Network-Aware Operator Placement for Stream-Processing Systems

Written by

Peter Pietzuch, Jonathan Ledlie, Jeffrey Shneidman, Mema Roussopoulos, Matt Welsh, Margo Seltzer

Presented by

Ee Lee Ng

Large-Scale Stream-Processing

Many geographically distributed data **sources**

- e.g., sensors, network routers, RFID tag readers, …
- High volume of real-time stream data

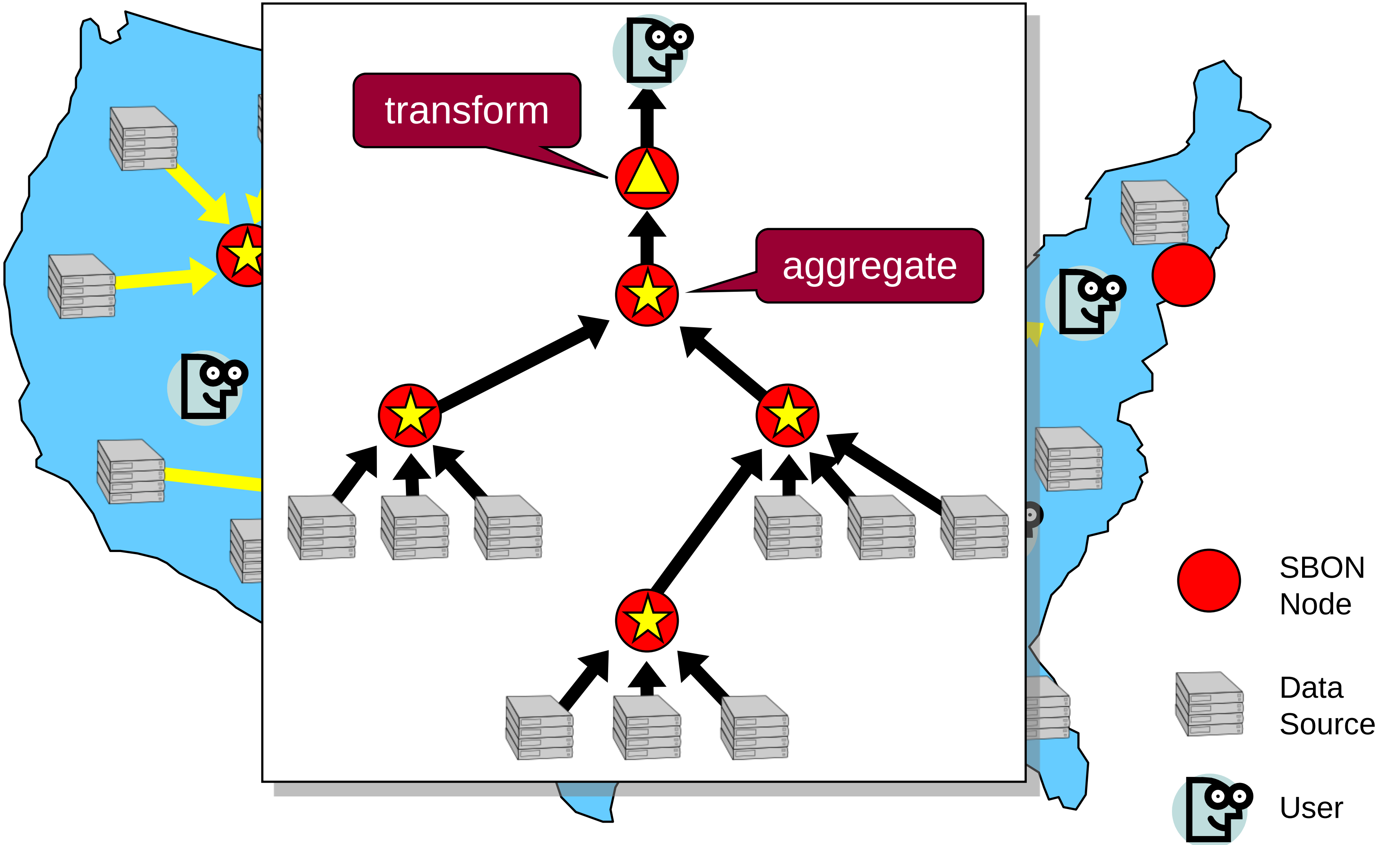
Many users, submitting individual stream **queries**

- Queries use the Internet for stream transport

Queries include **operators** for stream-processing

- e.g., join, filter, aggregate, XPath, image analysis, …
- Operators require nodes for execution
- In-network processing can often reduce data volume

Stream-Based Overlay Network



Operator Placement Problem

How do you map operators to overlay nodes?

Efficiency

- Node and network resources are limited and shared
- Operator placement must be **network-aware**
  - Consider link latency, bandwidth, congestion, jitter, …
  - Filter and aggregate data close to sources

Scalability

- Must scale to many sources, overlay nodes and queries
- No global view of the system

Adaptability

- Resource conditions change over time

Contributions

Stream-Based Overlay Network (SBON)

- Generic layer between network and stream-processing apps
- Shields applications from network complexity

Operator placement using a metric **cost space**

- Decentralized framework for minimizing network impact
- **Relaxation placement** algorithm for operator placement
- Adaptive to change in network conditions

