INTERNATIONAL JOURNAL OF PURE AND APPLIED RESEARCH IN ENGINEERING AND TECHNOLOGY

A PATH FOR HORIZING YOUR INNOVATIVE WORK

QUEUING DISCIPLINE

SHRAYU P. PACHGADE[1], PROF .K.G. BAGDE [2]

1. First year student M.E, Department of Computer Science & engineering H.V.P.M's college of Engineering & Technology Amravati, India
2. Asst. Professor, Department of Computer Science & engineering H.V.P.M's college of Engineering & Technology Amravati, India.

Accepted Date:

27/02/2013

Publish Date:

01/04/2013

Keywords

First-in-first-out (FIFO),
Fair Queuing (FQ),
Routers,
Bit-Round Fair Queuing (BRFQ),
Weighted Fair Queuing (WFQ).

Corresponding Author

Mr. Shrayu P. Pachgade

Abstract

This paper deals with queuing disciplines for a class of wide-area virtual- circuit data networks. Messages on each virtual circuit are transmitted logically as bit-by-bit. A small number of successive bytes are collected into a packet and provided with an address header. Packets from different virtual circuits are intermingled on trunk and are relayed from one node of the network to another. While protocol processing and flow control are handled at the network edges. In this paper we compare the performance of first-in-first-out and round-robin packet service discipline at the trunk node, including, if desired, priority service for single packet messages. Round-robin disciplines have better fairness properties than first-in-first-out disciplines. When the network is congested, in that they protect well-behaved users against the demands of uncontrolled users

INTRODUCTION:

An important component of an Integrated Service Architecture implementation is the queuing discipline used at routers. Routers traditionally have used a first-in-first-out (FIFO) queuing discipline, also known as first-come-first-served (FCFS), at each output port. A single queue is maintained at each output port. When a new packet arrives and is routed to an output port, it is placed at the end of the queue. As long as the queue is not empty, the router transmits packets from the queue, taking the oldest remaining packet next. There are several drawbacks to the FIFO queuing discipline:No special treatment is given to packets from flows that are of higher priority or are more delay sensitive. If a number of packets from different flows are ready to forward, they are handled strictly in FIFO order. IF a number of smaller packets are queued behind a long packet, then FIFO queuing results in a larger average delay pr packet than if the shorter packets were transmitted before the longer packet. In general, flows of larger packets get better service. A greedy TCP connection can crowd out more altruistic connections. If congestion occurs and one TCP connection fails to back off, other connections along the same path segment must back off more than they would otherwise have to do.

FAIR QUEUING (FQ):

To overcome some of the drawbacks of FIFO queuing, Nagle proposed a scheme called fair queuing [NAGL87][4]. In this scheme, a router maintains multiple queues at each output port. Nagle suggested maintaining one queue for each source(fig); it would also be possible to maintain one queue for each flow.

With fair queuing, each incoming packet is placed in the appropriate queue. The queue are serviced is round-robin fashion, taking one packet from each non-empty queue in turn. Empty queues are skipped over.

This scheme is fair in that each busy flow gets to send exactly one packet per cycle. Further, this is the form of load balancing among the various flows. Also note that there is no advantage in being greedy. A greedy flow finds that its queue becomes long, increasing its delay, whereas other flows are unaffected by this behavior.

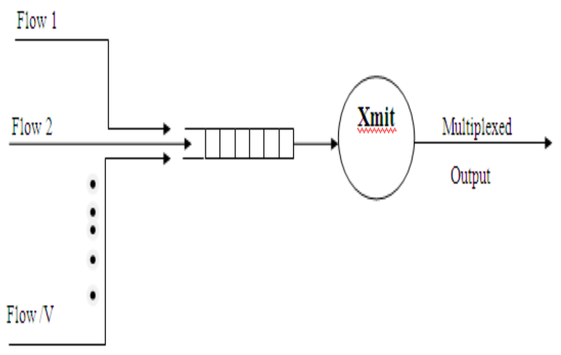

Fig.(a) FIFO queuing

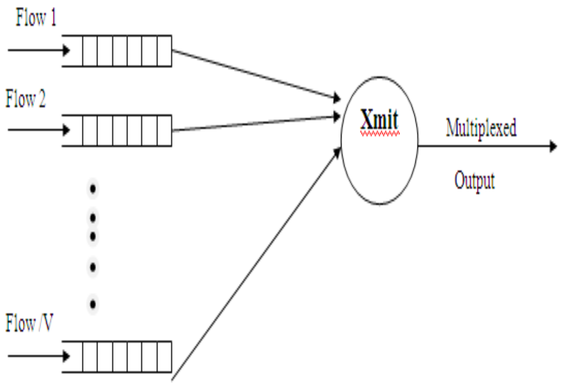

Fig.(b) Fair queuing

PROCESS SHARING (PS):

A serious drawback to the fair queuing scheme is that short packets are penalized: More capacity goes to flow with a longer average packet size compared to flows with shorter average packet size. The reason for this is that each queue transmits one packet per cycle.

