Fault Recovery for Realtime Networks

Anindo Banerjea
banerjea@tenet.berkeley.edu

The Tenet Group
University of California at Berkeley
and the International Computer Science Institute.

ILP Conference
March 10, 1994.

Outline

- Motivation - why fault management for realtime?
- Fault recovery model
- Framework for experiments
  - Factors affecting recoverability
  - Performance indices of recoverability
- Simulator
- Results
- Conclusions

Motivation

- Realtime service so far
  - Guaranteed service in the absence of faults
  - Uses reservation of available resources
  - Breaks down if resources are affected by fault

- Fault recovery mechanisms so far
  - Telecommunication networks
    - Identical calls
    - Call small compared to network capacity
    - Less state in the network
    - Capacity planning to allow network to perform well under fault
  - Computer networks
    - Best effort paradigm
    - Fault recovery focussed on maintaining connectivity but not performance bounds

Paradigm

- Centralized vs. Distributed
- Precomputation vs. Dynamic
- Link vs. End-to-end
- Application initiated vs. Automatic (N/W initiated)

Fault Recovery Model

- Automatic, distributed, dynamic
- Detection by node adjacent to fault
- Detector sends route update messages
- Detector instigates recovery by sending reroute messages to sources of channels affected
- Source node initiates reroute on channel

Framework of Experiments

- Aim
  - Manage realtime channels in the presence of network faults
  - Minimize the effect on service guarantees

- Methodology
  - Identify set of factors affecting recoverability
  - Pick a good metric to measure the performance of recovery
  - Use simulation to understand effect of factors on recoverability
  - Use above understanding in developing good recovery schemes

Factors Affecting Recoverability

- Network topology *(Capacity planning, over-engineering)*

- Workload *(Application mix, operating load)*

- Routing
  - Algorithm *(Djikstra's, Bellman-Ford ...)*
  - Link vs. Global vs. Hybrid *(locus of reroute constraints)*

- Timing *(Co-operation between nodes)*
  - Immediate vs. Sequential vs. Random

- Retries
  - Number
  - Timing

- Failures Model
  - Single or multiple faults

Workload

- Bandwidth does not capture length of route and delay

- RCSP amenable to workload characterization

- Queueing Delay Load Index is:
  - Linear in bandwidth
  - Monotonically decreasing in delay
  - Additive over links
  - Additive over channels

