Asian Journal of Information Technology 7 (2): 35-39, 2008
ISSN: 1682-3915

Multi-Agent Based Intelligent Routing System

[1]J. Arokia Renjit, [1]L. Ancy Geoferla and [2]Chandrasekhar Reddy
[1]Department of CSE, Jeppiaar Engineering College, Old Mamallapuram Road,
Chennai, Tamilnadu-600 119, India
[2]University of JNT, A.P., India

Abstract: In this study, a network routing algorithm, which has autonomous adaptability to network traffic conditions has been proposed. When a routing node has some different paths to a given destination, we can evaluate these paths in terms of their latency (delay time), which will be informed back from the destination node. By using latency evaluation of the path for selection, every node works as a distributed autonomous agent for adaptive routing. In forwarding packets, a routing node decides the next routing node according to the local rules with evaluation. There are 2 local rules for adaptive routing: rule 1 in which we select the next node with the shortest latency and rule 2 in which we select the next node at a probability inversely proportional to the latency value. A combination of these two rules results in the convergence to the optimal solution. Our network simulations show that this routing algorithm has good adaptability towards congested path avoidance.

Key words: Adaptive routing, routing agents, congestion, network simulation

INTRODUCTION

By a wide spread of the Internet connectivity, the total amount of network routing nodes is keeping on increasing. In such a situation, routing tables used to route IP packets will grow bigger and bigger if we continue to rely on conventional routing protocols. Moreover, it's rather difficult for these protocols to deal with network's environmental changes such as topological changes or unpredictable turbulence of network latency distribution (Vutukury and Garcia, 1999).

Here an adaptive routing algorithm is proposed in which every network node works as a distributed autonomous agent for network routing (Dorigo *et al*., 1999; Bonabeau *et al*., 1998; Di Caro and Dorigo, 1998). Generally each node may have some feasible paths to the final destination for the packets. Scores of these feasible paths are given by using the delay time, which will be informed back from the destination node later. Such a score of paths is shared and accumulated by the all-routing nodes along the path.

This approach is not based on a source routing algorithm in which a sequence of the nodes to visit is fully determined at the source node (Munetomo *et al*., 1998; Barolli *et al*., 2001). If we can specify the sequence of the node as a string, which is assigned with a score given by the delay time, stochastic genetic operations such as crossover or mutation are useful to search an optimal solution. But, the source routing is not suited for the Internet because IP packet routing is originally based on a distributed routing scheme.

The adaptive routing algorithm described here works in a fully distributed manner (Stojmenovic and Lin, 2001). At every node, we arrange an autonomous agent for adaptive routing. According to the latency values of paths obtained, each agent independently decides the best adjacent node toward the final destination specified in the packet. To escape from local minimum solutions, we also make use of probabilistic selection of adjacent nodes.

ROUTING AGENTS

Routing tables: In general, a routing node in a network has two types of data table. One is for network topology information, which is called a topology table and formed of n×n matrix (n is the total number of routing nodes). When two nodes are connected, they exchange their

Corresponding Author: J. Arokia Renjit, Department of CSE, Jeppiaar Engineering College, Old Mamallapuram Road, Chennai, Tamilnadu-600 119, India

topology table each other. Thus all routing nodes have a same topology table, which guarantees the packets reachability.

And the other is for packet routing, which is called a routing table. The routing table includes a sequence of latency values for each adjacent node. An entry in a routing table means an observed delay time specified in a feedback-packet, which is eventually replied by the final destination node. Hence, score values are separately given for every different destination.

Packet handling: Any packet traveling in a network has to include two characteristic data fields in its header. One is for describing a path (list of node IDs) through which the packet has traveled so far. The other is for describing the latency (delay time) between every adjacent nodes included in the path. Here we suppose that system clocks of all the nodes are globally synchronous. So, latency can be exactly measured by using a time stamp put on a packet.

When a packet reaches the destination node, the node immediately generates a feedback-packet. Such a feedback-packet may be implemented by extending a conventional ICMP packet. The feedback-packet includes information about the path and its latency observed. Then the destination node sends back this feedback-packet to the source node. This feedback packet visits the all nodes in a reverse order of the path description, from the destination to the source. Any node along the path can obtain the latency information from the feedback-packet arrived there.

