Efficient and Robust Streaming Provisioning in VPNs

Z. Morley Mao

David Johnson

Oliver Spatscheck

Kobus van der Merwe

Jia Wang

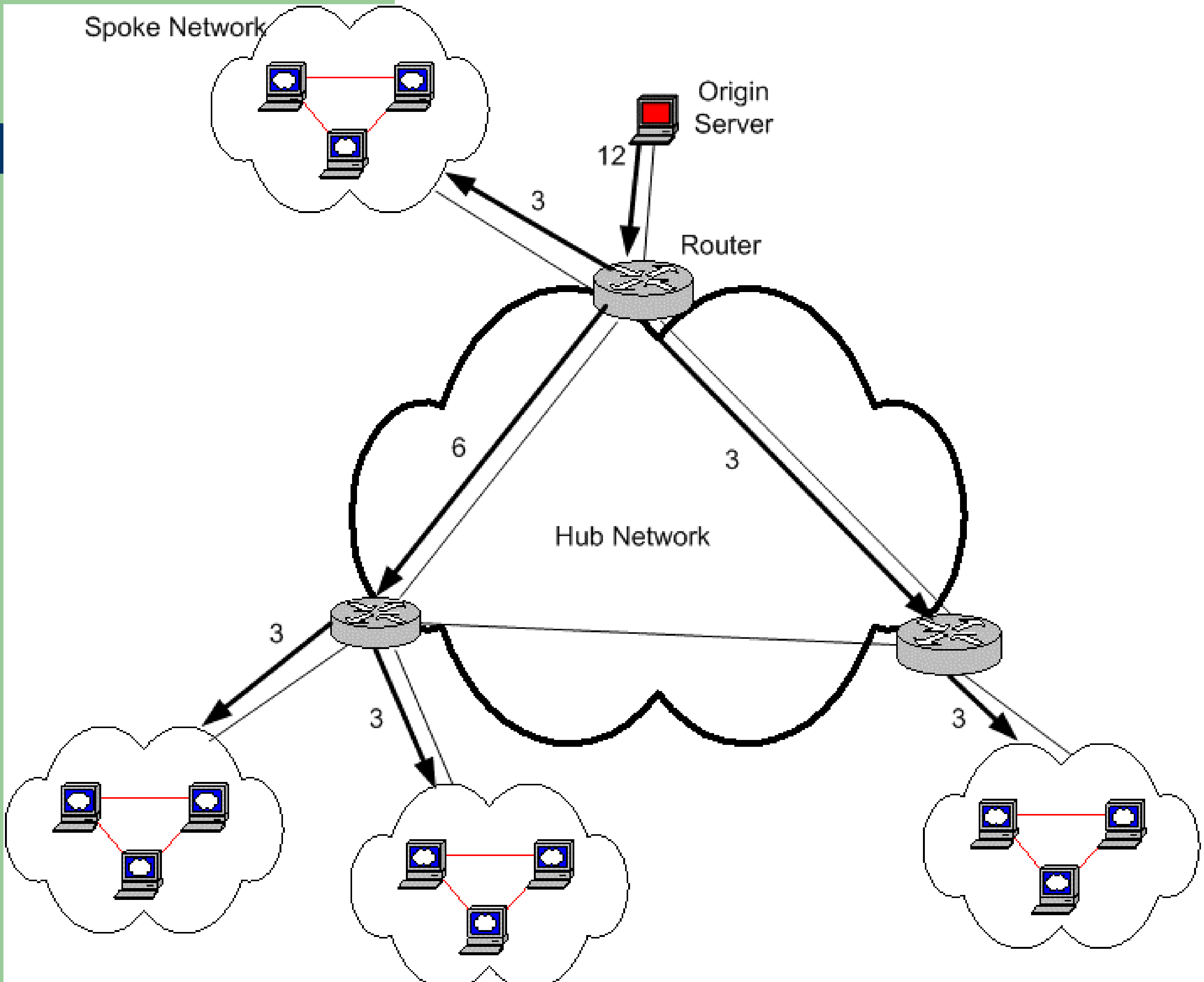
Motivation

- Live streaming in VPNs increasingly popular
  - E.g., CEO-employee town hall meeting
- Lack of layer 3 multicast
  - Requires unicast streaming
- Wide-area bandwidths are expensive and easily congested
- Solution proposal:
  - Streaming cache servers

What are VPNs?

