Speeding Up Short Data Transfers

Theory, Architectural Support, and Simulation Results

Yin Zhang, Lili Qiu — Cornell University

Srinivasan Keshav — Ensim Corporation

NOSSDAV'00, Chapel Hill, NC, June 2000

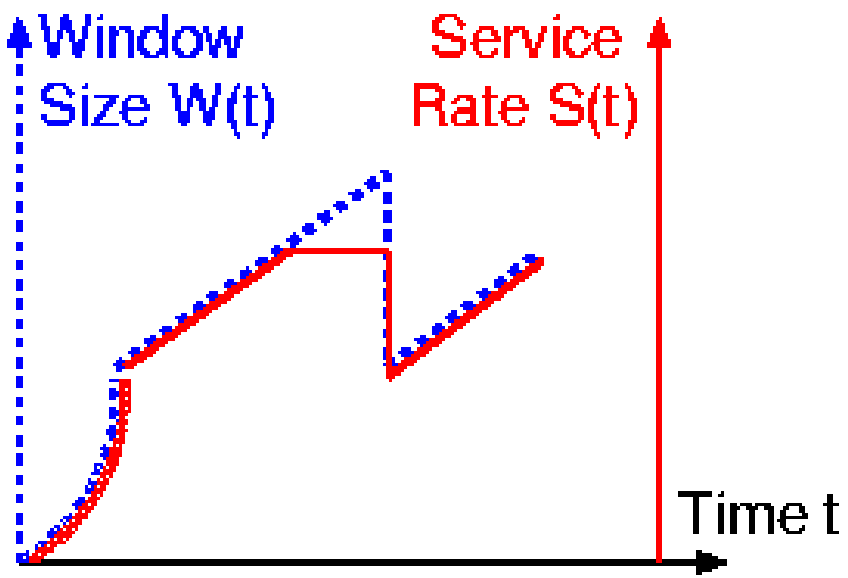
Outline

- Motivation
- Related Work
- Theory
- Architectural Support
- Simulation Results
- Conclusions and Future Work

Motivation

- Dominance of Web data transfers
  - Short & bursty [Mah97]
  - Small downloading time is important!
- Dominance of TCP
- Problem: Short data transfers interact poorly with TCP !

TCP/Reno Basics



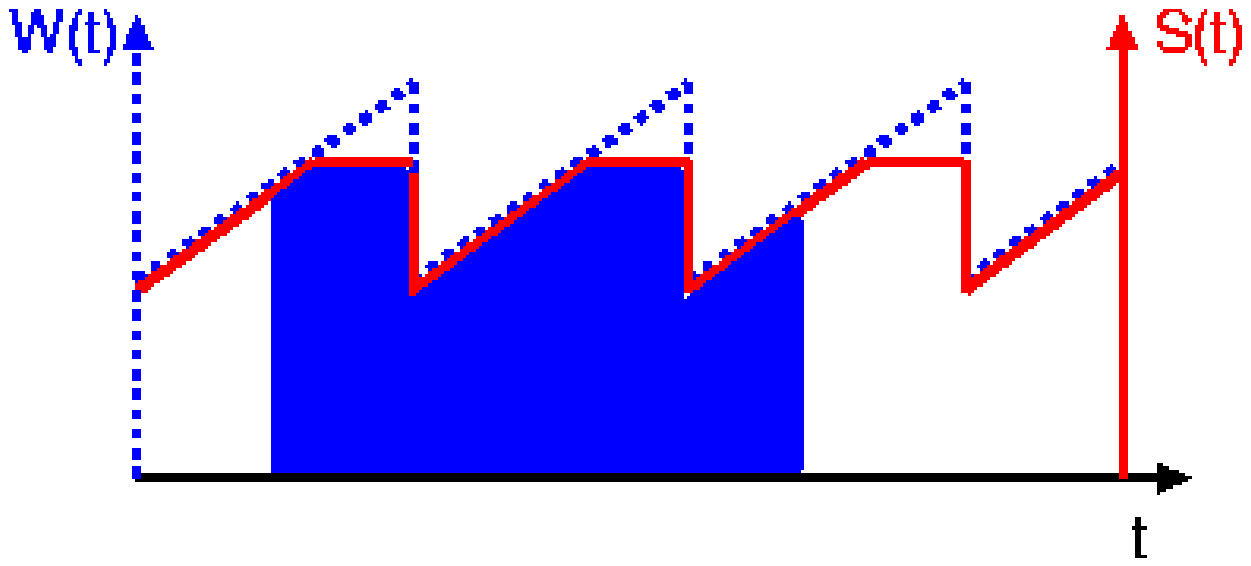
- Slow Start
  - Exponential growth in congestion window,
  - Slow: log(n) round trips for n segments
- Congestion Avoidance
  - Linear probing of BW
- Fast Retransmission
  - Triggered by 3 Duplicated ACK's

Related Work

- P-HTTP [PM94]
  - Avoid repeated probing only for components within the SAME page.
- T/TCP [Bra94]
  - Cache connection count, RTT
- TCP Control Block Interdependence [Tou97]:
  - Cache cwnd, but large bursts cause losses
- Rate Based Pacing [VH97]
- 4K Initial Window [AFP98]
- Fast Start [PK98, Pad98]
  - Most similar to our work, but need router support to ensure TCP friendliness

