Literature Review: Adaptive Analysis of High-Speed Router Performance in Packet-Switched Networks

N.M. Mosharaf Kabir Chowdhury

July 24, 2007

1 Introduction

The Internet is a global, publicly accessible, complex network of interconnected computer networks that uses the standard Internet Protocol(IP) to transmit data by dividing it into smaller units, called packets. These packets pass through routers, that connect separate networks, to reach their destinations. Whenever a packet reaches a router, that router has to decide on the next router on the path toward the destination of that packet. Even though it takes just a simple lookup operation in the 'router table', it becomes significant when the arrival rate of packets is larger than the speed of a lookup operation. This is exactly the case in the Internet backbone where high-speed routers are employed to handle this situation. The goal of this project is to establish an appropriate model to address this issue and to perform an adaptive analysis of the performance of these high-speed routers based on the 'niceness' of the input packet sequence. We say an input sequence is 'nicer' than another input sequence if the former one has fewer number of changes in packet destinations in consecutive packets of the sequence i.e. the total number of lookup operations needed is fewer. To the best of our knowledge, no such analysis have been done before in this context. In this literature review we summarize all the works related to problem. We review the framework, packet-switched networks in general, in Section 2, existing packet scheduling algorithms in Section 3 and finally we summarize the recent results in the differentiated services frameworks in Section 4. In Section 5, we give a formal description of our tentative model.

2 Packet-Switched Networks

A packet-switched network is an interconnected set of networks that are joined by routers. Each message is divided into smaller parts or packets which are routed separately using the address in their headers between nodes over data links shared with other traffic. Once all the packets forming a message arrive at the destination, they are merged together to form the original message. Unlike a circuit-switched network, which sets up a constant bit rate and constant delay connection

between the two nodes for their exclusive use for the duration of the communication, a packet-switched network does not guarantee anything and packets are queued or buffered and in some cases dropped in different nodes resulting in variable delay. Packet-switched networks are also known as *best-effort* networks.

When moving from one node to another using a data link in a packet-switched network, each packet uses the full link bandwidth. Hence it is not possible to physically divide the link between different data streams. Instead, *statistical multiplexing technique*, also known as *dynamic bandwidth allocation method*, is employed in routers to divide a physical communication channel into an arbitrary number of logical, variable bit-rate channels or data streams. There are some variants of packet-switched networks that use simple FIFO (First-In First-Out) order to advance packets through these streams. On the other hand, sophisticated forwarding techniques are employed in other variants, that use scheduling disciplines for *Fair Queuing* or *Differentiated* and/or *Guaranteed* QoS (Quality of Service).

Packet switching can be categorized into *datagram networks* (also known as *connection-less*) such as Ethernet and IP networks (Internet) and *virtual circuit switching* (also known as *connection-oriented*) like ATM, X.25 and Frame relay. We will mainly consider the performance of routers in the first category i.e. in the Internet, as it is the biggest of all packet-switched networks.

2.1 Routers and Routing

A router is a device that interconnects several logical subnets and determines the proper path for packets to travel between different networks. It can be seen as a specialized computer specifically tailored to perform the tasks of *routing* and *forwarding*. Most modern routers use specialized operating system (e.g. Cisco's IOS or Extreme Networks XOS) and multiple processors. High-end routers contain many processors and specialized ASICs (Application-Specific Integrated Circuits) and have to do a lot of parallel processing to manage the huge amount of packets that pass through them. For example, routers in the Internet backbone face packets arriving at a staggering rate of one packet per 8 nano second.

Routing is the process of selecting paths in a network while forwarding can be viewed as the relaying of packets through that path. Routing process usually directs forwarding on the basis of routing tables which maintain a record of the routes indexed by various network destinations. Whenever a packet arrives, it takes place in the input queue of the router as long as there is an empty space in the queue; otherwise it is dropped. The router continuously takes out packets from the queue and reads their header information to find their destination and then forwards it through the appropriate output link consulting the routing table. Even though this lookup operation is very fast (approximately 180 nano second for high-end routers), it's still much slower in comparison with the rate of arrival of packets.

2.2 Routing in the Internet

Traditional IP routing is relatively simple because it uses *next-hop* routing where the router only needs to consider the next destination of the packet, and does not need to consider the subsequent path of the packet on the remaining hops toward its actual destination. In this case, the performance of a router actually depends on how fast it can perform a lookup operation. A lot of efforts, both in hardware and software, have been given throughout the years to speedup this process.

