SEARCHING FOR LOW LATENCY ROUTES IN CPN WITH REDUCED PACKET OVERHEAD *

R. LENT AND P. LIU

Imperial College London,
London SW7 2AZ, UK
E-mail: {r.lent, p.liu}@imperial.ac.uk

Most applications consider network latency as an important metric for their operation. Latency plays a particular role in time-sensitive applications, such as, data transfers or interactive sessions. Smart packets in cognitive packet networks, can learn to find low-latency paths by explicitly expressing delay in their routing goal functions. However, to maintain the quality of paths, packets need to continuously monitor the round-trip delay that paths produce, to allow the algorithm learn any change. The acquisition of network status requires space in packets and lengthens their transmission time. This paper proposes an alternative composite goal consisting of path length and buffer occupancy of nodes that requires less storage space in packets, while offering a similar performance to a delay based goal. Measurements in a network testbed and simulation studies illustrate the problem and solution addressed in this study.

1. Introduction

Quality-of-Service (QoS) routing has become an important solution for Internet service providers and corporate networks to intelligently deal with the growing amounts of traffic and concurrent diversity of applications. While different applications expect different characteristics from the network, most applications demand at least low latency. Low latency is of particular importance in time-sensitive applications, such as mission critical services or interactive sessions.

The latency of an end-to-end communication is the product of a number of factors. These factors include the processing speed of nodes in the path, as well as the propagation delay, which becomes significant in networks of long links. However, in the majority of cases, the dominant factors in

*This work was supported by the Engineering and Physical Sciences Research Council, U.K. under Grant GR/S52360/01.

network latency are the waiting time of packets in nodes' buffers and the store-and-forward transmission delays.

An effective way to achieve low latency in a communication is by dynamically identifying and using the paths in the network that produce the lowest latency as an alternative to using pre-defined paths as intra-AS Internet protocols do [1,2]. Routing algorithms based on the link state or distance vector algorithms are in general not suitable to find delay sensitive paths because of the relative slow process of information exchange among neighbors.

Most algorithms determine path selection in the path length, which uses hop-count (number of nodes in a path) as metric. Shortest paths are usually desirable because they tend to minimize the use of resources, such as the number of transmissions or buffer occupancies. However, shortest paths may produce an undesirable distribution of traffic that may lead to higher buffer occupancies and therefore to higher delays.

It is worthy of note that in some cases QoS routing algorithms are required to satisfy more than one constraint. The problem of path selection with multiple constraints is a problem extensively studied as a general unicast routing problem in the literature. A few examples follow. Wang and Crowcroft proposed a bandwidth-delay constrained solution based on Dijkstra's shortest path algorithm [3]. The bandwidth and delay constrained path selection under imprecise knowledge of network states was undertaken by Guerin and Orda [4], who suggested the use of probability distribution functions to statistically determine the capacity for new connections. Awebuch et al. proposed an algorithm to try to maximize the long-term average throughput of the network by associating an exponential cost of the bandwidth utilization [5]. Salama et al. proposed a distributed heuristic algorithm for a delay constrained and cost constrained problem [6]. Chen and Ansari [7] showed that linear cost functions of additive metric can produce low computation complexity in their algorithm that is based on an extended Bellman-Ford. The effectiveness of these algorithms largely depends on the accuracy of the network state information that is available to them, which usually involves moving large amounts of information traffic and therefore limiting their application.

The advantage of cognitive packet networks (CPN) over other routing algorithms is in that network state collection is focused on active paths and that the algorithm is able to make decisions with limited information [8]. As the algorithm learns more information, decisions are refined and routes become better in their QoS sense. CPN uses smart packets as control packets that are sent out by source nodes to discover routes on demand. Discovered routes are exploited by streams of dumb packets, which transport user data and use source routing to reach their destinations. Packet flows create a collaborative feedback system via acknowledgement packets. These acknowledgments disseminate network status to the nodes on the routes the packets use. For this, packets include a storage area (cognitive map) in addition to a header and payload area, to collect and disseminate network knowledge. Knowledge is accumulated via reinforcement learning in random neural networks located in the nodes of the network and is utilized by future smart packets to make better routing decisions.

