table was adapted and trained and the selection of node 20 as a means between right and left segments of network was balanced, so the load was distributed to other nodes. This is shown in Figure 4 for all nodes.

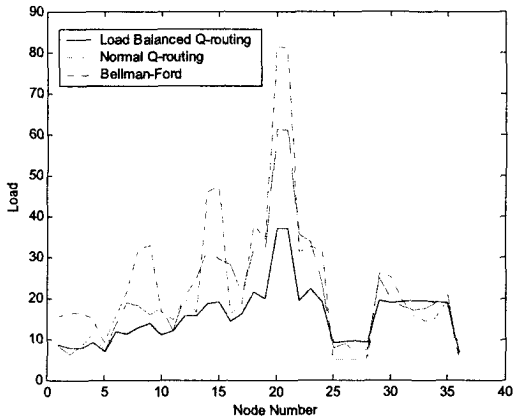

Figure 4: Load distribution over all nodes in the three algorithms

It can be observed that the load put over node 20 by the suggested algorithm is comparably smaller than the other two. Some of the load has been distributed over the other links between left and right which are nodes

Proceedings of the 17th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'05)
1082-3409/05 $20.00 © 2005 IEEE

32 and 33 and the load over these two nodes has enlarged.

4. DYNAMIC CHANGES IN NETWORK

The real computer networks are not static and different changes are dynamically applied to them. These changes may occur due to disconnection of some links or damage in a node or two; therefore result in topology change of network. The other changes which add to dynamics of network are different traffic patterns and load levels. Therefore, such issues were considered in the simulations:

4.1. Changes in Topology

In order to make changes in topology, some links were disconnected during simulation and some other disconnected nodes were linked together. The Q-routing algorithm learned and adapted to the changes very fast. One of the modifications tested was one also applied by Peshkin and Savova (2002) which changed the discussed 6*6 network to a more intricate case of figure 5.

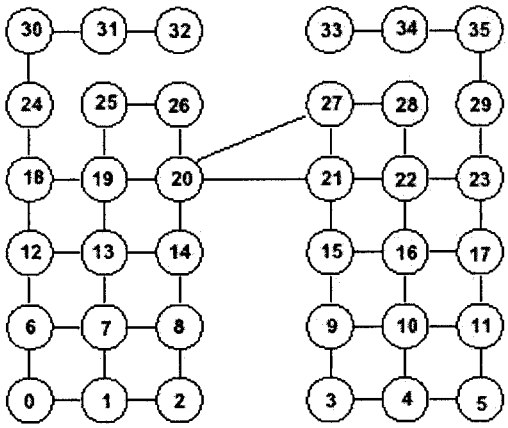

Figure 5: The changed network with a narrower bottleneck

In this network, node 20 is the only connection between left and right segments. In the test done, the Q-table was adapted and converged very quickly. The results are demonstrated in Figure 6.

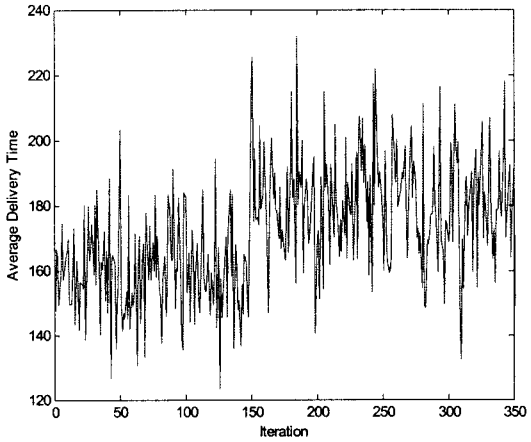

Figure 6: Convergence of Q-table after topology change

After the training period, the average delay time has normally increased since the network is now more complex. But the significance is in the fast convergence. The test showed that at first, the link between node 20-27 was not selected; but as the time passed, the rate of link's selection raised and in some cases got more than selection of link 20-21.

4.2. Traffic Patterns

Relating to Boyan and Littman (1994), some experiments were done to cause the simulation to oscillate between two very different request patterns: one in which the entire load was directed from the upper halves to the lower halves of the network, and one in which all traffic was directed between the right and the left segments.

