TCP PERFORMANCE WITH DEFLECTION ROUTING IN THE INTERNET

Jingyi He[†] *S.-H. Gary Chan*[‡]

[†]Department of Electrical and Electronic Engineering
[‡]Department of Computer Science
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong

ABSTRACT

Deflection routing has been well studied for optical networks with regular topologies. In this paper, we propose using deflection routing in the Internet and study its TCP performance. In particular, we show that when the difference between the delay of the deflection path and the shortest path (i.e., the deflection cost) is in a certain range, deflection routing can make almost full use of the free bandwidth in the deflection path and hence achieve substantial throughput improvement. In the worst case when the deflection cost is large, deflection routing can achieve an aggregate throughput no less than that without deflection routing. We analyze the underlying mechanisms of these characteristics. In order to extend the usefulness of deflection routing, we also propose a deflection routing scheme with adaptive deflection point and show via simulation that this scheme can achieve very high throughput as long as the deflection cost is no larger than a certain value. We show that deflection routing is friendly to the existing traffic in the deflection path. One possibly unfavorable requirement of deflection routing is that it should be enabled for either all or none of the flows contending for the same outgoing (congested) link.

1. INTRODUCTION

In the Internet nowadays, packets are routed to their destinations via shortest path routing. When a preferred outgoing link of the router is congested (e.g., when the buffer is full), the packet is simply dropped and hence is lost. With the Transmission Control Protocol (TCP), a lost packet would lead to the backoff mechanism which severely degrades the throughput. In this paper, we consider that a packet encountering congestion is temporarily misrouted, and hence "deflected" to some other outgoing link instead of being dropped. This is the so-called "deflection routing". If there is no buffer at all, deflection routing is referred to as "hot-potato" routing [1].

Deflection routing has been previously studied almost exclusively for optical networks with regular topologies [2], [3], [4], [5], [6], [7], [8]. This is due to two factors. Firstly, in contrast to the availability of relatively large buffers in the traditional store-and-forward electronic networks, optical buffers in the form of optical delay lines (ODL's) are expensive and hence an optical switch is of limited buffering capacity. Deflection routing can hence avoid packet dropping in this case with some path cost. Secondly, despite some studies on unslotted (asynchronous) deflection routing, deflection routing is generally implemented in a slotted (synchronous) network to simplify the operation [5], [6]. In this network, the number of incoming links and outgoing links at each node is the same so that every incoming packet can be forwarded in the next time slot. Therefore, deflection routing is previously studied in the context of regular network topology with the same in-degree and out-degree such as the Manhattan Street Network (MS-Net) and the ShuffleNet, and seldom, if not at all, studied with irregular topologies such as the Internet.

Note that deflection routing is essentially a congestion control scheme which tries to avoid packet losses at the cost that the deflected packets may have to take a longer path to its destination. In this paper, we show how such deflection routing mechanism may be used in the Internet and study its performance. A common problem that exists in both optical networks and the Internet is that the packets may arrive out of sequence at the destination, due to the longer deflection path(s). Most of the researches on deflection routing in optical networks assume that the receiver has an infinite reassembly buffer and a connectionless transport protocol, hence the out-of-order arrivals may not cause trouble. However, in the Internet nowadays, the connection-oriented transport protocol, TCP, is the most common, where the receivers with finite buffer actively participates in the flow control. Hence the interaction of deflection routing with TCP in the Internet needs to be carefully studied. As far as we know, there has been no work addressing the problem of deflection routing in the Internet. The only work that is relevant is on the use of deflection routing in wormhole routing networks [9], [10], [11]. However, wormhole routing networks are typically optical networks with regular topologies, and are usually used as high-speed local area networks (LANs). The performance of TCP in wormhole routing networks with deflection routing has been studied in [11], and the result shows that TCP can well tolerate out-of-order packet (or worm) delivery. However, this is mainly due to the special input control policy adopted in wormhole routing networks, which the Internet nowadays does not use.