- Virtual private networks
  - Connect remote locations of large companies
  - Implemented using technologies such as Frame Relay, MPLS, or IPSEC
  - Requires
    - privacy
    - performance isolation from public Internet
  - Typically hub and spoke topologies

Hub and spoke topology



Problem statement

1. What are the minimum number of cache servers and their placement to deliver unicast streaming content to a given population?
   - We prove the problem is NP hard
2. How to place the cache servers to minimize total bandwidth usage?

Assumptions for the General Case

- Known network:
  - topology, link capacity, user location
- Known origin server, bandwidth of the stream
- Request routing: from any cache server
- Cache location: at any router
- Application requirement
  - Bandwidth is the critical resource
- Bandwidth usage: cannot exceed link capacity
- Sufficient server capacity
- VPN topology: hub and spoke

Redirection overview

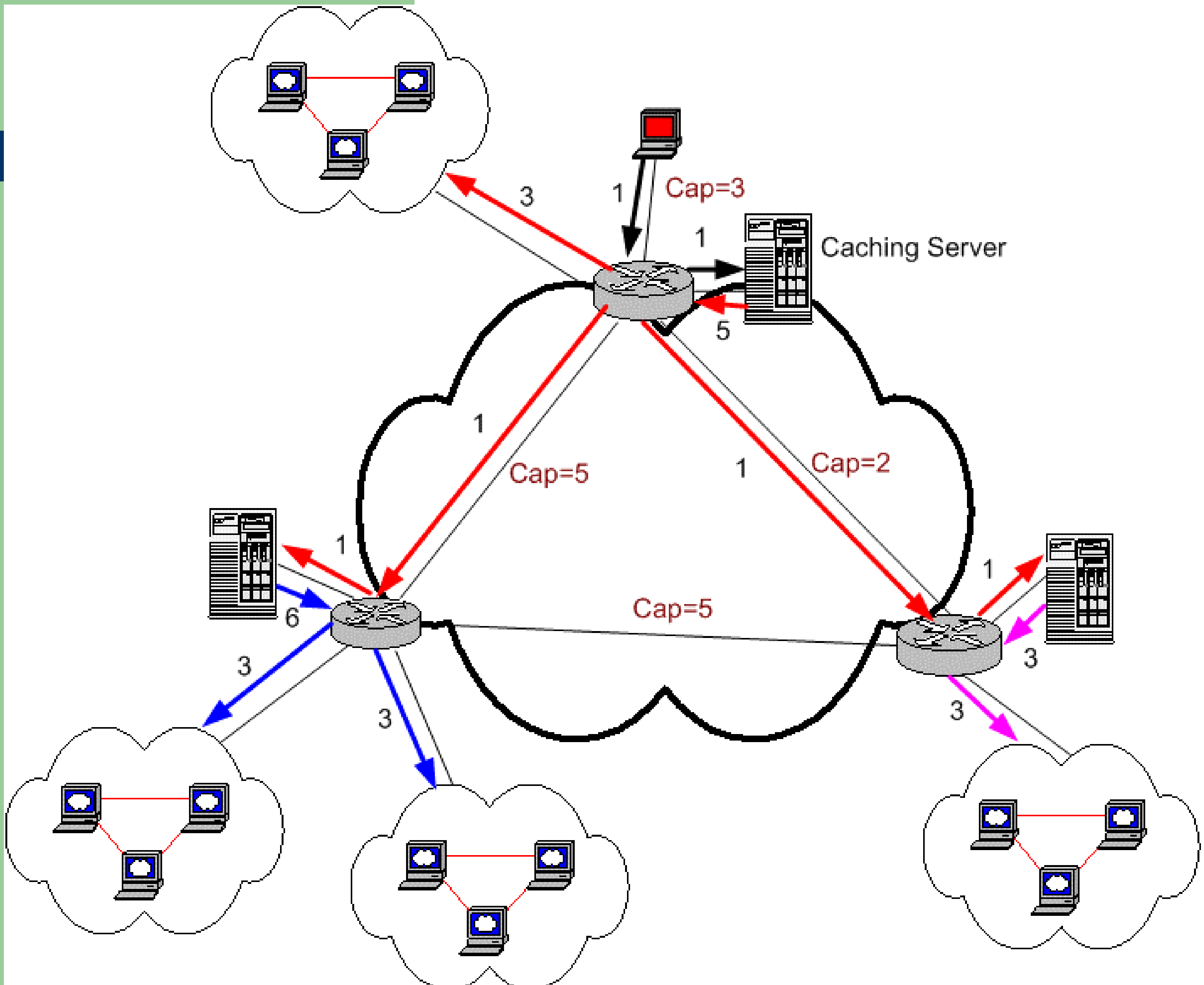
- Interception based
  - Clients request from origin server
  - Caches intercept requests
  - Optimal greedy algorithm: O(V)
- Router based redirection
  - Clients connected to the same router request from the same server
  - $O(|V|^2|E|)$
- Client based redirection
  - Each client can request from a different cache
- Flow-based redirection
  - End to end routing controlled

