A Naming Service for Overlay Networks

A Master of Computer Science Presentation

by

Gregory Mattes

Jörg Liebeherr, Advisor

Multimedia Networks Group
Department of Computer Science
University of Virginia

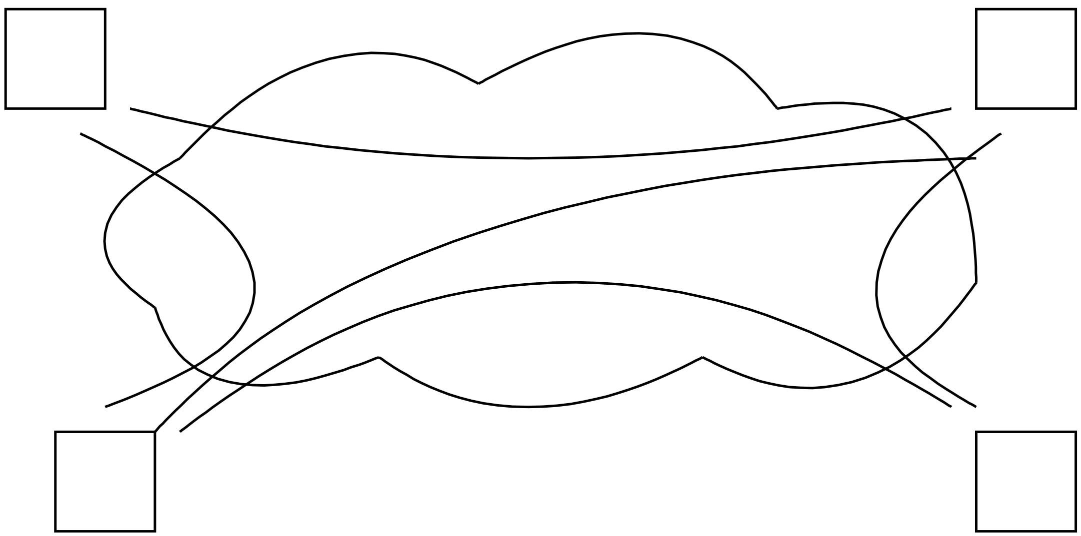
22 July 2005

Presentation Outline

- Overlay Network Addressing
- Naming Service Challenges
- Naming Service Solution
- HyperCast Naming Service
- Naming Service Evaluation
- Conclusion

Overlay and Substrate Networks

- Built by applications
- Self-organize to form network
- Called an overlay
- Uses underlay or substrate network for message transport: commonly the Internet



Logical Addressing

Logical Address is an address of an application in an overlay network used for overlay message routing.

- Bit String: `10011`
- Coordinate tuple: `(565, 359)`

Logical Address Limitations

- Cumbersome to use for application programmers
  - Applications should not be dependent on logical address scheme
  - Applications, services, and users are not identified by logical addresses
  - Logical addresses cannot be used to define arbitrary groups of network peers
  - Logical addresses of an application may be variable
- Applications prefer meaningful names

Mnemonic Names

- Names like in DNS (`www.example.com`). In DNS names are structured, but not as IP topology.
- Mnemonic names can overcome the limitations of logical addresses. Ex. *10011* $\leftrightarrow$ *Greg*
- Names have semantic information that identifies applications, services, and users. Ex. *(565, 359)* $\leftrightarrow$ *Police Officer*
- Names be used to identify groups of applications, services, and users. Ex. *(565, 359)* $\leftrightarrow$ *Police Officer*, *(234, 758)* $\leftrightarrow$ *Police Officer*
- Names are unstructured with respect to logical addressing schemes
- Names are independent of logical addressing schemes

Logical Address and Name Comparison

Property	Logical Address	Name
Useful for Message Routing	Yes	No (if unstructured)
Logical Address Scheme Independent	No	Yes
Application Specific Semantic Value	No	Yes
Can Identify Groups	No	Yes
Can Identify Users	No	Yes

Names give a user-level addressing scheme, similar to DNS in the Internet.

Challenges and Issues of Naming Service for Dynamic Overlay Networks

- **Goals**
  - Bind logical address to a name
  - Name service for dynamic overlay networks that resolves bindings
  - No assumption of a fixed infrastructure, directory, or central respository
  - Ability to define group names
  - Deal with frequent changes of logical address (peer mobility)
- **Issues**
  - Is a naming service in a dynamic overlay network viable?
  - How will it perform?
  - How can names be trusted with no trusted third party?
  - How to disseminate information on bindings?

A Naming Service For Dynamic Peer Networks

- **All peers** participate in the naming service in the same way
  - Completely symmetric
  - No centralized directory
  - No designation of particular naming service nodes
- Naming Service Operations
  - Resolves forward queries: name $\rightarrow$ logical addresses
  - Resolves reverse queries : logical address $\rightarrow$ names
  - Incorporates trust relationships between peers
  - Operation to exchange trust information

Name Binding

Maps a logical address to a mnemonic name

Auth Flag	Name Size	Name	Logical Address Size	Logical Address	Signer Name Size	Signer Name
1 byte	2 bytes	$>$0 bytes	1 byte	$>$0 bytes	2 bytes	$>$0 bytes

Timestamp	Logical Address Change Count	Digital Signature Size	Digital Signature
8 bytes	4 bytes	2 bytes	$>$=0 bytes