CPN can identify low-latency paths by explicitly expressing delay as a goal for smart packets in the reinforcement-learning algorithm. However, the use of delay implies a collection of timestamps along packets' paths, which add overhead to the packets that may become significant in long routes.

This paper proposes an alternative mechanism for CPN to achieve traffic balancing and to identify low-delay paths in a network. The proposal introduces a routing metric for smart packets, which combines path length and buffer occupancy information to replace the explicit expression of delay in the routing goal of smart packets. As a result, a significant amount of overhead can be removed from packets while achieving similar results.

2. QoS Goal Functions

We refer the reader to the literature for further details on the CPN algorithm [8]. This section focuses on defining new goal functions (the inverse of the reward function) for smart packets in the reinforcement learning of CPN.

Assuming that packets of a given flow move along path $P = (n_1, n_2, \ldots, n_h, \ldots, n_d)$, where n_h represents the h-th node on the path and (n_h, n_{h+1}) represents the link between nodes n_h and n_{h+1}. The goal computed at node n_h is:

$$G(n_h, n_d) = \sum_{j=h+1}^{d} \eta C(n_j) \tag{1}$$

where $C(n_j)$ represents the measured cost of a metric of interest when the packet visits node n_j and may be a metric of the node itself or of the link used to reach the node. η is a constant that can be used to balance the resulting function. For example, delay can be minimized by simply making the goal equal to this metric:

$$G(n_h, n_d) = \sum_{j=h}^{d-1} \eta D(n_j, n_{j+1}) \tag{2}$$

where $D(n_j, n_{j+1})$ is the latency in moving a packet from node n_j to n_{j+1}. CPN uses round-trip delay as a metric (instead of forward delay) to avoid synchronize nodes' clocks. Round-trip delay is calculated from the arrival time difference of packets and their corresponding acknowledgments.

Shortest paths in hop count terms can be achieved by defining:

$$G(n_h, n_d) = \sum_{j=h+1}^{d} \eta \times 1 = \eta(d-h) \tag{3}$$

This paper proposes a way to achieve traffic load balancing by composing hop count and buffer occupancy:

$$G(n_h, n_d) = \sum_{j=h+1}^{d} \eta_1 + \eta_2 Q(n_j) \tag{4}$$

where $Q(n_j)$ is the buffer occupancy of node n_j. Through load balancing, we attempt to indirectly make smart packets discover low delay paths. The benefit is in the significant savings that can be achieved in packet overhead after replacing equation (2) with (4).

3. Testbed Measurements

A performance comparison of smart packets either using delay or path length as goals was conducted in a network testbed of 26 nodes. Testbed nodes are PC computers equipped with 4 Ethernet network ports and configured to transmit at a nominal rate of 10 Mbps. The topology consisted of 6 rings of 4 nodes connected in a grid fashion with two additional nodes connected to the ends of the resulting cylinder, which acted as source and destination for the experiments. An implementation of CPN runs in a Linux 2.4.x kernel space as a network layer protocol.

The experiments consisted in observing the round-trip delay and path length of packets in a single traffic stream. A constant bit rate traffic (UDP) was established between these two nodes with packets of size 1024 bytes, which were sent either at a rate of 200 pps or 500 pps. After an initial route was established, one smart packet was sent every 4 dumb packets on average, of which 5% employed random search instead of RNN/RL driven search.

Fig. 3 depicts the path length as measured by consecutive dumb packets within the first 2 minutes of the experiments with a sending rate of 200 pps and 500 pps respectively. The curves compare the ensemble average of at least 20 samples along with the 95% interval of routing cases when path length (hop count) or delay was the routing goal of smart packets. Both curves approximately tend to the value of 8.3.

The ideal shortest path length between source and destination was 8. The observed variation from the ideal value is because of the 5% smart packets that randomly decided their next hop. Randomness is a mechanism in CPN that avoids trapping the algorithm in local minima. For reference purposes, the average path length when using pure random decisions was measured as 14 hops, which gives: $8 * 0.95 + 14 * 0.05 = 8.3$ hops–the value learnt by smart packets in the experiments.

