Packet scheduling research

Sally Floyd
DARTnet II Meeting
floyd@ee.lbl.gov

March 6, 1995

(Credits to Barbara Denny, Van Jacobson, Scott Shenker, Lixia Zhang)

DARTnet research program - packet scheduling algorithms

- CSZ (Clark/Shenker/Zhang) Scheduler [MIT, PARC]
- CBQ (Class-Based Queueing) [LBL]
- Virtual Clock (VC) [PARC, BBN]
- FairShare [BBN]
- Deadline Scheduling [UMass]
- SFQ (Stochastic Fair-Queueing) and SFQ/VC [SRI]
- IP Source Quench [ISI]
- MBONE enhancements [PARC]

Overview of talk:

- The general research questions.
- Stochastic Fair Queueing.
- CSZ.
- CBQ.
- The role of DARTnet.

If there is time:

- One perspective on some of the open issues.

[Note: I am not going to try to describe each of the packet scheduling algorithms that have been investigated on DARTnet.]

The general research questions for packet scheduling algorithms

- Real-time traffic (traffic with fixed or adaptive playback times).

- Best-effort traffic - beyond FIFO.
  - Fairness, protection from misbehaving users.
  - Different sensitivities to delay.

- Controlled link-sharing.
  - between organizations (e.g., agencies sharing the FAT pipe).
  - between protocol families (e.g., IP and SNA).
  - between traffic types (e.g., Mbone and TCP).

A Scheduling Algorithm for Best-Effort Traffic: Stochastic Fair Queueing [SRI]

- More efficient that strict fair queueing.
- A hash function to map from source/destination address pair to queue.
- Perturb hashing seed periodically for fairness.
- Packet-by-packet round-robin scheduling among queues.
- DARTnet experiments explore fairness, prevention of starvation, graceful degradation under overload, resource usage.
- Other researchers are extending Stochastic Fair Queueing with scheduling algorithms that are not packet-by-packet.

Integrated Services: the CSZ Service Model

Service model as the interface between applications and the network.

- Real-time applications:
  - ”Guaranteed service” for applications with fixed playback times.
  - ”Predictive service” for applications with adaptive playback times and loss tolerance.
- Elastic (best-effort) applications: ASAP (as soon as possible) service. Different priority levels.

Controlled link-sharing between firms, protocols, etc., where real-time traffic has precedence over the link-sharing requirements.

CSZ: Scheduling Algorithms for Guaranteed Service

- Fundamental principle: isolate flows from each other.
- Bound the burstiness of each flow.
- Allocate enough bandwidth to each.
- Many capable algorithms exist. The scheduling algorithm explored by CSZ is Weighted Fair Queueing [DKS 89], with end-to-end delay bounds proven by Parekh and Gallager.

CSZ: Scheduling Algorithms for Predictive Service

- Fundamental principle: Promote statistical sharing among flows of the same QoS.
- Use FIFO/FIFO+ to reduce the tail of the delay distribution.
- Applications can adapt to a changing delay.
- Admissions control procedure based on measurements of existing traffic.

CSZ: Why use FIFO to reduce the maximum delay of predictive traffic?

- With Fair Queueing or Round-Robin-based scheduling algorithms, flows are isolated from misbehaving users. At the same time, packets that arrive for a particular flow in a burst are spread out, and transmitted one at a time according to the FQ/RR scheduling algorithm. This does not reduce the worst-case delay.

- With FIFO, packets for a particular flow that arrive in a burst are transmitted in a similar burst. The admissions control procedure, with policing at the edge of the network, is assumed to protect flows from misbehaving users.

Integrated Services: Class-Based Queueing (CBQ)

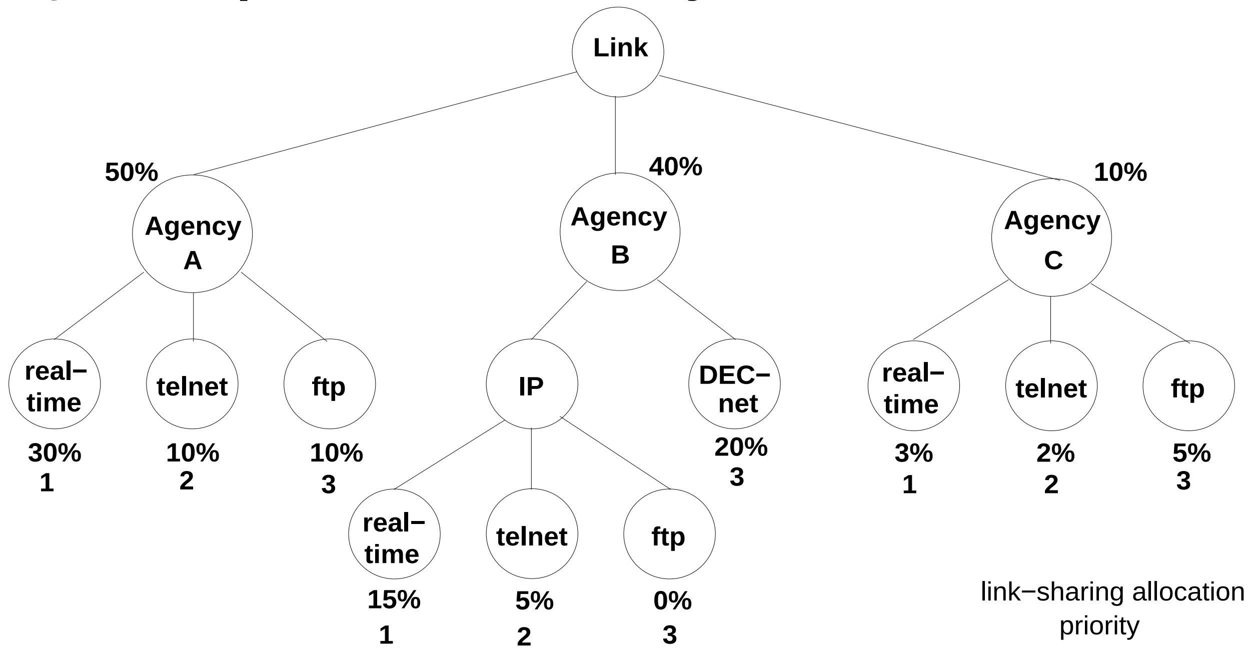
- Separate low-level mechanisms from high-level policy, to allow evolution.