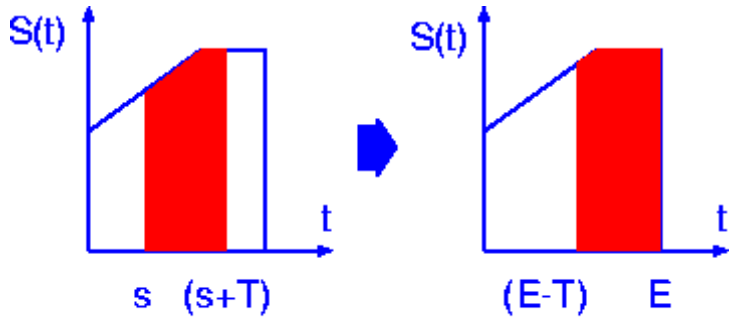
Our Approach

- Directly enter Congestion Avoidance
- Choose optimal initial congestion window
  - A Geometry Problem: Fitting a block to the service rate curve to minimize completion time

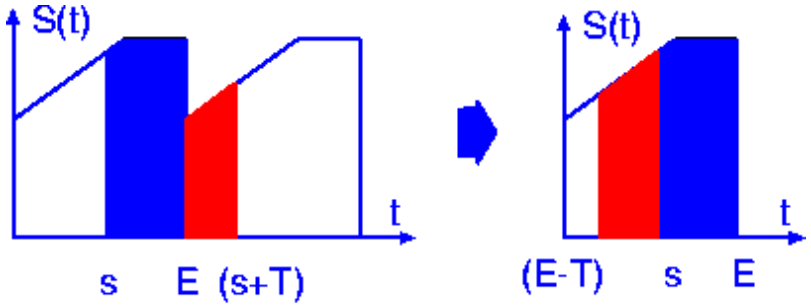


Optimal Initial cwnd

- Minimize completion time by having the transfer end at an Epoch boundary.



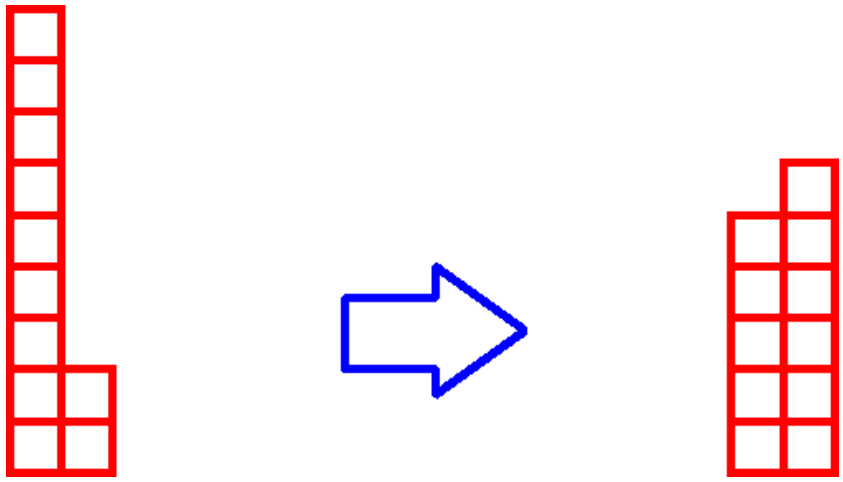
(a) s+T <= E



(b) s + T > E

Shift Optimization

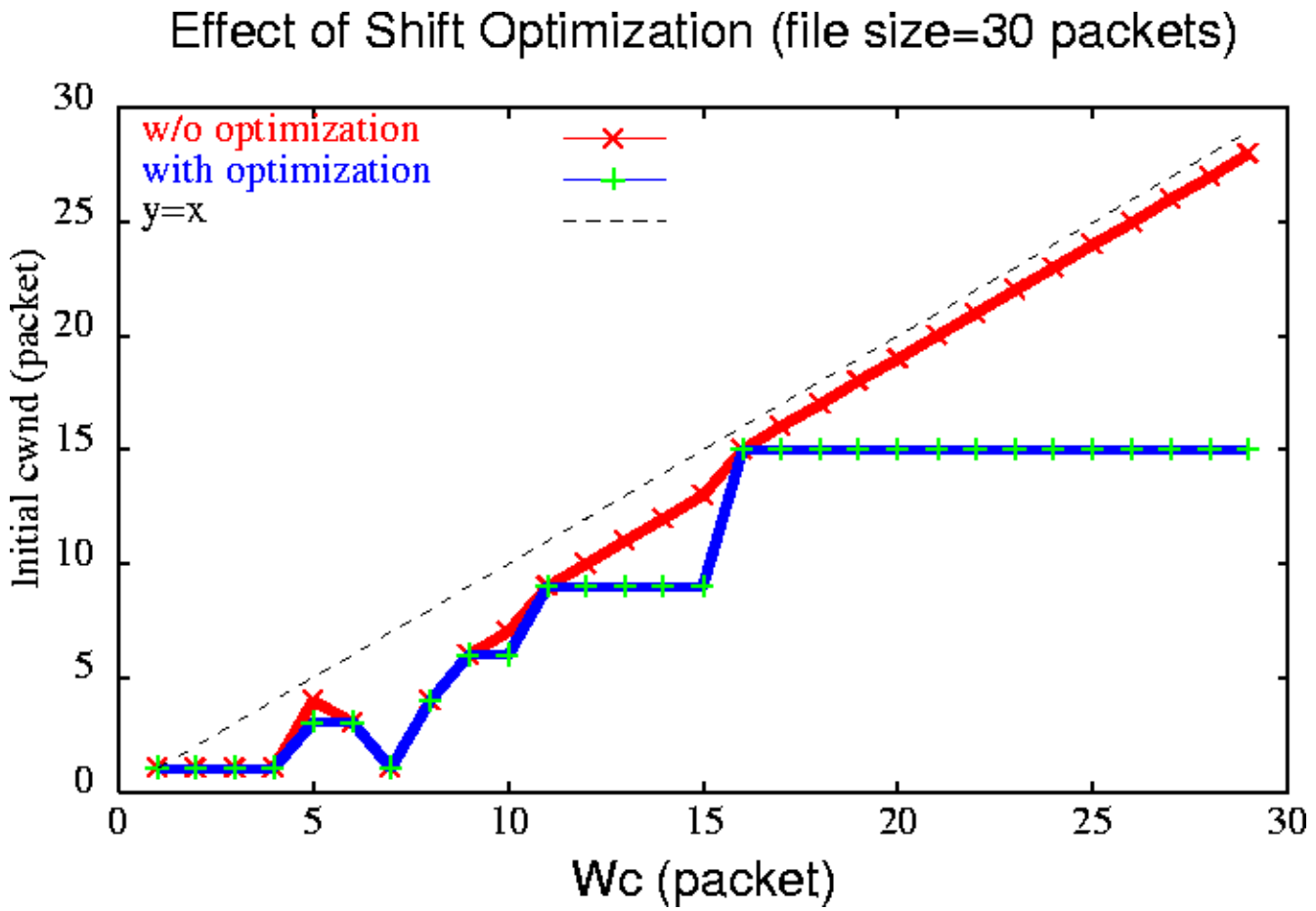
- Minimize initial cwnd while keeping the same integer number of RTT's