This paper specifically addresses the possibility of using deflection routing in the Internet to improve the TCP throughput. In particular, we start by examining the performance of a generic deflection routing scheme, in which the arriving packet is always deflected in case of congestion. We show that this scheme can effectively improve the throughput only for a certain range of *deflection cost*, which is defined as the difference of the delays between the deflection path and the shortest path. By analyzing the typical performance of this scheme, we propose a new deflection scheme

This work was supported, in part, by the Areas of Excellence (AoE) Scheme on Information Technology funded by the University Grant Council in Hong Kong (AoE/E-01/99), and by the Competitive Earmarked Research Grant of the Hong Kong Research Grant Council (HKUST6014/01E).

0-7803-7533-5/02/$17.00 ©2002 IEEE.

in which the deflection point is adaptively determined for different deflection costs. The new scheme can effectively improve the throughput for a large range of deflection cost. We finally study the fairness issues of deflection routing.

The rest of this paper is organized as follows. Section 2 presents and analyzes the performance of the generic tail deflection routing scheme. In Section 3, the deflection routing scheme with adaptive deflection point is proposed and studied. In Section 4, the fairness issues induced by deflection routing are addressed. Section 5 concludes the whole paper.

2. TAIL DEFLECTION

2.1. Scheme Description

We first describe a simple deflection routing mechanism. In this system, a router maintains a list of outgoing links for each destination, with the first outgoing link corresponding to the shortest path and other outgoing links for the potential deflection paths. Output queueing is used in which each outgoing link has a single queue shared by all the flows. When a packet arrives at the router, its preferred outgoing link (i.e., the shortest path) is first checked. If the queue is full, the next outgoing link (e.g., the second shortest path) is checked. The packet is forwarded to the first outgoing link of which the queue is not full. The packet is dropped only if the queues of all the outgoing links in the list are full. From the viewpoint of the link queue, this scheme simply replaces the "drop-tail" mechanism with a "deflect-tail" mechanism. In other words, the deflection point is always at the tail of the queue and hence the term "tail deflection".

With TCP, deflection routing may possibly avoid packet losses and hence retransmissions and the decrease of the TCP congestion window in times of network congestion, thus improving the TCP throughput. However, as mentioned before, a longer deflection path may cause packets to arrive at the destination out of sequence, which may lead to some backoff mechanism in TCP. For example, with Reno TCP, the receiver generates an *ACK* for the next packet it is expecting upon receiving a new packet. A deflected packet subject to a longer delay may arrive later than its following packets forwarded via the shortest path, causing the receiver to generate a number of duplicated acknowledgements (*dup ACK*s). If the sender receives more than a certain number (e.g., three) of such *dup ACK*s, it retransmits the deflected packet and reduces its congestion window by half. This inevitably decreases the TCP throughput. Therefore, the delay difference between the deflection path and the shortest path (i.e., the deflection cost) is a key affecting factor on the performance of deflection routing. We study the effect of different deflection costs on the aggregate TCP throughput in the following.

