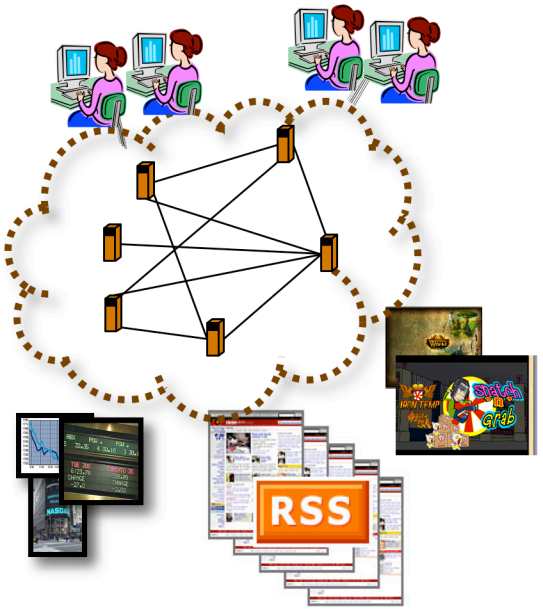
Extensible Optimization in Overlay Dissemination Trees

Olga Papaemmanouil,
Y. Ahmad, U. Cetintemel, J. Jannotti, Y.Yildirim
Brown University
Computer Science Department

Large-Scale Data Dissemination

- Growing set of dissemination-based apps
  - RSS feed distribution, content delivery networks, networked games, etc.
- …with disparate application-specific logic
  - different data/profile types and QoS targets
- …but similar "plumbing" needs
  - overlay network construction & management, data routing, membership management.



XPORT

(eXtensible Profile-driven Overlay Routing Trees)

- A generic "infrastructure" system for data dissemination
  - overlay network
  - profile-driven
- Extensibility is key!
  - performance goals
  - profile and data management

XPORT

(eXtensible Profile-driven Overlay Routing Trees)

- A generic "infrastructure" system for data dissemination
  - overlay network
  - profile-driven
- Extensibility is key!
  - performance goals
  - profile and data management

XPORT

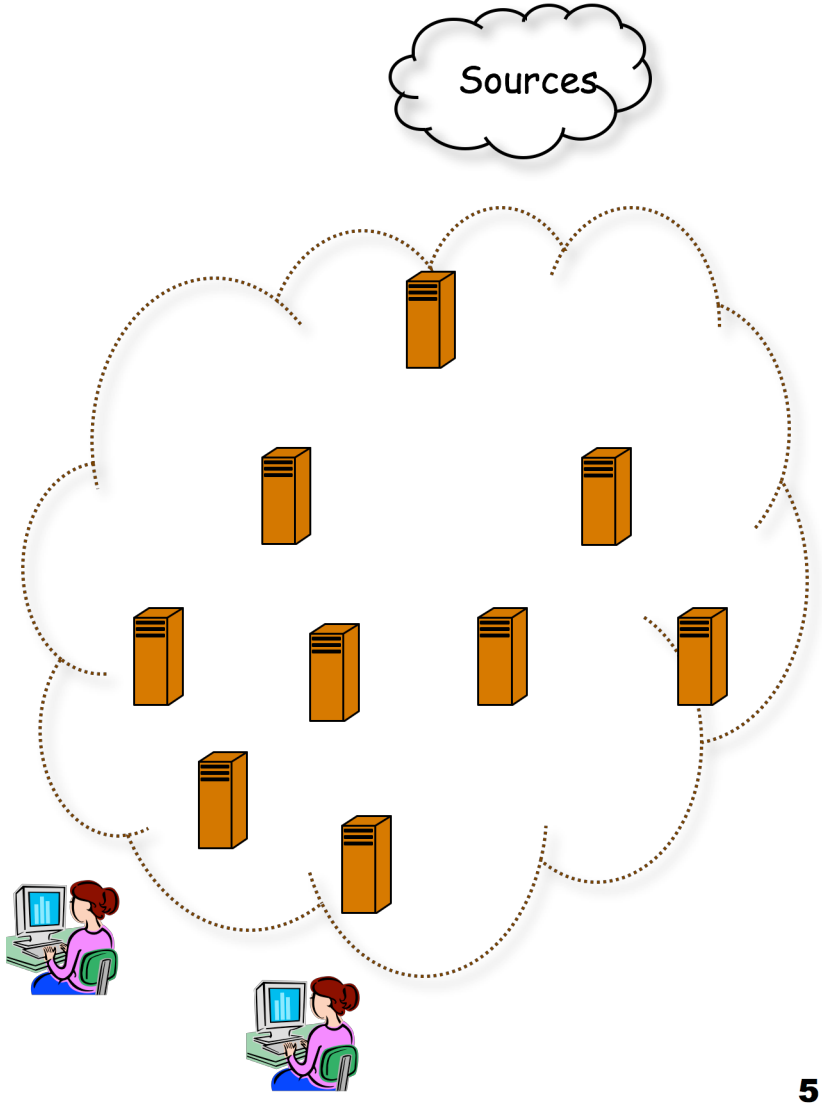
(eXtensible Profile-driven Overlay Routing Trees)