Hence, every node along the path knows how much the next node has an influence on the total latency from here to the final destination. The nodes append the latency value to the sequence of values associated with the next node and the destination node. The length of the latency values sequence is limited and is referred to as a time window. As a new latency value comes in, an oldest value goes out.

Local selection rules: A routing agent is assigned to each node and makes an autonomous decision for adaptive routing. A routing procedure for forwarding normal packets is as follows. When a packet arrives at a node, the agent refers its own routing table and finds out feasible next nodes, which leads to the packet's destination. The agent reads out all the latency value sequences of the feasible next nodes and calculates the weighted average latency Ai as follows:

$$A_i = \sum_{j=1}^{n} \alpha_j L_{dij},$$

Where, i is a next node's ID, d is the destination node's ID, Ldij means jth element in the history of latency time about destination node: d and next node: i. αj is a weighting array and n is a time window size (length of a latency values sequence).

The routing agent has two rules for local selection: rule-1 and rule-2. In the local selection rule-1, an adjacent node, which has the smallest Ai is selected as the next node to receive the packet. It is a straightforward approach to find out a short latency path. But when network's environmental change occurs, the generated paths may be the locally optimal solution.

The local selection rule-2 is a random selection weighted by the inverse of Ai. The reason why we use the inverse is to make the average of the total latency shorter and to obtain more feedback information about shorter total latency paths. By using this optional rule, the path can escape from the locally optimal solutions. These 2 local selection rules are implemented to each routing agent. Which rule to apply is heuristically determined according to the latency conditions given by feedback packets or agent-to-agent communication. Hereafter we refer the nodes as the same meaning term of a routing agent.

Implementation of local rules: All nodes initially act according to the local selection rule-1. When congestion occurs, the next node of the upstream side of the congested link changes its own rule to the local selection rule-2. If the random selection (local selection rule-2) does not make an efficient path change in a viewpoint of total latency during a time span given by ti, a random selection rule is applied to the next node in a upstream direction. And the former random selection node turns back to the deterministic rule. If the congestion is still detected in the downstream of the random selection node, a random selection rule is applied to the next node in a upstream direction again. And so in this case, the random selection node before turns back to the deterministic rule.

In such recursive applications of the local selection rule-2, the time span ti in which we make use of a random selection is given as follows,

$$ti = t0 \times i,$$

Where t0 is a constant value for random searching and I means the number of hops from the destination node.

RESULTS AND DISCUSSION

For a preliminary evaluation of our approach, we have implemented a small network simulator including 16 routing nodes connected in a mesh. In this simulation experiment, at first we focus on the behaviour of path generation and their total latency and compare them with the globally optimal solution theoretically obtained by Dijkstra's algorithm.

As shown in Fig. 1 the simulation results show that our algorithm adaptively makes up a new path with the shortest latency in a changing network environment. The weight (delay time) of some arbitrary links changes slightly. Whenever, a dynamic change of the links weight happens, the total latency rapidly converges to the best solution which is the same as what derived theoretically from Dijkstra's algorithm.

Although the random selection (application of the local selection rule-2) initially increases the total latency in a way of bursting for a time, a long-term average of the total latency is shorter than that of conventional routing algorithms.

To evaluate our adaptive routing algorithm on more realistic environmental conditions, we consider the processing limit for packets in every routing node. Suppose every node can't receive any packets beyond a certain number of packets per step and the remaining packets are pushed into a queue for later processing. At the routing node which has many queued packets, the average visiting time of packets easily becomes longer. In this simulation for the second evaluation, we suppose packets are issued from Node 6 to Node 9 at every 10 simulation steps on the network shown in Fig. 1. Then we suppose Node 5 and Node 11 exchange packets each other so frequently that Node 5, 8 and 11 will exceed their packet processing limits. We focus on the total latency from Node 6 to Node 9 and compare them with other unadaptable results.

