Evaluating the Overheads of Source-Directed Quality-of-Service Routing

Anees Shaikh Jennifer Rexford Kang G. Shin

University of Michigan
Ann Arbor, MI

AT&T Labs – Research
Florham Park, NJ

October 14, 1998

http://www.eecs.umich.edu/~ashaikh/research

Quality-of-Service Routing

- Support diverse traffic characteristics on a single backbone
- QoS routing considers network load and flow traffic spec
  - Optimize network resource usage (balance load)
  - Satisfy QoS requirements of flows
- Protocols consume bandwidth, CPU, and memory
  - Exchange information about link load
  - Compute suitable route for each connection

$\Rightarrow$ Basic trade-off between routing performance and overheads

Outline

- Source-directed link-state routing
- Simulation model
- Link-state update policies
- Link cost functions
- Summary of other results (traffic pattern and topology)
- Conclusions

Source-Directed Link-State Routing

Link-state QoS routing (e.g., QOSPF, PNNI)

- Source router computes path for each flow
- Route based on link-state database and flow requirements
- Link state represents available resources
- Link state updated periodically or triggered by large change

Tunable parameters affecting network overheads

- Link-state update frequency (bandwidth, CPU)
- Network size and connectivity (bandwidth, CPU)
- Route computation algorithm and parameters (CPU, memory) - *NOSSDAV '98*
- Route compuation frequency (CPU)

Route Computation

Compute a route to destination with bandwidth b

- (Optionally) prune infeasible links ($u_l + b > 1$)
- Compute minimum-hop path(s) to destination
- Extract a route with the minimum total cost ($\sum_l c_l$)

Link cost function

- Link costs exponential in utilization ($c_l \propto u_l^\alpha$)
- May be discretized to small number of levels

$\Rightarrow$ cheapest-shortest (feasible) path

Resource Reservation Model

Hop-by-hop signalling

1. Test admissibility ($u_l + b \leq 1$)
2. Reserve bandwidth ($u_l = u_l + b$)

Routing v. set-up failures

Routing failure: source cannot find a feasible path

- local at source

Set-up failure: resource reservation fails during signalling

- consumes network resources

Link-State Update Policies

Routers exchange information about available link bandwidth

Three parameter model

- Trigger on significant change in *available* bandwidth
  - more responsive as link nears capacity
  - difficult to predict (rapid fluctuations)
- Periodic refresh of link state
  - simplified provisioning of resources
- Hold-down timer to enforce minimum spacing

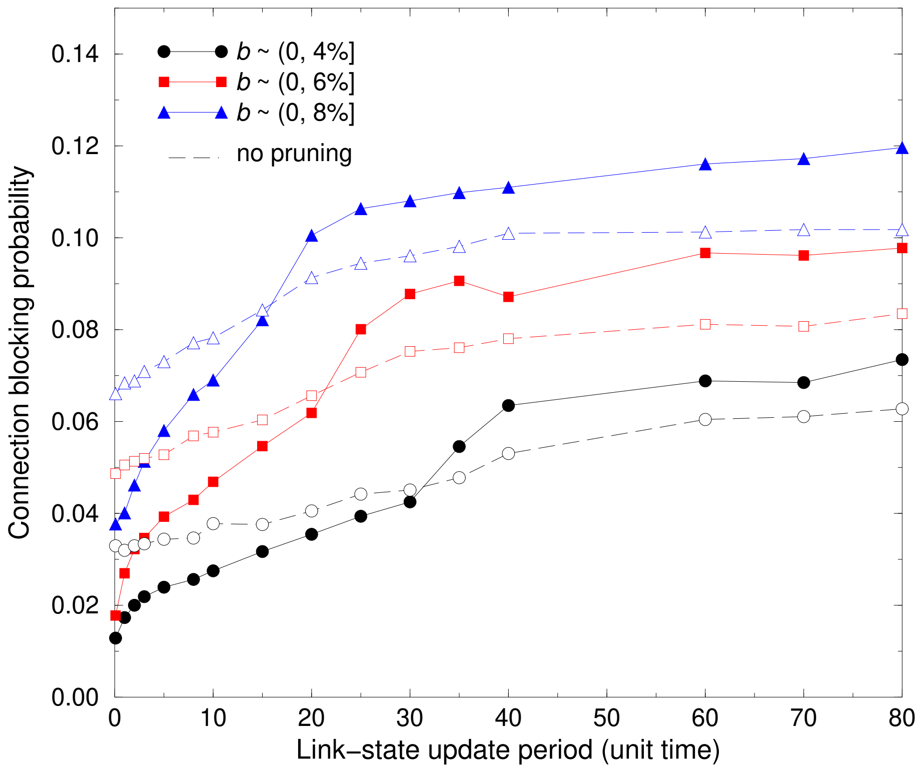
Link-state distribution method also influences overhead