Deployment of SBON and sample applications (Borealis extension) on PlanetLab

Operator Placement Goals

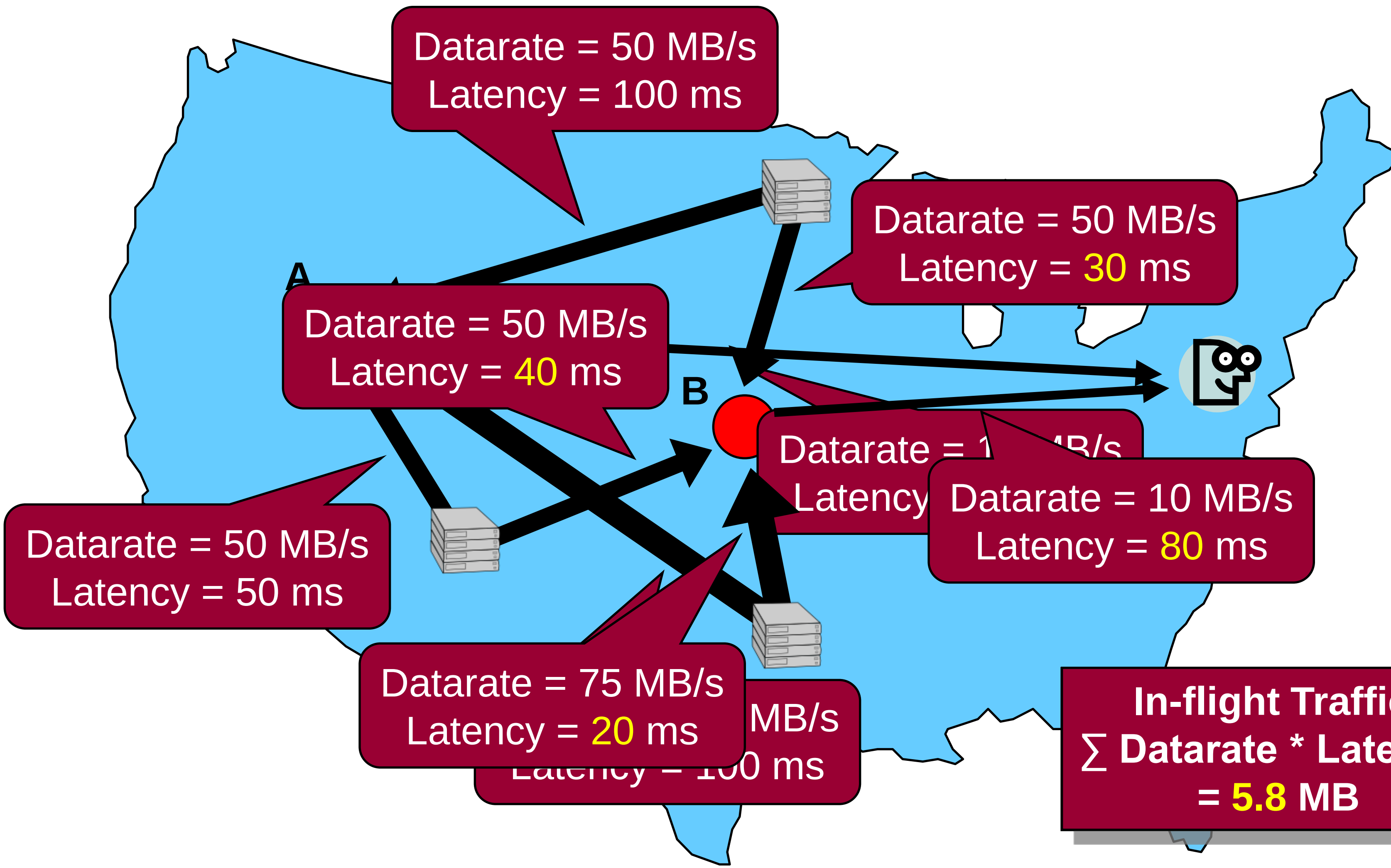
Two conflicting optimization goals:

1. Global system performance with concurrent queries
    - Minimize network usage
    - Bala

☛ **Minimize global network usage**

2. Individual query performance
    - Minimize data delay
    - Maximize stream throughput

Network Usage



In-flight Traffic:
∑ Datarate * Latency
= 5.8 MB

Network-Aware Operator Placement

Perform operator placement in a decentralized fashion

- Need information about data rate and latency

But measuring network metrics is expensive

- All pairs latency measurements are $O(n^2)$
- Network latencies change over time
- No global knowledge of measurements

Idea: Approximate optimal query with a cost space [NetDB'05]

1. Build metric cost space to encode current network latencies
2. Find query with minimal network usage in cost space
3. Map query back to physical Internet nodes and instantiate