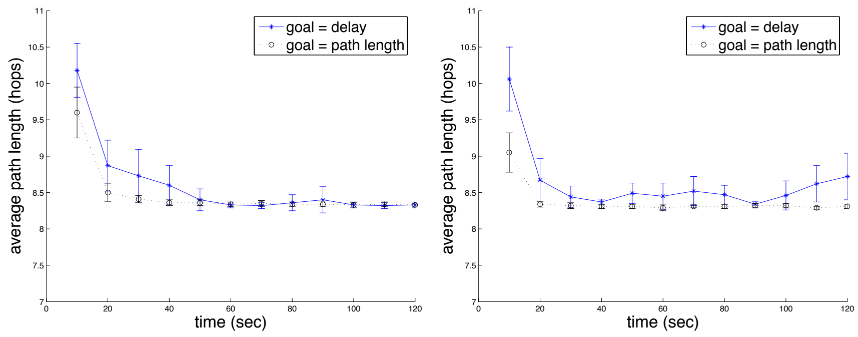

Figure 1. Average path length over time of dumb packets either when delay or path length (hop count) was the minimization goal for 200 pps and 500 pps respectively

Routing based on delay offers the advantage of load balancing while maintaining the path length close to the shortest path. The case is visible under heavier traffic load as shown in Fig. 3. However, routing based on delay requires significant higher overhead than routing based on path length as packets carry extra fields to collect timestamp information at each hop, which implies longer transmission times. In fact, when no other flows are present in the network, routing based on delay performs worse than routing based on hop count because of this extra overhead as reported in Fig. 3, which shows the corresponding average round-trip delay of the packets for the experiments.

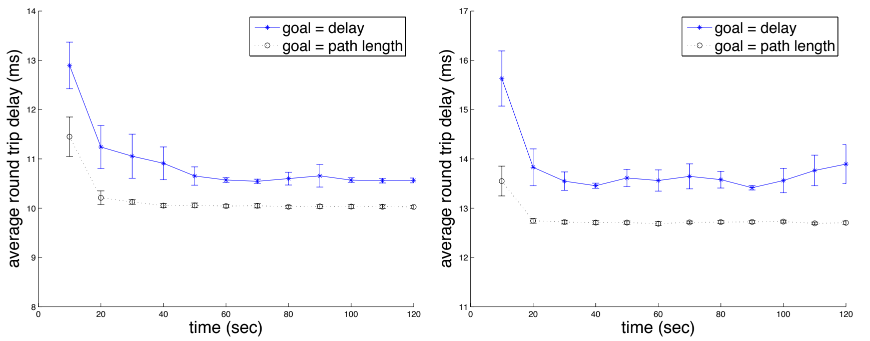

Figure 2. Average round-trip delay over time of dumb packets either when delay or path length (hop count) was the minimization goal for 200 pps and 500 pps respectively

These results confirm the ability of the algorithm to minimize a target cost function and shows the problem that this paper is addressing. In the next section, a simulation based study reports on the solution that we propose.

4. Wireless Ad Hoc Simulations

A simulation study was conducted in *Network Simulator 2* (NS-2) to test the algorithm in a wireless setting.

The algorithm differs in a few details to the one used in the experimental testbed study. Simulations were based on AHCPN [9] and not CPN, so a proportion of smart packets (5%) employed broadcast (flooding) to move in the network. The topology consisted of 7×7 equally-spaced stationary nodes producing a grid setting where only horizontal or vertical communications were possible. The smart to dumb packet ratio was fixed to 0.20 for all experiments and nodes' buffers prioritized smart packets and their acknowledgments.

Ad hoc CPN (AHCPN) is a mobile ad hoc network protocol [10]. As opposed to other protocols in the area [11,12,13,14], the AHCPN protocol restricts the use of flooding (broadcast) and replaces most of the routing decisions with decisions driven by the RNN/RL algorithm. The rate of broadcasts is controlled by a fixed parameter, which was defined as indicated before.

Four CBR traffic streams of 500 byte-length packets were established from each corner of the grid and destined to the opposite diagonal corner. Experiments were repeated to consider three different types of goals: round-trip delay, hop count or a composition of hop count and buffer occupancy. The capacity of each node's buffer was defined as 50 packets.