Example: “Foo”, `(34, 92)`, Non-Authoritative, 2 minutes old, 4 LA changes

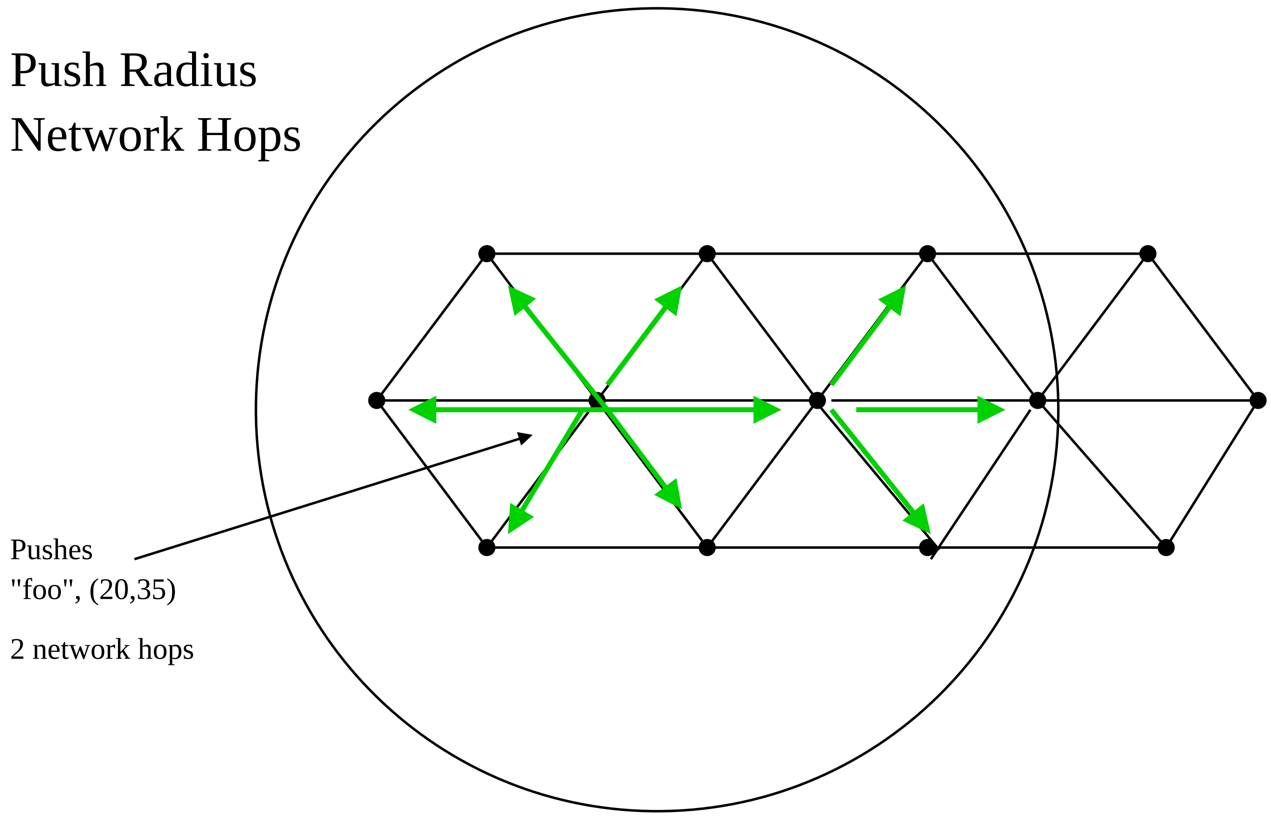
Naming Service Functions

- Create name bindings
- Store name bindings
- Exchange name bindings
  - Push a name binding
  - Pull a name binding
    - Logical address query (forward query)
    - Name query (reverse query)
- Invalidate name bindings

Application can invoke operations in any order

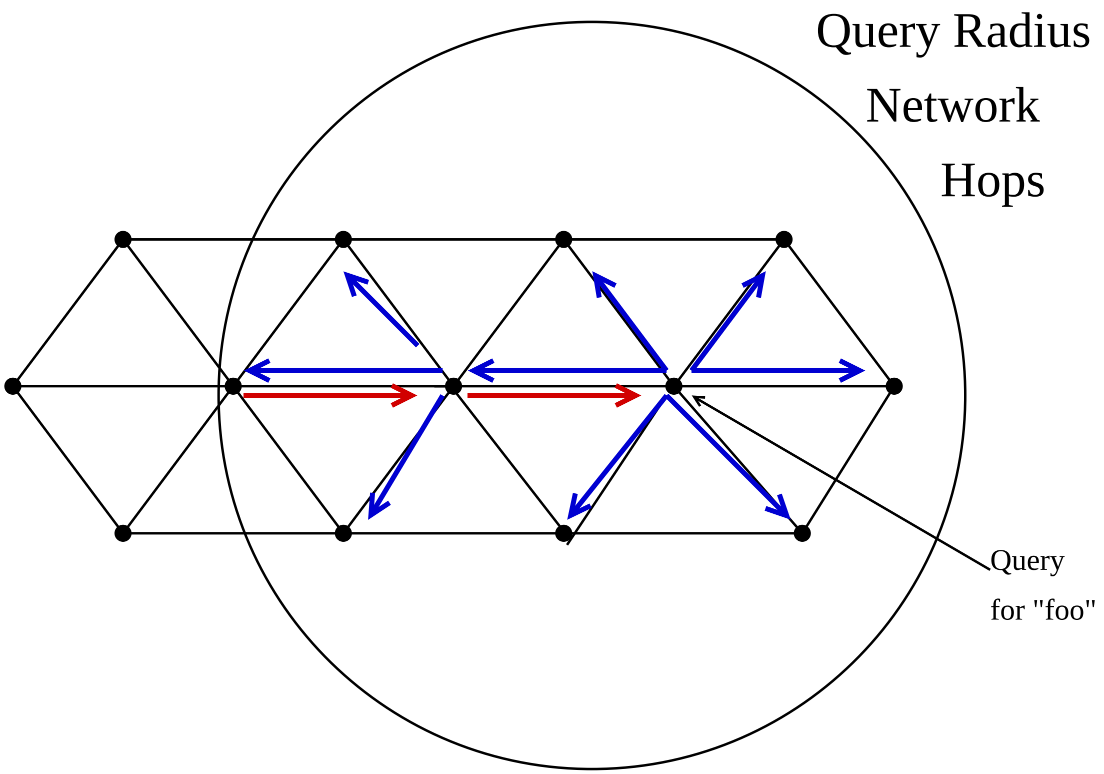
Pushing Name Bindings Operation

- The push name bindings operation disseminates name bindings when they are created (broadcast)
- Peers store name bindings in tables
- Traffic limited by radius (locality)



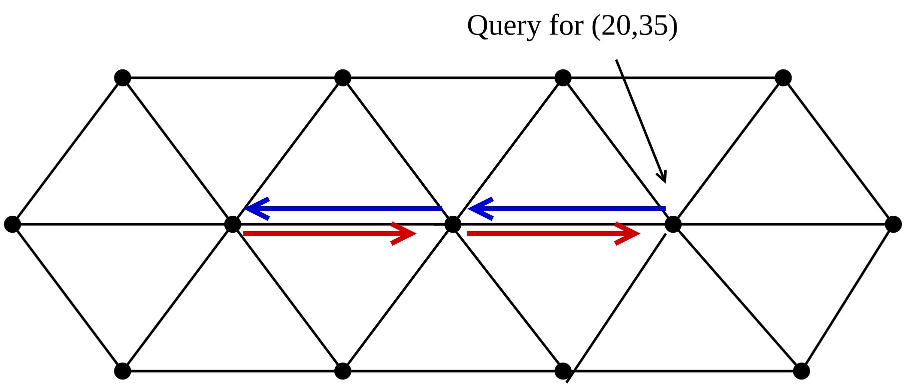
Pulling a Name Binding (Forward Query)

- A query initiated by a peer that wishes to learn the logical addresses associated with a given name.
- Hard: where to send query? Uses broadcast.