Instead of increasing the speed of the lookup operation, performance of the router can also be improved by decreasing the total number of lookup operations that are necessary on a given sequence of packets by properly scheduling the packets and dropping some of them if needed. Dropped packets will be automatically retransmitted from the source at a later time. Many of these packet scheduling algorithms have been studied under the context of QoS management.

3 Packet Scheduling Algorithms

Every router has a packet scheduler that plays the important role of determining the order of packets belonging to a particular flow to effectively ensure that the QoS service requirements are met. FIFO and Round Robin(RR)[18] are the simplest of all packet scheduling algorithms that can be employed. These two algorithms share one common problem; they cannot guarantee fair allocation of resources in case of contention. To handle this fairness issue some other scheduling algorithms have been proposed over the years. Fair Queuing(FQ)[12, 18], Weighted Fair Queuing(WFQ)[12], Generalized Processor Sharing(GPS)[1], Deficit Round Robin(DRR)[20], Worst-case Fair Weighted Fair Queuing(WF^2Q)[8], Fair Queuing with Round Robin(FQRR)[19] and Group Round Robin(GRR)[10] are some of those.

3.1 First-In First-Out (FIFO)

FIFO is the trivial packet scheduling approach. As the name suggests, all the packets that come in are put in the output link in exactly the same order they arrived. In case of congestions it drops packets but do not consider fairness issues.

3.2 Round Robin(RR)[18]

The idea of RR is pretty simple. For each host it keeps separate queues and put one packet from each queue, if available, into the output link in a round robin manner. It meets the Max-Min fairness criterion if all the packets have same size. Max-Min criterion defines an allocation to be fair if -

1. No user receives more than its request,
2. No other allocation scheme satisfying condition 1 has a higher minimum allocation, and

3. Condition 2 remains recursively true as we remove the minimal user and reduce the total resource accordingly.

3.3 Fair Queuing(FQ)[12, 18]

In order to decide which packet should be forwarded first, FQ estimates a *virtual* finishing time of all candidate packets (i.e., the packets at the head of all non-empty queues), based on the arrival time of the packet and the number of users who are sharing the buffer. Then FQ compares the virtual finishing time and selects the minimum one to be forwarded. It achieves Max-Min fairness without any restriction on packet sizes.

With a link data rate of R at any given time, N active data flows (the ones with non-empty queues) are serviced simultaneously at an average data rate of $\frac{R}{N}$.

3.4 Weighted Fair Queuing(WFQ)[12]

WFQ is a generalization of FQ. Contrary to FQ, WFQ allows different sessions to have different service shares. If N data flows are currently active, with weights $w_1, w_2, \ldots, w_N$, then the i^{th} data flow will achieve an average data rate of $\frac{Rw_i}{(w_1+w_2+\ldots+w_N)}$.

By regulating the WFQ weights dynamically, it can be utilized for controlling the Quality of Service.

3.5 Generalized Processor Sharing(GPS)[1]

GPS can be described as RR scheduling in bit level. It assumes that the network traffic is fluid, so whenever an application has packets in the queue, it will receive a fraction of the server processor. Even though it is impractical to implement a scheduling algorithm at that granularity, GPS is useful for benchmarking of practical algorithms. It is also possible to approximate the GPS conception with clever packet-based service disciplines. For example, WFQ has a close GPS approximation known as Packet-by-Packet Generalized Processor Sharing (PGPS)[1].

3.6 Deficit Round Robin(DRR)[20]

DRR uses stochastic fair queuing to assign flows to queues and employs round robin scheduling with a pre-specified quantum to serve those queues. Its difference from traditional RR algorithm is that if a queue was not able to send a packet in the previous round because its packet size was too large, the remainder from the previous quantum is added to the quantum for the next round and thus deficits are kept track off.

It achieves nearly perfect fairness in terms of throughput with $O(1)$ processing per packet in contrast to the $O(\log N)$ processing complexity in FQ related algorithms.

3.7 Worst-case Fair Weighted Fair Queuing(WF^2Q)[8]

WF^2Q is a closer approximation to GPS differing only in the maximum packet size. It is shown that there can be large discrepancies between WFQ or PGPS and GPS schedulers. WF^2Q shares both the bounded-delay and worst-case fairness properties of GPS.

4 Differentiated Services or DiffServ

DiffServ[9] is a simple, coarse-grained mechanism for classifying and managing network traffic and providing QoS guarantees on modern IP networks. It was developed to overcome the problems of *best-effort* networks that exist today and fail to support guaranteed QoS. DiffServ allows classification of packets and marks them using DS field of a IP Packet to receive a particular per-hop forwarding behavior on nodes along their path. Thus it provides a simple framework that, if intelligently used, can provide better QoS. All the sophisticated marking and classifying policies are implemented at network boundaries.

