QoS Routing for Traffic with Performance Guarantees

Qingming Ma   Peter Steenkiste

Carnegie Mellon University

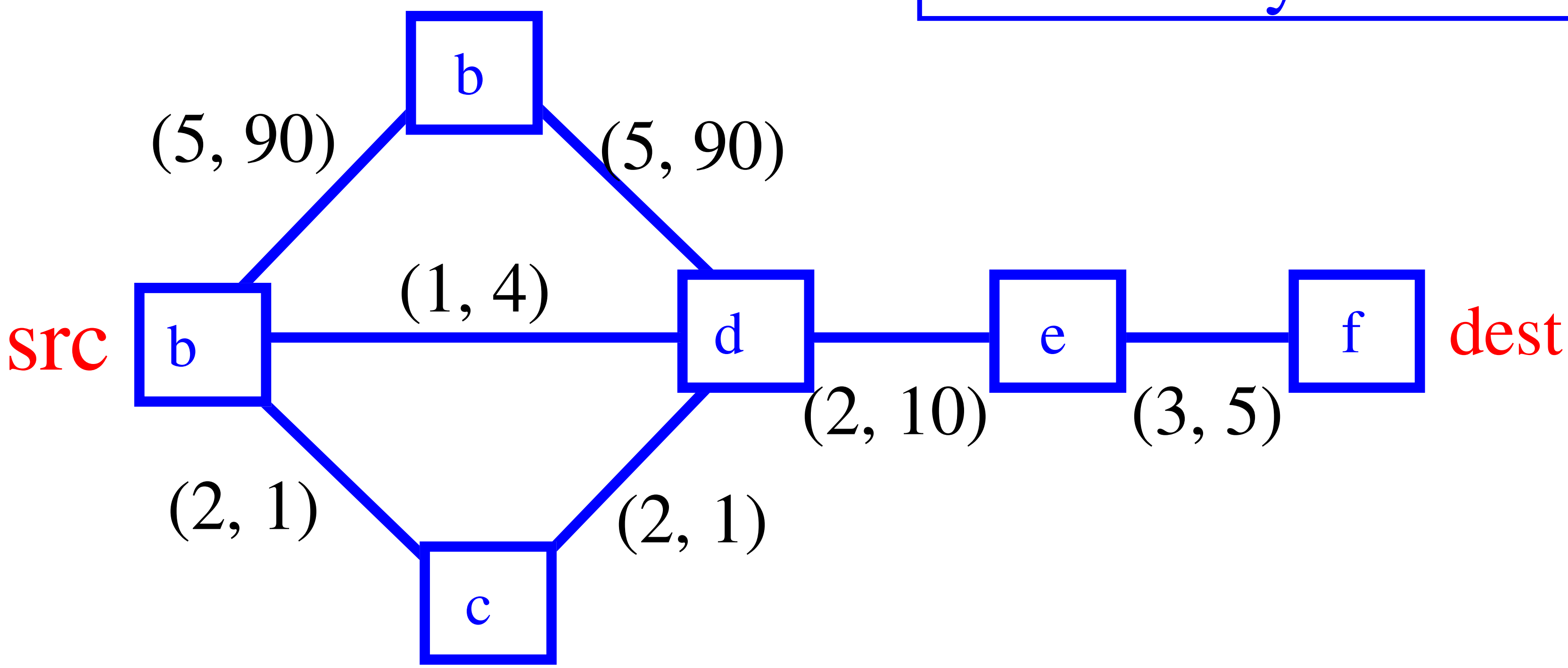
Motivations for QoS routing

- Real-time applications
  - stringent bandwidth and delay bounds
- Scheduling, admission control, reservation
  - assume a path with sufficient resources
- *Find a path to support QoS guarantees*
  - select one that meets QoS constraints
- *Achieve resource efficiency*
  - select a ***better*** path

An example

(bandwidth, delay)

bandwidth: 1.5 mb/s
delay: 200 ms



Two known results

- Bandwidth and delay constraints
  - bandwidth---minimum, delay---additive
  - can use Dijkstra or Bellman-Ford algorithms
- Solving with 2-additive-constraint is NP-complete [Wang & Crowcroft 96, Garey & Johnson 79]
  - e.g., delay and delay-jitter constraints

Both assume constraints are independent

Our approach

- QoS constraints are not independent
  - delay, delay-jitter, and buffer size depend on BW
  - Scheduling algorithms determine the relationship
- Make path selection aware of this relationship
- *Better, and simpler routing algorithms*
  - reduces complexity
  - selects more efficient path