- A generic "infrastructure" system for data dissemination
  - overlay network
  - profile-driven
- Extensibility is key!
  - performance goals
  - profile and data management

XPORT Model

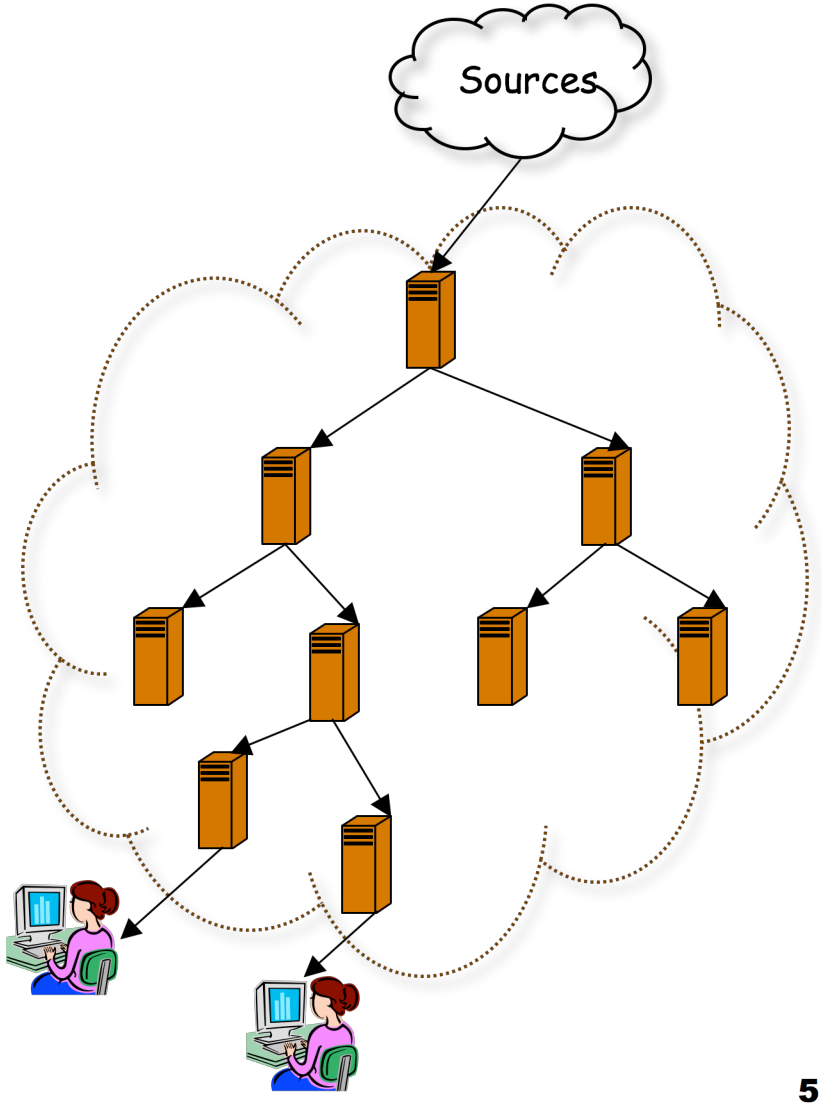
- Separates plumbing from app-specific dissemination logic:
  - Apps provide a small set of methods:
    - Data and profile definitions
    - Profile matching function
    - Performance goals and constraints
  - XPORT then automatically builds, maintains, optimizes an overlay dissemination network