- Can be used to compare sets of channels

- Single index of network load

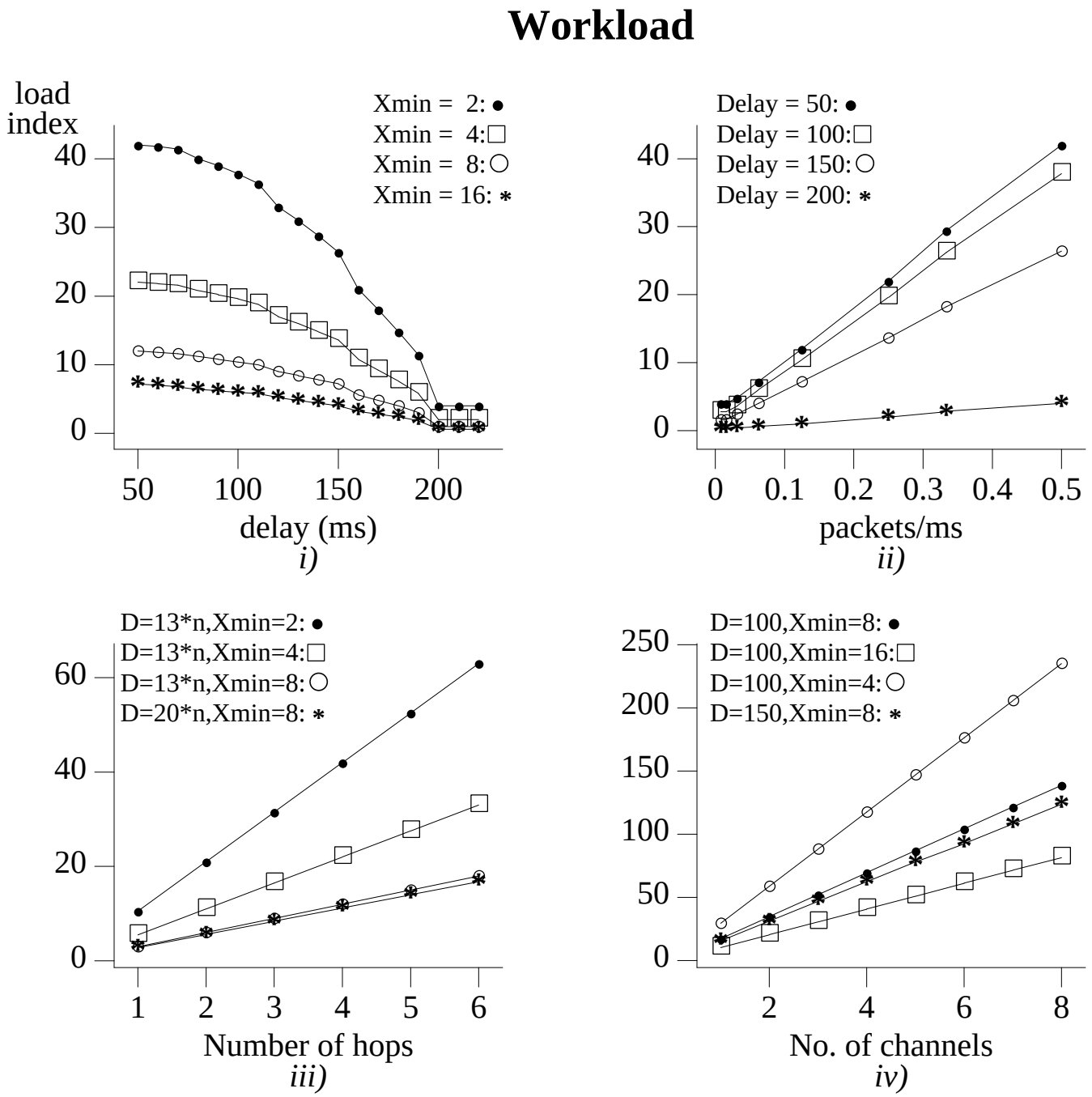


Figure 3. Queuing Delay index vs. other load indices.

Performance Indices

- How much was rerouted
  - Number of channels rerouted (%)
  - Bandwidth rerouted (%)
  - Workload rerouted (%) √

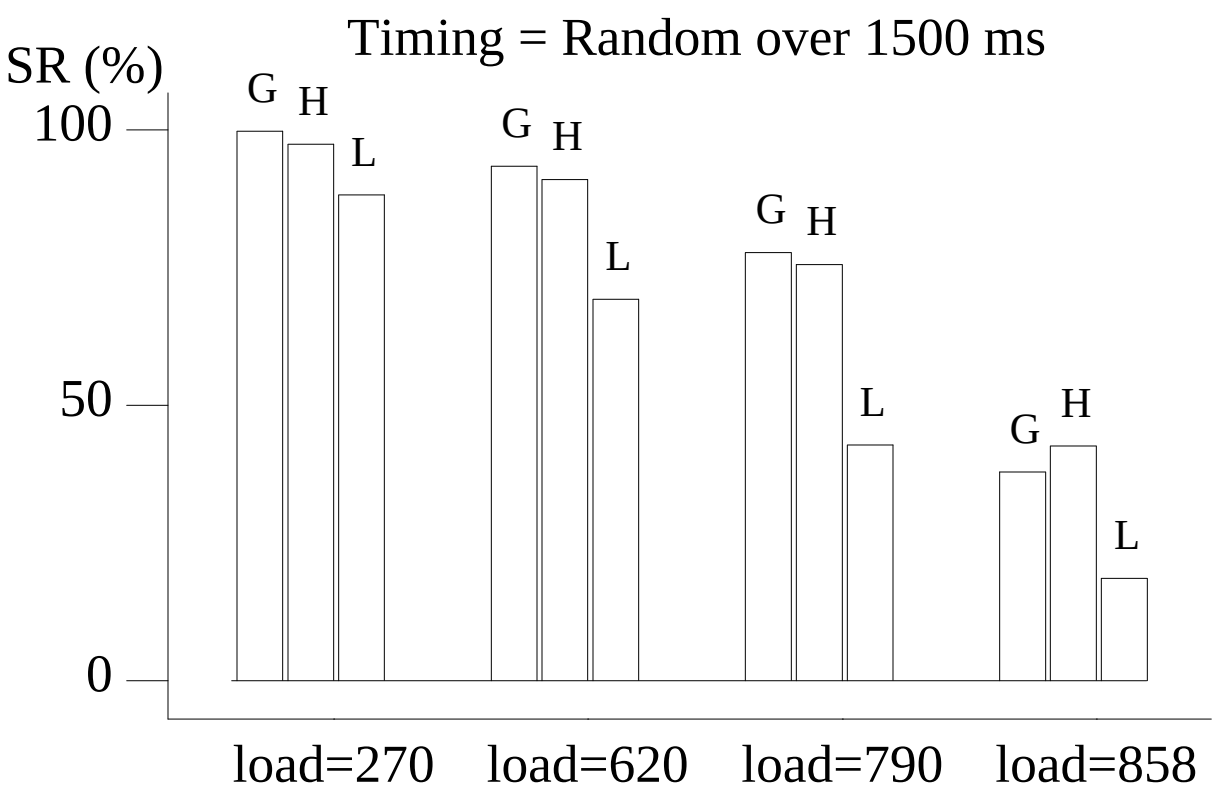
- Goodness of reroute
  - Length of reroute
  - Excess resources consumed √