Since the inception of DiffServ, a lot of research have been done on the buffer management policies of routers in the context of DiffServ. Most of them uses the concept of DS field and hence assume that there are packets of two different priorities. Based on this assumption, many existing packet dropping policies have been analyzed and their competitive ratios(CRs) have been compared to the optimal omniscient algorithm. In later years, multi-priority models have also been analyzed. Most important of these models are *Non-preemptive* model, *FIFO Preemptive* model and *Bounded Delay* model. All of these models initially considered single queue or buffer, which was later extended to multiple queue case. The objective of these algorithms is to maximize the total value of the output packets, where a packet has a value proportional to its priority.

4.1 Different Models

In FIFO model, sequence of transmitted packets has to be a subsequence of the arriving packets. *Non-preemptive FIFO* model does not allow dropping of a packet from a buffer to accommodate a packet, that has just arrived, with a higher value. *FIFO Preemptive* model, on the other hand, is much more liberal. It allows to drop packets with lower value from the buffer to accommodate fresh, higher valued packets at the end.

In the *bounded delay* model, for each packet p there is a specific time within which it must be removed from the buffer, either delivered or dropped. This time is also known as the *deadline* of the packet. And the most significant difference of this model from FIFO model is that packets can be re-ordered in the buffer. *s-uniform bounded delay* model is a special case of the normal bounded delay model where deadlines of all the packets are bounded by s.

4.2 Results for Single Queue Model

In single queue model there is only one queue or buffer where packets can be stored before they are delivered to a output link or dropped internally. It has been studied under non-preemptive model, preemptive FIFO model and s-uniform bounded delay model and there are already some tight upper and lower bounds on some of the algorithms considered under these models.

4.2.1 Aiello et al. (INFOCOM '00)[2]

In this model a benefit(value) of $\alpha \geq 1$ is given to each high priority packet and benefit of 1 is given to each low priority model. An upper bound on the CR of non-preemptive FIFO model is established, which is $(2 - \frac{1}{\alpha})$.

4.2.2 Kesselman et al. (STOC '01)[15]

They propose a simple *greedy* algorithm that drops the earliest packet among all the lower value packets. This idea is completely opposite to the *tail-drop* policy currently in use. For their analysis, they consider $\alpha(\geq 0)$ as the ratio of maximum to minimum value of a packet. The results are,

- FIFO Model
  Tight bounds of $(2 - \frac{1}{\alpha})$ on the CR of the greedy algorithm is given and it's also shown that this greedy policy is 1.5 times better than the tail-drop policy.

- Bounded Delay Model
  CR of any on-line algorithm for a uniform bounded delay buffer is shown to be bounded away from 1, independent of the delay size. And for general case, greedy algorithm has CR = 2. For more specific cases, it is shown that in 2-value model greedy algorithm has CR of $(1 + \frac{1}{\alpha})$, and in 2-uniform bounded delay model it has upper bound of CR to be $\phi(1.618)$.

- Lower bounds on the competitive ratio for uniform delay model is shown as 1.11, for bounded delay model it is 1.17 and for 2-uniform bounded delay model with unit bandwidth, CR has upper bound of 1.43. In 2-uniform case, lower bound on CR is 1.25 and in 2-variable model it is 1.414.

4.2.3 Andelman et al. (SODA '03)[3]

- Non-preemptive Model
  Competitive ratio of $\Theta(\log \alpha)$ is given and for 2-value model it is $\frac{2\alpha-1}{\alpha}$.

- FIFO Preemptive Model
  For buffer size(B) of 2, CR of the FIFO preemptive model is shown to be 1.434.

- Bounded Delay Model
  In the 2-uniform bounded delay model lower and upper bounds of CR is proved as 1.366 and

1.414, which are improvements over 1.25 and 1.43 respectively, presented by Kesselman et al.[15]. Also in 2-variable model, a tighter lower bound of CR is given, which is ϕ and matches the upper bound of [15].

4.2.4 Kesselman et al. (ESA '03)[16]

In this paper, they present the first algorithm for the FIFO model with arbitrary packet values that achieves a competitive ratio better than 2; $(2 - \epsilon)$ to be exact, for a constant $\epsilon > 0$. They achieved this by slightly changing the greedy algorithm[15] by adding a threshold while preempting a packet.