Name Query Operation

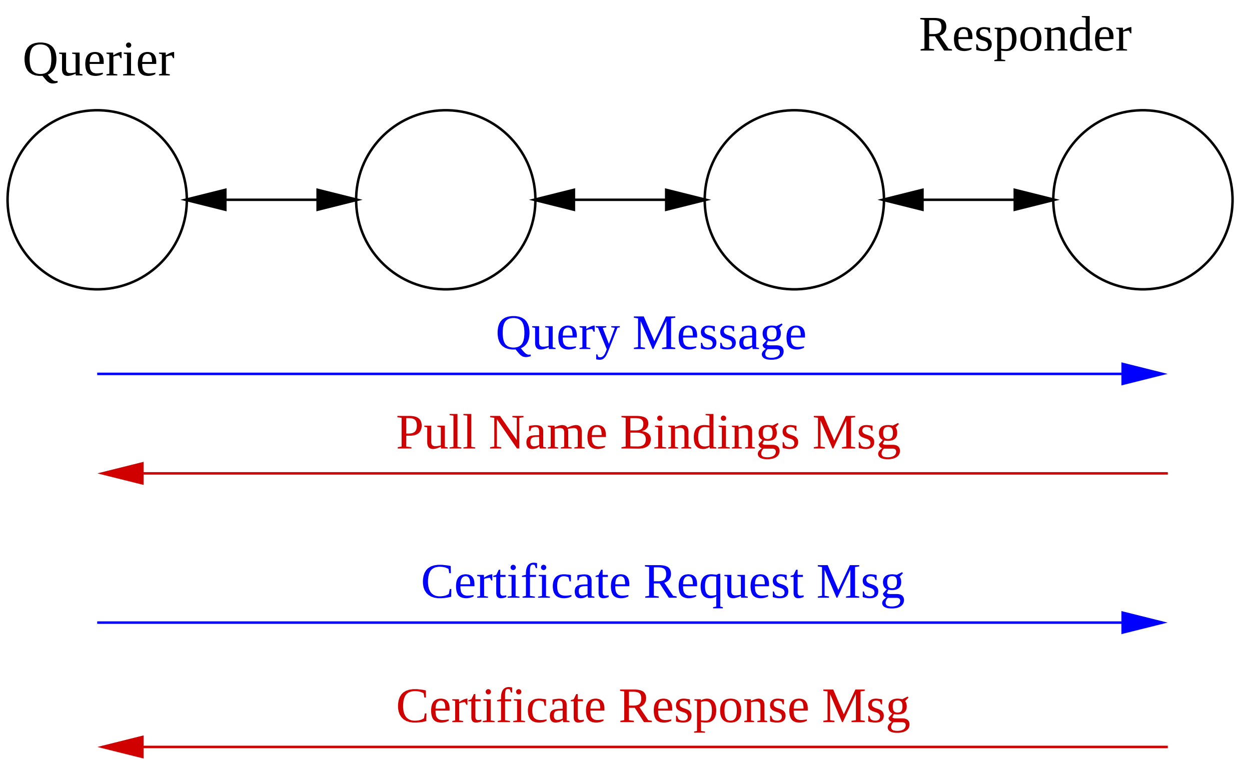
- A query initiated by a peer that wishes to learn the logical addresses associated with a given name
- Name query contains a logical address used in query routing
- No broadcast



Adding Trust to the Naming Service

- In the absence of a trusted server, why/how should names be trusted?
- Ensures integrity and authenticity of a name binding
- Exchang trust information with peers to establish trust of name bindings
- Verifies trust “on-the-fly”
- Builds trust chains (series of certificates that terminates at trust anchor)
- Compute digital signature for each binding
- Verify digital signature for each binding

Query Operation with Trust



Naming Service Implementation in the Hypercast System

- Overlay Sockets
- Unicast and multicast operations
- Naming service implemented as a network service inside HyperCast overlay socket
- Solutions to all previously stated issues are implemented
- Names are bound to logical addresses **not** sockets
- Uses extensible network services architecture with finite state machine paradigm
- Naming Service API

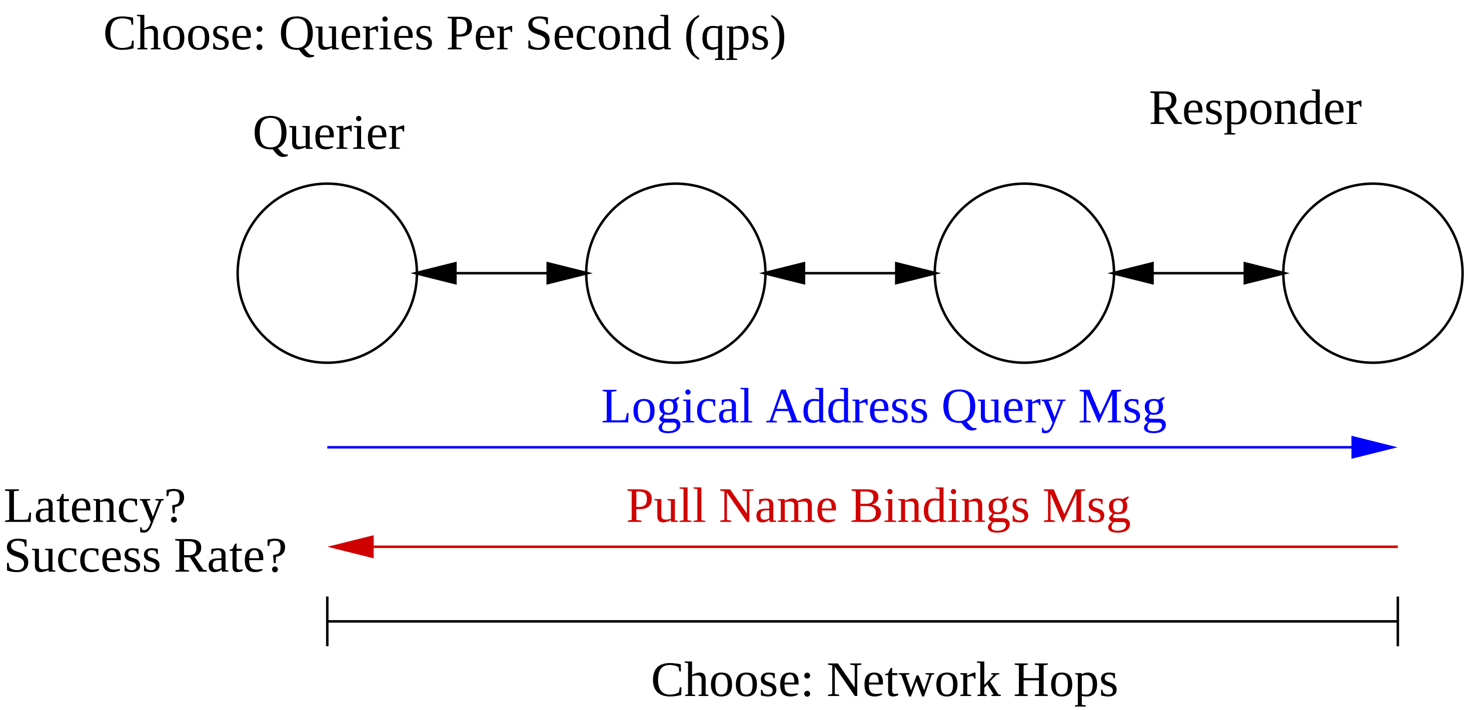
Example Program: Naming API

```
HyperCastConfig config =
    HyperCastConfig.createConfig ("hypercast.xml");
I_OverlaySocket socket =
    config.createOverlaySocket (null);
socket.joinOverlay();
socket.setName ("foo");
I_LogicalAddress [] logicalAddresses =
    socket.getLogicalAddressByName ("bar");
for (int i = 0; i < logicalAddresses.length; ++i)
      System.out.println ("LA for bar:  " +
                                logicalAddresses[i]);
```