- Speed of reroute
  - Average time to reroute √
  - Time to completion √

Simulator

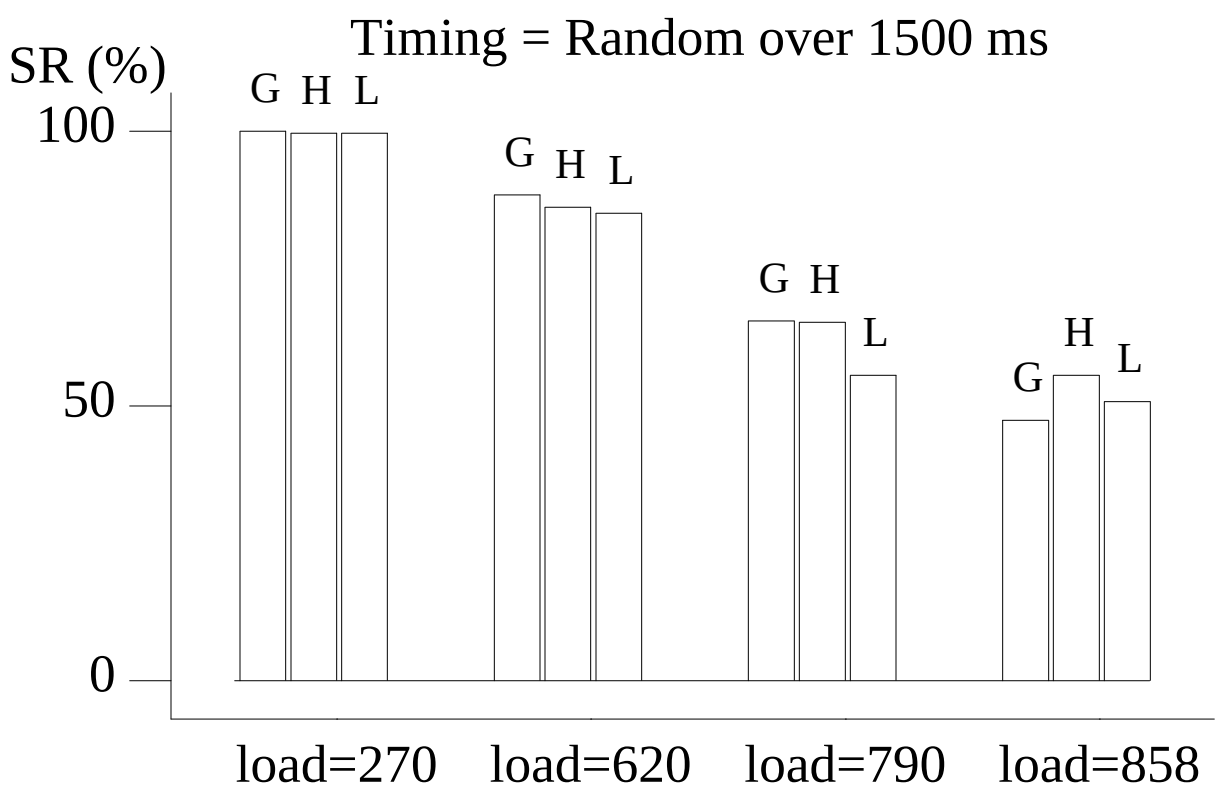
- Based on simulator written by Hui Zhang
- Dynamic Channel Management algorithms written by Colin Parris
- Fault management module added
  - Specify fault events
  - Affected channels are automatically rerouted in remaining network
  - Locus of reroute - Global, Local, Hybrid
  - Timing - Immediate, Random, Sequential
  - Retries
    - Variable number
    - Timing

