Packet Routing in Dynamically Changing Networks

Arpana Alka

1   Introduction

Objective of this project is to find which adjacent node the current node should send its packets to get the final destination as early as possible. The performance measure of a policy is its total time to deliver a packet. We can easily observe all the essential characteristics in this problem to model it as a reinforcement learning task, especially in the case of dynamically changing networks.

1. A routing policy should balances minimizing the number of "nodes" a packet will take with the possibility of congestion along popular routes. The congestion rate varies dynamically in a network, thus algorithms with fixed behaviour will not fit for this problem, it should have good adaption capability.

2. The learning should be continual and online.

3. The algorithm should be robust enough to handle irregular and dynamically changing network topology.

A Reinforcement Learning based algorithm definitely meets all these requirments and hence it is a natural fit for this problem. Moreover, it uses only local information for building the routing table, thereby reducing the computational overhead and increasing the effectiveness in a network with high load.

2   Modelling the Routing problem

This Reinforcement Learning based Learning Algorithm is called Q-routing because its underlying principles are same as that of Q-Learning. Consider a network topology shown in figure. Here each node in the network can be considered as a state in the Reinforement Learning terminology. The actions are links to the adjacent nodes from the current node. Unlike the classical Q-learning algorithm, here we don't have any reward function to give immediate reward for every action the agent takes. The reason is because the performance of a policy can only be measured if the packets are reached at their intended destination, so there is no intermediate evaluation for the action we take Let $Q_x(d, y)$ be the time that a node x estimates it takes to deliver a packet P with destination d by routing it through the neighbouring node y(this includes the time that P spends in node x's queue for processing). When the packet reaches the node y, y immediately sends back its estimate of the Q-value to x(ie, the remaining time in the trip starting from node y) namely

Figure 1:

$$t = minQ_y d, z$$

If the packet spends q units of time in x's queue and s units of time in transmission between nodes x and y, then node x can revise its Q value as follows

$$Q_x^{new}(d, y) = q + s + t$$
$$\Delta Q_x(d, y) = \alpha(q + s + t - Q_x(d, y))$$

where α as the learning rate parameter. With this estimated $\Delta Q_x(d, y)$ value, node x can revise its $Q_x(d, y)$ as follows:

Hence the objective of the algorithm is to minimize $Q_x(d, y)$.

3   Q-Routing Algorithm

The $Q_x(d, y)$ value is learned by the algorithm is stored in the form of a table. The final output of the algorithm is the $Q_x(d, y)$ table.

1. Generate a packet, insert it to a random node(say x), assign a random destination.

2. When a node x receives a packet, it sends that to y, where y is

$$y = arg \min_{z \in neighbour of x}\{Q_x(d, z) + s\}$$

Here s is the transmission time between nodes x and z.

3. 3. When the packet reaches y, y sends back its estimate of the Q y (d , z) value(say t) to x.

$$y = arg \min_{z \in neighbour of x}\{Q_y(d, z)\}$$

4. $\Delta Q_x(d, y) = \alpha(q + s + t - Q_x(d, y))$

5. $\Delta Q_x(d, y) = Q_x(d, y) + \Delta Q_x(d, y)$
   terminate the process if Q x (d , y) converges. Otherwise goto next step.

   if y is the destination d, goto step 1. Otherwise Assign x=y, goto step 2.