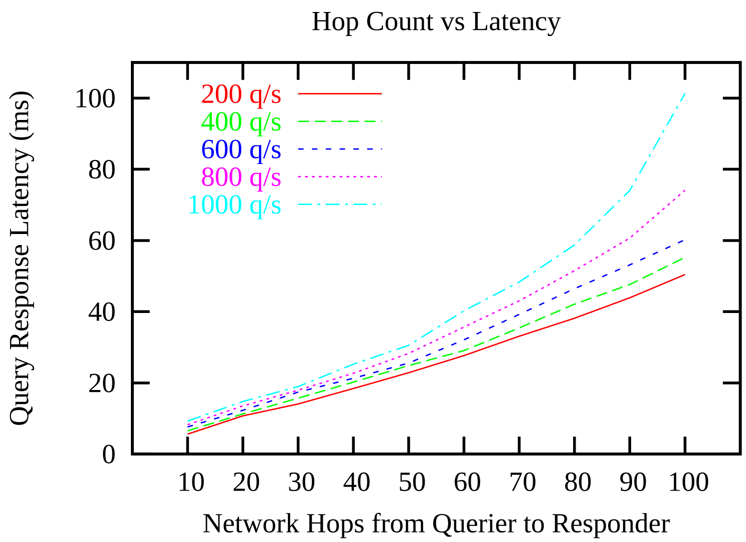
Experiments

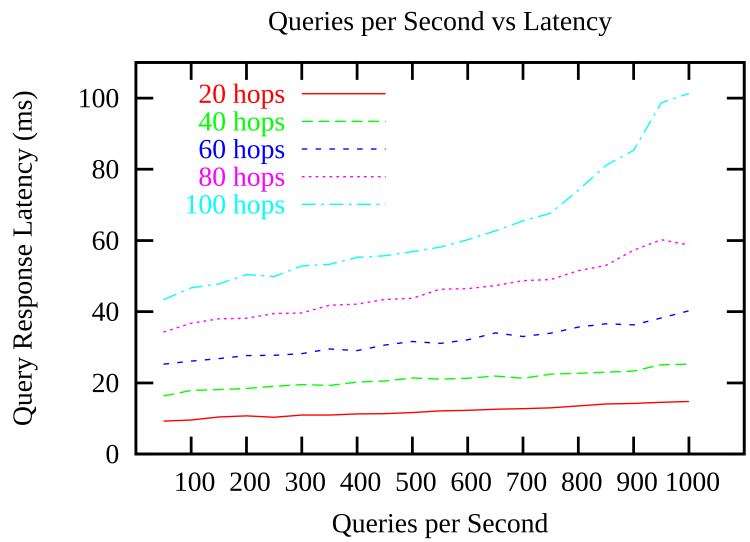
- Test Bed
  - Cluster of 20 Sun Microsystems Sunfires running Linux
  - Dual 2.8 GHz Xeon processors
  - 512 MB of physical memory
  - 1 Gbps ethernet interface
  - Connected by a single 1 Gbps ethernet switch
  - UDP datagrams are used for message transport in all experiments
- Experiment Configurations
  - “Linear” experiments
  - 40 row $\times$ 40 column “grid” experiments