The Fig. 2 shows a relation between a simulation time which a packet leaves Node 6 and the total latency of this packet. This total latency is averaged every span of 500 simulation steps. At the simulation step 500, a network congestion happens at the Node 5, 8 and 11. In a routing algorithm with no adaptability, total latency linearly grows up to 18 and then remains constant after that. In contrast, our adaptive algorithm shoots over 18 in an early period.

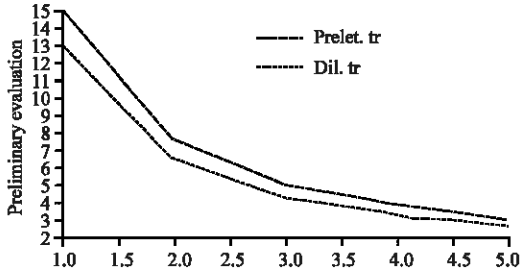

Fig. 1: Adaptive changes of latency in the preliminary evaluation

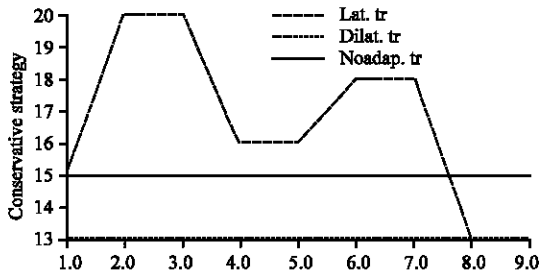

Fig. 2: Adaptive changes of latency in the detour generation with the conservative strategy

But, as the random selection finds out better paths, the total latency gradually converges to the optimal solution.

At the beginning, all packets which left the Node 6 go along the path 6-10-7-11-8-12-9 (minimum delay) as shown in Fig. 3a.

After some simulation steps, Node 7 knows the congestion around Node 11 by the feedback packets, then changes its own local rule to the random selection.

Node 7 continues to search more suitable (smaller latency) paths for a certain period given by ti. For example, suppose we have new paths 6-10-7-4-8-12-9 and 6- 10-7-4-2-5-9, Now 6-10-7-4-8-12-9 path is selected as shown in Fig. 3b.

Now Node 4 finds out the congestion around Node 8, then path change by 6- 10-7-4-2-5-9 as shown in Fig. 3c.

When the random selection can't make an efficient path, a node to be applied the random selection is transferred to another node in the upstream direction.

That is, the random selection is applied to Node 10. Finally, we have the optimal solution path 6-10-13-15-14-12-9 shown by Fig. 3d. If the constant value t0 is smaller, we may have faster convergence. But, too small t0 brings about such a situation that nodes can't evaluate whether the selected next node will be able to cause good changes on the total latency. In fact, the adaptive path generation didn't converge to the optimal solution in the case of t0 = 20.