Simulation Parameters

Class	Parameters
Path selection	Pruning/not-pruning, cost levels C, and exponent α
Link-state updates	Period, trigger, and hold-down timer
Network topology	Random graph, regular topology, or MCI backbone
Flow characteristics	Uniform random bandwidth b, Pareto or exponential holding time ℓ
Traffic pattern	Poisson or Weibull arrival rate λ, uniform random destinations
Performance metrics	Blocking, routing vs. set-up failures, and link-state update rate

- Evaluate with a connection-level event-driven simulator

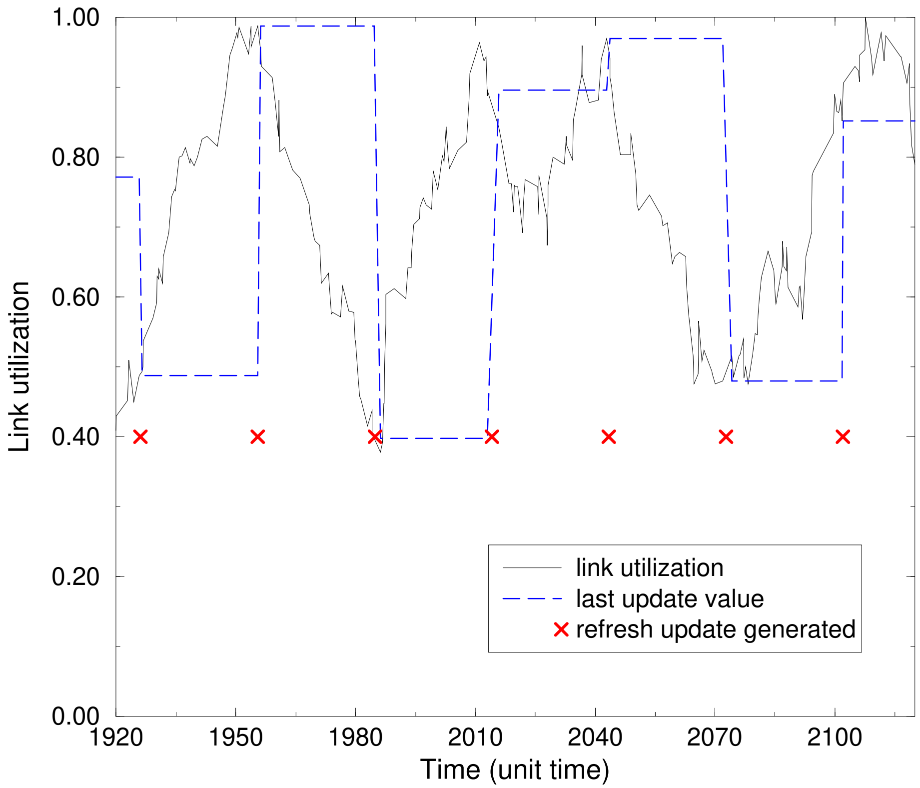
Staleness Due to Periodic Updates



$$\rho = 0.75,\ \lambda = 1,\ \alpha = 1$$

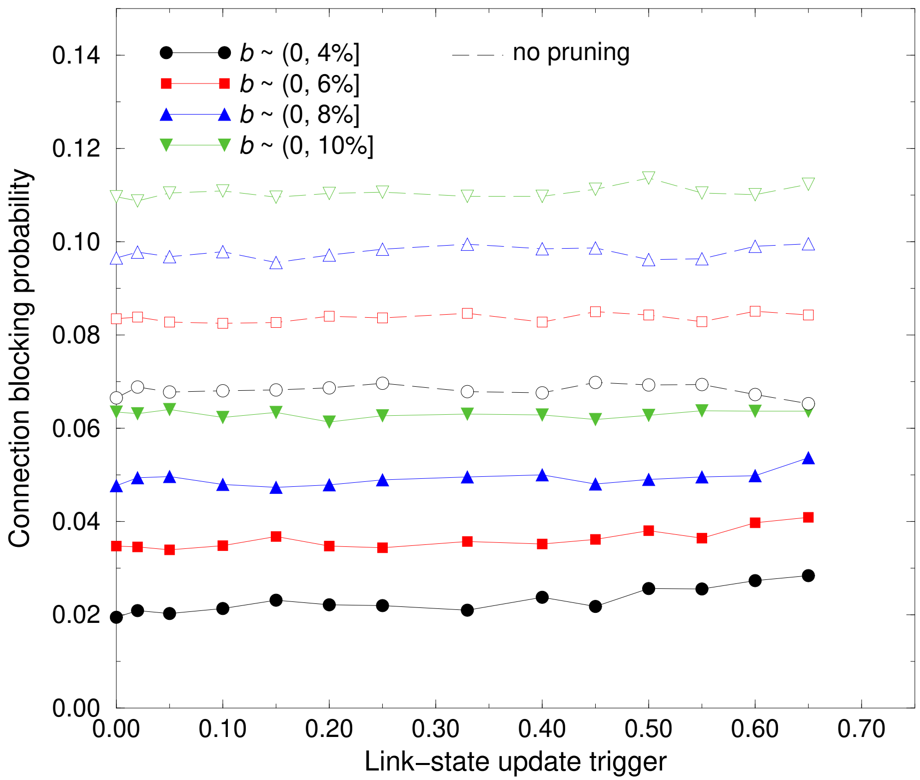
- Difficult to route high-bandwidth connections
- Pruning hurts when information is stale (and load is high)
- Low blocking requires very low update period