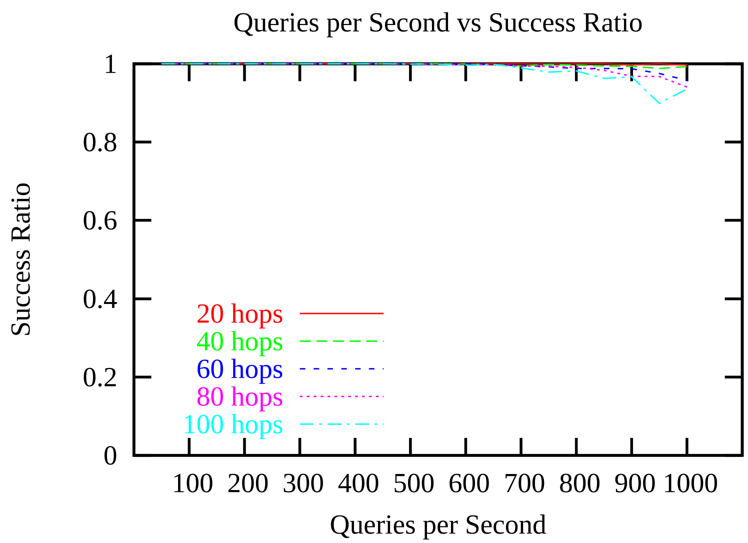
Linear Experimental Setup



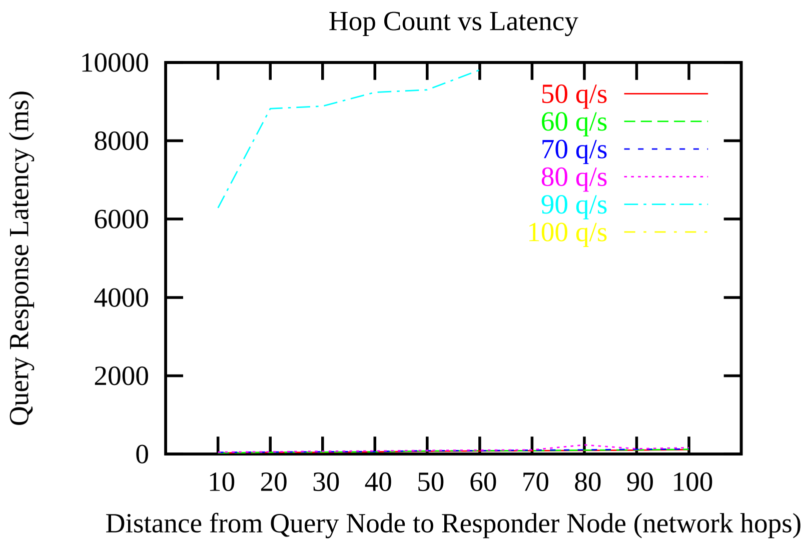
Linear Network Experiments

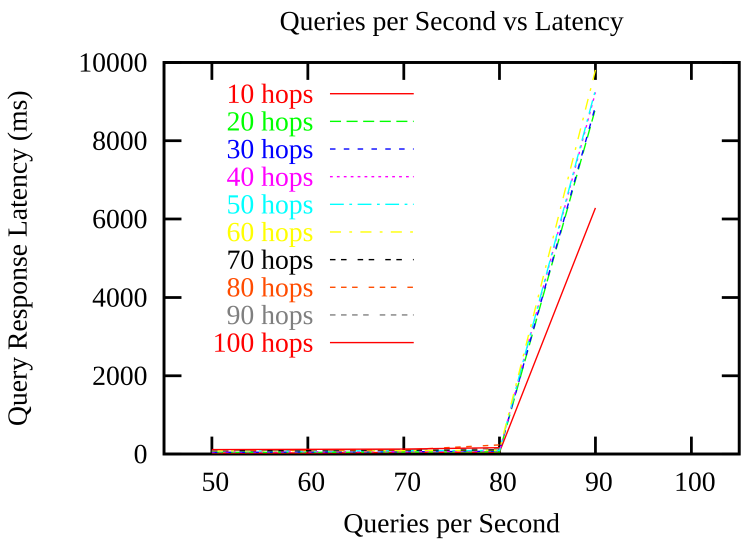


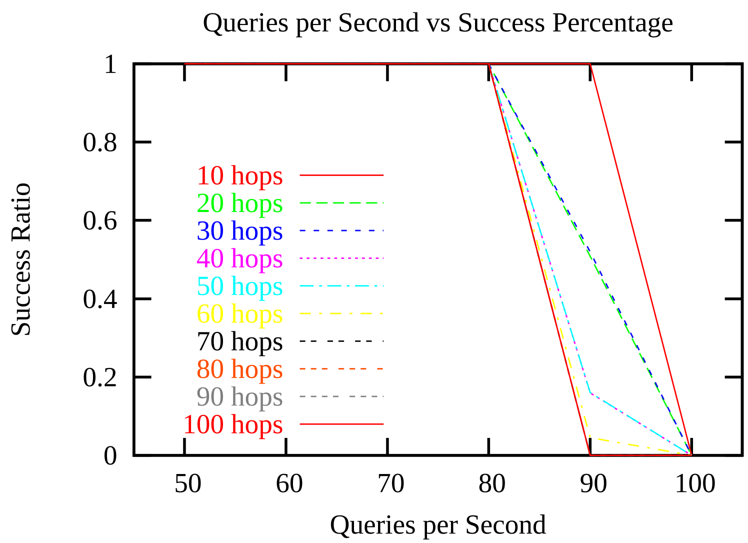




Linear Network Experiments: Trust

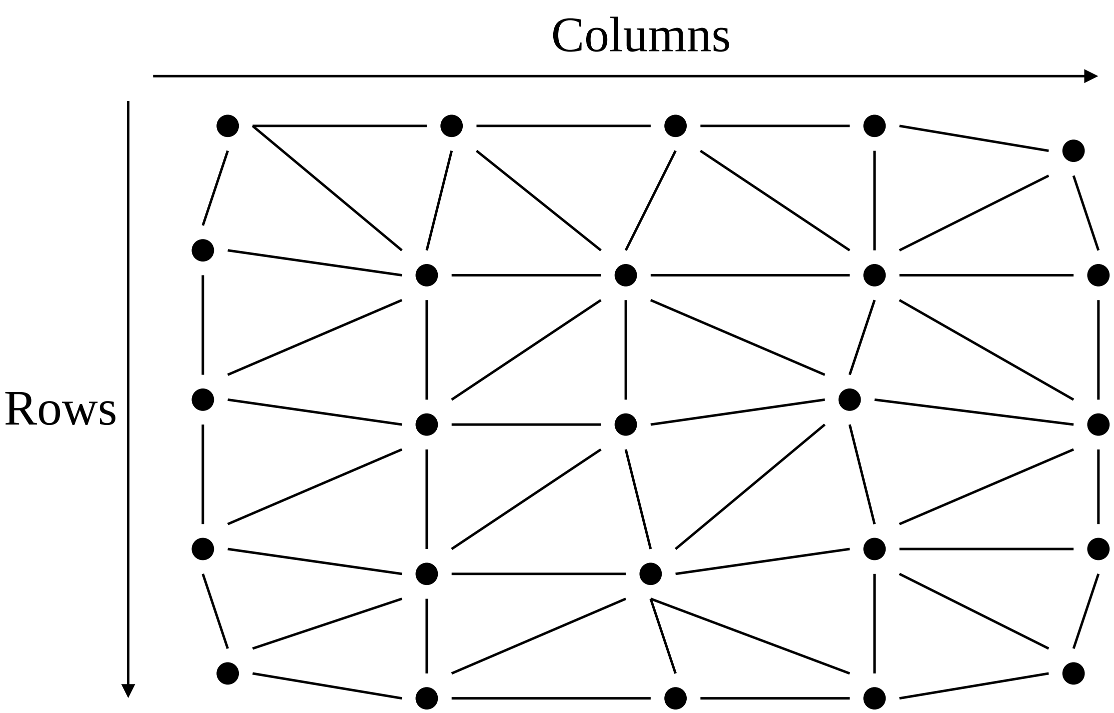






Grid Experimental Setup

- Regular grid
- Trade-off of Push vs. Pull
- All sockets query; all sockets respond