Profile-driven Data Dissemination



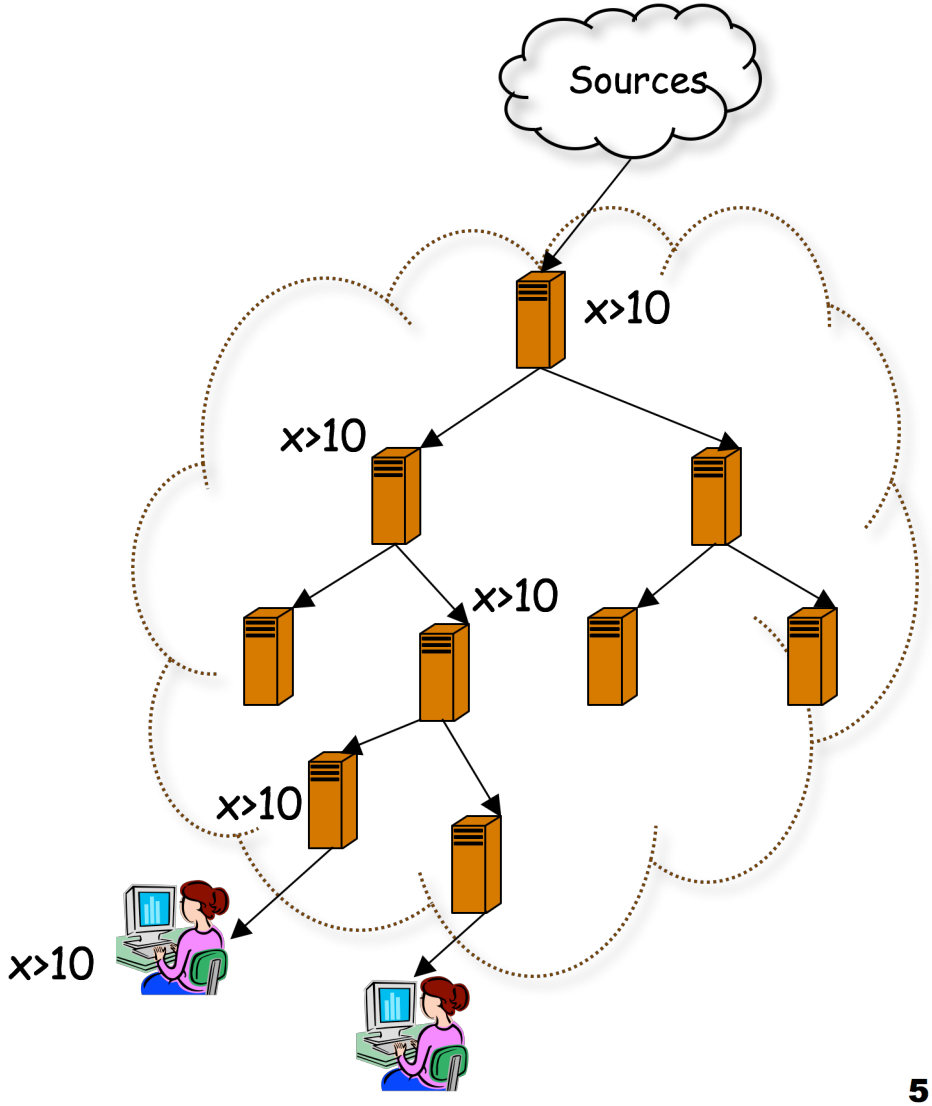
Profile-driven Data Dissemination

- Overlay tree construction



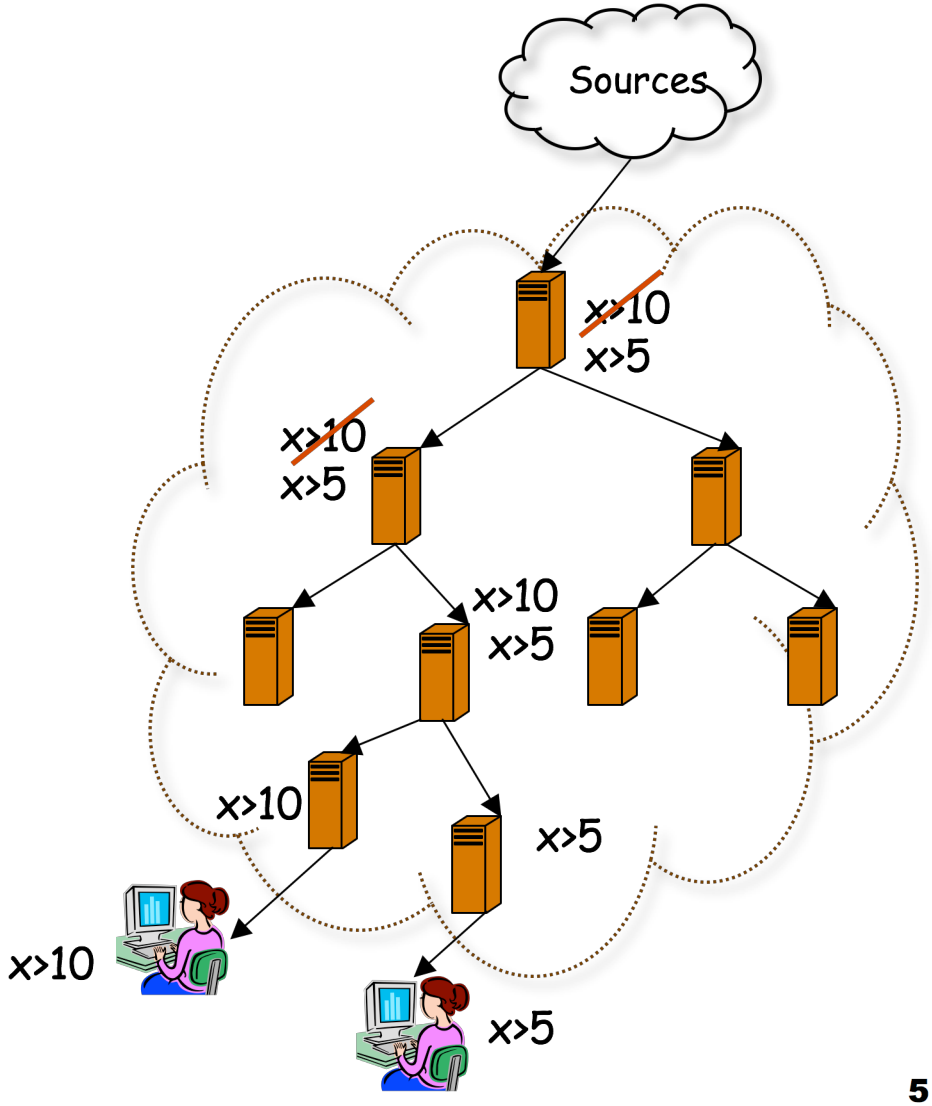
Profile-driven Data Dissemination