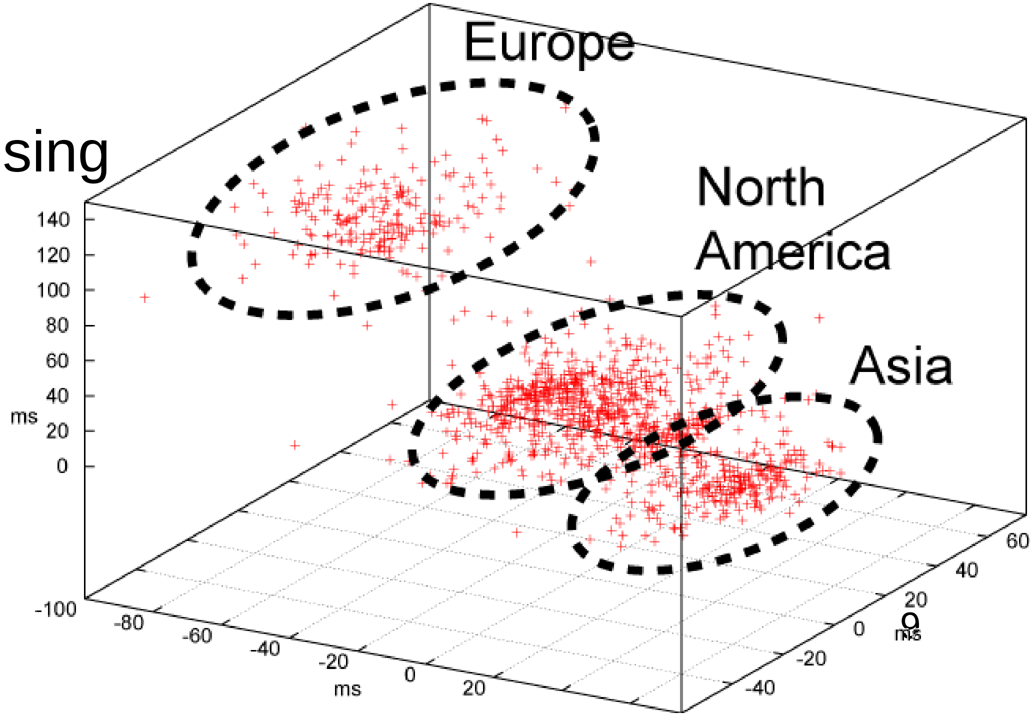
Cost Space

Embed latency measurements into a metric space

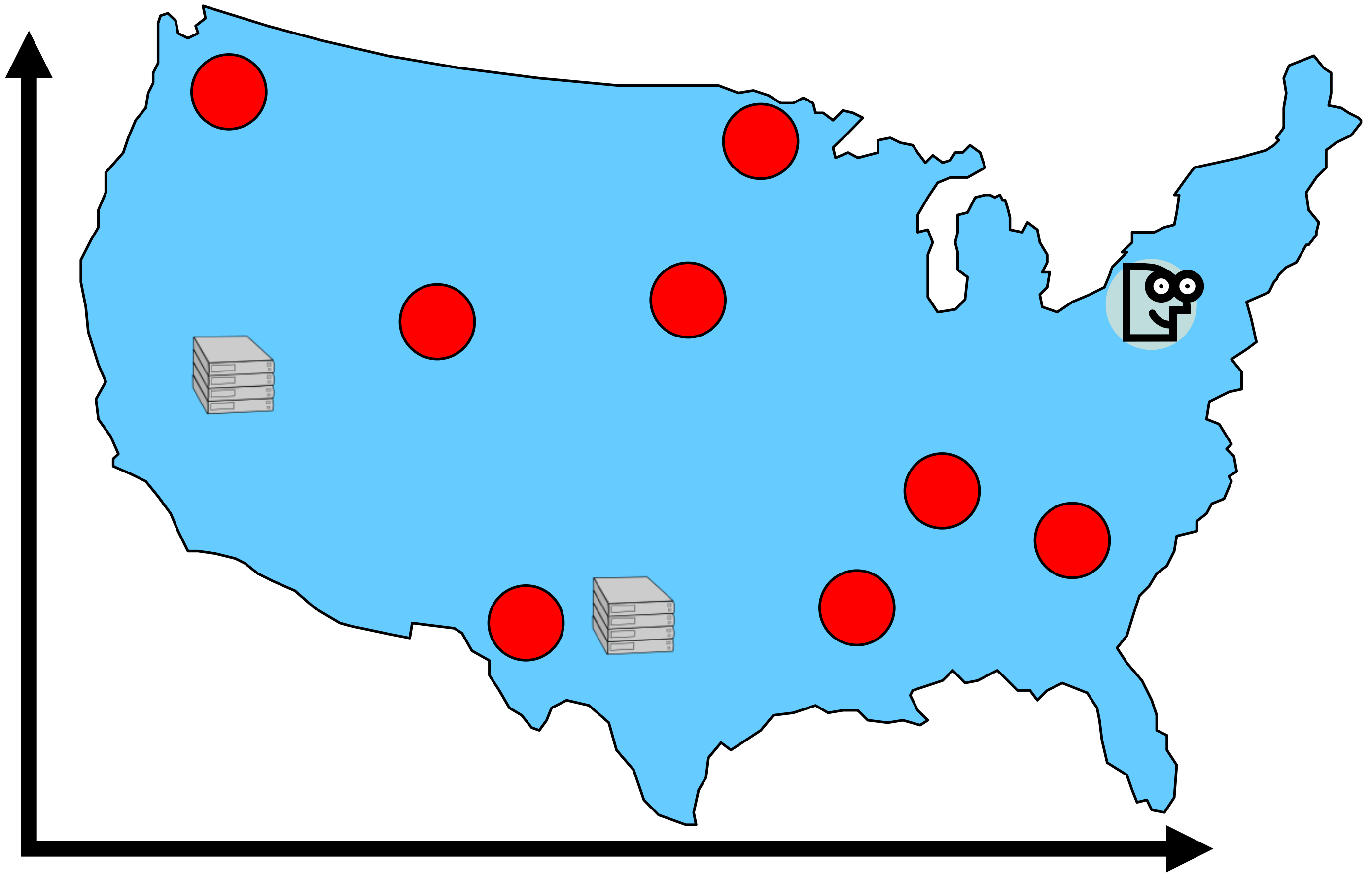
- Assign each SBON node a coordinate in a cost space
- Euclidean distance ≈ network latency
- **Vivaldi** algorithm [MIT]
  - Repeated measurements to refine local coordinate

Advantages

- Mathematical model for using geometric algorithms
- Optimization decisions without global knowledge
- Adaptive to change