Before optimization:
cwnd = 9

After optimization:
cwnd = 5

Effect of Shift Optimization



TCP/SPAND

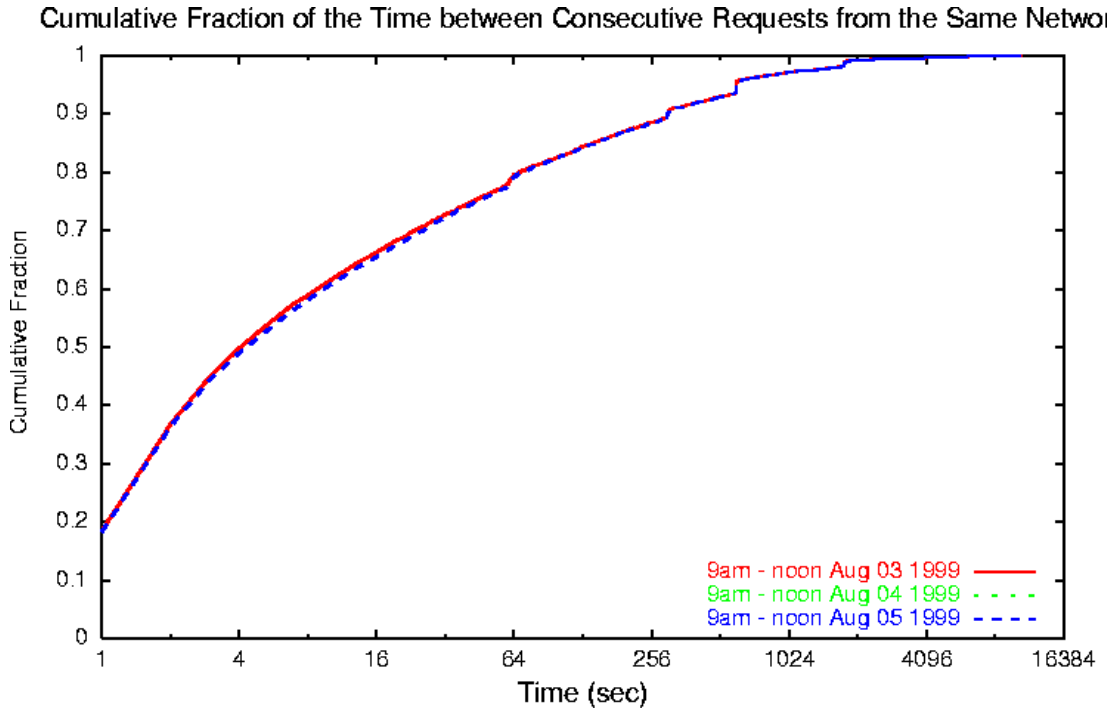
- Estimate network state by sharing performance information
  - SPAND: Shared PAssive Network Discovery [SSK97]
- Directly enter Congestion Avoidance, starting with the optimal initial cwnd
- Avoid large bursts by pacing

Implementation Issues

- Scope for sharing and aggregation
  - 24-bit heuristic
  - network-aware clustering [KW00]
- Collecting performance information
  - New TCP option, Windmill's approach, …
- Information aggregation
  - Sliding window average
- Retrieving estimation of network state
  - Explicit query, active push, …
- Pacing
  - Leaky bucket based pacing