This disadvantage is overcome with bit-round fair queuing (BRFQ), which uses packet length as well as flow identification to schedule packets; this technique is described in [DEME90][7].

To understand BRFQ let us first consider an ideal policy that is not practical to implement. Set up the multiple queues, as in FQ, but now transmit only one bit from each queue on each round. In this way, longer packets no longer receive an advantage, and each busy source receives exactly the same amount of capacity. I particular, if there are N queues always has a packet to send. Then each queue receives exactly 1/N of the available capacity.

This bit-by-bit capacity known as process sharing (PS). To understand what follows, it will be useful o define some terms:

R(t) = number of rounds in the PS service discipline that have occurred up to time t, normalized to the output data rate

N(t) = number of nonempty queues at time t

P_i^α = transmission time for packet I in queue α, normalized to the output data rate

T_i^α = arrival time for packet i in queue α

S_i^α = value of R(t) when packet I in queue α begins transmission

F_i^α = value of R(t) when packet I in queue α ends transmission

We can think of R(t) as a virtual time, which records the rate of service seen by the packet at the head of a queue. An equivalent definition is as follows:

$$R'(t)=\frac{d}{dt}R(t)=\frac{1}{max[1,N(t)]}$$

The following recurrence relationships summarize how the PS system evolves in virtual time:

$$F_i^\alpha = S_i^\alpha + P_i^\alpha$$

$$S_i^\alpha=\max[F^\alpha_{i-1}, R(T_i^\alpha)]$$

(Eqn.1)

Note that from these equations, we can compute a packet's virtual finishing time the moment it arrives. However, we cannot compute the packet's real finishing time on its arrival because the real finishing time depends on future arrivals.

BIT-ROUND FAIR QUEUING (BRFQ):

We wish to transmit entire packets rather than individual bits. BRFQ is design to emulate a bit-by-bit round-robin discipline. BRFQ is implemented by computing virtual starting and finishing times on the fly as if PS were running. The BRFQ rule simply this: Whenever a packet finishes transmission, the next packet sent is the one with the smallest value of F_i^α.

GENERALIZED PROCESSOR SHARING (GPS):

BRFQ is an improvement over FQ or FIFO in that it fairly allocates the available capacity among all activity flow through a node. However, it is not able to provide different amounts of the capacity of different flows. To support QoS transport, this differential allocation capabilities is needed. The approach that is received the widest acceptance is an enhancement of BRFQ known as weighted fair queuing (WFQ). Again, it will clarify the description if we first look at a bit-by-bit round-robin version.

To take into account the differing demands of different sources, we can generalize the PS discipline to allow for arbitrary capacity allocations. With GPS, each flow α is assigned a weight ϕ_α that determines how many bits are transmitted from that queue during each round. Thus, if the weight for a given flow is 5, then during each round that the queue is nonempty, 5 bits will be transmitted. Some thought should convince you that we can model this process by modified equation (Eqn.1) as follows:

$$F_i^\alpha = S_i^\alpha + (P_i^\alpha / \Phi_\alpha)$$

$$S_i^\alpha = \max[F^\alpha_{i-1}, R(T_i^\alpha)]$$ **(Eqn. 2)**

In effect, we set the effective packet length to $1/\phi_\alpha$ times the true packet length. It is easy to see that, at any given time, the service rate g_i for a nonempty flow I is

$$g_i = (\Phi i / \textstyle\sum_j \Phi j) C$$ **(Eqn. 3)**

Where the sum is taken over all active queues and C is the outgoing link data rate.