- Overlay tree construction
- Clients register



Profile-driven Data Dissemination

- Overlay tree construction
- Clients register
- Profile ***merging***

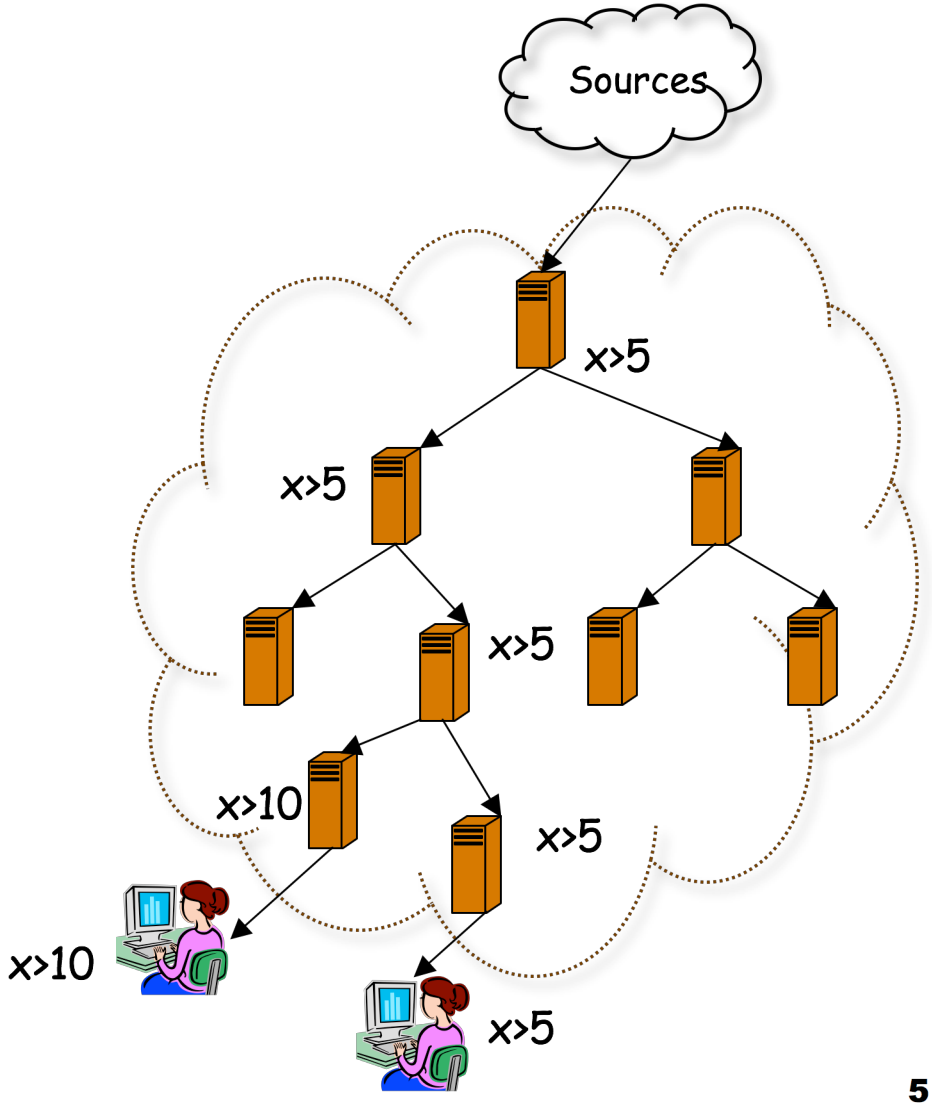


Profile-driven Data Dissemination