Results - Success ratio



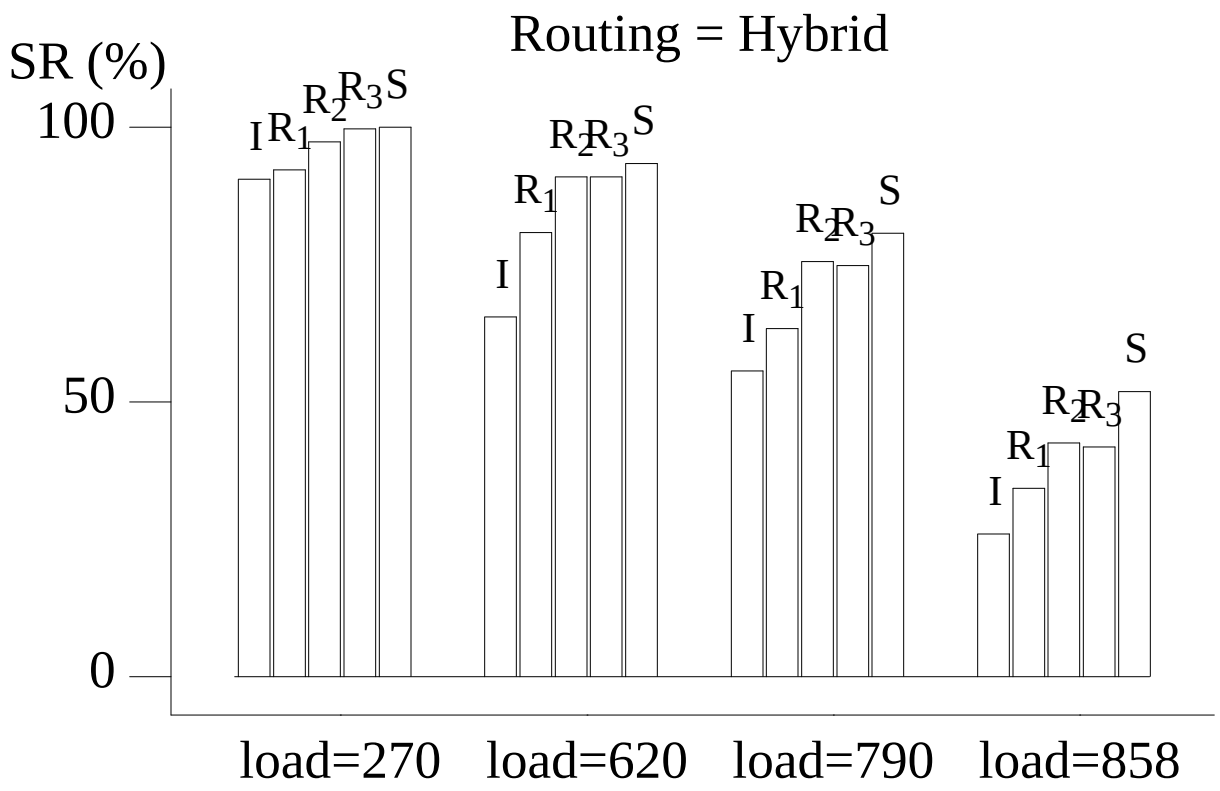
- Rerouting success of Global, Local, and Hybrid locus of reroutes (Single fault)
- Local ≪ Hybrid ≈ Global

Results - Success ratio



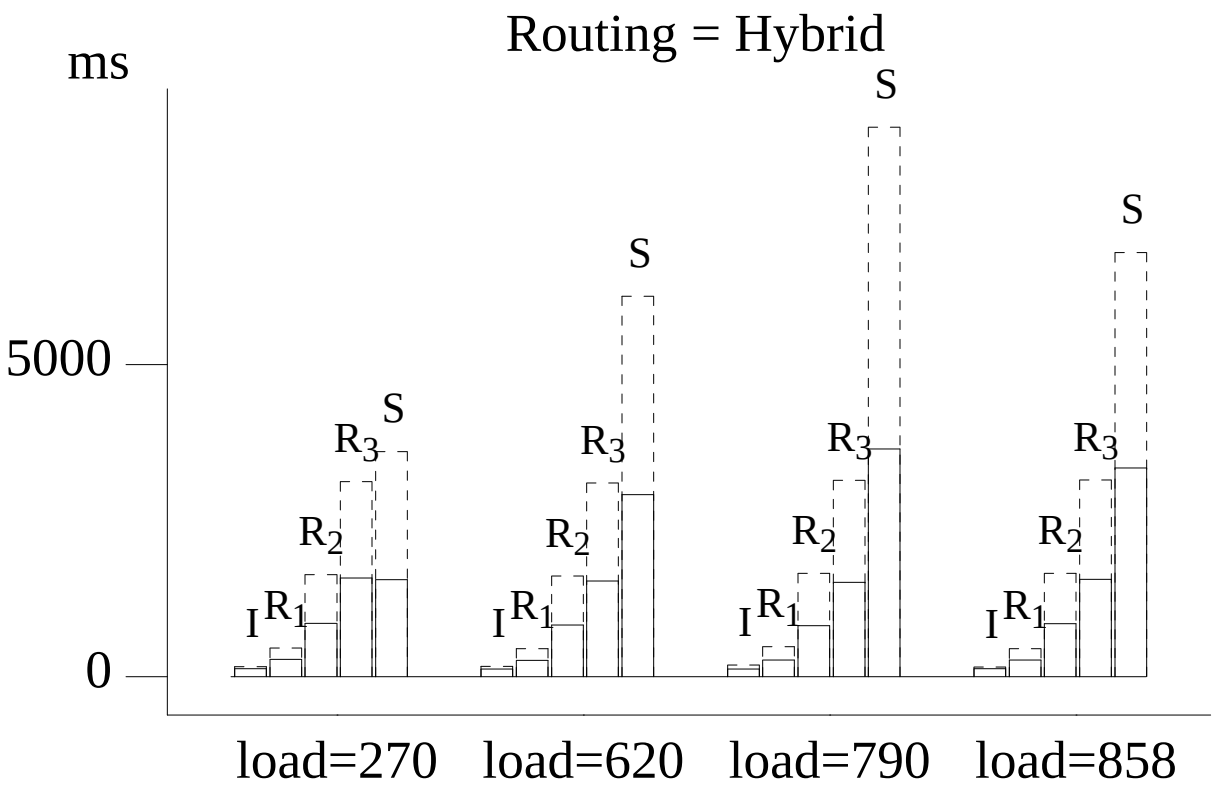
- Rerouting success of Global, Local, and Hybrid locus of reroutes (Multiple fault)
- Now Global and Local are comparable, Hybrid seems to do slightly better at high load.