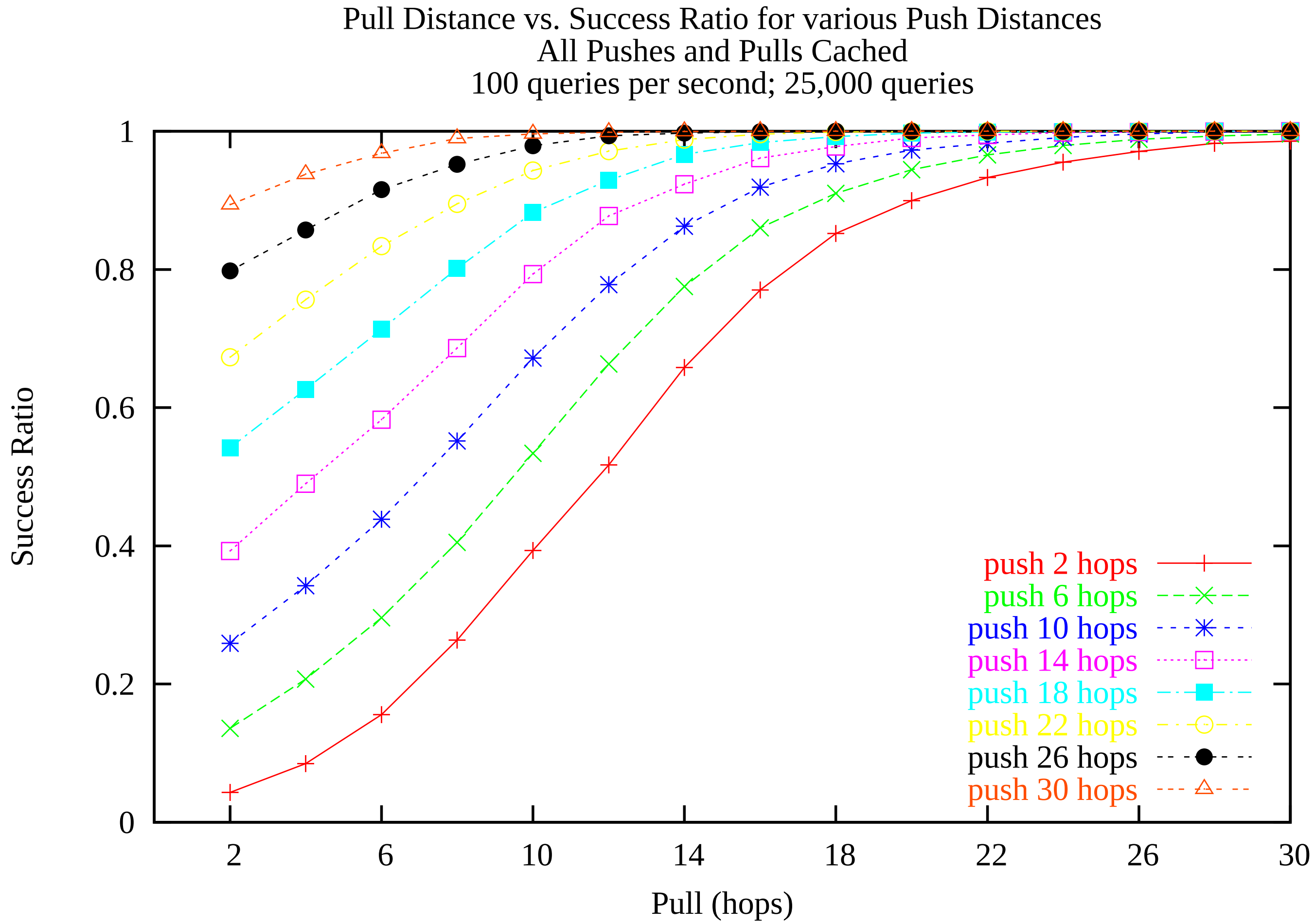


Choose:
Push Radius
Pull Radius

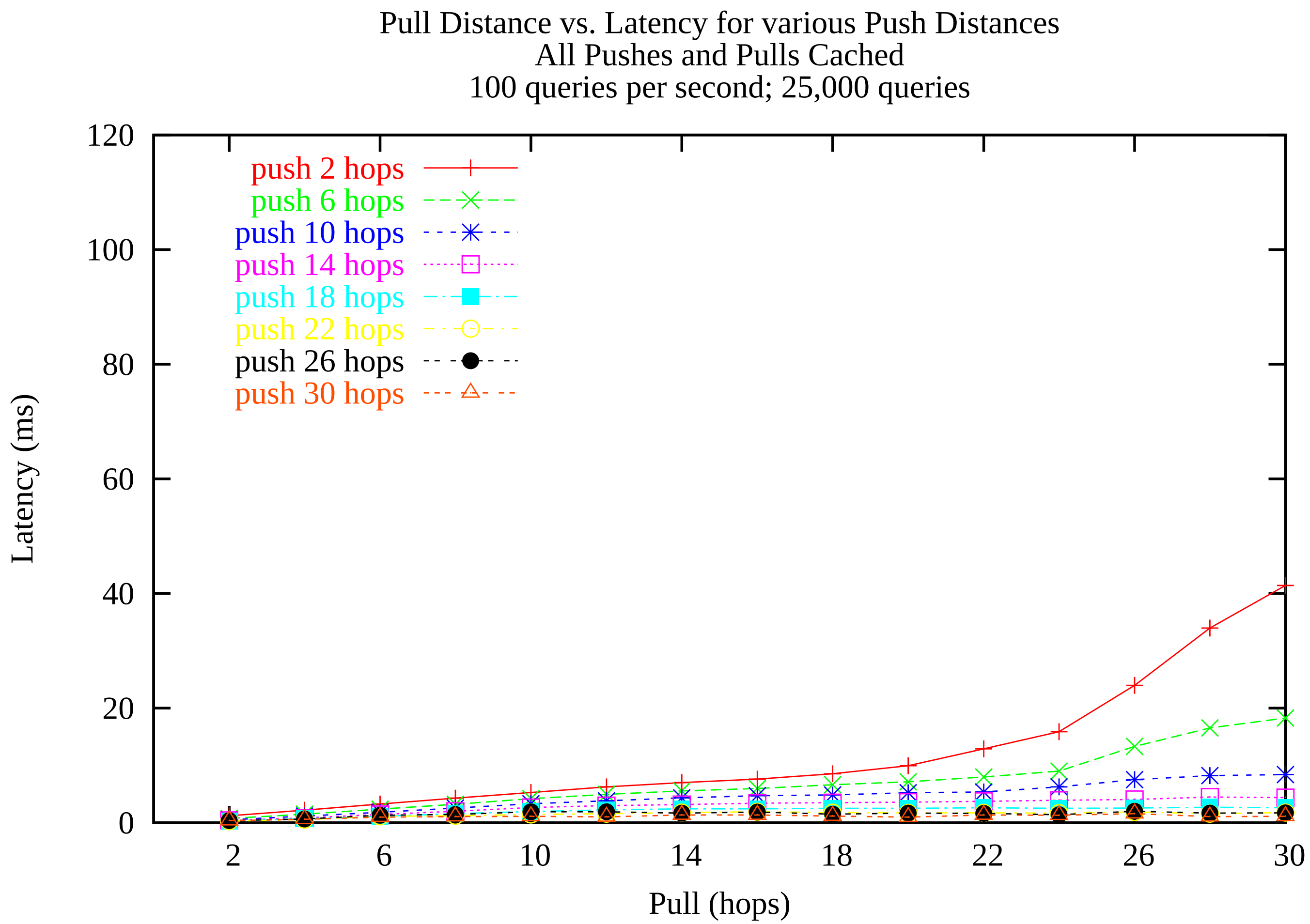
Network: 40 X 40
qps fixed

Latency?
Success Ratio?