Increasing implementation complexity, But fewer cache servers

Interception proxy algorithm

- Greedy algorithm
  - Walk the tree from the leave nodes to the root
  - At each depth, place a cache at overloaded nodes
  - Overloaded node:
    - Demand from children exceed incoming link capacity
- Assigns the minimum number of caches assuming flows are restricted to the distribution tree T built from the origin server
- Running time
  - O(|V|): visit each link once.
- Algorithm is optimal for interception proxies

Interception proxy algorithm



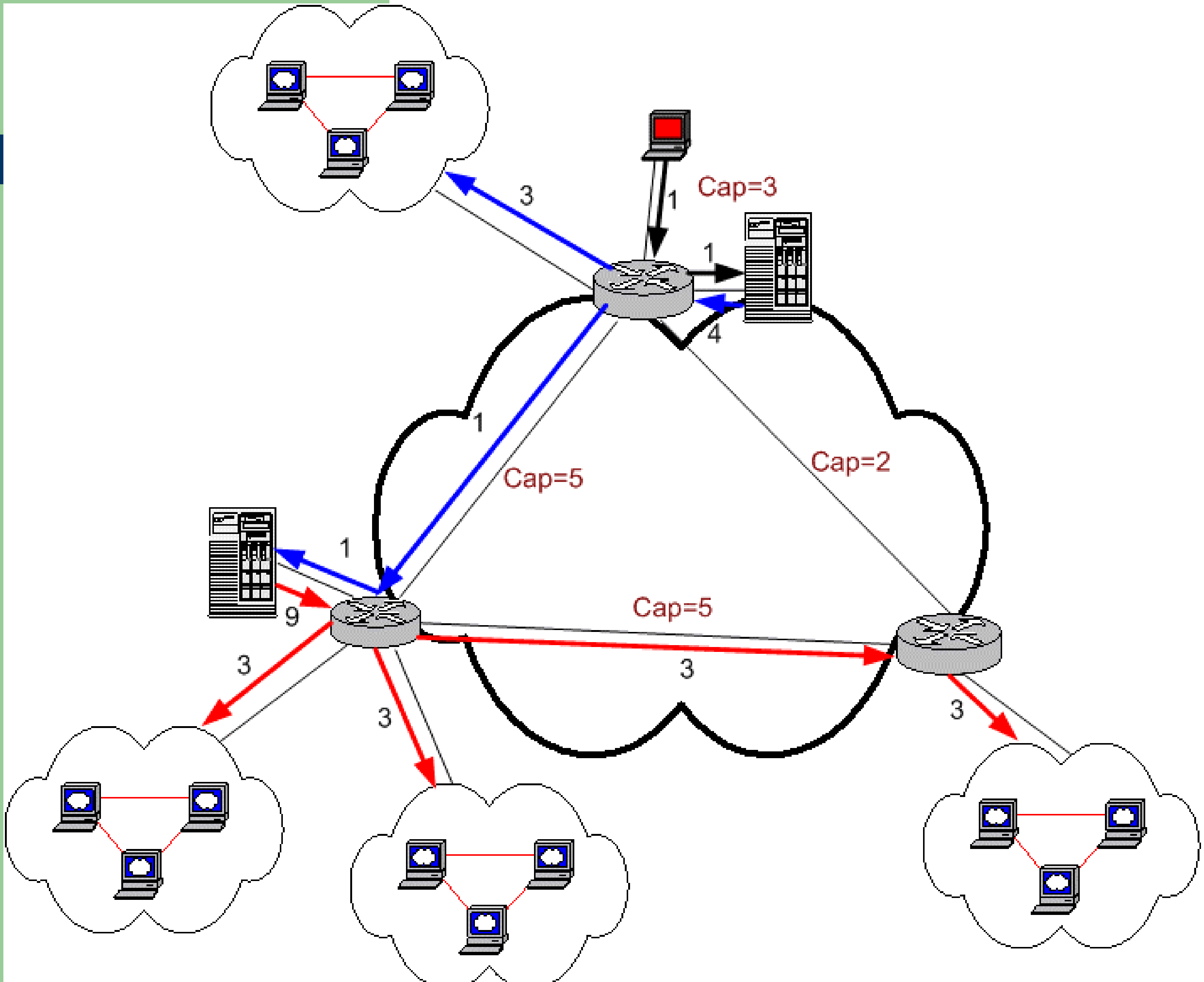
Interception proxy algorithm -- Minimizing bandwidth