The average results for the four flows in round trip delay of both smart packet and dumb packets are reported in Fig. 3. The average delay experienced by smart packets and their acknowledgments is much smaller than the corresponding delay of dumb packets because of the priority queuing for these packets (i.e. a high priority packet moves to the head of the queue regardless of the number of packets waiting). The delay observed in packets, either smart or dumb, is better when delay was an explicit metric in the goal of smart packets than when using hop count.

The exposed terminal problem present in the network, may starve nodes located in the vicinity of congested areas. These nodes will report empty buffers even if their channels are congested because of others' transmissions. However, smart packets were able to make the right decisions and the observed performance of the combined metric of hop count and buffer occupancy was almost indistinguishable from that of explicit round-trip delay. An explanation for this effect is in the feedback flow that acknowledgements create. If a smart packet selects a path containing starved nodes, future dumb packets will start congesting the buffers of the path. The information about congested areas will eventually become available to future smart packets, which will have the necessary information to avoid these areas.

Fig. 5 shows the packet loss ratio for smart and dumb packets. Fig. 6 shows the loss ratio for their respective acknowledgments for the experiment. The loss experienced by smart packets and their acknowledgments is approximately constant because of their higher priority in the nodes' queues. Hop count as the unique metric performed the worst of the three cases. The other two cases offered a very similar packet loss ratio.

5. Conclusions

This paper has proposed the use of alternative metrics for CPN routing, expressed as a combined goal formula for smart packets that effectively produce a balancing of network traffic and low-delay paths for end-to-end streams of dumb packets. A routing goal that combines path length and buffer occupancy in nodes offers the advantage of producing approximately the same performance as that of using delay but with a smaller packet overhead. In some cases, as demonstrated by measurements done in a network testbed, delay based routing may produce higher latency than path length-

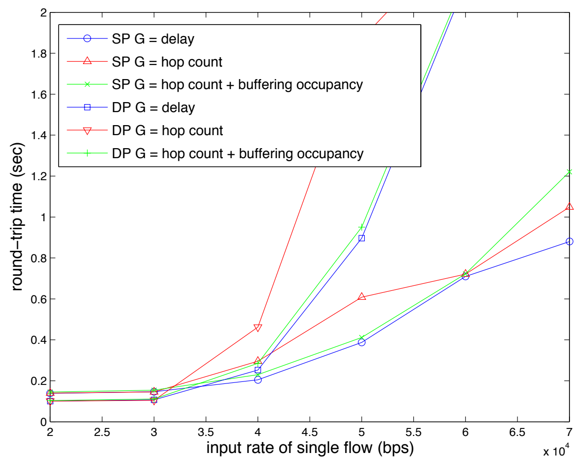

Figure 3. Round-trip latency of smart and dumb packets

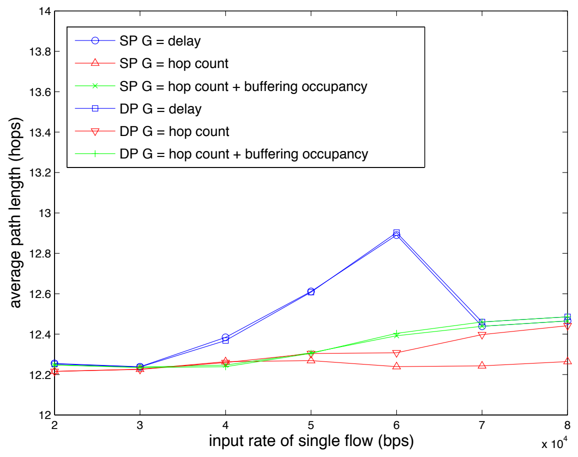

Figure 4. Average path length of smart and dumb packets. Note that after 60 Kbps, congestion becomes network-wide and delay based goal routing starts preferring again the shortest path.

based route as the extra overhead in packets impose a longer transmission time. Simulations have shown that smart packets that consider individual nodes' congestion can discover paths with similar performance dynamics to considering delay, but with a significant saving in packet length and transmission time.