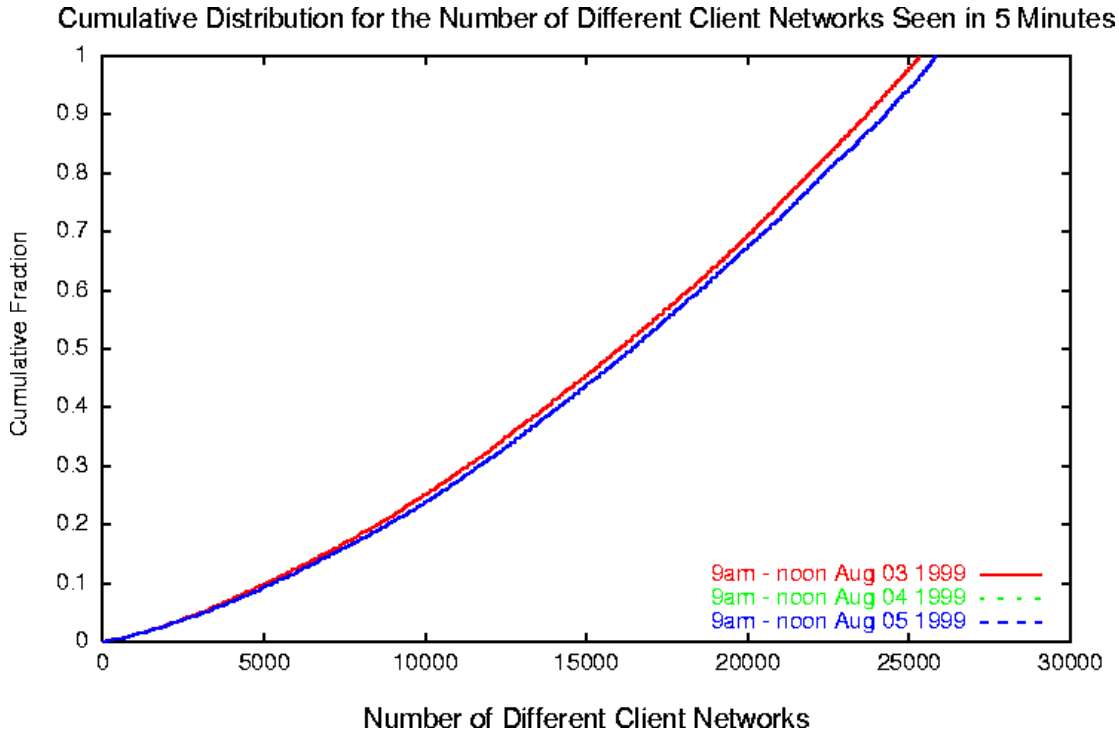
Opportunity for Sharing

- MSNBC: 90% requests arrive within 5 minutes since the most recent request from the same client network (using 24-bit heuristic)



Cost for Sharing

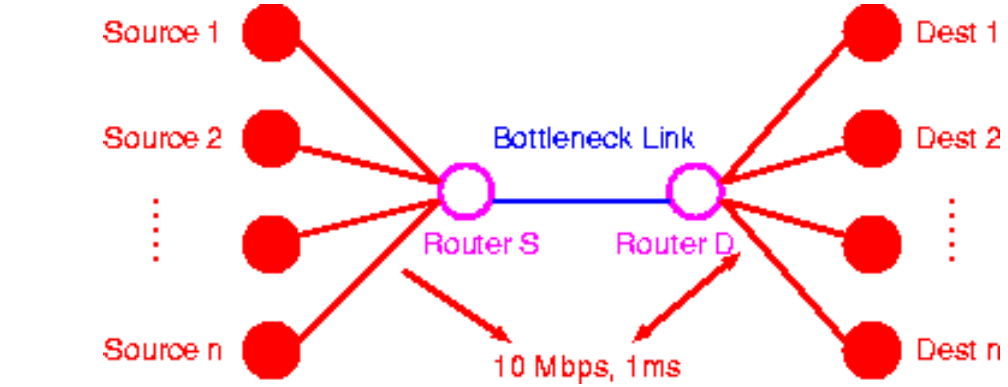
- MSNBC: 15,000-25,000 different client networks in a 5-minute interval during peak hours (using 24-bit heuristic)