2.2. Illustrative Simulation Results and Analysis

We use the *ns*-2 network simulator to study the performance[12]. The network topology used has the dumb-bell structure, with an additional deflection path as shown in Fig. 1. We consider five TCP flows, with flow i having source s_i and destination d_i, for $i = 1, \ldots, 5$. The nodes r_1,r_2, and r_3 are routers. Without deflection routing, all the flows go from their sources to their destinations via the direct link between r_1 and r_2. In the case that the shortest path $r_1 - r_2$ is congested and deflection routing is used, the flows take the path $r_1 - r_3 - r_2$. All the links between the sources and the router r_1, and between the destinations and the router r_2, have bandwidth of 3Mbps and delay of 5ms. The bottleneck link between r_1 and r_2 has bandwidth of 4Mbps and delay of 20ms. The link between r_1 and r_3 and that between r_3 and r_2 have the same bandwidth of 4Mbps, and delay of $a$ms and 5ms respectively. All the queues associated with the links are drop-tail queues. We set the size of the queues to be 50 packets, which is the default value in *ns*-2, except that the queue size of link $r_1 - r_2$ is set to be 20 packets to make deflection more easily happen. Each packet is 8000bits long in our simulation. Some of these parameters will be used in our later analysis, and we define a set of symbols for them in the following. B_i denotes the bandwidth of the link from each of the sources to r_1, and $B_i = 3$Mbps. B_s denotes the bandwidth of the shortest path $r_1 - r_2$, and $B_s = 4$Mbps. B_d denotes the bandwidth of the deflection path $r_1 - r_3 - r_2$, and $B_d = 4$Mbps. D_s denotes the delay of the shortest path $r_1 - r_2$, and $D_s = 20$ms. D_d denotes the delay of the deflection path $r_1 - r_3 - r_2$, and $D_d = a + 5$ms. C deontes the deflection cost, and $C = D_d - D_s$. S denotes the packet (segment) size, and $S = 8000$bits. Q deontes the queue size of link $r_1 - r_2$, and $Q = 20$packets. The deflection cost C is adjusted in our simulation by changing the value of a. For example, when $a = 15$, the deflection path has the same length as the shortest path, and hence the deflection cost is zero. By increasing the value of a beyond 15, different deflection costs result. We use the Reno TCP in our simulations, and all the TCP connections are bulk data transfer (FTP). The TCP receiver window is set to be 20 packets, so as to meet the bandwidth-delay product of the network for each flow.[1] To avoid global synchronization, all the TCP connections have their start time uniformly distributed in $[0.1, 0.2]$ second. We take statistics a certain period, say 5 seconds, after the simulation begins, when the initial slow-start stage has ended and the system is in steady state.

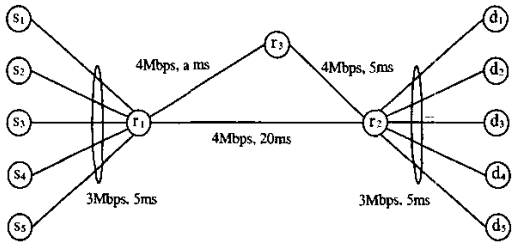

Figure 1: The network topology used in simulation.

We compare in Fig. 2 the aggregate TCP throughput with and without deflection routing versus the deflection cost. When deflection routing is not used, the TCP throughput does not change with the deflection cost, since the deflection path is not used at all. On the other hand, with deflection routing, there is a certain range of deflection cost that leads to substantially larger aggregate TCP throughput (almost doubles in this case) compared with that without deflection routing, which is referred to as the *best range* henceforth. With the same bandwidth of the deflection path as of

[1] With the given parameters, the round-trip time (RTT) of the shortest path (including the queueing delay at the bottleneck link) for all the flows is around 100ms. Assuming the five flows fairly share the 4Mbps bandwidth of the bottleneck link, each flow can have a bandwidth of 0.8Mbps. Therefore, the bandwidth-delay product is about 80Kbits (i.e., 10 packets). Hence, a TCP window of 20 packets will be enough even if taking the bandwidth of the deflection path (another 4Mbps) into account.

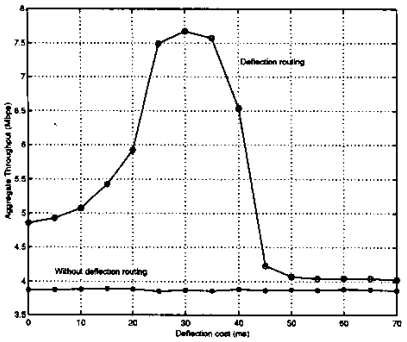

Figure 2: TCP throughput versus deflection cost for tail deflection.

the original shortest path, this means that deflection routing essentially makes almost full use of the bandwidth of the deflection path when the deflection cost is in the best range. When the deflection cost is out of the best range, lower aggregate throughput results. In particular, when the deflection cost is larger than the best range, the aggregate throughput drops quickly to some value, which is close to (but still larger than) that without using deflection routing. When the deflection cost is lower than the best range, the throughput decrease is less sharp and we still see a throughput improvement of at least 25% compared with that without deflection routing.