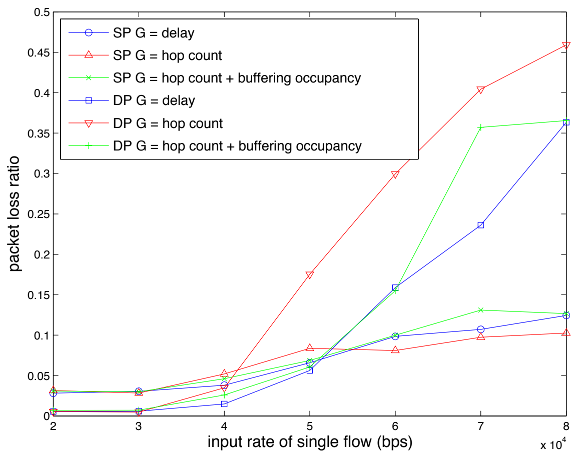

Figure 5. Smart and dumb packets loss ratio

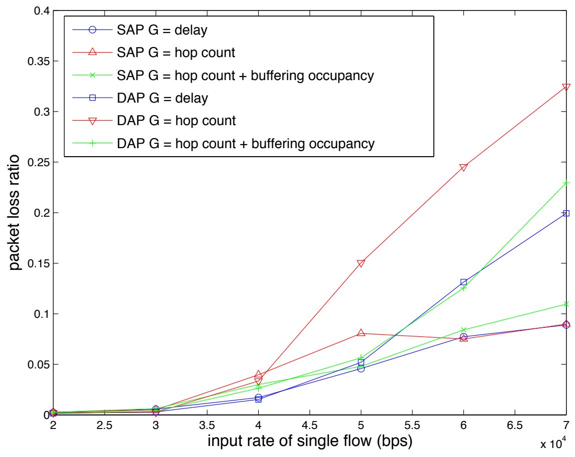

Figure 6. Acknowledgement loss ratio of smart and dumb packets

Acknowledgments

The authors wish to express their gratitude to Prof. E. Gelenbe for his support in the completion of this research work.

References

1. G. Malkin, "Routing information protocol version 2", *Internet RFC 2453*, November 1998.
2. J. Moy, "Open shortest path first version 2", *Internet RFC 2328*, April 1998.
3. Z. Wang and J. Crowcroft, "Qos routing for supporting resource reservation", *IEEE Journal of Selected Areas in Communications*, vol. 14, no. 7, pp. 1228–1234, 1996.
4. R. Guering and A. Orda, "Qos-based routing in networks with inaccurate information", in *Infocom '97, Japan*, April 1999.
5. B. Awerbuch, Y. Azar, and S. Plotkin, "Throughput-competitive online routing", in *Proceeding of the 34th Symposium Foundations of Computer Science*, April 1993, pp. 32–40.
6. H. F. Salama, D. S. Reeves, and Y. Viniotis, "A distributed algorithm for delay-constrained unicast routing", in *Infocom '97, Japan*, April 1997.
7. G. Chen and N. Ansari, "Multiple additively constrained path selection", *IEE Proceedings*, vol. 149, no. 5, October 2002.
8. Erol Gelenbe, Ricardo Lent, and Zhiguang Xu, "Design and performance of cognitive packet networks", *Performance Evaluation 46 (2-3)*, pp. 155–176, 2001.
9. Erol Gelenbe and Ricardo Lent, "Ad-hoc wireless and wireline cognitive packet networks", in *Proceedings of the Second IEEE Workshop on Applications and Services in Wireless Networks, Paris, France*, July 2002.
10. Elizabeth Royer and C-K. Toh, "A review of current routing protocols for ad-hoc mobile wireless networks", *IEEE PERSONAL COMMUNICATIONS*, April 1999.
11. David B. Johnson, Davis A. Maltz, Yih-Chun Hu, and Jorjeta G. Jetcheva, "The dynamic source routing protocol for mobile ad hoc networks", IETF DRAFT, March 2001.
12. Charles E. Perkins and Elizabeth M. Royer, "Ad hoc on demand distance vector routing", in *2nd IEEE Workshop on Mobile Computing Systems and Applications*. February 1999, IEEE.
13. Chai-Keong Toh, "A novel distributed routing protocol to support ad-hoc mobile computing", in *Conference Proceedings of the 1996 IEEE Fifteenth Annual International Phoenix Conference on Computers and Communications*. 1996, pp. 480–486, IEEE.
14. S. Corson, S. Papademetriou, P. Papadopoulos, and V. Park, "An internet manet encapsulation protocol (imep) specification", IETF DRAFT, August 1998.