- Link resources are associated with classes.
  Each class has a priority and a throughput allocation.
  CBQ makes no assumptions about how traffic is assigned to classes (it could be by connection, protocol, agency, etc.).

- Construct a hierarchy of classes to simultaneously capture both QoS and link-sharing constraints.

- Avoid extensive per-conversation parameterization.

CBQ, Example class hierarchy:



- Link-sharing between organizations, protocol families, and/or traffic types. We believe that needs met by link-sharing are fundamental (in the absence of free, infinite bandwidth).

- Hierarchy allows:
  - simultaneous QoS and sharing constraints;
  - flexible multiplexing via controlled ”borrowing”;
  - delegation.

- Different links in the network (e.g., an ISDN line to the home, a link in a backbone network) will have different class structures.

CBQ, Mechanisms:

- Classifier: Map arriving packets to classes, using information in the packet header.
- Estimator: Compute a short-term estimate of the class’s bandwidth.
- Selector: Find the class that is allowed to send the next packet. (In CBQ, look for the highest priority class, then use round-robin within classes of the same priority.)
- Delayer: For a class that is over its link-sharing allocation and contributing to congestion, compute the next time this class is allowed to send a packet. A delayed class is rate-limited to its allocated link-sharing bandwidth.

CBQ: Priority scheduling in a link-sharing framework.

- Simulations with delay-sensitive and throughput-sensitive traffic, to investigate the advantage (or lack of advantage) of incorporating priority-based scheduling in the link-sharing framework.

- Results: Priority sometimes does good and never does harm. Priority reduces delay for a high-priority class that is both bursty and low rate relative to the link bandwidth (e.g., a video stream, interactive traffic), without reducing the throughput of lower priority classes.

CBQ: Related work.

- CBQ implemented and tested on DARTnet [Jacobson].
- Simulations exploring behavior and modifications [Floyd].
- Work on the classifier, an implementation for the FAT pipe, and a public distribution of the CBQ code. [UCL, “Implementing Real Time Packet Forwarding Policies using Streams”, Wakeman et al., Usenix, January 1995.]
- Work on the LBL/UCL/Sun CBQ kernel. [Hoffman, Implementation report on the LBL/UCL/Sun CBQ kernel, Toronto IETF, July 1994.]

The Role of DARTnet:

- A wide range of packet scheduling algorithms has been investigated on the DARTnet testbed, using stock CPUs and T1 lines.
- The research has led to significant contributions to the wider Internet community.
- In addition to the experiments themselves on the DARTnet testbed, the DARTnet community has been an important forum for discussions of issues of common interest (including disagreements on these issues).

Open Questions:

- What mechanisms are needed in the Internet (if any) so that we can experiment with emerging real-time applications (e.g., rate-adaptive video) whose service requirements are not yet fully defined?

- ?

CBQ/CSZ, are there differences? (A personal answer.)

- Broad agreement about the importance of predictive service, the requirement for link-sharing, priority-based scheduling between different classes of best-effort (e.g., elastic) traffic, the need for a classifier, etc. Similar abilities to provide guaranteed service.

- Differences of emphasis between the primacy of the service model (CSZ), and the separation of mechanism and policy to allow flexibility and evolution (CBQ).

- For a class of predictive service traffic, if the admission control procedure's prediction of future traffic is incorrect, and the predictive service class becomes oversubscribed, the choice at the gateway is to limit the bandwidth of the predictive service class (CBQ), or to allow starvation of lower-priority classes (CSZ).

CBQ/CSZ, are there differences? (Continued.)

The classifier maps arriving packets to the appropriate service class.

- The role of the classifier is the same in CSZ and in CBQ.

- CSZ advocates that the mapping from packets to service class should come from specific service requests from the application, based on the service model, and not from network decisions based on port numbers and such. CBQ takes no position about policy for binding traffic to classes.