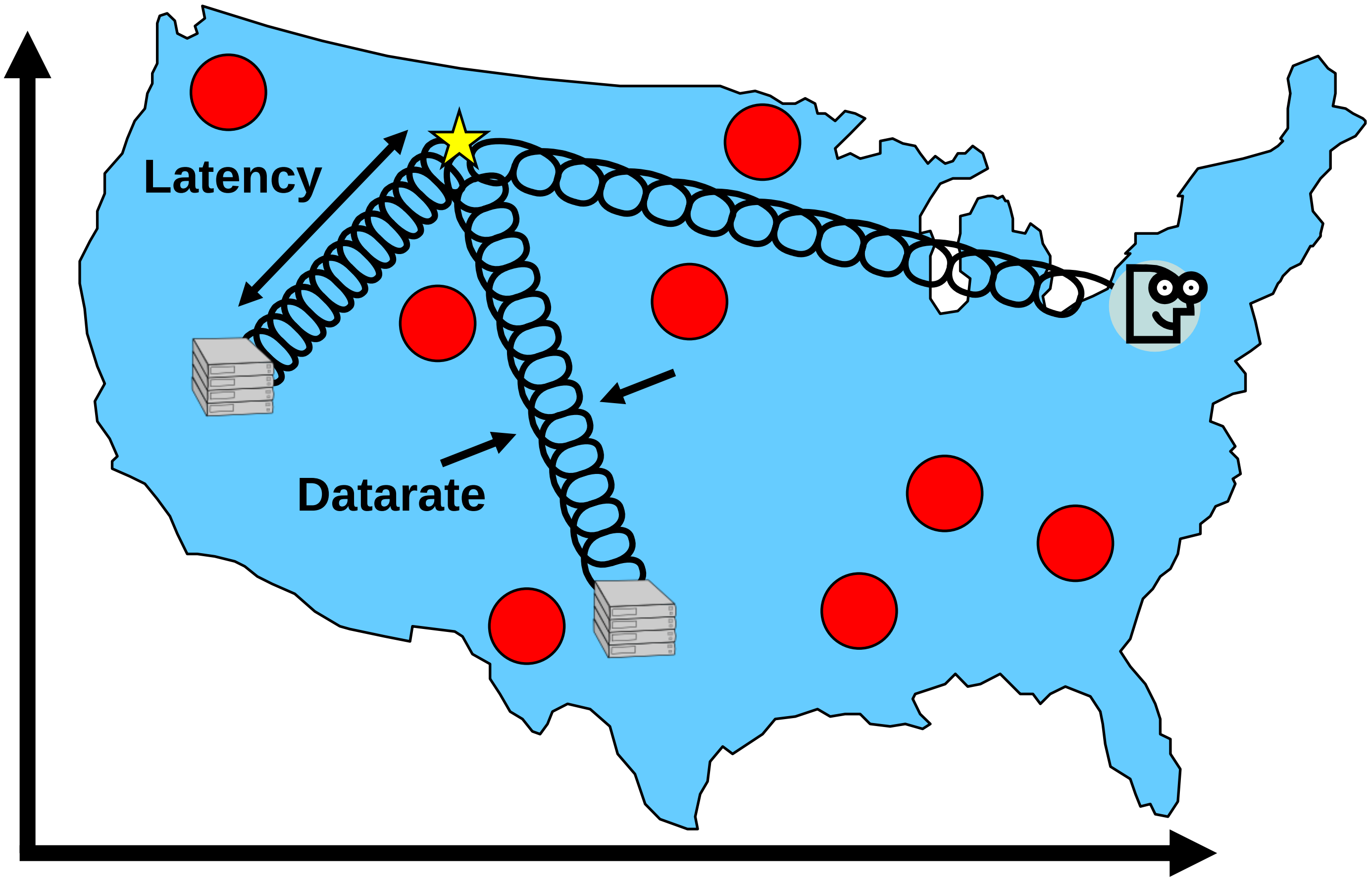


Relaxation Placement



Find a location for an operator that reduces network usage

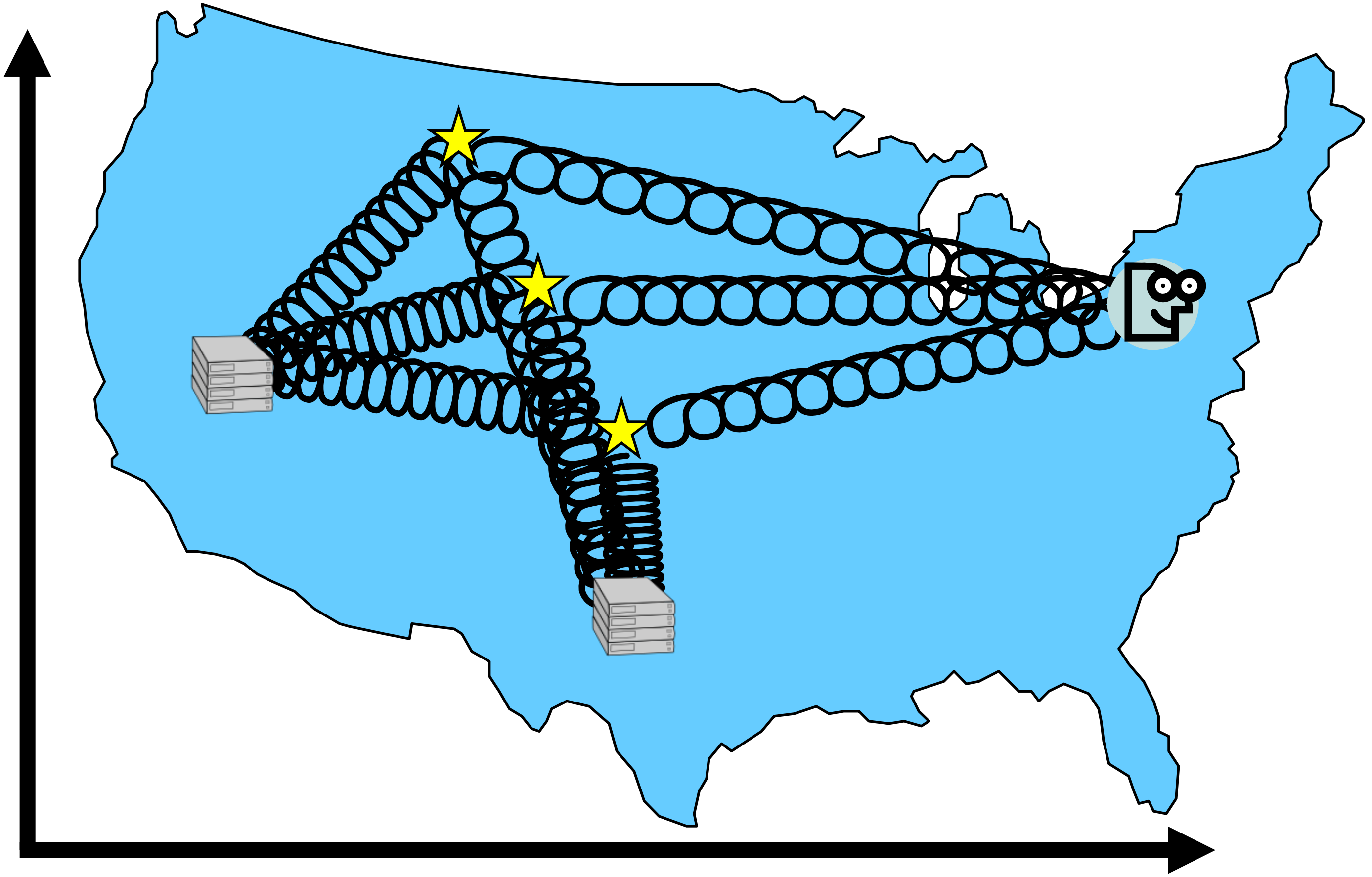
Relaxation Placement



Use spring relaxation technique to find best location

- Spring extension ≈ latency
- Spring constant ≈ data rate

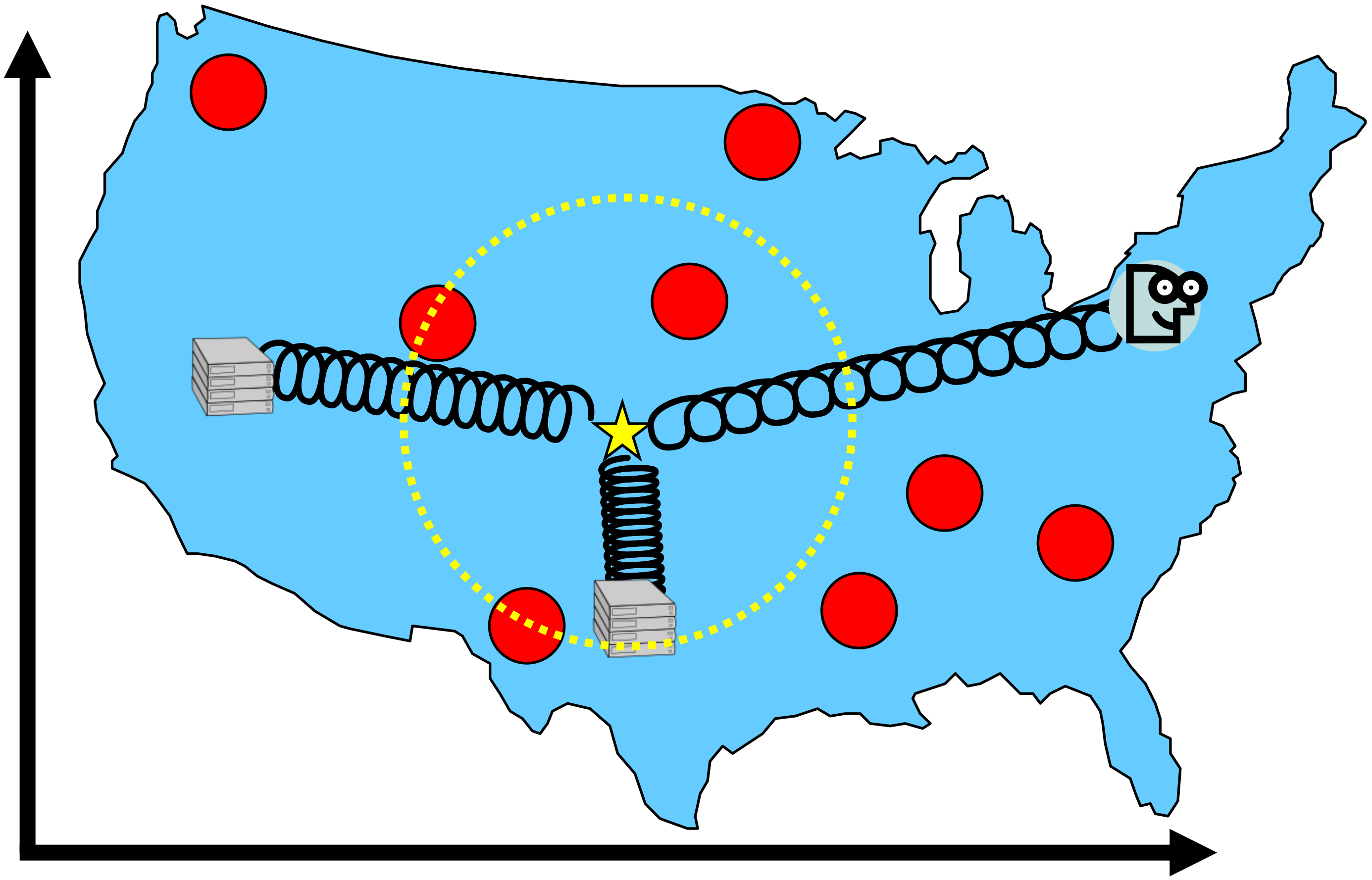
Relaxation Placement



Use spring relaxation technique to find best location

- Springs “relax” to low energy state, minimizing network usage
- Dynamically adapts to changes in cost space

Relaxation Placement



Uses nearest k-neighbor search for mapping of coordinates

- Interesting problem in decentralized context
  - Geometric routing [HUJI], DHT range queries [UCB], …

Relaxation Placement

Any SBON node can perform the placement for a new query

- Local computation without global state
  - Inputs are coordinates of nodes and data rates in query
- Supports placement of arbitrary complex queries
  - Model multiple queries as networks of spring