Results - Success ratio



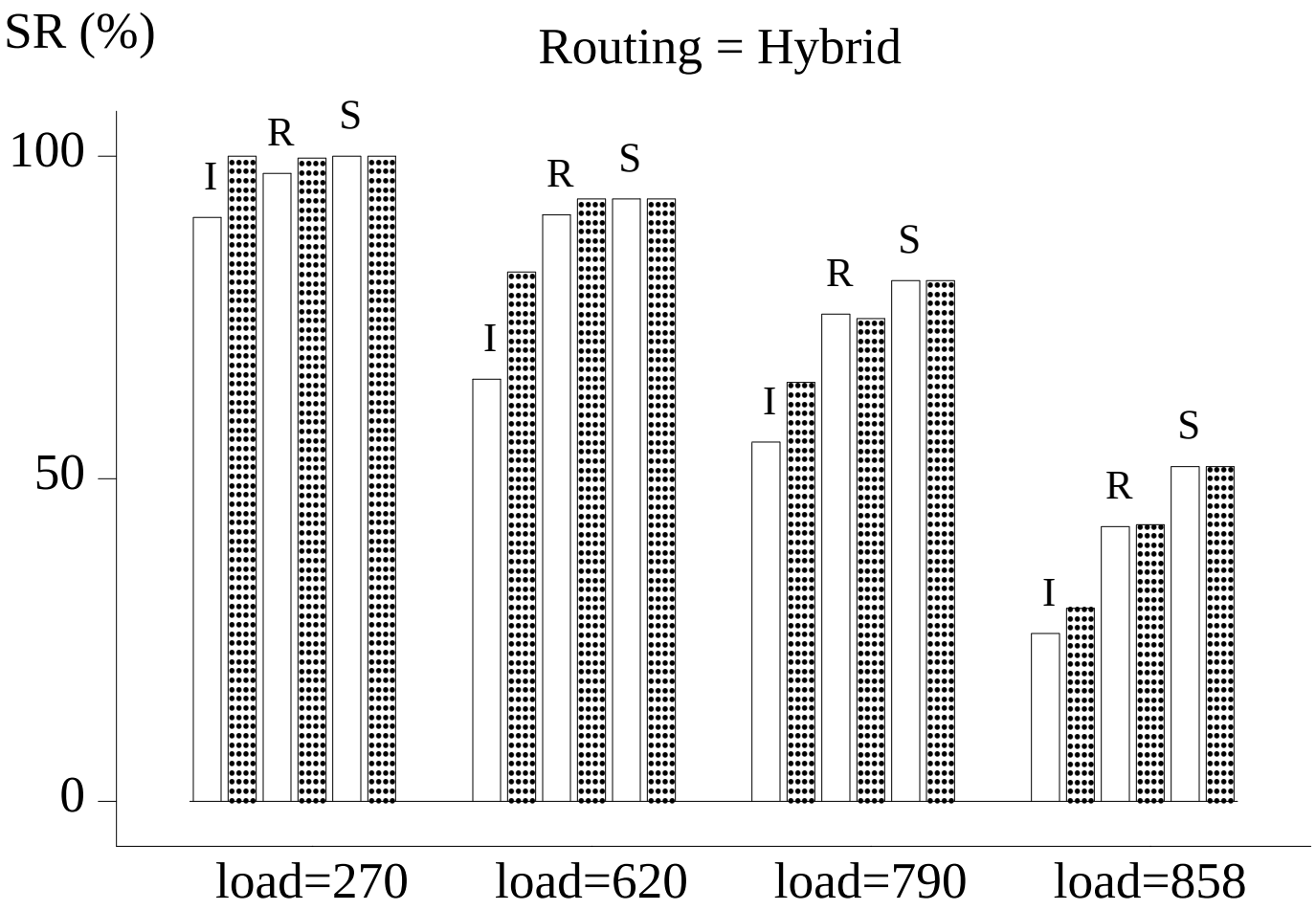
- Rerouting success of Immediate, Sequential, and Random over 300ms, 1500ms, and 3000ms timing (Single fault)
- Immediate ≪ Random ≪ Sequential
- Success improves with increasing Randomization interval
- Sequential has the highest rerouting success