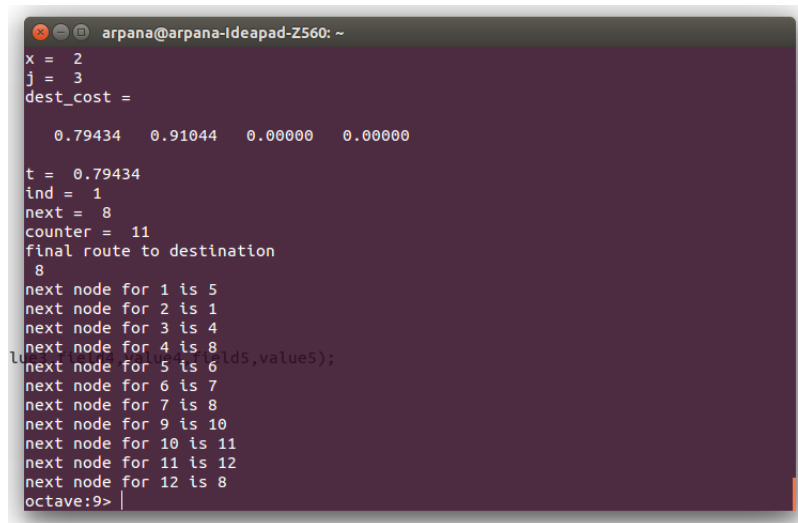# 4 Results

Tool used: matlab
Network structure is shown below

$1 - -2 \quad 3 - -4$
$| \qquad \quad |$
$5 - -6 - -7 - -8$
$| \qquad \qquad |$
$9 - -10 - 11 - 12$

Simulation of the network is done as follows

1. initial $Q_x(d, y)$ values are generated randomly for each nodes.

2. the adjacent nodes of the destination node are set with minimum cost of delivery.

3. the transmission delays are generated randomly from a uniform distribution.

4. the process terminates when there is no change in next node chosen for all nodes in the network over ten successive iterations.

This simulation gives low traffic path for one specific destination node at a time, hence to determine all such paths for all nodes we have to run it many times with different destination nodes. The path learned by the algorithm for destination node 8 is
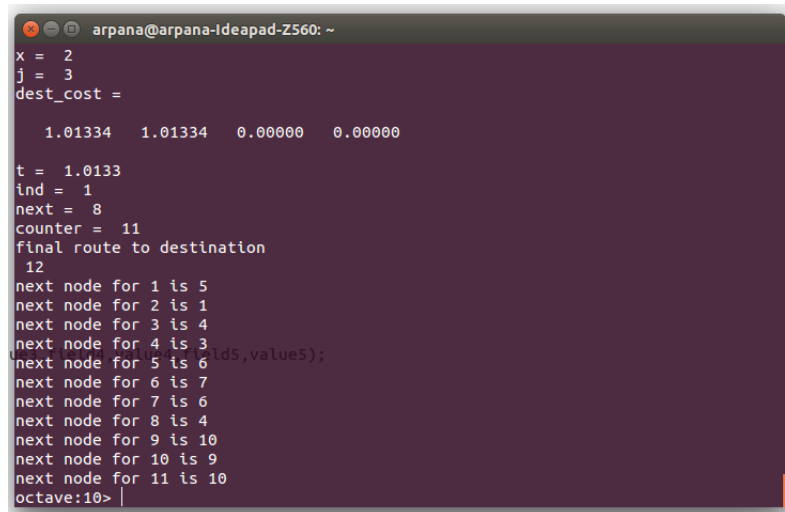 next node for 1 is 5

Figure 2:

next node for 2 is 1
next node for 3 is 4
next node for 4 is 8
next node for 5 is 9

3

next node for 6 is 5
next node for 7 is 6
next node for 9 is 10
next node for 10 is 11
next node for 11 is 10
next node for 12 is 11
 final route to destination 12

Figure 3:

next node for 1 is 5
next node for 2 is 1
next node for 3 is 4
next node for 4 is 8
next node for 5 is 6
next node for 6 is 7
next node for 7 is 8
next node for 8 is 12
next node for 9 is 10
next node for 10 is 11
next node for 11 is 12

5   Reference

[1] Justin A. Boyan, Michael L. Littman, "Packet Routing in Dynamically Changing Networks: A Reinforcement Learning Approach", Advances in Neural Information Processing Systems 6, 671-678, 1994. [2] Suciu Andrei, "A Survey on Network Routing and Reinforcement Learning", 2000.