- Greedy gives minimum number of caches
  - Flows restricted to original tree
- Bandwidth can be reduced
  - By pushing caches towards leaves
- Algorithm is optimal interception proxies

Router based redirection

- Algorithm:
  - Calculate for each overloaded node its merit value
    - Merit based on how many overloaded nodes it can alleviate if there is a cache placed there
    - Requirement: all hosts of the same router need to request from the same cache
  - Walk the tree from leaf nodes to root
  - Pick the node at each depth with the max merit
  - $O(|V|^2|E|)$

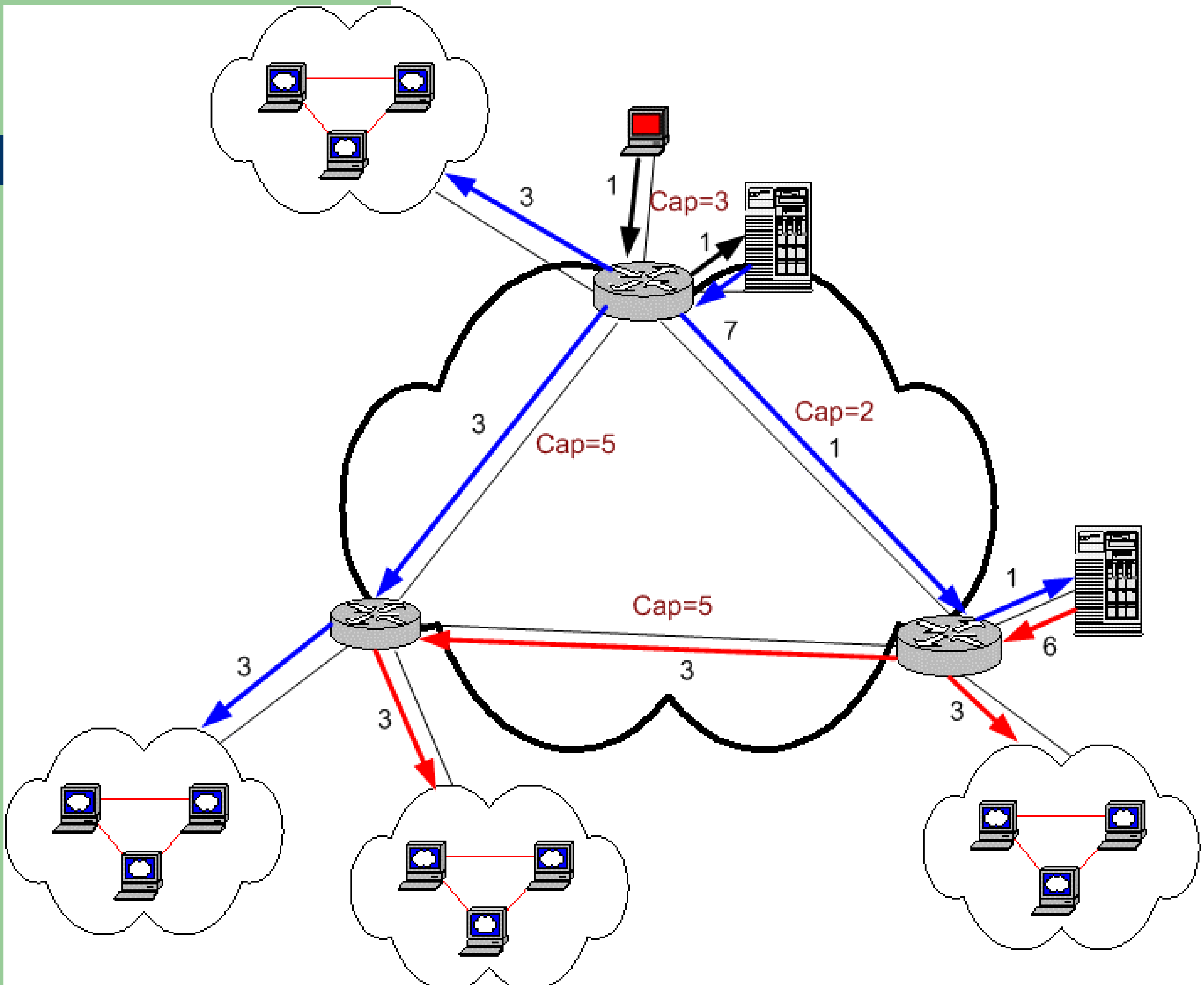
Router based redirection



Client based redirection

- Relax the requirement of router based redirection
  - Each client can choose its own cache server
- More fine grained redirection