4.2.5 Bansal et al. (ICALP '04)[6]

They further improved the result of [16] to 1.75 by taking into account some more information along with the threshold. One interesting aspect of their analysis is the use of linear programming theories to reduce the instance space.

4.2.6 Bartal et al. (STACS '04)[7]

The most important aspects of this paper and the paper after this one is the reduction of this problem to job scheduling problem. This is the first paper where a randomized algorithm for general case is given with a competitive ratio of 1.582. Also, an algorithm for 2-bounded delay model is given that has CR 1.25 with matching lower bound.

4.2.7 Chrobak et al. (ESA '04)[11]

They presented a analysis of s-uniform bounded delay model with an accompanying 1.838 CR algorithm. But this result is not applicable to FIFO model. For 2-uniform model, they proposed a 1.377 CR algorithm with matching lower bound.

4.2.8 Li et al. (SODA '05)[17]

Competitve ratio of s-uniform model is further improved by analyzing the agreeable deadline model, which is a generalization of s-uniform model and an algorithm with competitive ratio ϕ is presented. As a result, CR of s-uniform model also improved.

4.2.9 Englert and Westermann (ESA '06)[13]

For 2-value bounded delay model they presented an optimal algorithm with CR 1.282, if the buffer size is infinite and for a finite buffer it has a CR of 1.303. Finally, for the FIFO preemptive greedy model they gave tight upper bound 1.732 and lower bound 1.707 of CR improving from ϕ.

4.3 Results for Multiple Queue Model

Unlike the single queue model, this model has more than one queue or buffer where packets can be stored before they are delivered to a output link. In this case the problem actually consists of two dependent sub-problems, *admission control* or *buffer management*, i.e. which packet to admit into buffer and *scheduling* between different buffers to single output port.

4.3.1 Azar and Richter (STOC'03)[5]

- They presented a generic technique to transform any buffer management policy for a single queue (both preemptive and non-preemptive) to a scheduling and buffer management algorithm (preemptive and non-preemptive, respectively) for a switch with m queues, where CR will be twice the single queue case.
- They also provided a general 3.5 CR algorithm for m queues and a $\frac{e}{e-1}$ CR randomized algorithm for unit-value case.

4.3.2 Azar and Richter (STOC'04)[4]

Here they presented a general 3 CR algorithm for m queues which is an improvement over their 3.5 CR previous algorithm[5]. One of the most important contribution of this paper is the *Zero-One Principle*, which is quoted below -

Zero-One Principle: *In order to prove that an algorithm achieves c-approximation it is sufficient to prove that it achieves c-approximation with respect to sequences composed solely of 0s and 1s, where ties between packets with equal values, i.e. between 0 to 0 and 1 to 1, are broken in all possible ways (even if the algorithm breaks ties in a certain way).*

4.3.3 Itoh and Takabashi (IEICE'06)[14]

In this paper they provide the best known upper bound for the CR of multi-queue preemptive model which is $(3 - \frac{\alpha}{\alpha})$, where $\alpha \geq 1$. They also show that the CR of any multi-queue preemptive QoS algorithm is at least 1.514.

5 Model for Adaptive Analysis

In this section, we present a tentative generalized model to analyze the performance of a high-speed router. We define, *input queue* as a link that supplies packets into a router and *buffer* as a collection of internal spaces in the router where packets can be saved for rearranging. We use packet *type* to denote packets with different destinations. All the packets that have same destination are considered to have same type. And there is *cache*, which is very high speed small-amount of memory to hold a portion of the routing table.

Let us assume, there are N_Q input queues into a router, $\mathcal{R}$. $\mathcal{R}$ has N_B buffers, each with spaces to hold B_i packets ($1 \leq i \leq N_B$). Also there are N_T types of packets, $\{p_1, p_2, \ldots, p_{N_T}\}$ in the system under consideration and a boolean variable B_D which controls the permission to drop a packet from any of the buffers. It should be noted that we only consider the packets that are dropped inside the router, not those that are dropped because input queue was full. D denotes the number of such dropped packets and there is also an upper limit, D_{MAX} which defines how many packet drops can be tolerated without hampering the performance. If $D > D_{MAX}$, there is a cost associated with each of the $(D - D_{MAX})$ dropped packets. And N_C is the size of the high speed cache i.e. how many entries from the routing table the cache can hold.

Finally, the term *flip* is defined as the change of packet types in two consecutive packets in the input packet sequence, $\mathcal{S}$ and denoted by F. So the lower the number of flips the fewer lookup operations are needed.