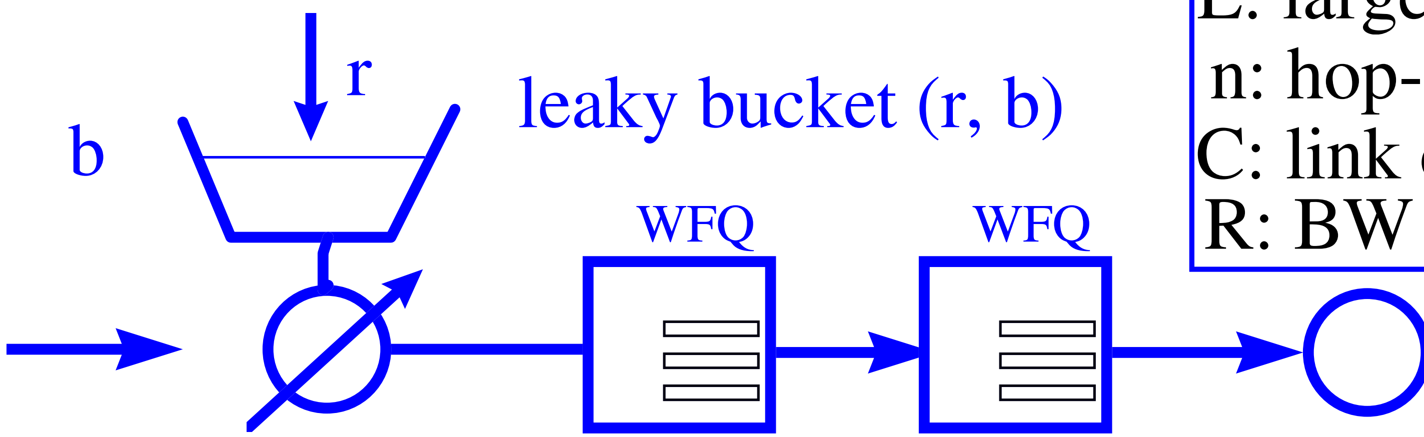
Relating QoS constraints

- Delay = queueing delay + propagation delay
- Queueing delay is determined by
  - bandwidth R to reserve
  - scheduling algorithms
- Worst delay-jitter $\leq$ maximum queueing delay
- Buffer size is determined by R and hop count

WFQ-like scheduling algorithms

- Weighted Fair Queueing
  - a session can be isolated from others
  - mathematical provable delay bounds
  - widely used (e.g., IETF, FORE, NETSTAR)
- A class of similar scheduling algorithms
  - virtual clock (Zhang 91)
  - worst-case fair queueing (Bennett & Zhang 96)
  - self-clocked fair queueing (Golestani 91)

Delay & buffer bounds using WFQ



L: largest pkt size
n: hop-count
C: link capacity
R: BW to reserve

- Delay $= \dfrac{b + n*L}{R} + \sum_{i=1}^{n} \dfrac{L}{C_i} + \sum_{i=1}^{n} prop_i$
- Delay-jitter = (b + n*L) / R
- Buffer space B for a node = b + n*L

QoS routing algorithms

	Known R	Unknown R	Heterogeneous R
delay	shortest path	iterative shortest path	iterative shortest path
delay, jitter	Bellman-Ford (limit hops)	iterative BF (limit hops)	NP complete
delay, buffer	Bellman-Ford (limit hops)	iterative BF (limit hops)	iterative-BF (limit hops)
delay, jitter, buffer	Bellman-Ford (limit hops)	iterative BF (limit hops)	NP complete

Case 1: delay+jitter, with R known

- Given delay bound D and jitter bound J.
- Define a link-metric function:

  $dist(i) = L/R + L/C_i + prop_i$
- Find a path with distance $\leq$ (D - b/R)
- To ensure jitter $\leq$ J, we need $(b + n*L)/R \leq J$

  let $N = (R*J - b) / L$

  jitter $\leq$ J iff hop-count $\leq$ N
- Thus, use Bellman-Ford SP with hop count $\leq$ N