Results - Time to reroute



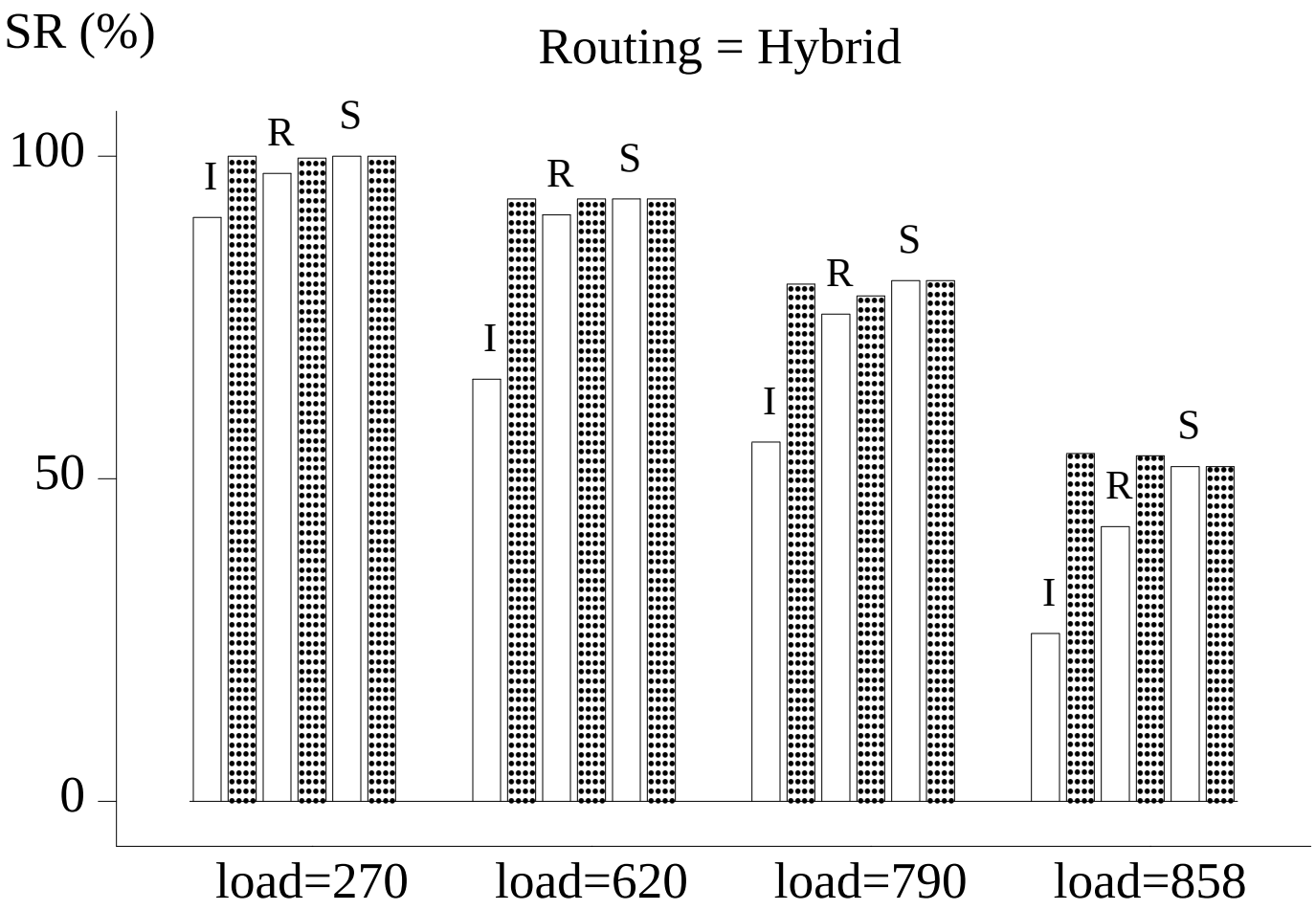
- Time to reroute of Immediate, Sequential, and Random over 300ms, 1500ms, and 3000ms timing (Single fault)
- Immediate ≪ Random ≪ Sequential
- Increasing the randomization interval increases time to reroute