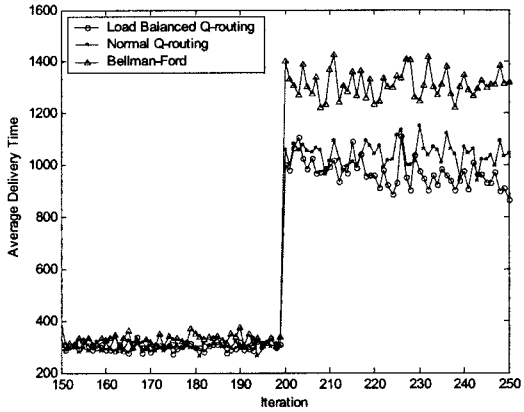

Figure 7: Convergence of Q-table after topology change

In a test which the result of one period is shown in Figure 7, first the traffic was directed from upper halves to lower halves. Since in this pattern the results were so close (average delivery time for load balanced Q-routing was 303.77; average delay time for normal Q-

routing was 306.22 and average delay time for Bellman-Ford was 336.22) the diagram lacks clearness. Of course this pattern does not include the special property of network; but in the rest of the period which the direction of traffic is from left to right (or vise versa), the suggested algorithm distinctively takes advantage over the other two.

4.3. Different Load Levels

The other issue about network traffic is the change in load level. When during the simulation, the total load of the network was highly increased, the routing algorithm adapted rapidly and directed the load to less congested nodes. However by decreasing the load again, the convergence did not take place as fast. This is due to reduction of number of explorations followed by the reduction of number of packets in network. In future work, number of explorations can be intelligent and adaptive with the load level, so the convergence of Q-table would then appear faster and with higher adaptation.

5. CONCLUSION

The focus of the work was on Q-routing. In the suggested algorithm, we showed that reinforcement learning can be a successful tool for adaptive network routing. This algorithm, without requiring the knowledge of the entire topology of the network and its traffic patterns, and without need of central routing, is able to find efficient and relevant paths in dynamically changing networks.

Admittedly, the simulation for network routing here is far from being realistic. To have a more realistic approach, matters such as dynamic change of queue size, different bandwidth of the links, collision of packets and many other factors could be considered. But it claims that the field of network routing by applying reinforcement learning can be considered as a valuable and valid field.

REFERENCES

[1] L. Peshkin & V. Savova, "*Reinforcement Learning for Adaptive Routing*", In Proc. of the International. Joint Conf. on Neural Networks, IJCNN,2002.

[2] J. Boyan and M. L. Littman, *"Packet Routing in Dynamically Changing Networks: A Reinforcement Approach"*, Advances in Neural Information Processing Systems, volume 7, pages 671-378, 1994.

[3] N. Tao, J. Baxter, and L. Weaver, *"A Multi-Agent, Policy-Gradient approach to Network Routing"*, Proceeding of the 18th International Conference on Machine Learning, 2001.

[4] M. E. Harmon & S. S. Harmon, "*Reinforcement Learning: A Tutorial*", www.nada.kth.se/kurser/kth/2D1432/2004/rltutorial.pdf,2004.

[5] Bruce S. Davie, Larry L. Peterson, *"Computer Networks: A Systems Approach"* Morgan Kaufmann, second edition, June 2000.

[6] M. Baglietto, T. Parisini and R. Zoppoli, *"Distributed-Information Neural Control: The Case of Dynamic Routing in Traffic Networks"*, IEEE Transactions on Neural Network, Vol. 12, No.3, May 2001.

[7] A. A. Smeda and M. E. El-Hawary, *"Application of Hopfield Neural Network in Routing for Computer Networks"*, Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering, May 1999.

[8] S. Pierre, H. Said and W. G. Probst, *" A Neural Network Approach for Routing in Computer Networks"*, Proceedings of IEEE Canadian Conference on Electrical and Computer Engineering, CCECE'98, 24-28 May, 1998, Waterloo, Canada, Vol. II, pp. 826-829.

[9] C. W. Ahn, R. S. Ramakrishna, C. G. Kang and I. C. Choi, *"Shortest Path Routing Algorithm using Hopfield Neural Network"*, Electronics Letters, Vol. 37, No. 19, September 2001.

[10] M. K. Mehmet Ali and F. Kamoun, *"Neural Networks for Shortest Path Computation and Routing in Computer Networks"*, IEEE Transactions on Neural Network, Vol. 4, No. 6, November 1993.

[11] A. Pasupuleti, A. V. Mathew, N. Shenoy and S. A. Dianat, *"A Fuzzy System for Adaptive Network Routing"*, Proceedings of SPIE Vol. #4740, SPIE's 16th Annual International Symposium on Aerospace/Defense Sensing, Simulation, and Controls, Orlando, Florida USA. 1-5 April 2002.

[12] P. F. Tsuchiya, *"The Landmark Hierarchy: A New Hierarchy for Routing in Very Large Networks"*, ACM SIGCOMM Computer Communication Review Vol. 18, Issue 4, August 1988.