- Overlay tree construction
- Clients register
- Profile ***merging***

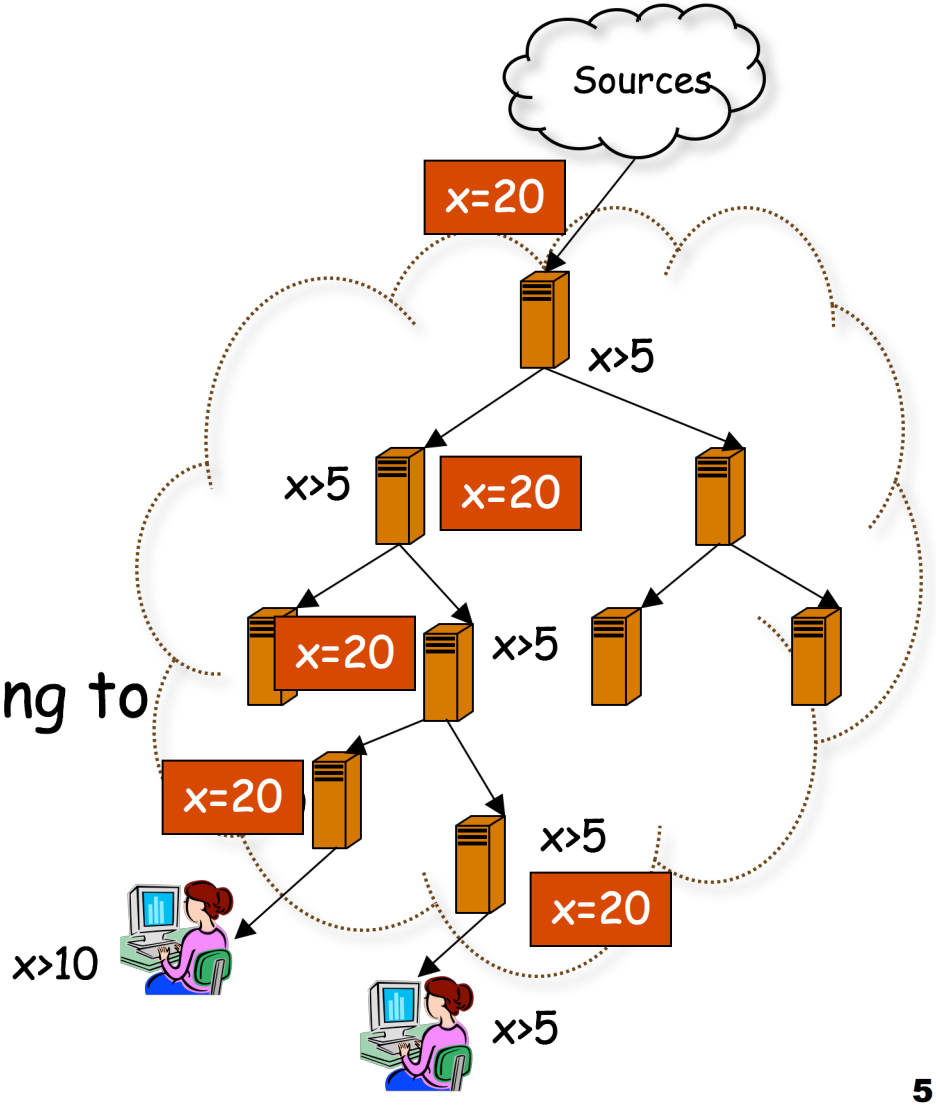
Profile-driven Data Dissemination

- Overlay tree construction
- Clients register
- Profile ***merging***