Results - Success ratio with retries



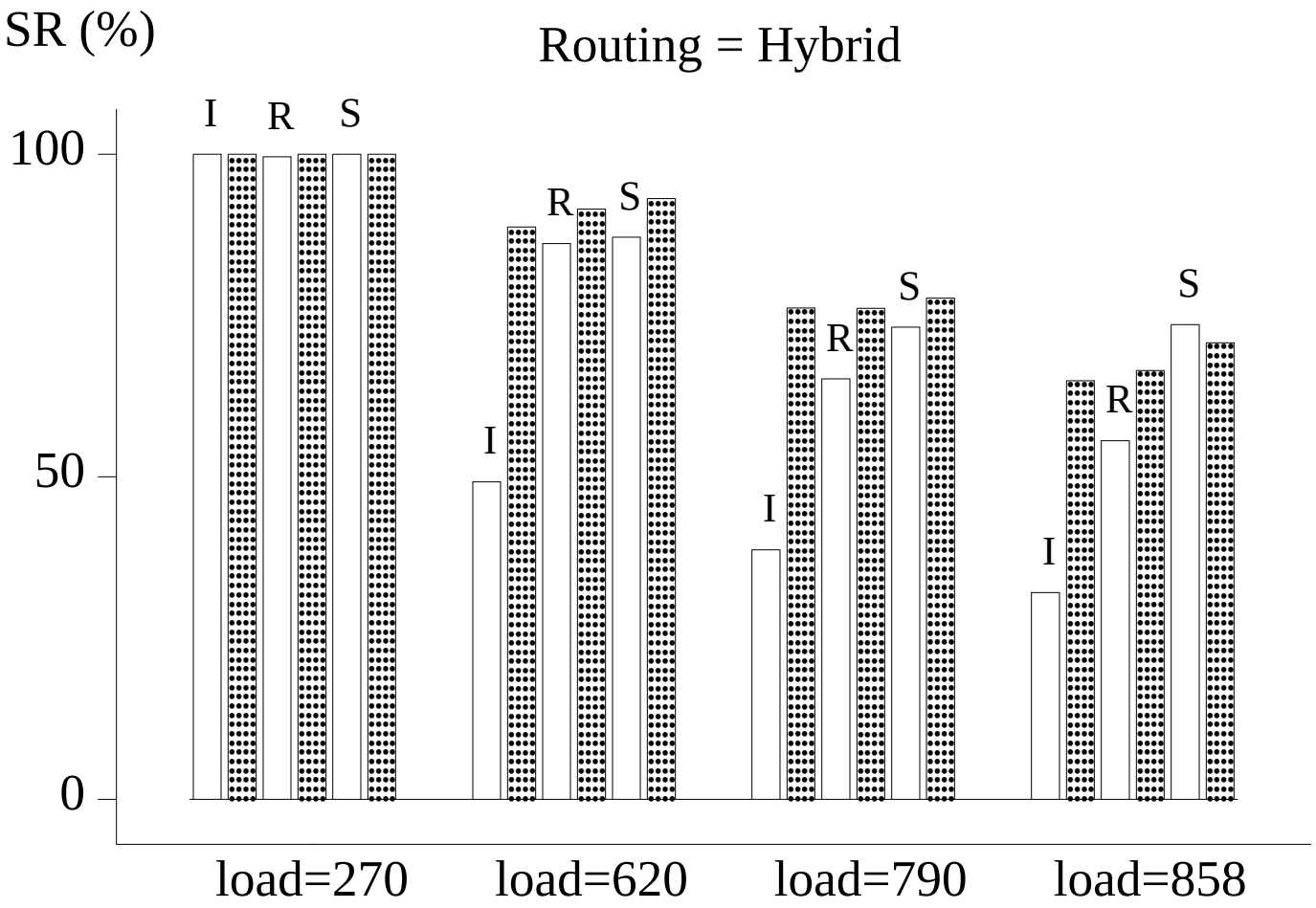
- Immediate retries do not have much effect