The above performance of deflection routing is typical, and can be explained as follows. As mentioned before, with Reno TCP, three or more *dup ACK*s cause the sender to retransmit the packet and cut the congestion window by half, leading to a throughput decrease. With deflection routing, multiple *dup ACK*s may be generated due to a either too large or too small deflection cost. When the deflection cost is larger than the best range, each time a packet is deflected, it is likely to arrive later than a number of its following undeflected packets, causing three or more *dup ACK*s and hence the decrease in throughput. This effect is similar to packet losses, hence the aggregate throughput is close to that without deflection routing. On the other hand, when the deflection cost is smaller than the best range, multiple deflected packets may arrive earlier than an undeflected packet before them, hence also causing three or more *dup ACK*s and the decrease in throughput. This is possible because a packet is deflected only if its preferred outgoing link is congested, which means that the packet before it is very likely to be subject to a large queueing delay if not deflected. Therefore, although the deflected packet takes a path of larger link delay, it may enjoy a lower queueing delay, hence arriving at the destination earlier. However, the multiple earlier arrivals of deflected packets in case of small deflection cost is not as likely to happen as the later arrival of a deflected packet in case of large deflection cost. Hence throughput improvement is more significant when the deflection cost is smaller than the best range than that when the deflection cost is larger than the best range. When the deflection cost is in the best range, the overall delays (including both link delay and queueing delay) of the shortest path and the deflection path are close to each other. The *dup ACK*s due to out-of-order packet delivery seldom exceed two, hence the congestion window is not likely to be decreased, leading to a high aggregate throughput. In the following, we present an approximate analysis on the condition under

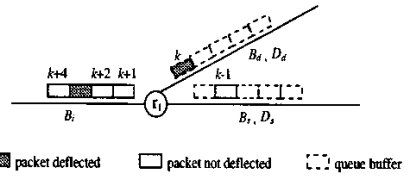

Figure 3: A deflection situation when the deflection cost is large and the deflected packet arrives at the destination later than its following packets via the shortest path.

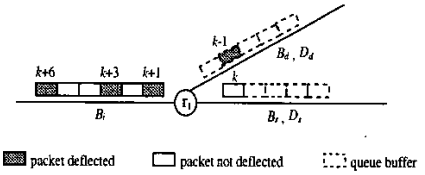

Figure 4: A deflection situation when the deflection cost is small and multiple deflected packets arrive at the destination earlier than the packet before them via the shortest path.

which the throughput "drop" occurs.

We first consider the case when the deflection cost is larger than the best range. An example is shown in Fig. 3, where the deflected packet with sequence number k arrives at the destination later than the three undeflected packets with sequence number $k+1$, $k+2$, and $k+4$ respectively. We assume that n packets have arrived at r_1 when the nth packet is the third packet that is not deflected after k for generality ($n = 4$ in Fig. 3). We define T_0 as the time that packet k arrives at r_1, T_d as the time that packet k arrives at the destination, and T_s as the time that packet $k+n$ arrives at the destination. We then have:

$$T_s = T_0 + \left(\frac{n}{B_i} + \frac{q_s+1}{B_s}\right) S + D_s, \tag{1}$$

and

$$T_d = T_0 + \left(\frac{q_d+1}{B_d}\right) S + D_d, \tag{2}$$

where q_s and q_d are the average queue lengths, as seen by an arriving packet, of link $r_1 - r_2$ and link $r_1 - r_3$ respectively. If $T_d > T_s$, three *dup ACK*s will be generated causing a decrease in the throughput. With the deflection cost defined as $C = D_d - D_s$, $T_d > T_s$ implies

$$C > \left(\frac{q_s+1}{B_s} - \frac{q_d+1}{B_d} + \frac{n}{B_i}\right) S. \tag{3}$$