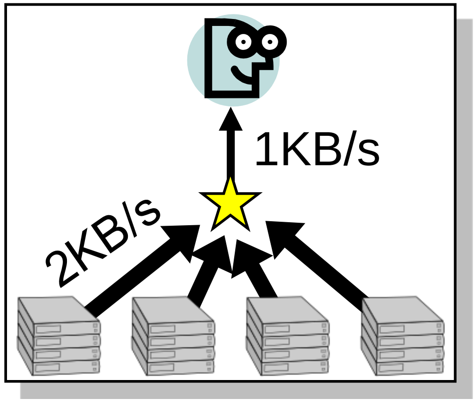
Each node is then responsible for the operators it is hosting

- Periodically re-execute Relaxation placement
- Dynamically migrate operator to reflect new placement
  - Adapts to changes in latency and data rate

Simulation Setup

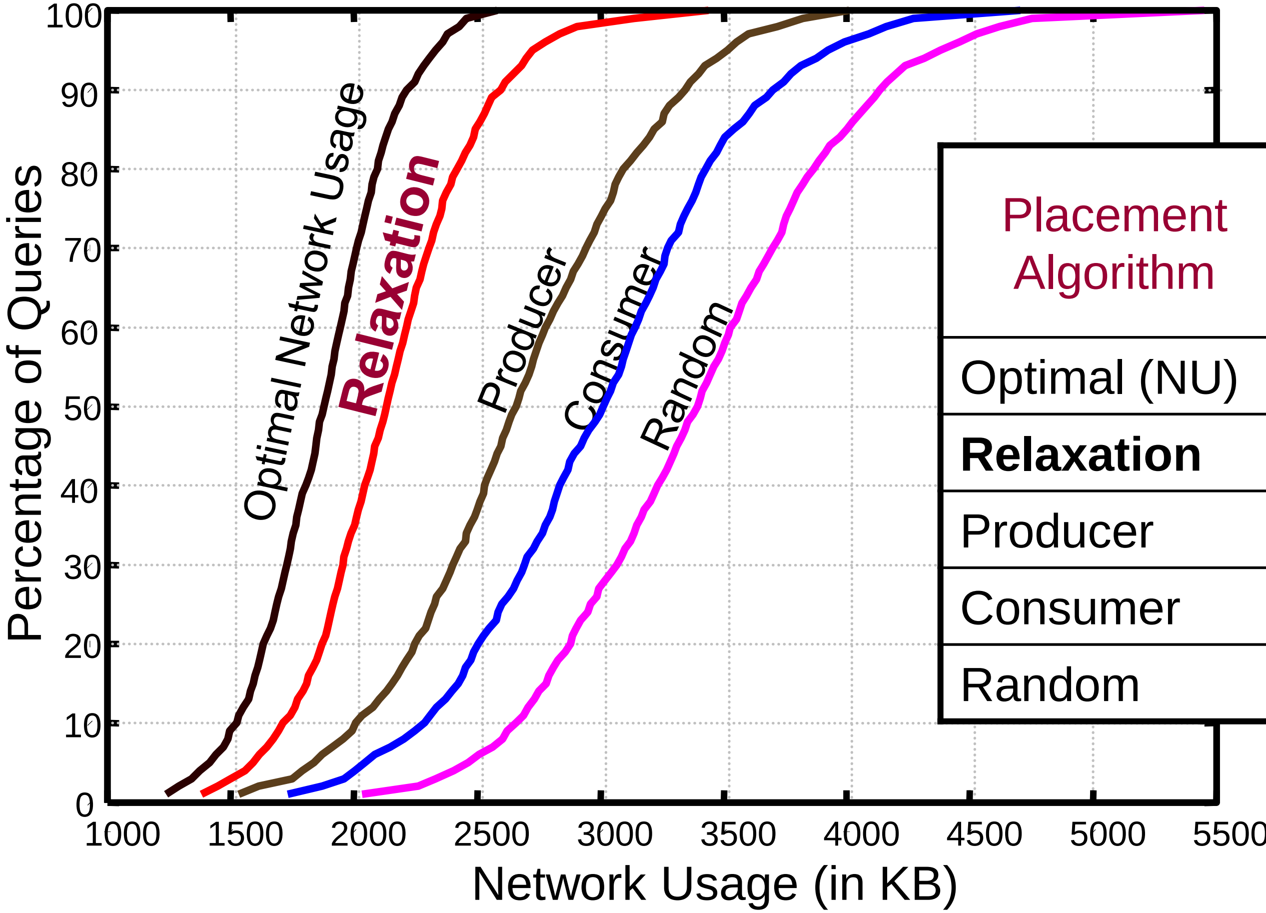
Discrete event simulator to evaluate placement algorithms

- GATech transit-stub topology with 1550 nodes
  - 10 transit domains and 150 stub domains
  - Realistic Internet routing tables
- 1000 queries with 5 random endpoints
- Comparison of **Relaxation** placement to 4 other algorithms



Optimal	Exhaustive search
Producer	Common strategy
Consumer	Central data warehouse
Random	Worst case