Results - Success ratio with delayed retries



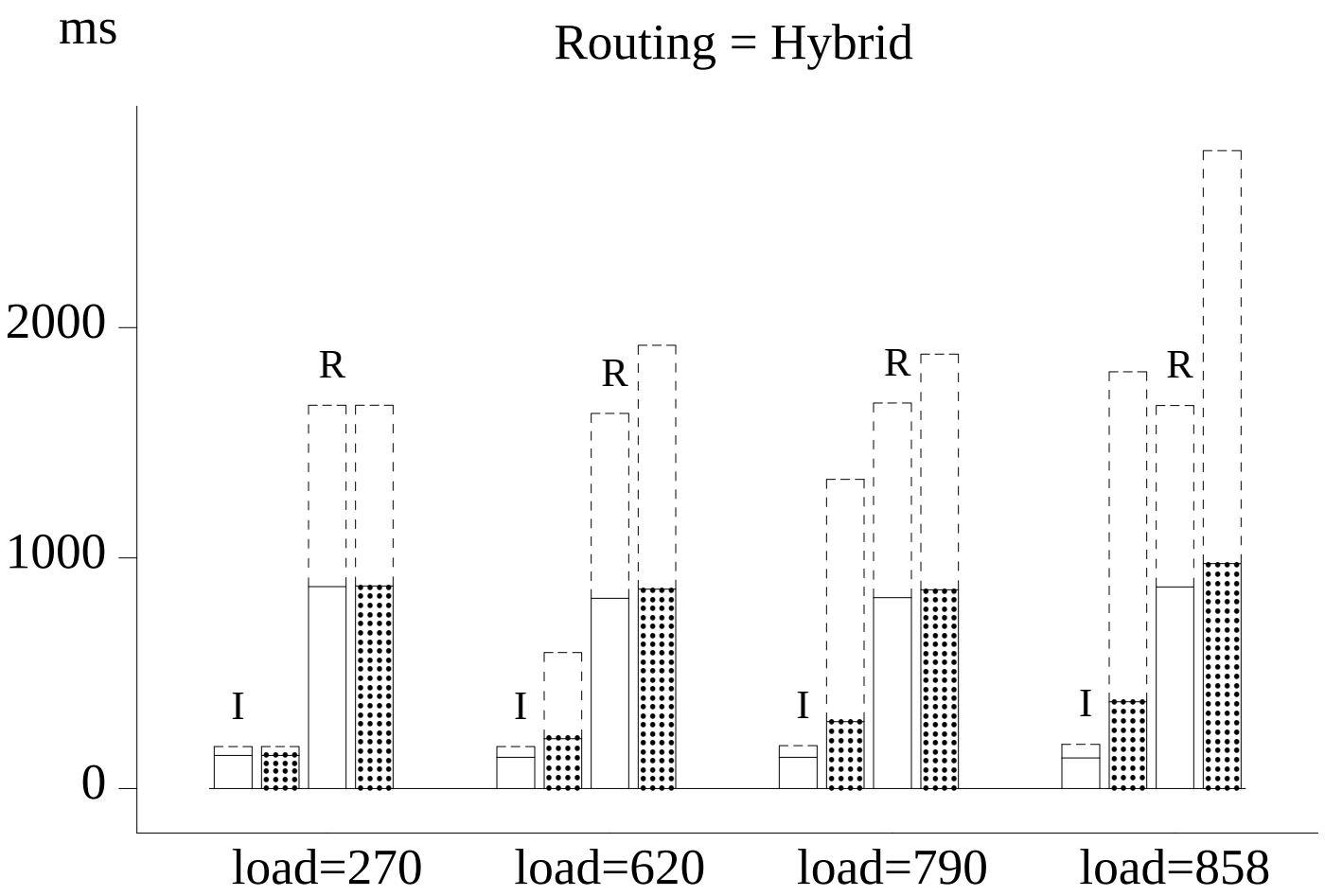
- Delayed retries are as good as Sequential

Results - SR of delayed retries with multiple faults



- Works for multiple faults too

Results - Timing of delayed retries multiple faults



- Average ≈ 200ms
- Very few points beyond 800ms
- Max depends on particular delayed retry mechanism (This is for exponentially increasing intervals)

Conclusions

- Randomization is useful to prevent collisions

- Local routing has some merit for localizing the effect of faults, especially with multiple faults. But lower probability of success. In Hybrid form it appears promising.

- Retries without delay do not increase success, with randomized delays give excellent success, even with Immediate reroute timing.

- Immediate rerouting (even with delayed retries) offers short response time (Av ≈ 200ms) with good rerouting success.

- Hybrid Immediate delayed retries provides the best combination of high rerouting success with low response time and excess resources.