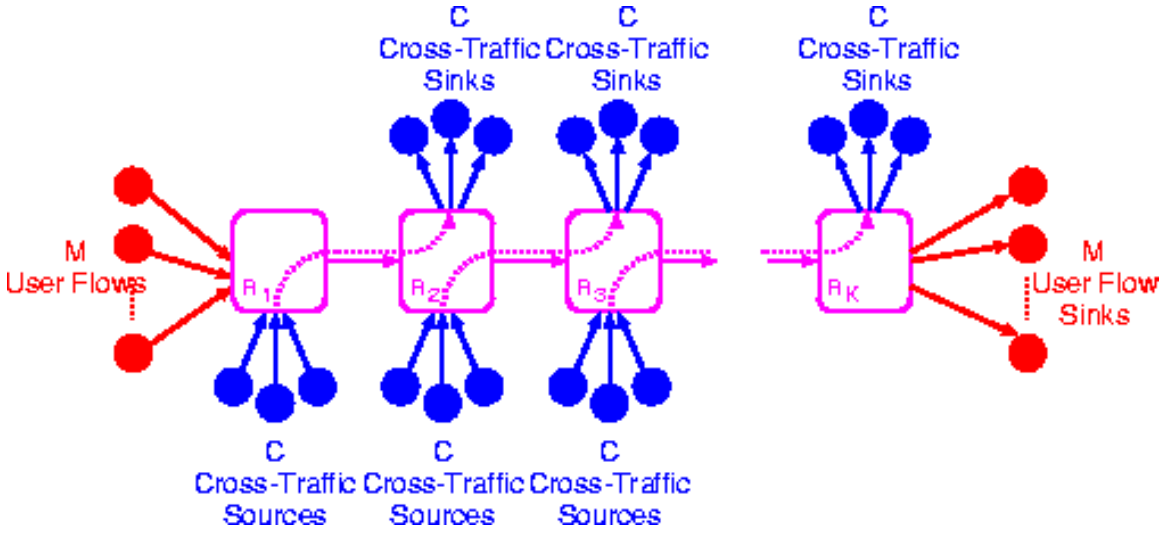


Simulation Results

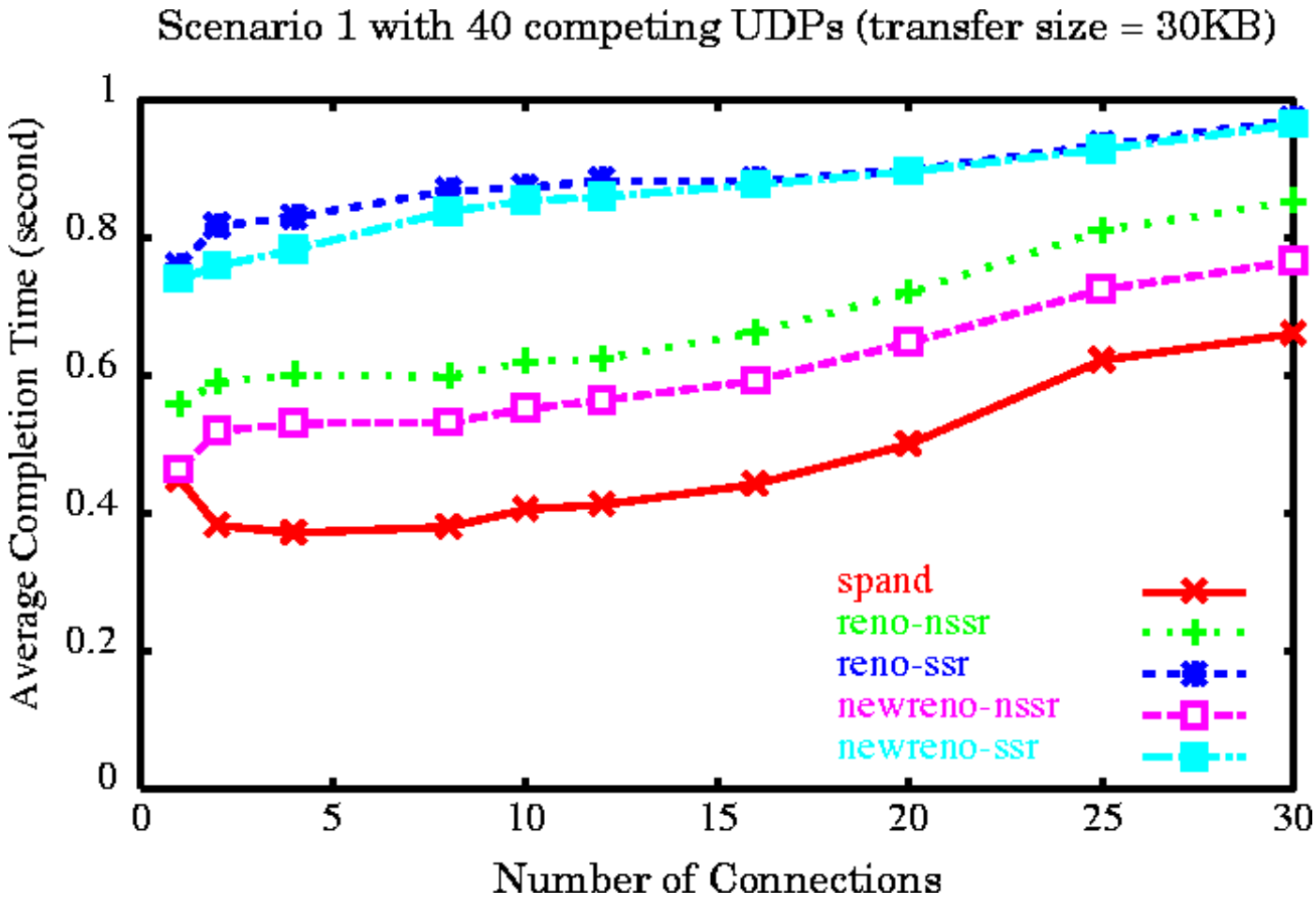
- Methodology
  - Download files in rounds
- Performance Metric
  - Average completion time
- TCP flavors considered
  - reno-ssr: Reno with slow start restart
  - reno-nssr: Reno w/o slow start restart
  - newreno-ssr: NewReno with slow start restart
  - newreno-nssr: NewReno w/o slow start restart

