CS-534: Packet Switch Architecture
Spring 2008

Department of Computer Science

Exercise Set 11:
Flow Control, Per-Flow Queueing

Assigned: (norm.: wk.11) 21/5/2008 (wk.12) - Due: (normally: wk.12) Wed. 4 June 2008 (wk.F)

11.1 Window Size Calculation

Compute the "window size" (throughput times round-trip-time), in bytes, for a flow passing through a link in the 48 = 4 x 6 x 2 combinations of cases for the following three parameters (make two tables, *(i)* and *(ii)*, with 4 columns (flow peak rate) and 6 rows (hop distance) each):

- The desirable peak rate of the flow is: (a) 10 Mbps, (b) 100 Mbps, (c) 1 Gbps, (d) 10 Gbps;
- The link (hop) distance is: (1) 10 cm (two adjacent chips on the same board), (2) 1 m (two chips in the same box), (3) 10 m (SAN), (4) 1 km (LAN), (5) 10 km (MAN), (6) 1000 km (WAN). In all cases, assume that the speed of light in the link medium is 200 Mm/s.
- The credit-packet processing time, to be added to the link round trip time, is as follows. At the downstream switch, the credit departs (in the upstream direction) concurrently with the packet departure (in the downstream direction). At the upstream switch:
    - the serial-to-parallel conversion delay for the arriving credit is **4** clock cycles;
    - the synchronization delay is another **4** clock cycles;
    - the delay for the scheduler to be notified of the credit arrival can be as much as: *(i)* **5** clock cycles (hardware case), or *(ii)* **800** clock cycles (software case);
    - the delay for a packet to start departing through an idle link, after the time the scheduler has been notified that the only active flow of this link has now acquired the credit that it was waiting for, is: *(i)* **5** clock cycles (hardware case), or *(ii)* **192** clock cycles (software case).
    - At the downstream switch, again, the delay from the moment a packet has started arriving until the time it can start departing (provided it finds, just in time, a credit and a just-going-idle output link) is: *(i)* **12** clock cycles (hardware case), or *(ii)* **1000** clock cycles (software case).

  The clock frequency is **400 MHz** in all cases.

In each of the above cases, how many windows can fit in a buffer memory of capacity (A) 4 MByte (small SRAM buffer case), or (B) 1 GBytes (DRAM case)? Ignore linked-list pointer overhead, and only consider raw packet size bytes.

11.2 Shared Queue Unfairness

A given switch S has 8 inputs and 8 outputs, of throughput 100 Mbps each, and uses output queueing, with one (logical) queue per output shared among all flows going out through that link. Input 1 carries flow 1, which originates from a 64 Kbps source and goes out through output 1. Flow 1 carries a message of size 1 MByte, whose packets were transmitted consecutively in time at peak source rate (64 Kbps).

(a) If flow 1 is the only flow going out through output 1, what is the delay from the moment the first byte of flow 1 arrives at S, to the moment the last byte of flow 1 departs from S? Assume that the enqueueing and dequeueing delay of S is negligible.

(b) Now assume that output 1 carries 8 active flows: flow 1 as before, plus flows 2 through 8 which arrive on inputs 2 through 8, respectively. Flows 2 through 8 were idle until the moment the first byte of flow 1 arrived, but then, unfortunately, they all became suddenly active, carrying traffic at their peak arrival rate, which is 100 Mbps for each of them. Assume that the buffer of output 1 is large enough so that it does not overlow during the time that it takes for all packets of flow 1 to arrive. How many bytes of packets belonging to flows 2 through 8 will be queued in output queue 1 between the first and the last byte of flow 1's message? (Assume, for simplicity, that the bytes of individual flows arrive uniformly spread in time, i.e. ignore their clustering in packets of varying sizes). How long will it take for all these bytes to depart through output 1? Subsequently, what is now the delay for the message of flow 1? How does it compare to the delay in (a)?

(c) Now assume that output 1 uses *per-flow queueing* , i.e. it has (more than) 8 separate (logical) queues, one for each of flows 1 through 8, and that it performs round-robin scheduling among its non-empty queues. For simplicity, assume that all 8 flows carry packets of the same size, and that this common packet size is 1500 Bytes. How long does it take for one packet to depart through output 1? Call this a "packet time". In the worst case, a packet of flow 1 that arrives into an empty flow-1 queue will have to wait for 7 packet times before it receives service, because it happened to arrive just after the round-robin pointer passed over its queue and found it empty. What is this worst-case queueing delay for a flow-1 packet, assuming it was the head packet in its queue? Every how often does a new packet of flow 1 arrive through input 1? Given your last two answers what do you conclude: when each flow-1 packet arrives, does it find an empty or a non-empty flow-1 queue? Based on the above findings, what is now the delay for the message of flow 1? How does it compare to the delays in (a) and (b)?

Up to the Home Page of CS-534

Last updated: 21 May 2008, by M. Katevenis.