Client based redirection



Flow based algorithm

- All existing algorithms use IP routing
  - Certain links may be underutilized
- Assume controlled end-to-end routing
  - Through MPLS, OSPF weight setting
- Algorithm:
  - Given Greedy's cache placement
  - Try to delete each cache and test for max flow
  - Delete if demand satisfied

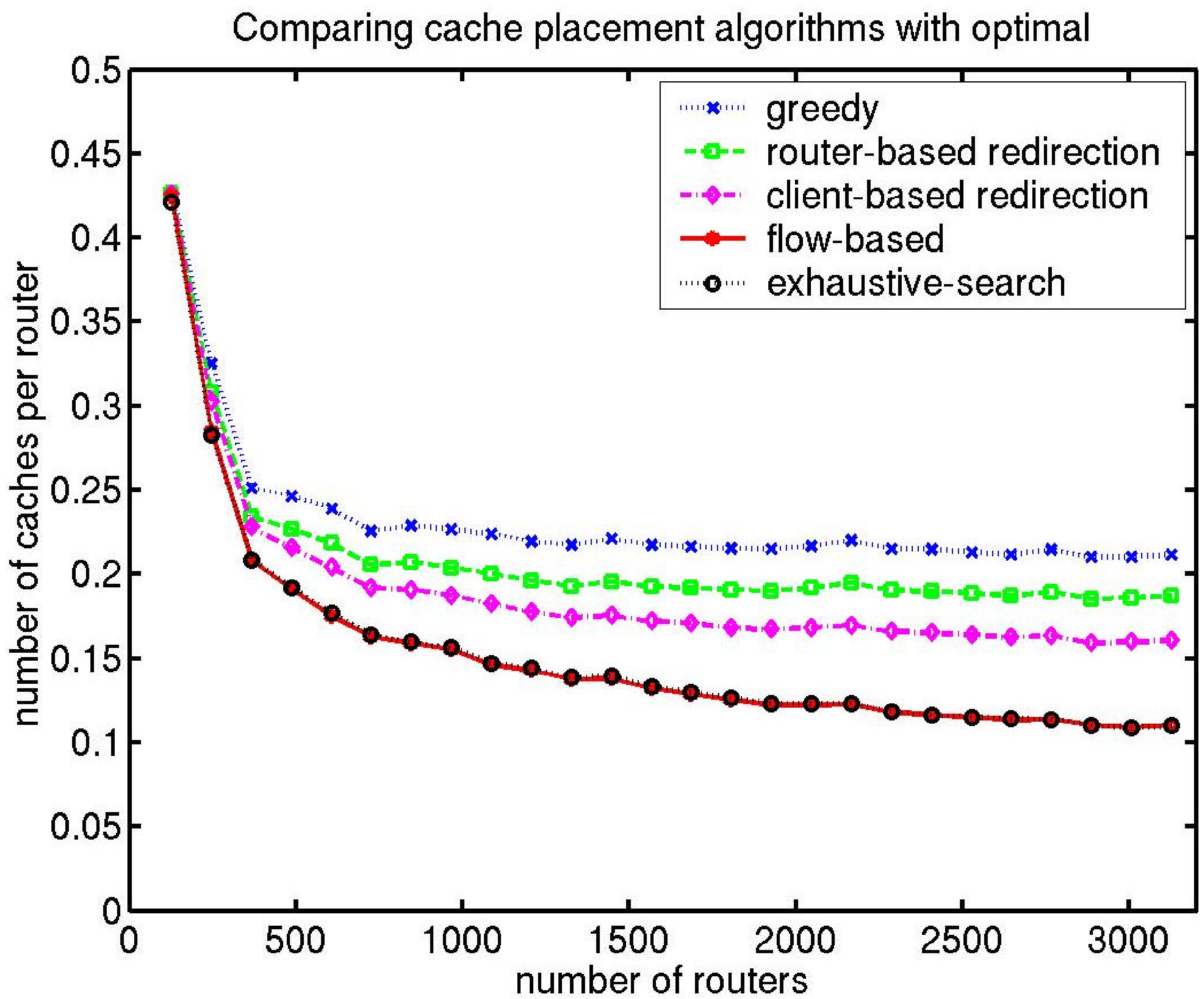
Local exhaustive search

- General problem is NP-hard
- Exhaustive search takes exponential time
  - Infeasible for large topologies
- Local exhaustive provides an upper bound
  - Assume every hub node contains a cache
  - Exhaustively search each stub network
  - Sum up total number of caches
- Assumes controlled end-to-end routing