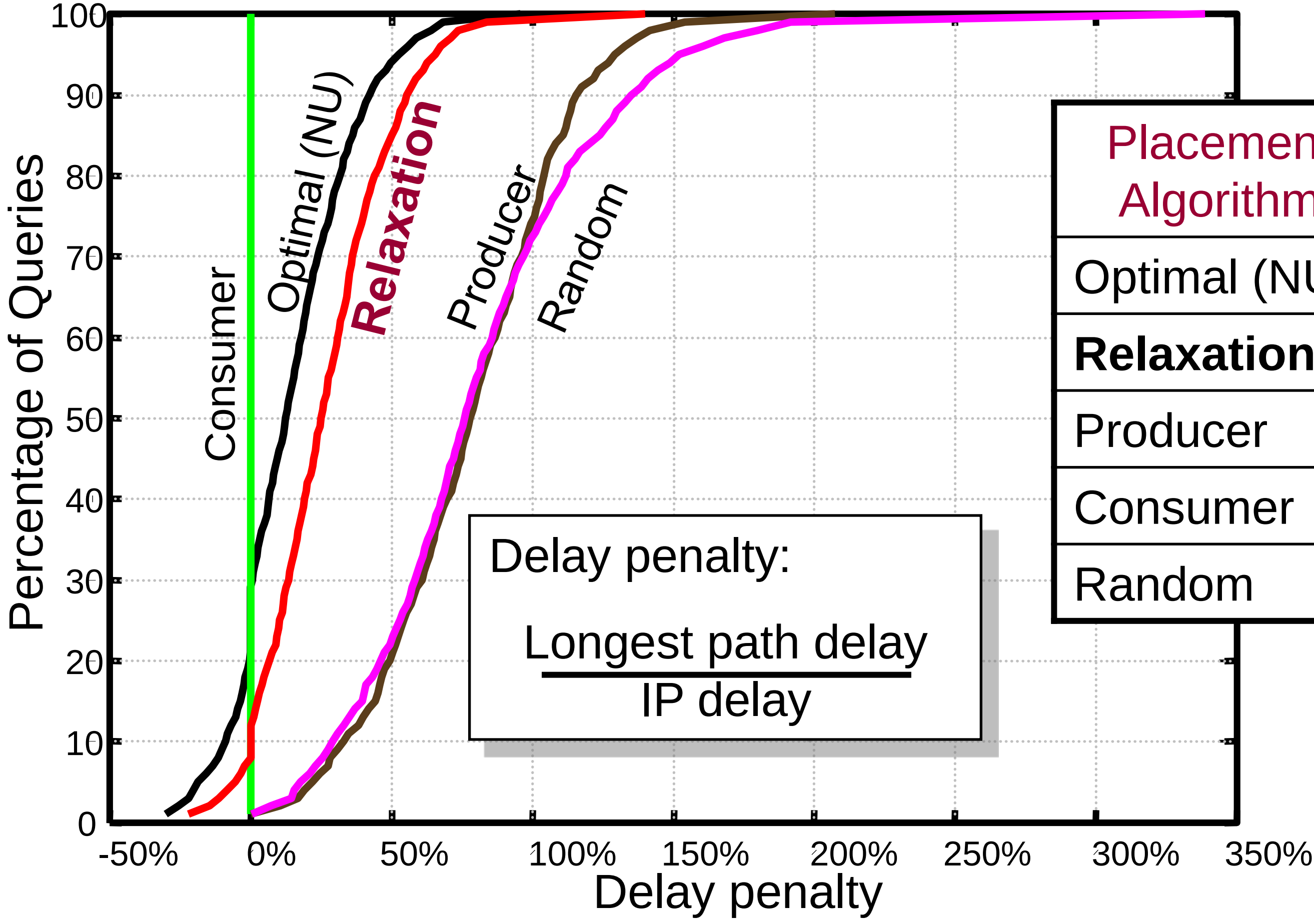
Global Network Usage



Placement Algorithm	Avg. Network Usage Penalty
Optimal (NU)	0%
Relaxation	**15%**
Producer	43%
Consumer	60%
Random	81%

- Relaxation placement performs close to Optimal

Application Delay Penalty

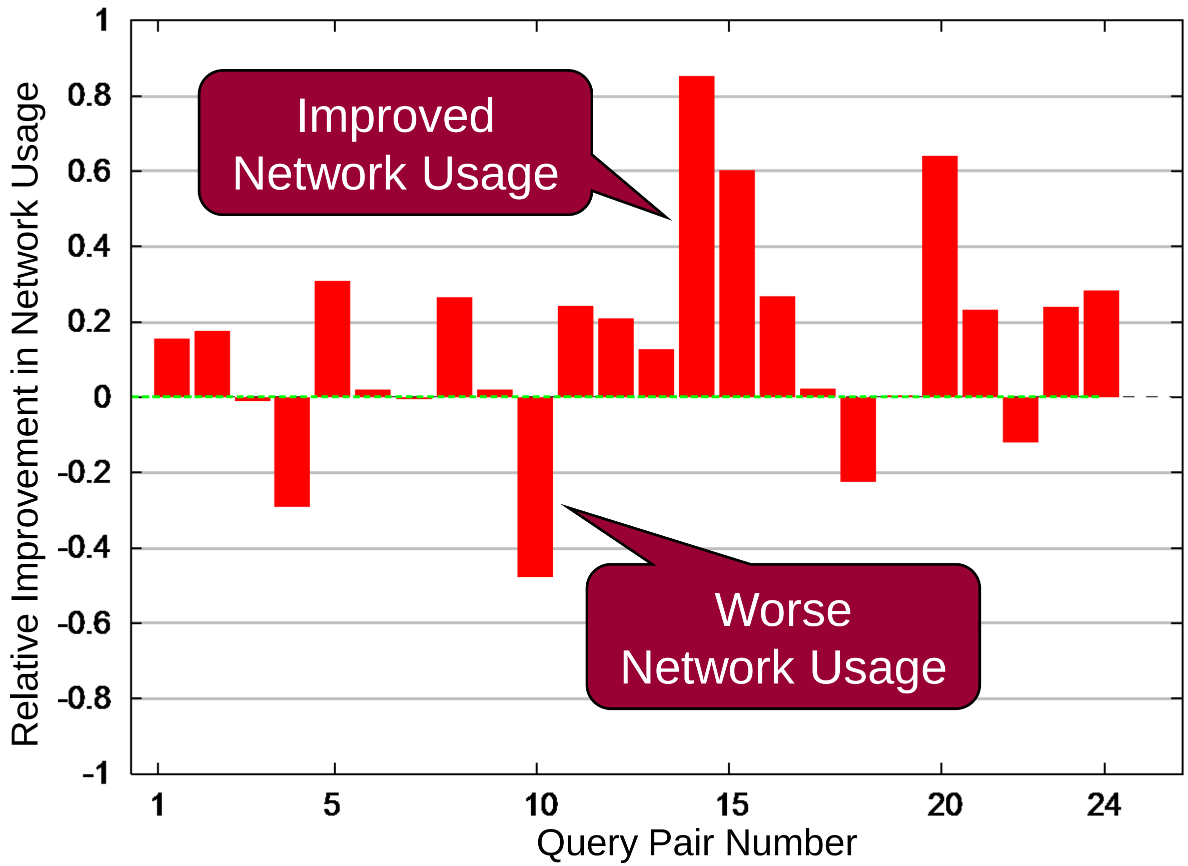


Placement Algorithm	Avg. Delay Penalty
Optimal (NU)	13%
Relaxation	**24%**
Producer	75%
Consumer	0%
Random	76%