So, the performance of the router, $\mathcal{R}$ can be expressed in terms of the number of flips (F) it makes and the number of packets $(D - D_{MAX})$ it drops over the upper limit.

References

[1] Robert G. Gallager Abhay K. Parekh. A generalized processor sharing approach to flow control in integrated services networks: The single-node case. *IEEE/ACM Transactions on Networking*, 1(3):344–357, 1993.

[2] William Aiello, Yishay Mansour, S. Rajagopolan, and Adi Rosen. Competitive queueing policies for differentiated services. In *INFOCOM*, pages 431–440, 2000.

[3] N. Andelman, Y. Mansour, and A. Zhu. Competitive queueing policies for QoS switches. In *Proceedings of the 14th ACM-SIAM Symposium on Discrete Algorithms(SODA'03)*, pages 761–770, 2003.

[4] Y. Azar and Y. Richter. The zero-one principle for switching networks. In *Proceedings of ACM Symposium on Theory of Computing(STOC'04)*, pages 64–71, 2004.

[5] Y. Azar and Y. Richter. Management of multi-queue switches in QoS networks. *Algorithmica*, 43:81–96, 2005.

[6] Nikhil Bansal, Lisa K Fleischer, Tracy Kimbrel, Mohammad Mahdian, Baruch Schieber, and Maxim Sviridenko. Further improvements in competitive guarantees for QoS buffering. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP'04)*, pages 196–207, 2004.

[7] Y. Bartal, F. Chin, M. Chrobak, S. Fung, W. Jawor, R. Lavi, J. Sgall, and T. Tichy. Online competitive algorithms for maximizing weighted throughput of unit jobs. In *Proceedings of the 21st Symposium on Theoretical Aspects of Computer Science (STACS'04)*, 2004.

[8] J. C. R. Bennet and H. Zhang. WF^2Q: Worst-case fair weighted queueing. In *Proceedings of IEEE INFOCOM'96*, pages 120–128, 1996.

[9] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated service, 1998.

[10] Bogdan Caprita, Jason Nieh, and Wong Chun Chan. Group round robin: Improving the fairness and complexity of packet scheduling. In *Proceedings of the 2005 Symposium on Architecture for Networking and Communications Systems*, pages 29–40, 2005.

[11] Marek Chrobak, Wojciech Jawor, Jiri Sgall, and Tomas Tichy. Improved online algorithms for buffer management in QoS switches. In *Proceedings of the 12th Annual Europian Symposium on Algorihtms(ESA'04)*, pages 204–215, 2004.

[12] A. Demers, S. Keshav, and S. Shenkar. Analysis and simulation of a fair queuing algorithm. *Journal of Internetworking Research and Experience*, pages 3–26, 1990.

[13] M. Englert and M. Westermann. Lower and upper bounds on FIFO buffer management in QoS switches. In *Proceedings of the 14th Annual European Symposium on Algorithms(ESA'06)*, pages 352–363, 2006.

[14] Toshiya Itoh and Noriyuki Takahashi. Competitive analysis of multi-queue preemptive qos algorithms for general priorities. *IEICE Transaction Fundamentals*, E89-A(5):1186–1197, 2006.

[15] Alexander Kesselman, Zvi Lotker, Yishay Mansour, Boaz Patt-Shamir, Baruch Schieber, and Maxim Sviridenko. Buffer overflow management in QoS switches. In *Proceedings of ACM Symposium on Theory of Computing(STOC'01)*, pages 520–529, 2001.

[16] Alexander Kesselman, Yishay Mansour, and Rob Van Stee. Improved competitive guarantees for QoS buffering. In *Proceedings of the 11th Annual Europian Symposium on Algorihtms(ESA'03)*, pages 361–372, 2003.

[17] Fei Li, Jay Sethuraman, and Clifford Stein. An optimal online algorithm for packet scheduling with agreeable deadlines. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms(SODA '05)*, pages 801–802, 2005.

[18] J. B. Nagle. On packet switches with infinite storage. *IEEE Transactions on Communications*, 35(4):435–438, 1987.

[19] A. Sen, L. Mohammed, R. Samprathi, and S. Bandyopadhyay. Fair queuing with round robin: A new packet scheduling algorithm for routers. In *Proceedings of the Seventh International Symposium on Computers and Communications*, pages 120–128, 2002.

[20] M. Shreedhar and G. Varghese. Efficient fair queuing using deficit round robin. *IEEE/ACM Transactions on Networking*, 4(3):375–385, 1996.