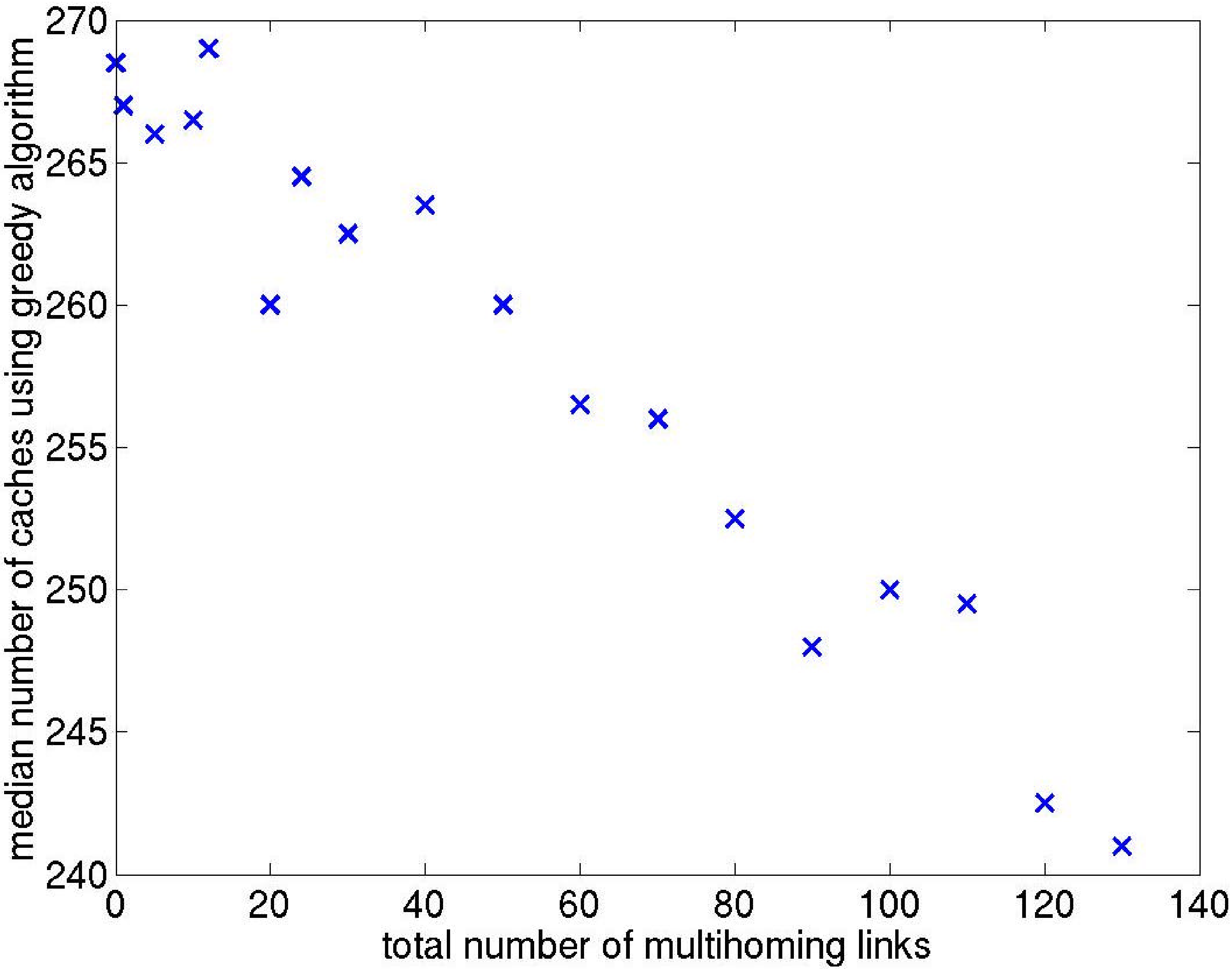
Results overview

- Simulation methodology
  - Algorithms implemented on typical hub-spokes
    - Three classes of VPNs: large companies, retail stores, engineering firms
  - Simulator based on GT-ITM topology generator, Stanford GraphBase
  - Empirical error distribution for link capacity estimates
    - Based on 600 measurements using Java and activeX based client side measurement tools