We now consider the second case when the deflection cost is smaller than the best range. An example is given in Fig. 4, where the deflected packets with sequence number $k+1$, $k+3$, and $k+6$ arrive at the destination earlier than the undeflected packet with sequence number k. Similarly, we assume that m packets have arrived at r_1 when the mth packet is the third packet that is deflected after k, and define T_0 as the time that packet k arrives at r_1, T'_s as the time that packet k arrives at the destination, and T'_d as the time that packet $k+m$ arrives at the destination. We now have:

$$T'_s = T_0 + \left(\frac{q_s+1}{B_s}\right) S + D_s. \tag{4}$$

and

$$T'_d = T_0 + \left(\frac{m}{B_i} + \frac{q_d + 1}{B_d}\right) S + D_d. \quad (5)$$

Three *dup ACK*s will be generated if $T'_d < T'_s$, which implies

$$C < \left(\frac{q_s + 1}{B_s} - \frac{q_d + 1}{B_d} - \frac{m}{B_i}\right) S. \quad (6)$$

In our simulation, B_i = 3Mbps, $B_s = B_d$ = 4Mbps. When retransmission happens, the aggregate throughput is not likely very high, hence the deflection path is not likely congested and therefore $q_d = 0$. On the other hand, the shortest path is congested and we set $q_s = Q - 2$, where Q is the queue size of the link $r_1 - r_2$ and $Q = 20$ in our simulations.[2] Based on the observations in our simulation, we let $n = 3$ and $m = 6$. Now we can get the region of the deflection cost C, in which retransmission and decrease of congestion window may happen: $C > 44$ms or $C < 20$ms. Comparing these two values with the curve in Fig. 2, we can see that the analysis roughly captures the characteristics of deflection routing. Further proof will be given in the following section.

3. DEFLECTION ROUTING WITH ADAPTIVE DEFLECTION POINT

3.1. Motivation

The deflection routing mechanism discussed so far employs tail deflection, i.e., the arriving packet is always deflected if its preferred outgoing link is full. The simulation result and the analysis show that under this scheme deflection routing can substantially improve TCP throughput for a certain range of deflection cost (i.e., the best range). However, the throughput drops when the deflection cost is either higher or lower than the best range. As we have seen that the drop in TCP throughput at low deflection cost is due to the large queueing delay of the undeflected packet in the congested link, this suggests a possible improvement in the scheme to extend the best range given a knowledge of the deflection cost: picking a packet of the same flow already in the queue to deflect while enqueueing the arriving one, since the packet before the deflected one now has a smaller queueing delay.[3] In other words, we move the deflection point forward towards the queue head depending on the deflection cost.

Before describing the method of adaptively determining the deflection point according to the deflection cost, we first examine three schemes, each of which has a fixed deflection point irrespective of the deflection cost. Specifically, we maintain for each flow a sorted list of the packets in the (shared) queue, according to their arrival time. When a new packet arrive at the router and the queue for its preferred outgoing link is full, we may deflect either the arriving packet (if the list is empty) or a packet in the list of this flow (if the list is not empty). Under the three schemes, the deflection points are at the list head, one third from the list head, and two thirds from the list head, respectively. We show in Fig. 5 the performances of the three schemes together with that of the generic

[2]The reason for $q_s = Q - 2$ is that in the implementation of drop-tail queues in *ns*-2, an arriving packet is first enqueued in the buffer, then the buffer is checked whether full. If yes, the packet is dropped. Therefore, the effective queue size is actually $Q - 1$. Since we are looking at an arriving packet that is forwarded along the congested link, it sees $Q - 2$ packets ahead of itself.