Simulation Topologies

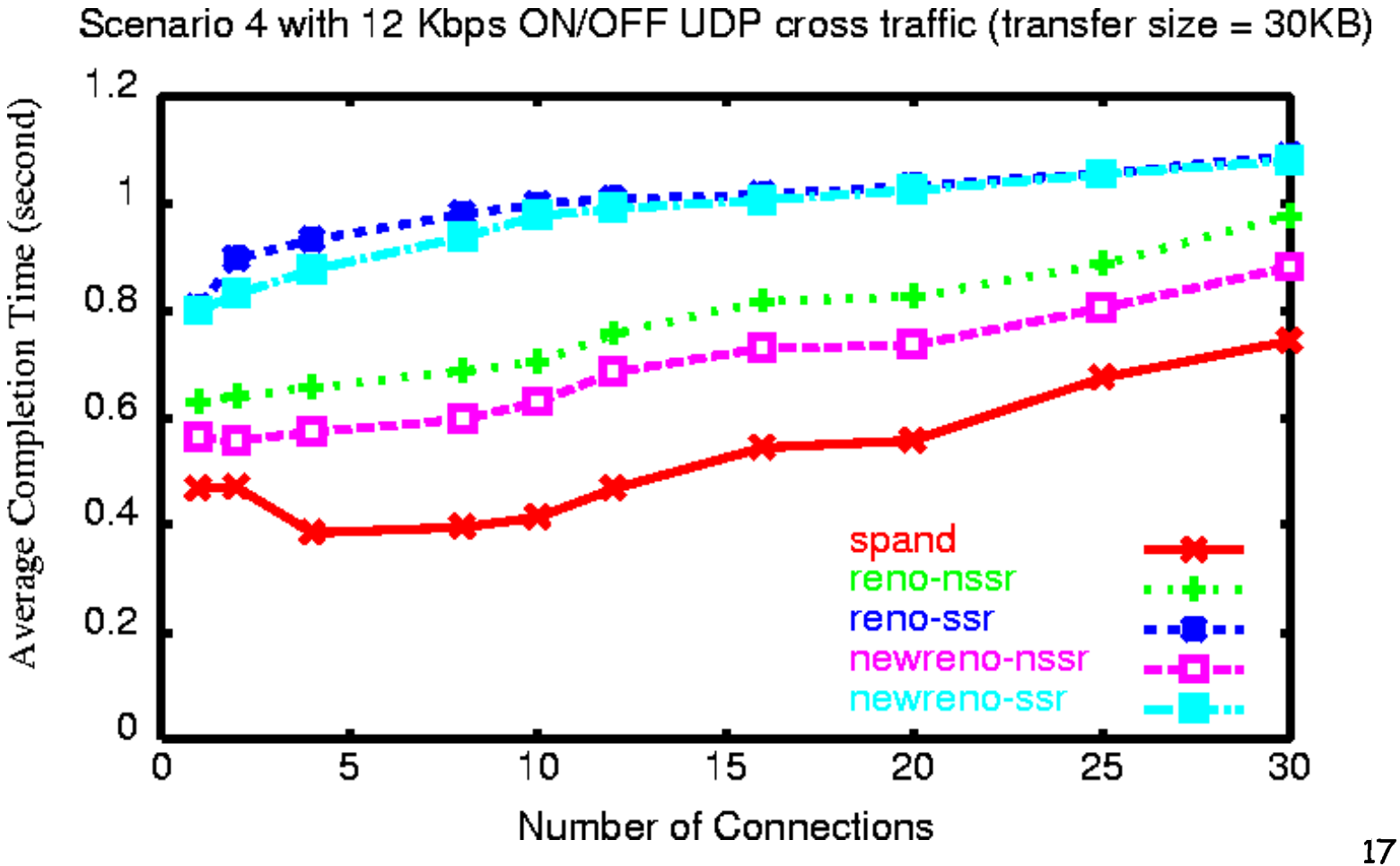




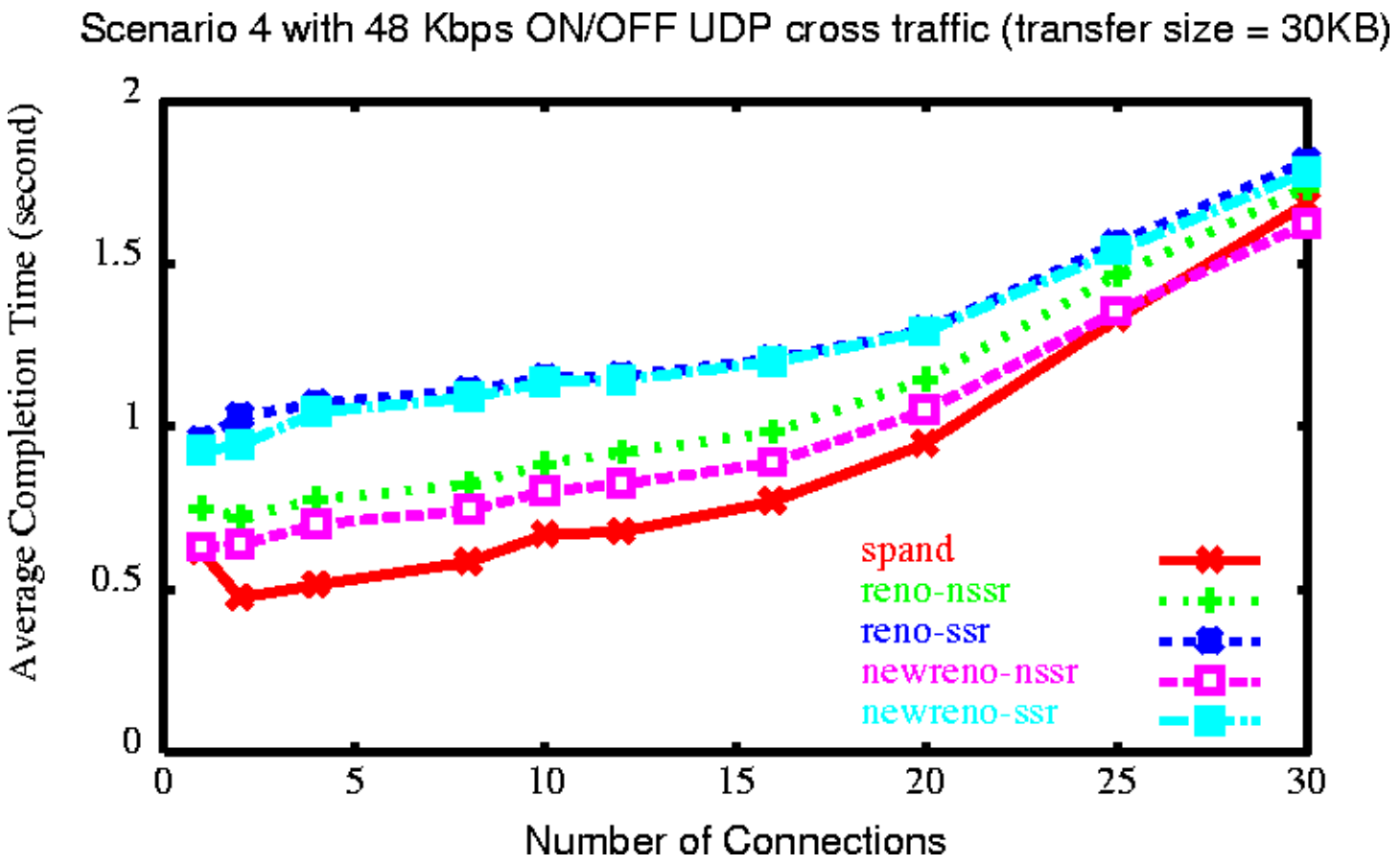