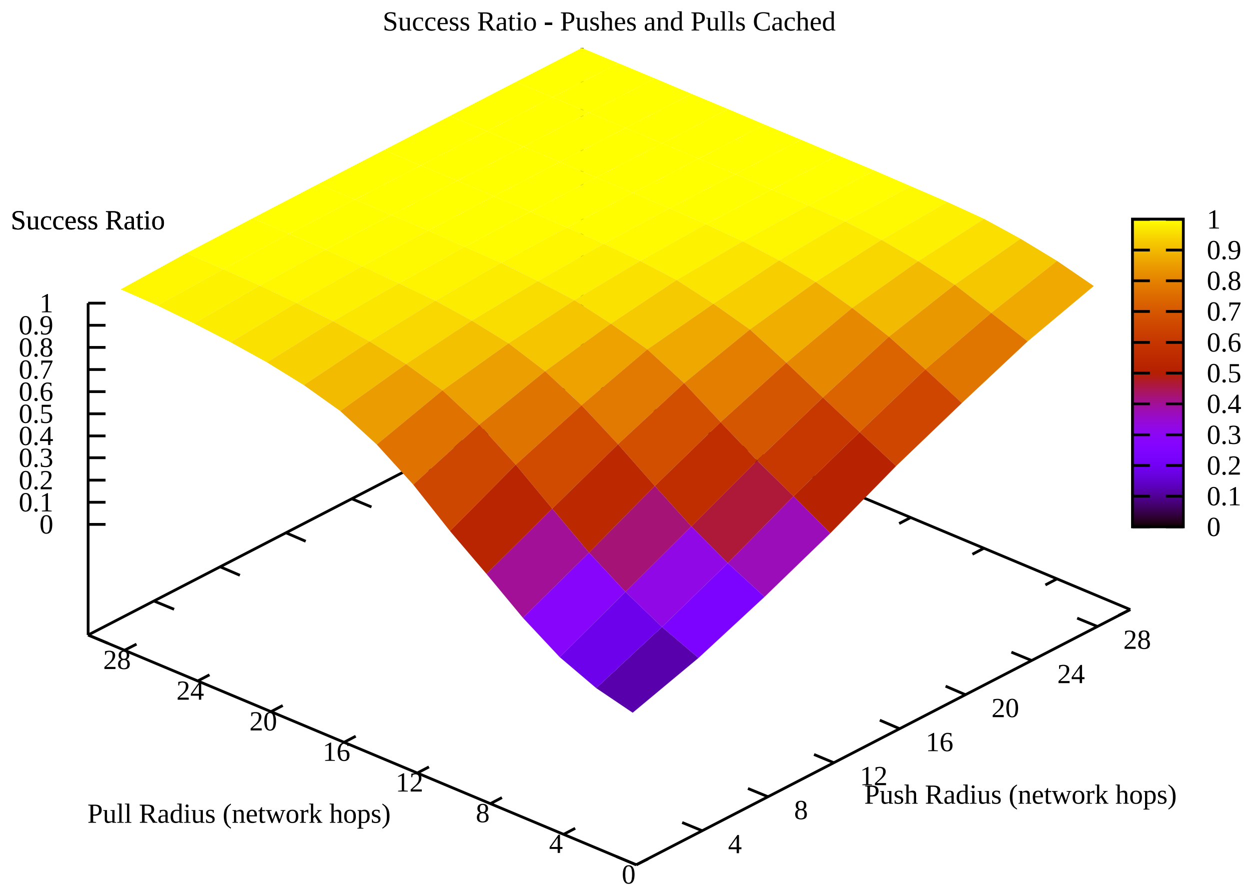
Grid Experiments: Success Ratio



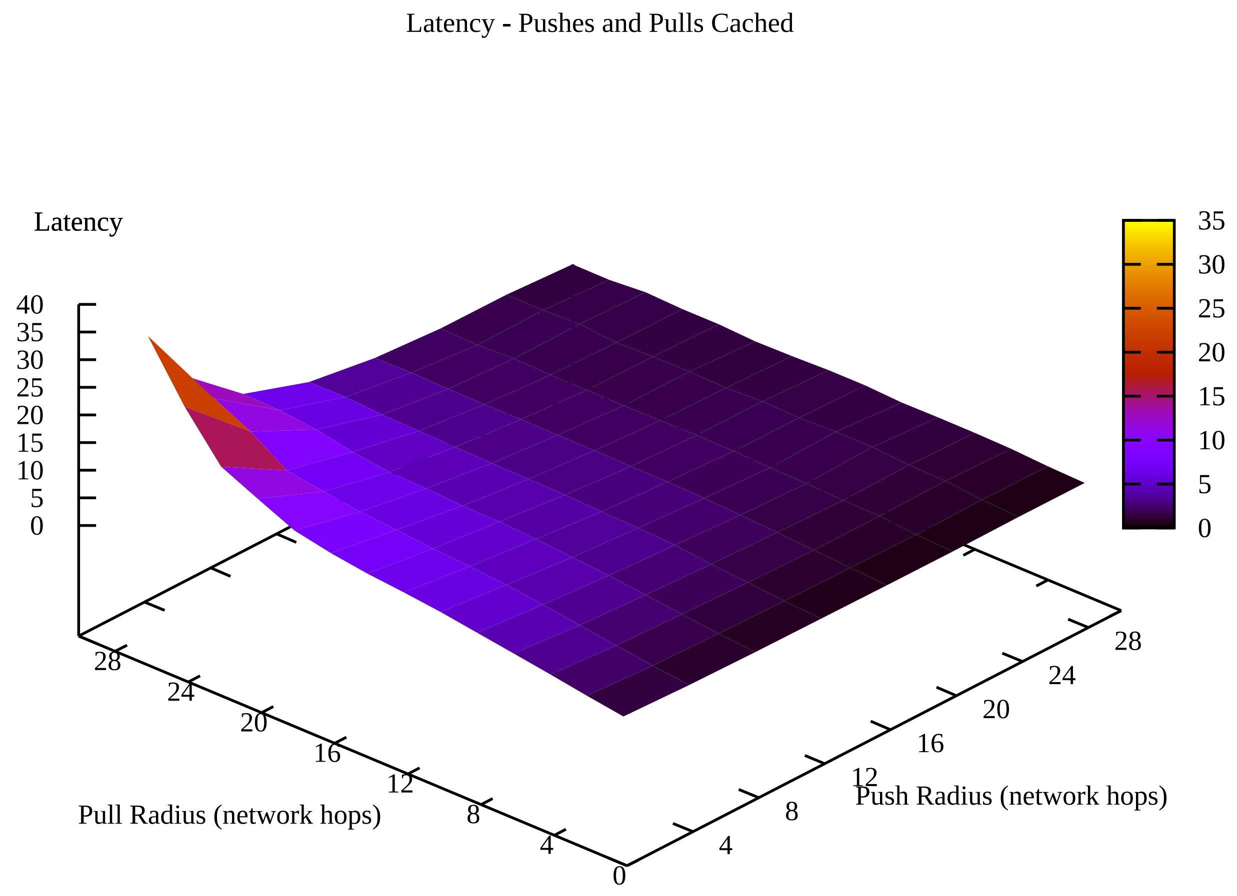
Grid Experiments: Latency



Grid Experiments: Success Ratio

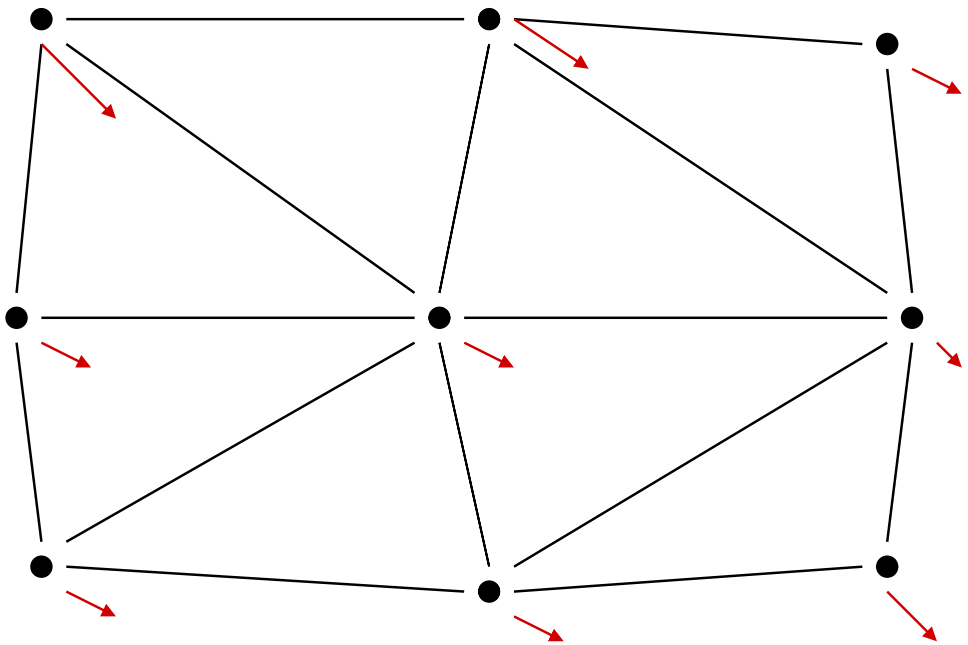


Grid Experiments: Latency



Grid Experimental Setup: Mobility

- Simulates mobile senario
- Structure of network does not change
- Not measuring overlay protocol's ability to reconfigure



Choose:

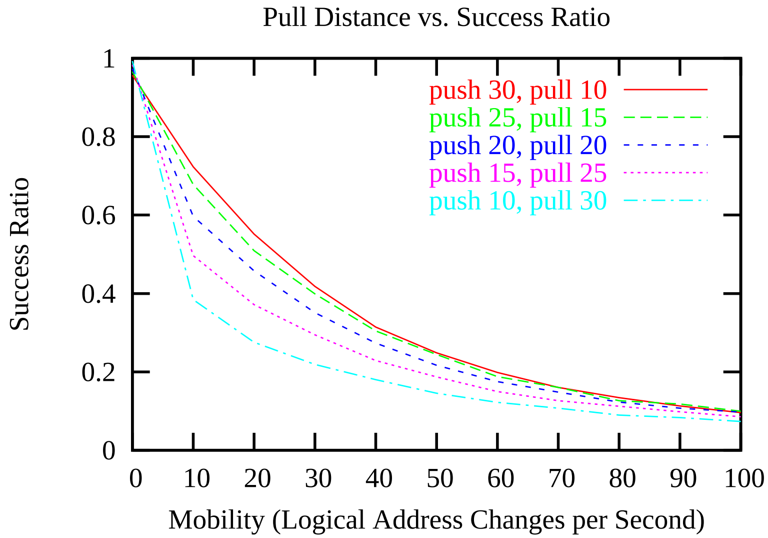
Mobility

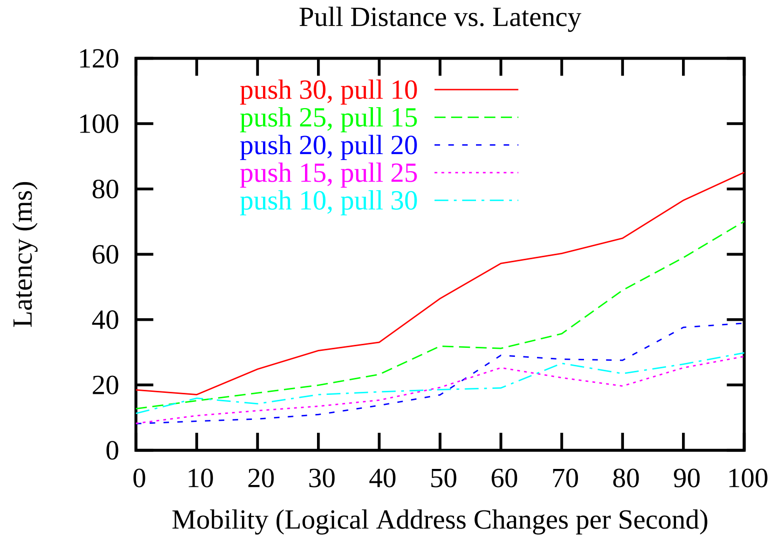
Push + Pull = Diameter

Latency?

Success Ratio?

Grid Experiments: Mobility





Related Work

- Internet
  - MAC/IP - ARP: find MAC of IP
  - IP/Domain Name - DNS: find IP of name
- Overlay
  - Distributed Hash Table (DHTs): use overlay for lookup - built a better DNS
  - Content Addressable Networks (CAN) - names used for routing
  - Intentional naming system - attribute-based query, separate naming overlay

Our work: if the world ran on an overlay network, what would DNS look like?

Conclusions

- The design and development of a naming service for dynamically changing application layer overlay networks without access to fixed infrastructure
- Implemented in HyperCast
- Solution for trust with no trusted third party
- Insights into trade-offs between push/pull, caching, and mobility
- Demonstrated viability
- Experiments performance evaluation
- Open questions
  - Scaling (limited by experimental resources)
  - Groups not fully explored (subgroups)