[3]Although deflecting a packet of a different flow in the queue also works in the case that all the flows share a single deflection path, it is not applicable in general when the flows have different deflection paths.

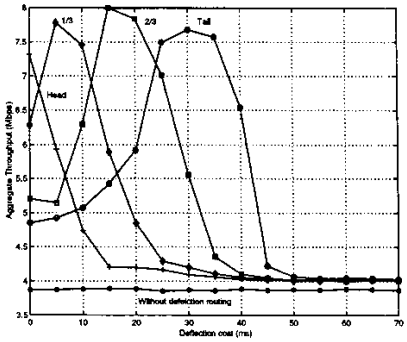

Figure 5: TCP throughput versus deflection cost for four deflection routing schemes with different fixed defleciton points.

tail deflection. We see that moving the deflection point forward has the effect of shifting the performance curve leftward, with differences however the characteristics reserved. In particular, for a fixed deflection point, there exists a best range of deflection cost that very high throughput can be achieved. This suggests that given the deflection cost, it is possible to achieve a high throughput by appropriately selecting the deflection point. We show how the appropriate deflection point can be determined in the following.

3.2. Analysis and Illustrative Simulation Results

We first show that when the deflection point is moved forward toward the queue head, the analysis in Sect. II remains to be valid, if only q_s is redefined as the queue length from the head to the deflection point in the queue. This can be understood just assuming the queue is shortened to the deflection point. For example, when the deflection point is two thirds from the list head, the corresponding deflection point in the shared queue $q_s = \frac{2}{3}(Q - 1) \approx 13$ (with $Q = 20$), assuming that the packets of each flow are roughly uniformly distributed in the queue. Hence the throughput drop condition on the deflection cost is $C > 34$ms and $C < 10$ms according to (3) and (6). This agrees with the simulation result shown in Fig. 5. In the case of head deflection, $C > 8$ms and $C < -16$ms results. Since the negative value has no practical meaning here, only $C > 8$ms makes sense, and it is also verified by the simulation result shown in the figure.

With these proofs, we conclude that the best range of the deflection cost can in general be expressed as $C \in \left[\left(\frac{q_s+1}{B_s} - \frac{q_d+1}{B_d} - \frac{m}{B_i}\right) S, \left(\frac{q_s+1}{B_s} - \frac{q_d+1}{B_d} + \frac{n}{B_i}\right) S\right]$. In this range, high TCP throughput can be achieved. We take the mid-point of the best range as the appropriate relation between the deflection point and deflection cost, i.e.,

$$C = \frac{q_s + 1}{B_s} - \frac{q_d + 1}{B_d} + \frac{n - m}{2B_i}, \quad (7)$$

or, equivalently,

$$q_s = \left(C + \frac{q_d + 1}{B_d} - \frac{n - m}{2B_i}\right) B_s - 1. \quad (8)$$

Given the deflection cost C, q_s tells the deflection point in the

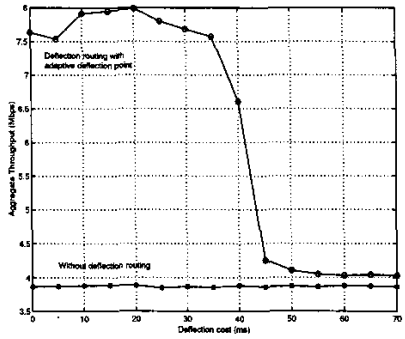

Figure 6: TCP throughput versus deflection cost for the deflection routing scheme with adaptive deflection point.