Link-State Fluctuations and Route Flapping



$\rho = 0.75$, $b \sim (0.0, 0.06]$, $\lambda = 1$, $\alpha = 1$, period = 30

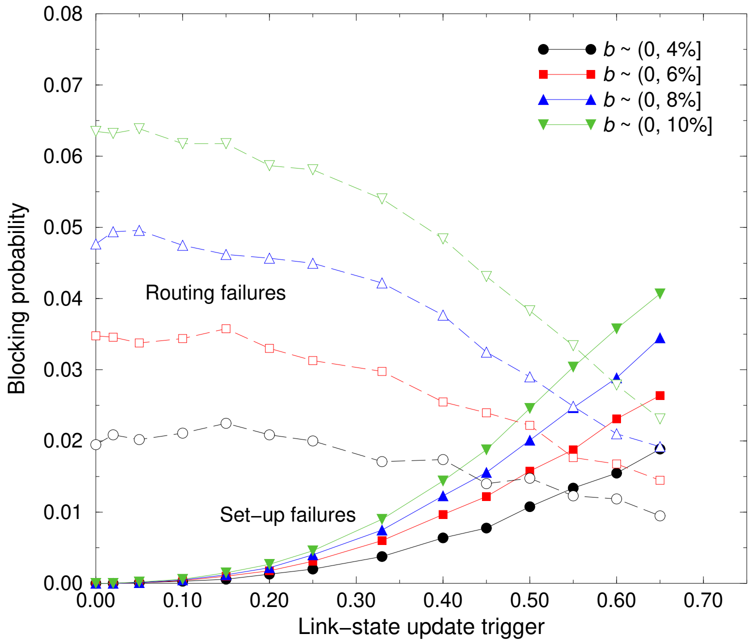
Blocking Insensitivity to Triggers

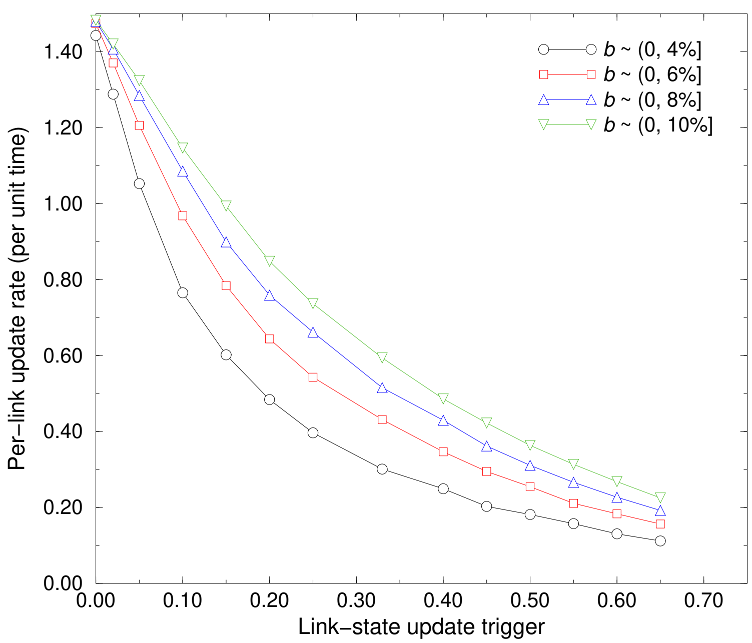


$$\rho = 0.70, \lambda = 1, \alpha = 1$$

- Blocking unaffected by trigger in wide variety of experiments
- High-bandwidth, pruning, and low connectivity inflate blocking
- Hold-down timers decrease updates but increase blocking