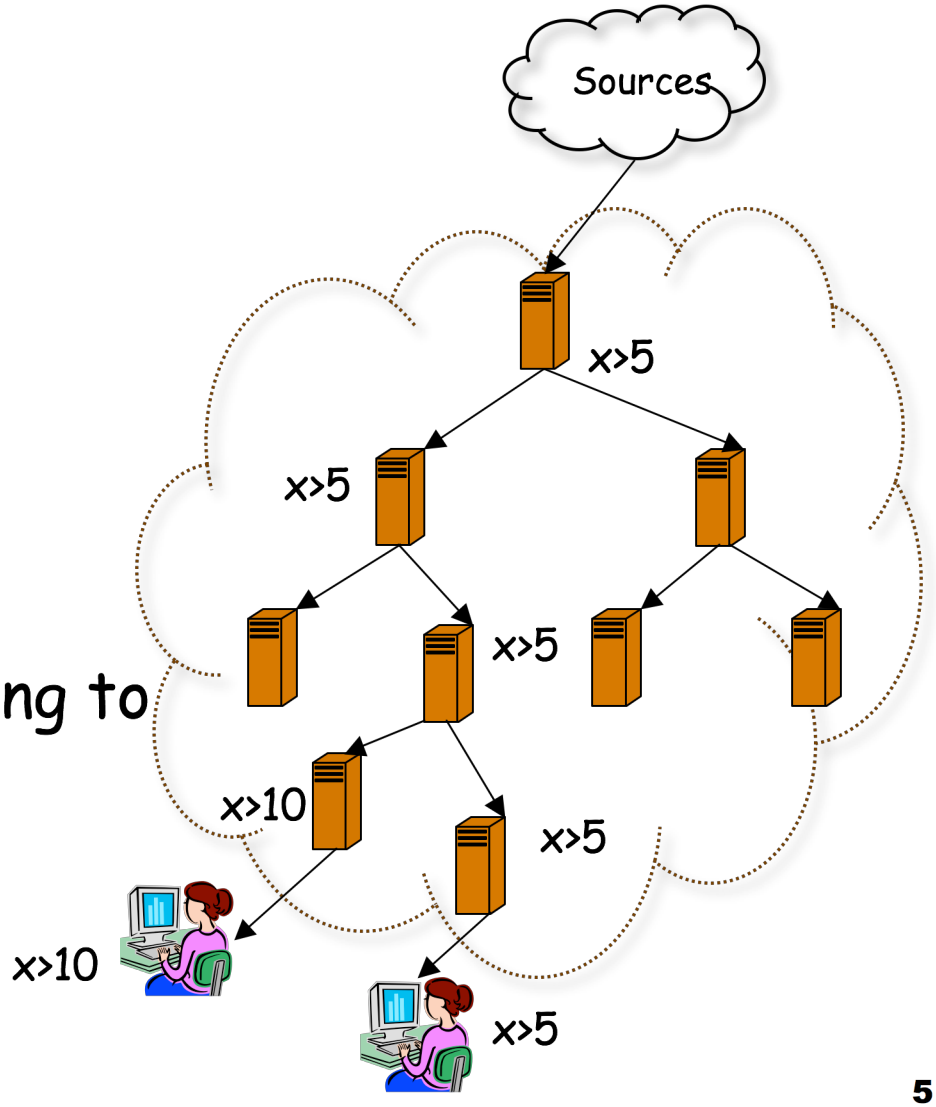
Profile-driven Data Dissemination

- Overlay tree construction
- Clients register
- Profile ***merging***
- Message ***matching***
  - Content-based routing to interested clients