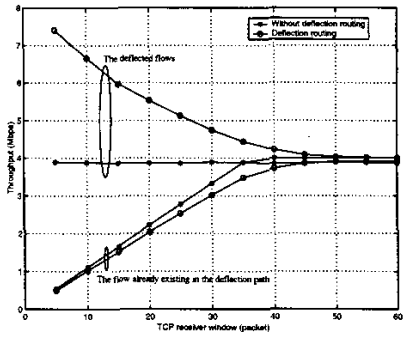

Figure 7: TCP throughput versus the TCP receiver window size of the flow existing in the deflection path.

shared queue.[4] Based on the assumption that the packets of each flow are roughly uniformly distributed in the queue, the packet that should be deflected can be determined as follows. If an arriving packet finds the queue is full and there are l packets of the same flow in the queue (i.e., the list length is l), the $\left[\frac{q_s}{Q-1}l\right]$th packet in the list is deflected, where $[x]$ means the closest integer to x.

We show in Fig. 6 the performance of this deflection routing scheme with adaptive deflection point. We see that it achieves high aggregate throughput as long as the deflection cost is no larger than the best range of the tail deflection scheme. In other words, the best range of the deflection cost has been substantially extended. This is favorable since it means that as long as the deflection cost is not too large, such a deflection routing scheme can effectively exploit the free bandwidth of the deflection path, and hence improve the TCP throughput.

4. FAIRNESS ISSUES

So far, we have considered a rather simple scenario where there is no traffic in the deflection path and hence all the bandwidth is available for the deflected flows. Moreover, we have also considered that all the flows can be deflected in case of congestion. In this section, we examine the fairness issue between the deflected flows and other flows without these conditions.

We first study the effect of the deflected traffic on the existing traffic in the deflection path. We add another TCP flow in our simulation scenario shown in Fig. 1, which has the source r_1 and the destination r_3, representing the existing traffic in the deflection path. The traffic is still FTP, and the start time is also uniformly distributed in $[0.1, 0.2]$ second. The traffic intensity of this flow is adjusted by changing its TCP receiver window. We run simulation with deflection cost $C = 20$ms (i.e., $a = 35$ms) using the deflection routing scheme with adaptive deflection point, and study the throughput of the flows as the TCP receiver window of the new flow increases. The simulation result is shown in Fig. 7. Without deflection routing, the throughput of the new flow increases to the capacity of the deflection path as its TCP receiver window increases, while the aggregate throughput of other flows does not change since they take the shortest path and do not matter with the deflection path. When deflection routing is used, the throughput of the new flow only has a small decrease compared with that when no deflection happens, while the aggregate throughput of the deflected flows decreases as the traffic intensity of the new flow increases. This means that deflection routing only gracefully makes use of the free bandwidth in the deflection path, which is a desirable characteristic.

We next study the problem arising from the possibility that some flows may not be deflected, for a foreseeably possible implementation of deflection routing, either because of the flow specification or the routing entry at the router. We study this partial deflection case by only allowing two of the five flows to be deflected in Fig. 1. All the parameters are not changed, except that the TCP receiver windows of the flows are enlarged to 40 packets to be able to match the bandwidth-delay product. The tail deflection routing scheme is used. The aggregate throughput of the deflected flows and undeflected flows with and without deflection routing are shown in Fig. 8. Without deflection routing, the deflected flows and undeflected flows fairly share the capacity of the shortest path. However, with deflection routing, the deflected flows take away almost all the bandwidth from the undeflected flows, especially when the deflection cost is in the best range. This suggests that for the flows that contend for the same outgoing link at a router, deflection routing should be enabled either for all or for none. This may seem to be a rigid requirement, however, this can be done at the router simply by adding additional entries for each destination in the routing table.

5. CONCLUSIONS

This paper investigates using deflection routing in the Internet as a mechanism for congestion control. The rationale lies in that a packet encountering congestion at a router may still arrive at the destination in time by being temporarily misrouted (i.e., deflected) to a longer path. In this way, the packet avoids being dropped. We have studied the typical performance of deflection routing on TCP throughput in the Internet with a dumbbell topology. The results show that deflection routing substantially improves TCP throughput by taking advantage of the available bandwidth in the deflection path. However, this improvement is achieved only for the deflection cost to be within some region. We have analyzed the underlying reason for the existence of this region, and shown the

[4]We assume that the deflection cost C can be known by the node where deflection happens dynamically, maybe through some control messages.