Set-up Failures and Update Overheads

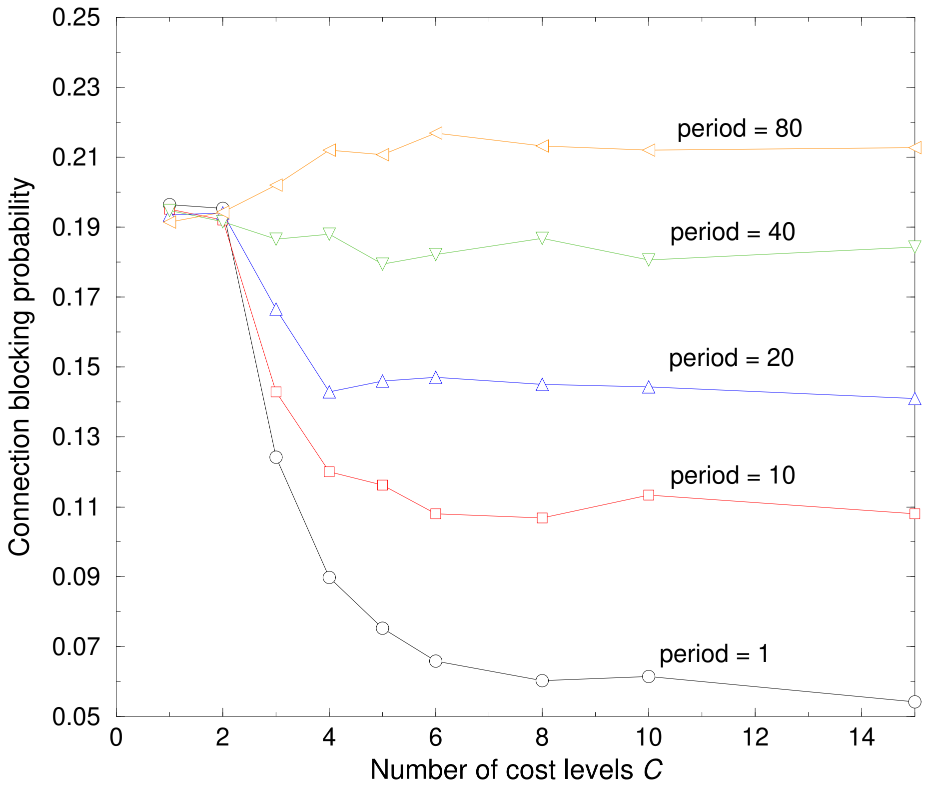




$$\rho = 0.70, \lambda = 1, \alpha = 1$$

- Coarse triggers increase expensive set-up failures
- Large triggers also decrease update overhead

Link Cost Parameters and Staleness



$$\rho = 0.85,\, b \sim (0.0, 0.1],\, \lambda = 1,\, \alpha = 1$$

- Fine-grained costs not much help with stale information
- Coarse-grained costs increase “ties” and reduce complexity
- Discrete costs require careful setting of α

Key Findings

Update policies

- Periodic updates cause flapping and set-up failures
- Triggered updates present a blocking vs. update rate trade-off

$\Rightarrow$ Use coarse trigger with hold-down timer

Link cost function

- Coarse-grained link costs perform well and reduce overhead
- Exponent α has little effect with fine-grained costs
- α should be tuned to cost discretization

Other Observations – Topology

- Network topology impacts performance

  $\Rightarrow$ topology should influence link-state update policy

- Long path lengths worsen blocking and overhead

  $\Rightarrow$ limit topology worst-case path length

- Pruning infeasible links is harmful when path choices are poor

  $\Rightarrow$ disable or control pruning in large networks

Other Observations – Traffic

- Bursty arrivals exacerbate blocking
  - many bad decisions when information is stale
  - much worse for high-bandwidth flows
- High-bandwidth flows are problematic
  - fragmentation causes increased connection blocking
  - but, effect not worsened as staleness grows
- Short connections require more frequent updates

  $\Rightarrow$ limit QoS routing to long-lived flows

Conclusions

Two routing extremes

- Static shortest-path routing for all flows
  - static link weights
  - poor performance when traffic shifts
  - relatively low overhead
- QoS or load-sensitive routing for all flows
  - high adaptability
  - high overhead
  - complex interactions

Combine static routing with load-senstive routing for a subset of flows