T1 Terrestrial WAN Link with Single Bottleneck



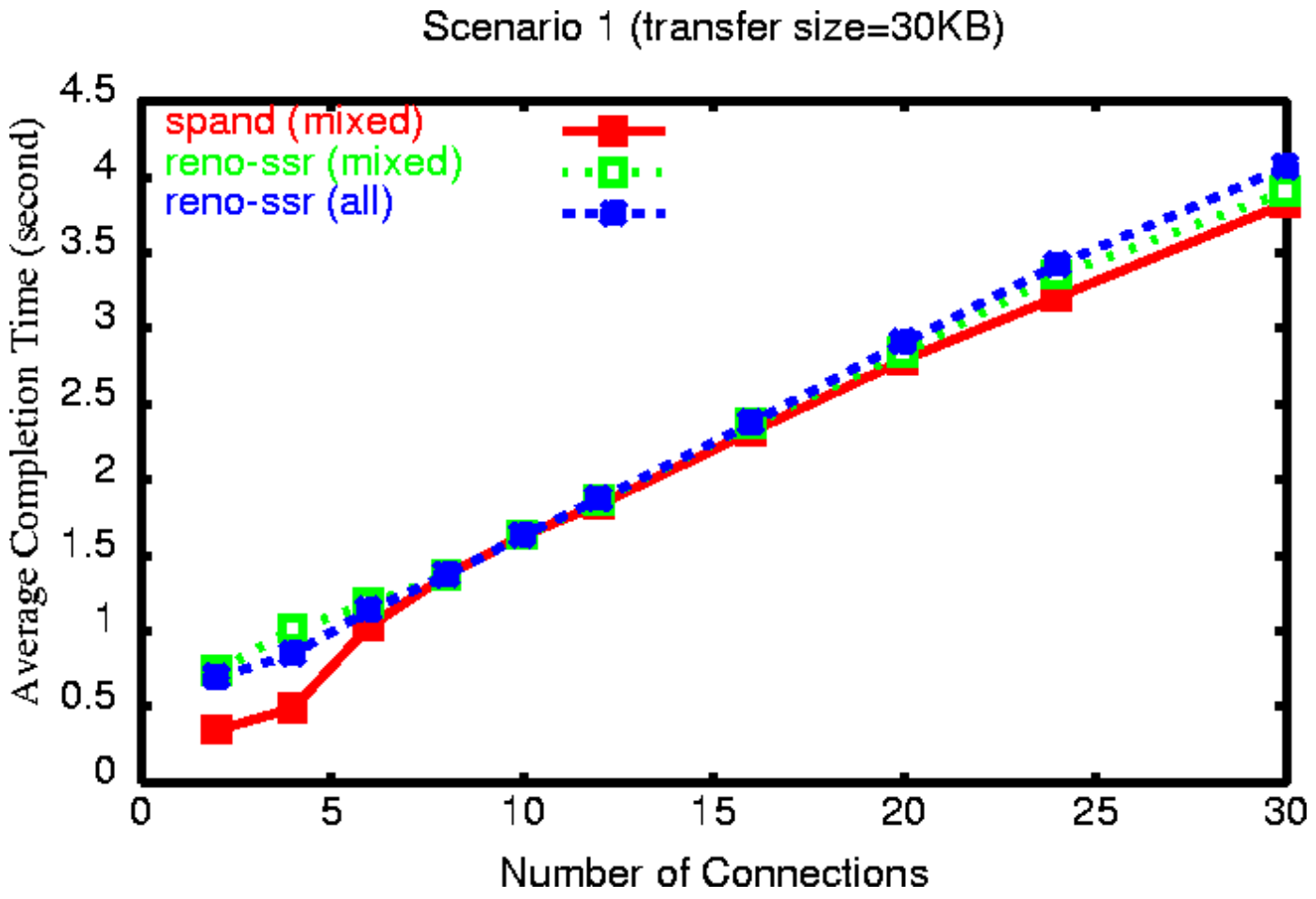
T1 Terrestrial WAN Link with Multiple Bottlenecks