Case 2: delay, with R unknown

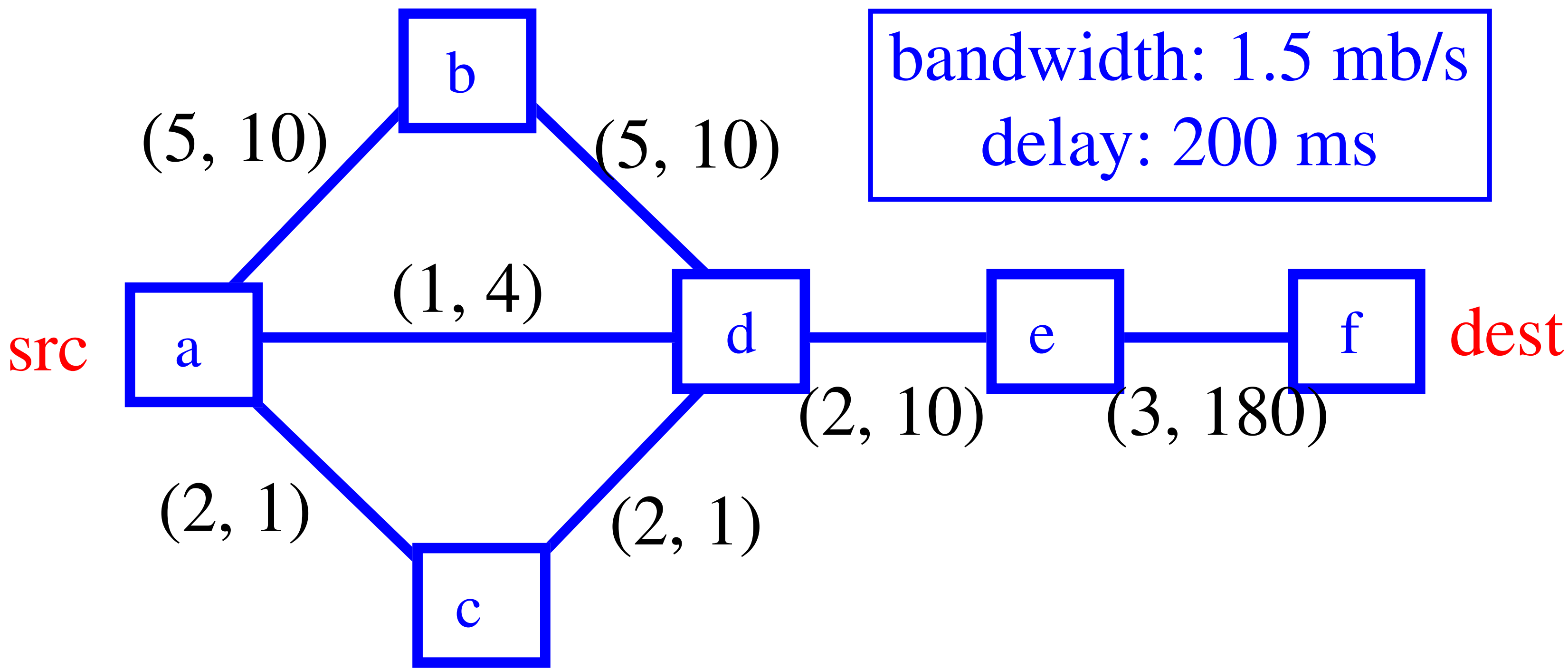
- Iterate SP over all values v of link residual BW
- At each iteration, consider only links with $R \geq v$

```
for (all values v of link residual BW) {
        dist(i, v) = L/v + L/C_i  +  prop_i;
        shortest_path(src, dest, v, dist);
        …
}
```