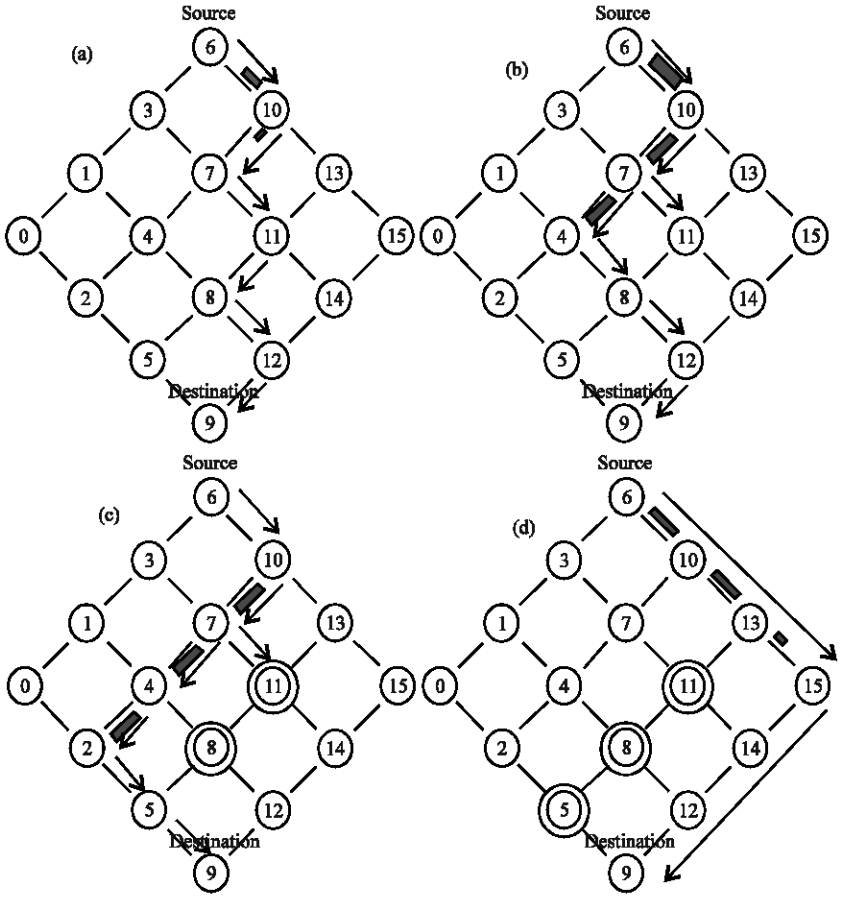

Fig. 3: Path changes in the adaptive detour evaluation

CONCLUSION

We have proposed an adaptive routing algorithm in which a node works as a distributed autonomous agent. Scores of the routing paths are evaluated in terms of the latency and shared by all the nodes along the path. According to the scores of paths obtained, every node independently selects the best adjacent node toward the final destination. To avoid local minimum solutions, we also make use of probabilistic selection and factitively aggressive selection of adjacent nodes.

We have implemented a small network simulator to evaluate our approach. By using simple simulation evaluations, proposed routing algorithm can select the shortest latency paths adaptively to the changes of the network traffic environment.

As a result of using deterministic and random rule we can expect to get a global optimal solution for routing in networks. The heuristic combination of the above two rules give better results when compared other routing algorithms.

ACKNOWLEDGEMENT

We take pleasure in thanking our Chairman Dr. Jeppiaar, the Directors of Jeppiaar Engineering College Mr. Marie Wilson B. Technology, M.B.A. and Mrs. Regeena Wilson B. Technology, M.B.A. and the Principal Mr. Sushil Lal Das M.Sc (Engineering), Ph.D., for their continual support and guidance. We would like to extend our thanks to my Guide, our Friends and Parents without whose inspiration and support our efforts would not have come true. Above all we would like to thank God for making all my efforts success.

REFERENCES

Bonabeau, E., F. Henaux, S. Guerin, D. Snyers, P. Kuntz and G. Theraulaz, 1998. Routing in Telecommunications Networks with Smart Ant-Like Agents. Proc. Intelligent Agents for Telecommunications Applications, pp: 98.

Barolli, L., A. Koyama, T. Yamada and S. Yokoyama, 2001. An Integrated CAC and Routing Strategy for High-speed Large-scale Networks Using Cooperative Agents. Trans. Inform. Proc. Soc. Japan, 42: 222-233.

Di Caro, G. and M. Dorigo, 1998. An Adaptive Multi-Agent Routing Algorithm Inspired by Ants Behavior, Proceedings of 5th Annual Australasian Conference on Parallel and Real-Time Systems.

Dorigo, M., G. Di Caro and L.M. Gambardella, 1999. Ant Algorithms for Discrete Optimization. Artificial Life, 5: 137-172.

Munetomo, M., Y. Takai and Y. Sato, 1998. An Adaptive Routing Algorithm with Load Balancing by a Genetic Algorithm. Trans. Inform. Proc. Soc. Japan, 39: 219-226.

Stojmenovic, I. and X. Lin, 2001. Loop-Free Hybrid Single-Path/Flooding Routing Algorithms with Guaranteed Delivery for Wireless Networks. IEEE. Trans. Parallel and Distrib. Sys., 12: 10.

Vutukury, S. and J.J. Garcia-Luna-Aceves, 1999. A Distributed Algorithm for Multipath Computation. Proc. GLOBECOM, 3: 5-9, 1689-1693.