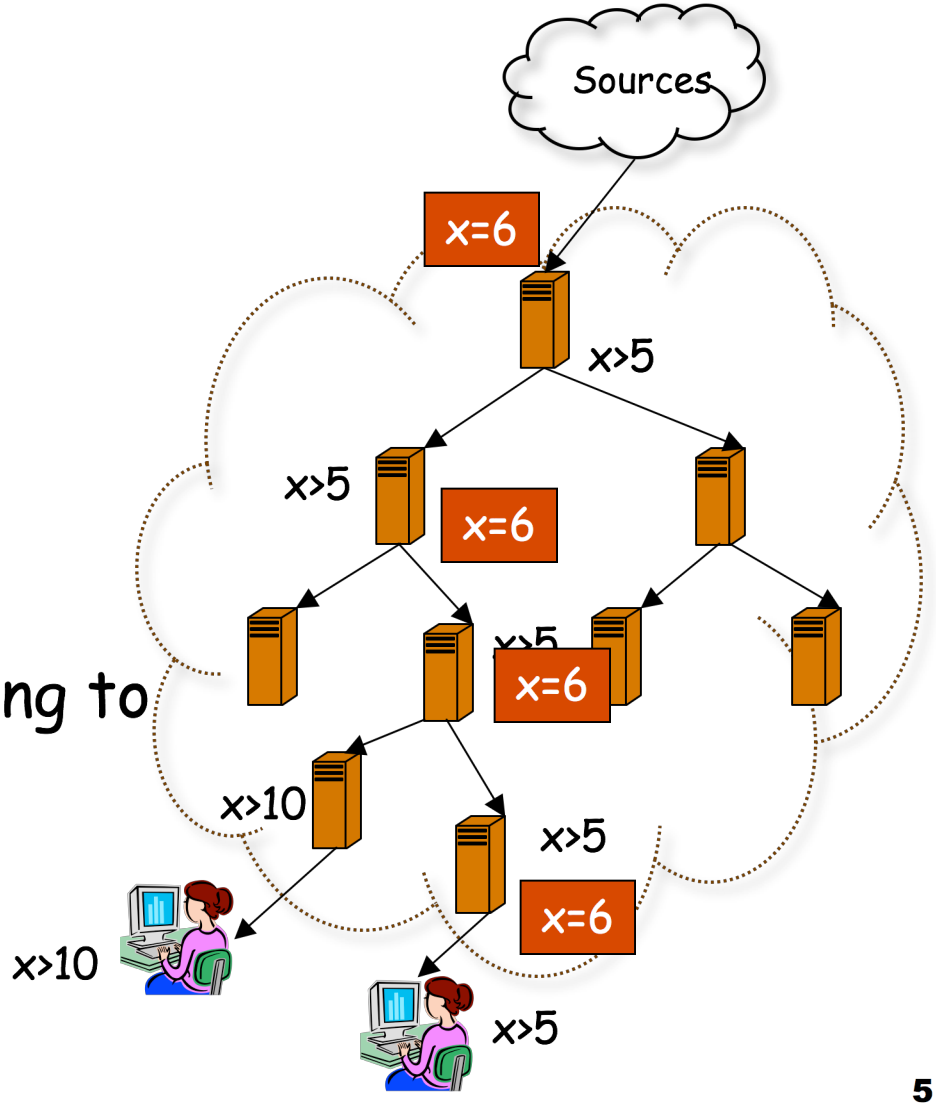
Profile-driven Data Dissemination

- Overlay tree construction
- Clients register
- Profile ***merging***
- Message ***matching***
  - Content-based routing to interested clients



Profile-driven Data Dissemination

- Overlay tree construction
- Clients register
- Profile *merging*
- Message *matching*
  - Content-based routing to interested clients



Profile Extensibility in XPORT

- Applications provide methods for handling their profile and data types

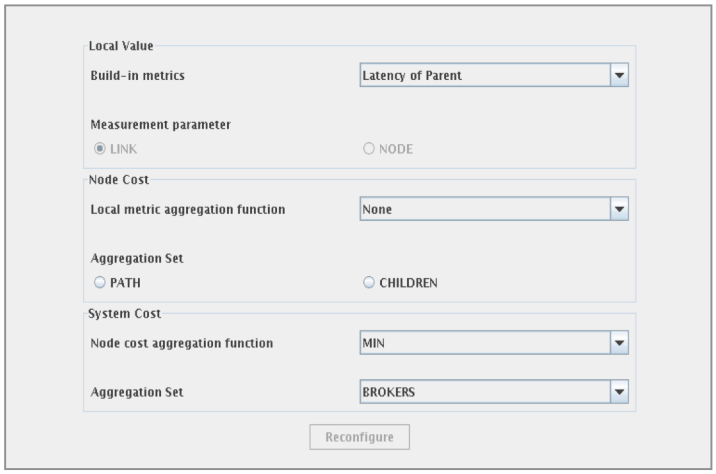
Core methods	match (*message*, *profile*)	Returns true if *message* matches *profile*
	merge (*profile*, *profile*)	Creates a more general profile
Index-based methods	initializeIndex() add (*profile*) remove (*profile*)	Initializes *index* Adds *profile* to the *index* structure Removes *profile* from the *index* structure
	match (message, *index*)	Returns entries in the *index* that match *message*