- Determine the minimum R to reserve

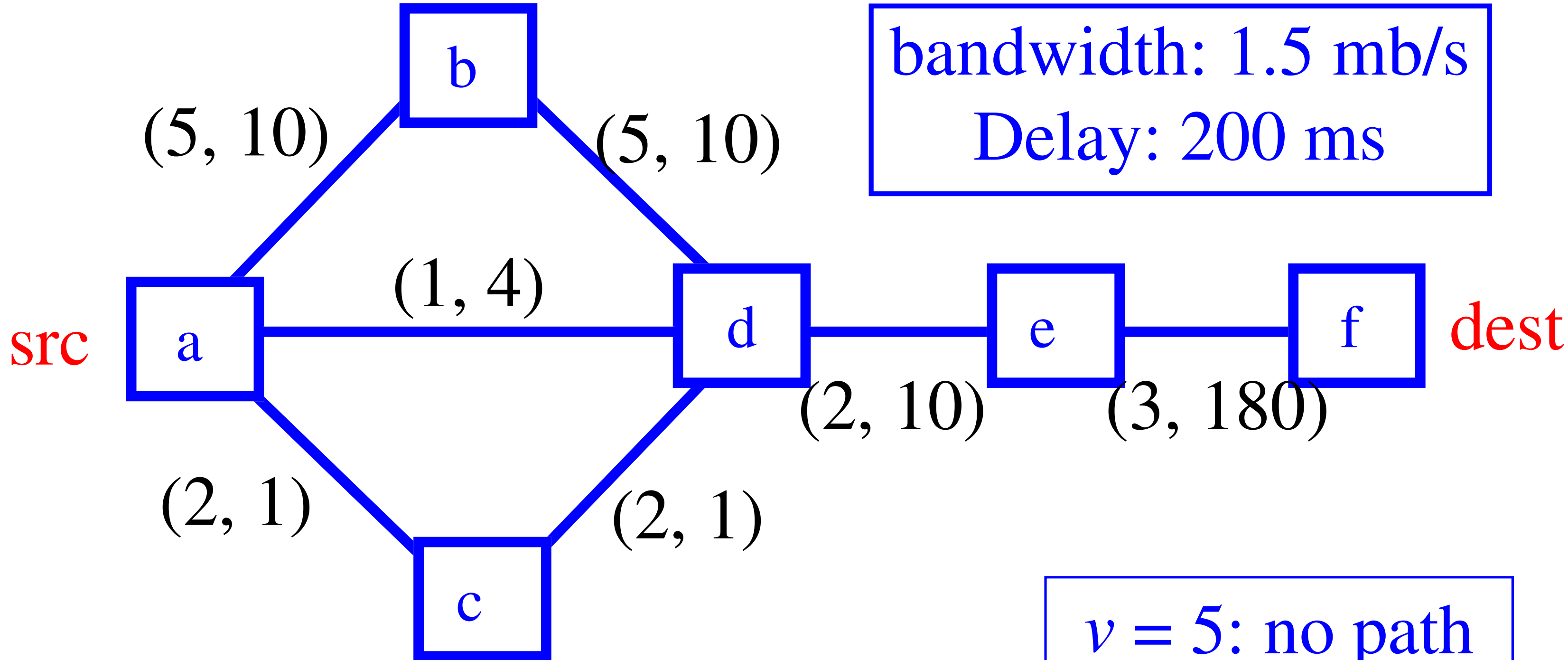
Case 2: delay, with R unknown

- Simple shortest path does not work



Both Dijkstra and Bellman-Ford will pick a->b->d->e

An example of iterative SP



bandwidth: 1.5 mb/s
Delay: 200 ms

v = 5: no path
v = 3: no path
v = 2: a-c-d-e-f

$\text{dist}(i, v) = L/v + L/C_i + \text{prop}_i$

Selecting among paths found

- Selection criteria
  - bandwidth consumption
  - path length (hop-count)
  - current path load
- Any “better” paths not found by iterative BF?
  - no path requiring less bandwidth
  - no path with less hops
  - no path with lighter load

Three path-selection alternatives

- Shortest-delay path
  - achieve minimum delay if maximum BW reserved
  - can be inefficient: pick a long path with a lot of BW
- Optimal-bandwidth path
  - reserve minimum R
  - can be inefficient: pick a loaded path
- Widest-shortest path
  - minimum hop ==> less bandwidth
  - maximum reservable BW ==> lighter load

An added benefit of WSP

Can be implemented efficiently

- Exchange the order of iterations
- Order bandwidth values decreasingly

```
for (hop =1; hop < N; hop++) {
    for (all values of link residual bandwidth) {
        ...
    }
}
```

- The first path found is WSP!

Heterogeneous R across links

- R is usually homogeneous, but what if not?
- $$D = \frac{b}{\min\{R_i\}} + \sum_{i=1}^{n} \frac{L}{R_i} + \sum_{i=1}^{n} \frac{L}{C_i} + \sum_{i=1}^{n} prop_i$$
- If delay only
  - can use the iterative Bellman-Ford algorithm
- If delay and delay-jitter
  - problem is NP-complete
  - reduce two-additive NP-complete problem to this one

Related work

- [Wang & Crowcroft 96]: NP-complete results
- [Przygienda 95]: a P-subset of NP
  - one additive & multiple minimum constraints
- [Guerin & Orda 97]: inaccurate information
  - similar algorithm to case 2
  - but for delay only
- [Pornavilai 97]
  - considered only the case when R is known

Conclusions

- Routing with delay, jitter, and buffer constraints is solvable in polynomial time, if
  - consider relationship among constraints
  - network uses WFQ-like scheduling algorithms
- Solution is the iterating Bellman-Ford SP
  - iterative over all values of link residual BW
  - limit path hop count of a path to meet jitter bound
  - limit node hop count to meet buffer space constraint