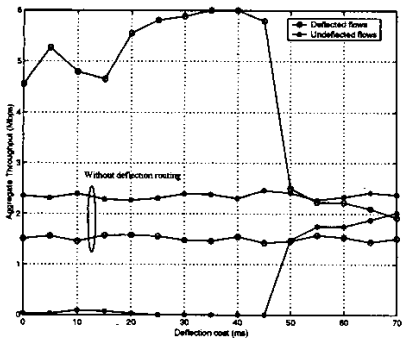

Figure 8: TCP throughput versus deflection cost with only two of the five flows deflected.

relation between this region and the deflection point. In any case, deflection routing can achieve an aggregate throughput at least no less than that without deflection routing, no matter how large the deflection cost is. Based on the analyses we have also proposed and studied a deflection routing scheme with adaptive deflection point, which appropriately determines the deflection point given the knowledge of the deflection cost, hence extending the usefulness of deflection routing to a large range of deflection cost. We have finally studied the fairness issues induced by deflection routing, showing that deflection routing is friendly to the existing traffic in the deflection path. However, the fairness between the contending flows for the congested link should be carefully handled.

In the future, we will address the problems such as the effect of multiple deflection paths, and the deflection strategy when packets may be subjected to multiple deflections.

6. REFERENCES

[1] P. Baran, "On distributed communications networks," *IEEE Trans. Commun. Syst.*, vol. 12, pp. 1–9, Mar. 1964.

[2] A. S. Acampora and S. I. A. Shah, "Multihop lightwave networks: A comparison of store-and-forward and hot-potato routing," in *Proc. IEEE INFOCOM'91*, pp. 10–19, Apr. 1991.

[3] N. F. Maxemchuk, "Comparison of deflection and store-and-forward techniques in the Manhattan street and shuffle-exchange networks," in *Proc. IEEE INFOCOM'89*, pp. 800–809, Apr. 1989.

[4] A. K. Choudhury, "A comparative study of architectures for deflection routing," in *Proc. IEEE GLOBECOM'92*, pp. 1911–1920, Dec. 1992.

[5] F. Borgonovo, L. Fratta, and J. A. Bannister, "Unslotted deflection routing in all-optical networks," in *Proc. IEEE GLOBECOM'93*, pp. 119–125, Nov. 1993.

[6] F. Borgonovo, L. Fratta, and J. A. Bannister, "On the design of optical deflection-routing networks," in *Proc. IEEE INFOCOM'94*, pp. 427–431, Jun. 1994.

[7] F. Forghieri, A. Bononi, and P. R. Prucnal, "Analysis and comparison of hot-potato and single-buffer deflection routing in very high bit rate optical mesh networks," *IEEE Transactions on Communications*, vol. 43, pp. 88–98, Jan. 1995.

[8] S.-H. G. Chan and H. Kobayashi, "Packet scheduling algorithms and performance of a buffered shufflenet with deflection routing," *Journal of Lightwave Technology*, vol. 18, pp. 490–501, Apr. 2000.

[9] E. Leonardi, F. Neri, M. Gerla, and P. Palnati, "Congestion control in asynchronous, high-speed wormhole routing networks," *IEEE Communications Magazine*, vol. 11, pp. 58–69, Nov. 1996.

[10] C. Roche, P. Palnati, M. Gerla, F. Neri, and E. Leonardi, "Performance of congestion control mechanisms in wormhole routing networks," in *Proc. IEEE INFOCOM'97*, pp. 1365–1372, Apr. 1997.

[11] A. Bianco, E. Leonardi, M. Munafo, and F. Neri, "Performance of TCP connections in wormhole routing and ATM networks," in *Proc. 2000 Symposium on Performance Evaluation of Computer and Telecommunication Systems*, pp. 171–184, Jul. 2000.

[12] http://www.isi.edu/nsnam/ns.