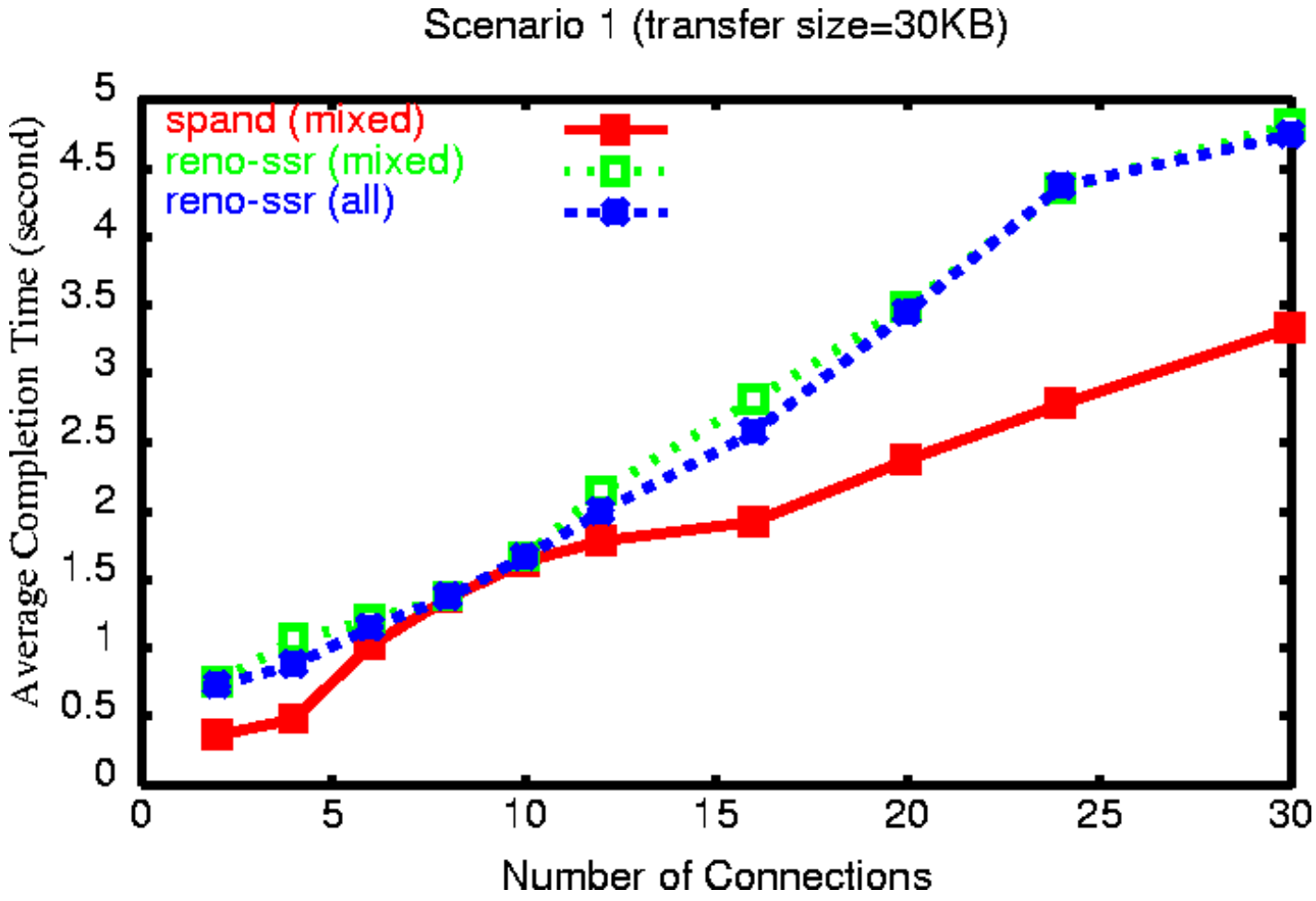
T1 Terrestrial WAN Link with Multiple Bottlenecks and Heavy Congestion



TCP Friendliness (I)
Against reno-ssr with 50-ms Timer



TCP Friendliness (II)
Against reno-ssr with 200-ms Timer



Conclusions

- TCP/SPAND significantly reduces latency for short data transfers
  - 35-65% compared to reno-ssr / newreno-ssr
  - 20-50% compared to reno-nssr / newreno-nssr
  - Even higher for fatter pipes
- TCP/SPAND is TCP-friendly
- TCP/SPAND is incrementally deployable

Future Work

- Real implementation for TCP/SPAND
- Better information aggregation
  - Exponential decay when there is not enough feedback
- Understand pacing for short flows

Acknowledgement

- Brad Karp
- Geoffrey M. Voelker
- Venkata N. Padmanabhan