Two-Pass or Not Two-Pass.....

Scott Shenker

Xerox PARC

Basic Problem

Local Behavior vs. Global Requirements

- QoS is an End-to-End Concept
- Reservation is a Per-Link Concept

How do we ensure that the set of per-link reservations meets the end-to-end service needs of a receiver?

What Do Receivers Want to Know?

Getting the Service:

- End-to-End service that results from a given service request
  - Allows estimation of interactive performance and buffer requirements
- Spectrum of end-to-end services available
  - Rough price/performance tradeoff
  - Most applications do not have a single delay bound target

Using the Service:

- Resulting end-to-end service
  - Nonadaptive playback applications need bound to set playback point
  - Adaptive playback applications might use bounds to guide adaptive algorithm

Traditional Approaches

One-Pass (current RSVP)

- Receiver specifies per-link service
  - Predictive or Guaranteed
  - Traffic Characteristics (r,b)
- Set-up protocol delivers this request to each switch along path
- Resulting end-to-end delays never specified

Two-Pass (e.g., Tenet)

- Sender specifies desired end-to-end delay and jitter
- Network computes the needed per-link services
- This requires the “second-pass”
- Receiver does not know the spectrum of services available

New Approach

One-Pass with Advertising (OPWA)

- Network advertises mapping between per-link service requests and resulting end-to-end service
- Receiver makes per-link service request already knowing what the end-to-end result will be
- No need for "second-pass"

Notes

We are only concerned with delay:

- Bandwidth-related services (AASAP) have trivial mappings between per-link and end-to-end services
- Delay related services have nontrivial mappings

Mechanism is rather general:

- Applies to any QoS that can be computed iteratively along the path
- Specific assumptions make the presentation easier

Assumptions

Traffic described by token rate and bucket size

Two kinds of service:

- Predictive
- Guaranteed

Guaranteed:

- Per-Link service:
  - Rate r (for a particular flow)
  - Link error parameters A_i and B_i
    * $\frac{A_i}{r} + B_i$ bounds deviation from fluid model
  - Latency L_i
- End-to-End Delay
  - $\frac{b}{r} + \sum_i \left(\frac{A_i}{r} + B_i + L_i\right)$
  - Modeled on Parekh's result

Predictive:

- Per-Link service:
  - Classes α
    - hiding a level of indirection
    - these are *logical* classes
  - Delay bound PD_i^α (fairly reliable)
  - Latency L_i
- End-to-End Delay
  - $\sum_i \left(PD_i^\alpha + L_i\right)$
  - (controversial)

Advertising

Add the following fields to PATH messages:

- $\sum_i A_i$
- $\sum_i B_i$
- $\sum_i L_i$
- $\sum_i PD_i^\alpha$ (for each α)

Sums are computed iteratively at each hop

Advertisements enable receiver to compute the end-to-end delay and jitter from each source

Service Request:

- Traffic: token rate and bucket size
- G: rate r
- P: class α

Back to the Basic Problem

Local Behavior vs. Global Requirements

- QoS is an End-to-End Concept
- Reservation is a Per-Link Concept

Who does the mapping between global requirements and local service?

Contrasting the Mechanisms

- One-Pass
  - receiver specifies per-link service
  - end-to-end service never defined
  - global to local mapping never made
- Two-Pass
  - receiver specifies end-to-end service
  - network figures out required per-link services
  - network does global to local mapping
- OPWA
  - receiver is told mapping between per-link service and end-to-end service
  - receiver specifies per-link service
  - receiver does global to local mapping

Paradigm Shift

Single "Flowspec":

- e.g., RFC 1363

Multiple Specifications:

QSpec: end-to-end quality of service

RSpec: service request

HSpec: per-hop service

TSpec: traffic specification
(policed by network)

Usage in OPWA:

- RSpec carried by RSVP
- Converted to HSpec and TSpec reservation requests at each hop
- QSpec not used in any interface
- Receiver can compute QSpec based on advertisements

Advantages

- vs. OP: can meet delay/jitter targets
- vs. TP: simpler mechanism (esp. mcast)
- vs. both: provides price/performance menu

Disadvantages

- Requires PATH messages
- Extra overhead in PATH messages
- Hard to reroute transparently
- ??

Unresolved Issues

Nonuniform predictive levels

Multisource Flows:

- Advertisements give delay from single source to receiver
- Do we need anything more?

Hybrid Reservations:

- Mixture of P and G along path
- INTSERV: Should we allow it?
- RSVP: What to advertise?

Additional Advertisements:

- Current Reservation State?
- Other?
  - Average Predictive Delays
  - Average ASAP Delays

Any Questions?

Finer Granularity for Predictive Service

Currently can only get same level of Predictive service at each link

- Receivers only care about resulting end-to-end delay/jitter
- Big difference in delay between classes (5x?)
- Want to offer intermediate delays

Possible solution: Extra Delay (ED) field

- Specify desired Predictive class and the ED you can tolerate
- Switch can reserve lower class and subtract the delay difference from the ED field

Practical problems:

- Per-hop monotonicity in the ED to HSpec mapping must be preserved
  - Mapping is independent of other reservations
  - ED is zeroed by first hop that cannot use it

- Need(?) to advertise the end-to-end service that results from each value of ED
  - Can be done, but length of PATH messages increases linearly with length of path
  - Note that the difference between the requested bound and the real bound is between 0 and $MAX[PD_i^{\alpha-1} - PD_i^{\alpha}]$

Interaction with QoS Routing

Receiver specifies "quality of route" to routing service

- Coarse description of quality of service
  - Target bandwidth
  - Routing goal (e.g., minimize latency)
- Policy constraints, etc.

Once route is established, the receiver...

- sees advertising messages
- evaluates the available end-to-end QoS
- requests the appropriate per-link service, or requests an alternate path
- can request to see advertising along alternate path before activating that path