- Consumer has smallest delay penalty
- Relaxation has low delay penalty for an overlay network

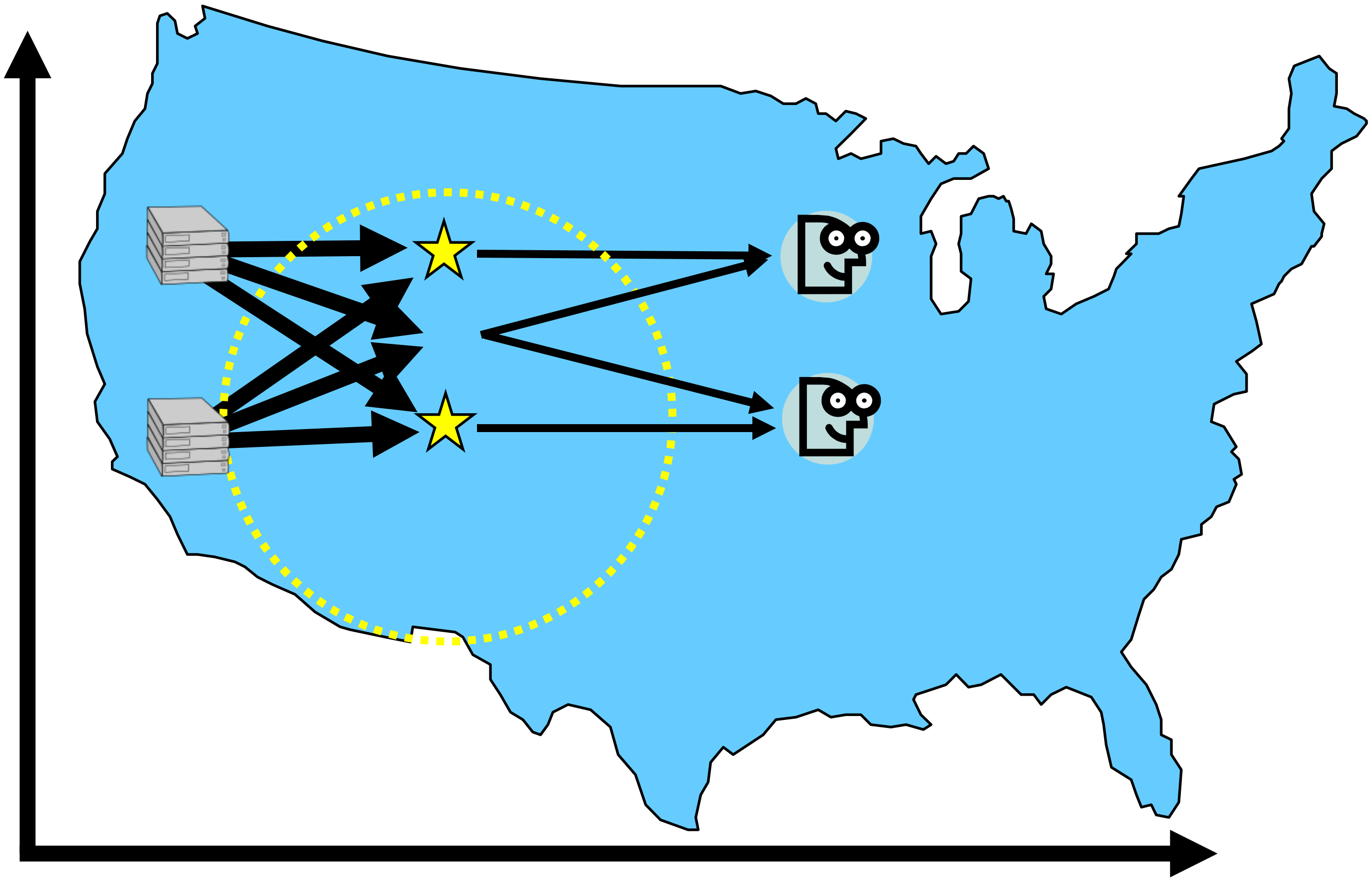
Operator Migration on PlanetLab

- 48 concurrent queries on 130 nodes
- ½ of the queries could migrate
- Same initial place-ment for migrating and non-migrating queries
- Change in network usage of migrating queries after 5 hours



- Migration decreased network usage for 75% of queries
  - 17% less network usage and 11% lower application delay

Operator Reuse



Share operators between overlapping sub-queries

- Use cost space to bound search effort for reuse

Related Work

Borealis [MIT, Brown, Brandeis] , Medusa [MIT], Gates [Ohio]

- Focus on high-availability and load management
- Wide-area operator placement specified by user

SAND [Brown] , PIER [UCB]

- Operator placement at edge (prod/cons) or in-network
- Exploit DHT routing paths for operator placement
  - Can lead to poor placement efficiency [IPTPS'05]

IrisNet [Intel]

- Hierarchical placement following DNS structure

Summary

Large-scale stream applications need new systems support

- SBON: Infrastructure for stream-processing applications
- Provides network-aware stream query optimization

Cost space approach for query optimization

- Metric space for decentralised optimization decisions
- Express query optimization as geometric problem

Relaxation placement algorithm for operator placement

- Scalable placement decisions reducing network usage
- Continuous optimization as network conditions change

Thank You. Any Questions?