GPS is attractive because it provides a means of responding to different service requests. If a source requests a given service rate g_i for a flow, then the node can grant the request if sufficient capacity is available and can assign the proper weight to guarantee the service. Equally important, GPS provides a way of guaranteeing that delays for a well-behaved flow do not exceed some bound. Consider a set of flows that are defined by , and limited to, the token bucket specification described in section fig.1, where B_i and R_i are the bucket size and token rate, respectively, for flow i. Now let the weight assigned to each flow equal to token rate; that is $\phi_i = R_i$. Then the maximum delay experienced by flow I, D_i, is bounded by

$$D_i \leq (B_i / R_i)$$ **(Eqn. 4)**

The proof of this is given in [PARE94][5]; here we give an intuitive argument. Assume a situation in which all of the flows have been low or idle for some time and that all of the buckets full. Then all of the flows begin to transmit at the maximum rate. The network is configured, by reservation, to handle the maximum rate R_i from each flow. At this rate, tokens are added to the bucket as fast as they are drained. If the node keeps up with the flow, then the queue length at the node will not exceed the bucket size. So the delay experienced by the flow through the node will not exceed the bucket size divided by the token rate, which is Equation (Eqn.4).

WEIGHTED FAIR QUEUING (WFQ):

Again, we wish to transmit entire packets rather than individual bits. Just as BRFQ emulates the bit-by-bit PS, WFQ emulates the bit-by-bit GPS. The strategy is the same: Whenever a packet finishes transmission, the next packet sent is the one with the smallest value of F_i^α . In this case Equation (Eqn.2) governs the calculation of F_i^α. Equation (Eqn.4) must be modified as follows:

$$D_i \leq ((B_i/R_i) + (((K_i - 1)\, L_i)/R_i) + \sum_{m=1}^{Ki}(\mathrm{Lmax}/\mathrm{Cm}))$$

(Eqn. 5)

Where,

D_i = maximum delay experienced by flow i

B_i = token bucket size for flow i

R_i = token rate for flow i

K_i = number of nodes in the path flow I through the internet

L_i = maximum packet size for flow i

L_{max}= maximum packet length for all flows through all nodes on the path of flow i

C_m = outgoing link capacity at node m

The first term carries over from the GPS case and accounts for delay due to bucket size, which is the same as delay due to burstiness. The second term is proportional to the delay experienced at each node for each packet by this flow. The final term reflects the consequence of packet- by-packet rather than bit-by-bit transmission. Again, we give an intuitive explanation. In both BRFQ and WFQ, a packet may leave some time later than it would have under bit-by-bit processing (PS or GPS). The reason is that the node can only choose for transmission among all the packets that have already arrived. If the next packet is chosen relatively large and small packet arrives during this transmission, it may be that under GPS, this small packet has the earliest finish time and therefore should have been transmitted first under WFQ.

Because it did not arrive, it was transmitted and has to be delayed up to at most the full length of the longest packet that moves through this node.

Equation (Eqn.5) is important in the design of an Integrated Services Architecture (ISA). It says that there is an easy way to set parameters in router to guarantee a given rate of service. Further, at this rate, an upper bound on delay can be granted to the user. Finally, [PARE94][5] shows that the maximum queue size needed at each node is proportional to the maximum delay defined in Equation (Eqn.5); in particular, it approaches g_iD_i. Thus, the node can easily determine the resources required to grant a particular reservation.

CONCLUSION:

We have studied the queuing discipline of FIFO and round-robin type of bit-by-bit data networks. The principle conclusion are as follows, under normal traffic, high speed trunk substantially reduce queuing delays. Almost any queuing discipline will give acceptable delay if the backbone network is enough faster than the access lines. Both the packet FIFO and the round robin discipline can be augmented with a priority queue with expedites single-packet message, which may carry network control signals echoplex characters. In discipline of FIFO type, the mean delays of messages do not go through the priority queue depend on the overall message length distribution. The sprinkling of the very long message can significantly increase the mean delay of other messages. In discipline of round-robin type, the mean delay of each message type is not affected by the presence of very long message of other type.

REFERENCES:

1. High Speed Networks and Internets: (Performance and Quality Service), William Stallings (Second Edition).

2. Armitage, G. Quality of Services in IP Networks. Indianapolis, IN: Macmillan Technical Publishing, 2000.

3. Greenberg, A., and Madras, N. “How Fair is Fair Queuing?” Journal of the ACM, July 1992.

4. Nagle, J. “On Packet Switches with Infinite Storage.” IEEE Transactions on Communications, April 1987.

5. Parekh, A., and Gallager, G. "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Multiple Node Case." IEEE/ACM Transaction on Networking, April 1994.

6. Zhang, H. "Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks." Proceedings of the IEEE, October 1995.

7. Demers,A.; Keshav, S.; and Shenkar, S. "Analysis and simulation of a Fair Queueing Algorithm." Internetworking: Research and Experience Sept.1990.