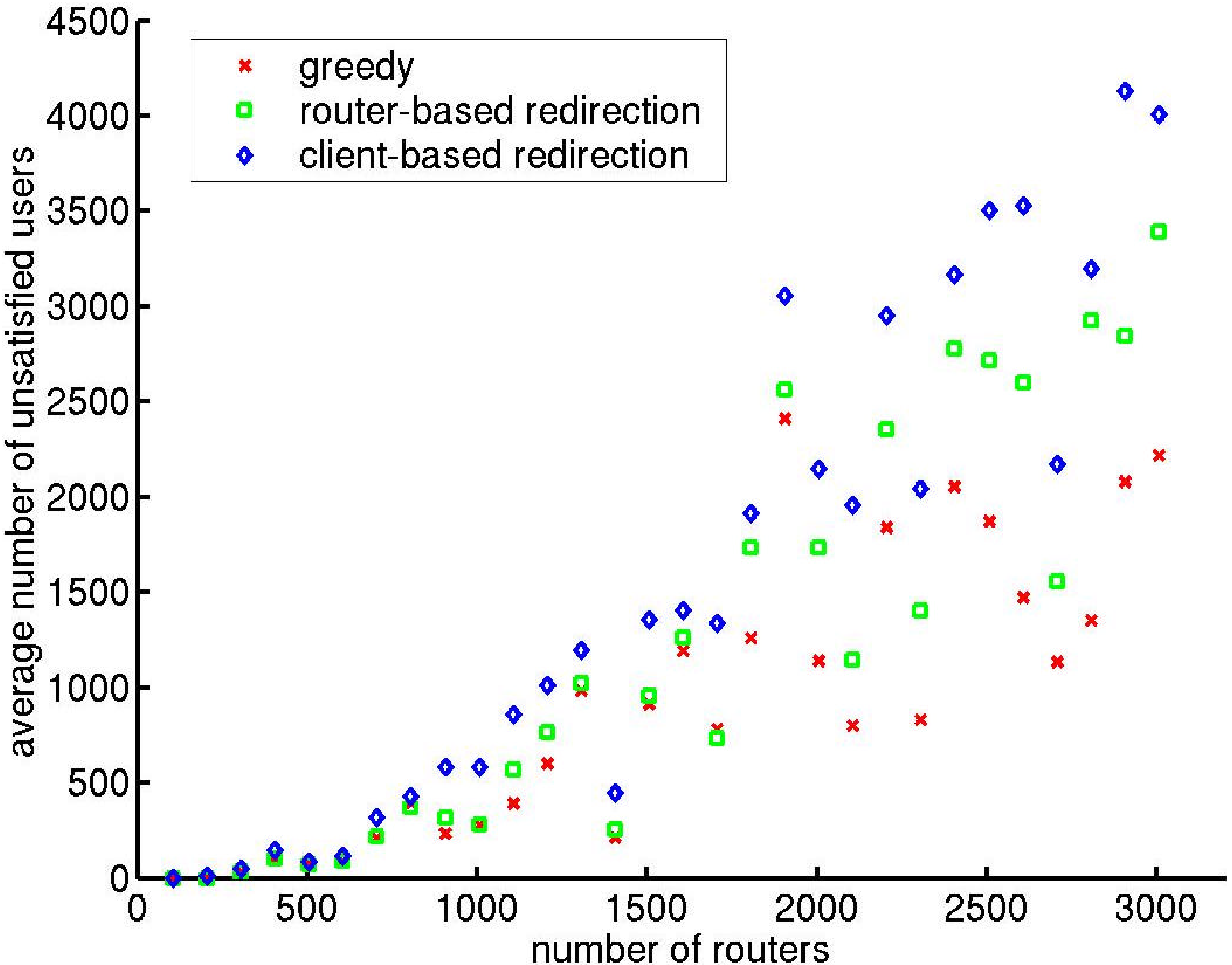
Compare the algorithms



Effect of multihoming



Error resilience



Concluding remarks

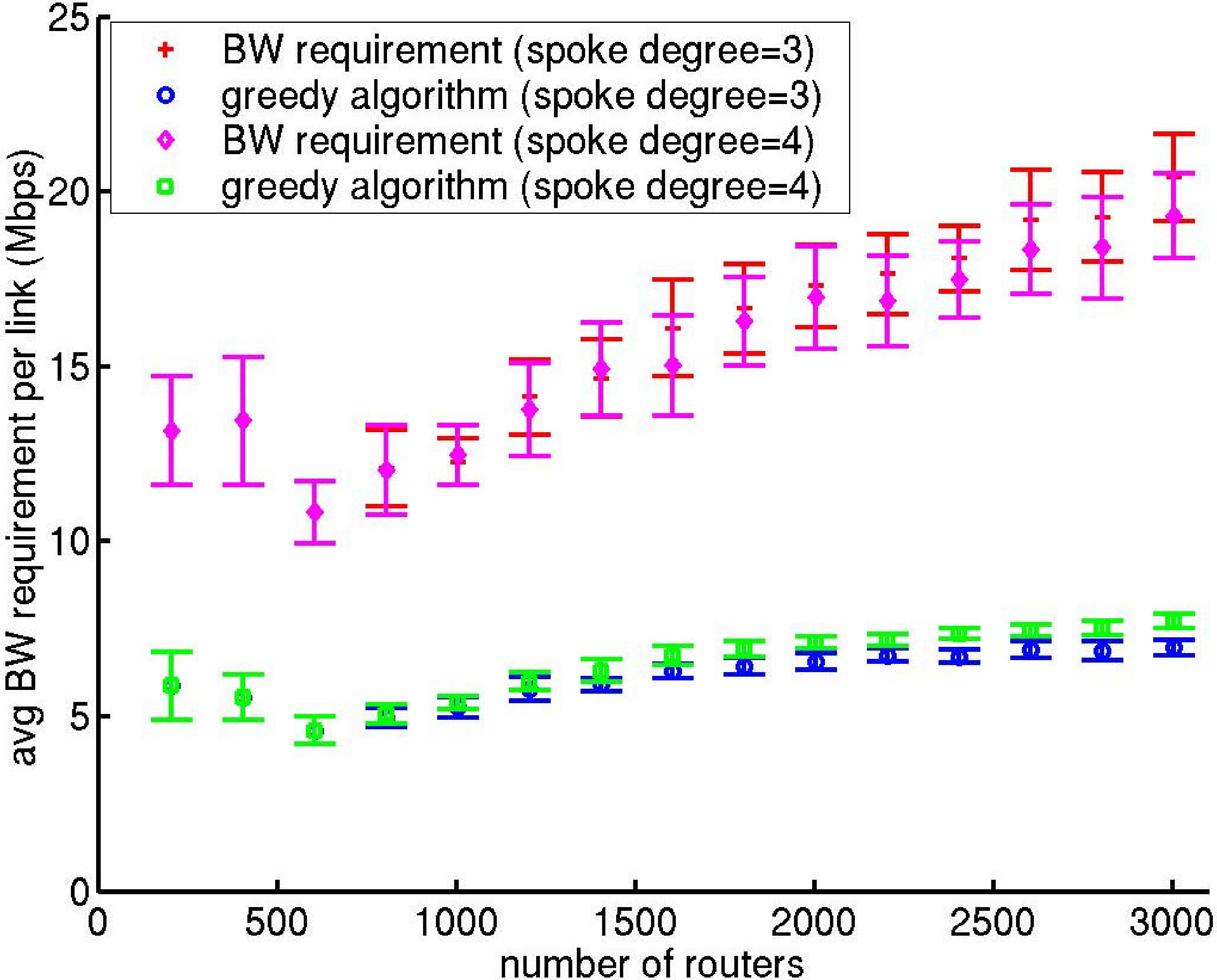
- Study the problem of cache server placement in VPNs for unicast based streaming
- Developed provably optimal algorithm
  - Minimum number of caches
  - Minimum total bandwidth usage
  - Assuming interception based algorithm
- General problem is NP-hard
  - Router based redirection
  - Client based redirection
  - Flow based algorithm: very close to optimal

Related work

- Cache placement for web traffic
- Server placement in overlay networks
- Assumptions of previous work
  - Ignoring network constraints
- Main distinction of our work:
  - VPN environment
  - Minimum number of caches for a known user population
  - Consideration of robustness of algorithm in face of imperfect input data

Extras

Effect of spoke domain size



Error resilience: using robust algorithm