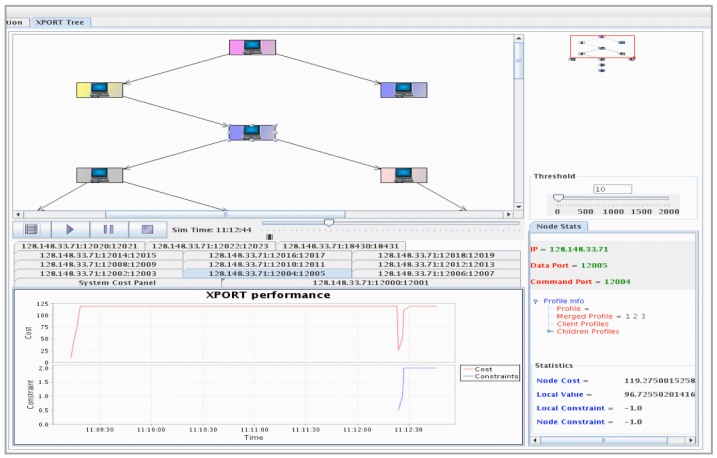
Cost Extensibility in XPORT

- Applications define:
  - System cost
  - Constraint metrics
  - e.g., "minimize total bandwidth consumption while keeping dissemination latencies under 100ms."
- Distributed optimization framework
  - Metric-independent
  - Based on semantics of cost functions
  - Extensible set of optimization rules

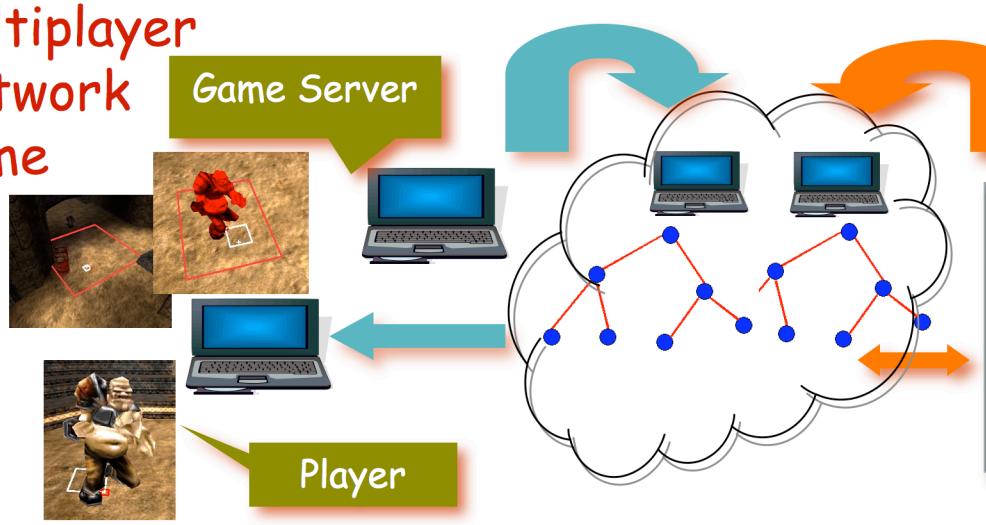
XPORT Prototype

Network configuration & Visualization Tools

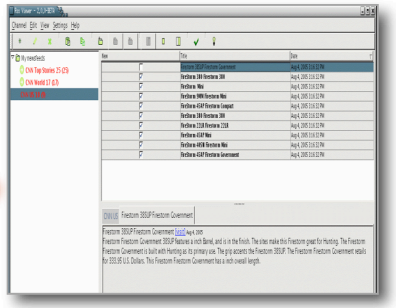




Multiplayer Network Game



RSS Feed Dissemination



Road Map

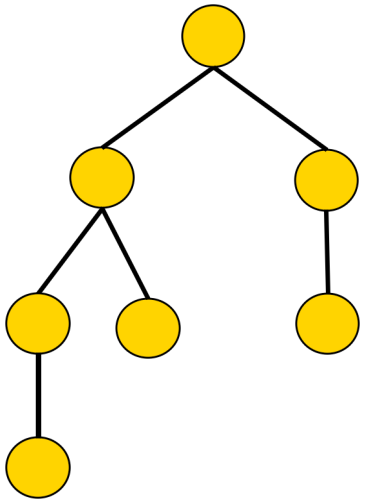
- Motivation
- Basic XPORT system model
- Extensible optimization framework
- Experimental results
- Future work

Metric Definition in XPORT

- Applications define performance metrics and constrained metrics
- Metrics defined by a 2-level aggregation model
  - Level 1: Defines the local node cost
    - Aggregation of a metric over nodes in a "scope":
      - Path to the root
      - Children
  - Level 2: Defines the global system cost
    - Aggregate costs of all nodes
- XPORT includes grammar for metric specification
  - Using different metrics & functions we can define a variety of performance measures & constraints
    - e.g., MIN/MAX, SUM, AVG, VAR…

Metric Examples

- Optimization goal:
  - "Minimize total bandwidth consumption while keeping dissemination latencies under 100ms."
- In XPORT:
  - MIN(SUM(SUM(children, in_data)),brokers)
  - while SUM(ancestors, link_latency) < 100ms.



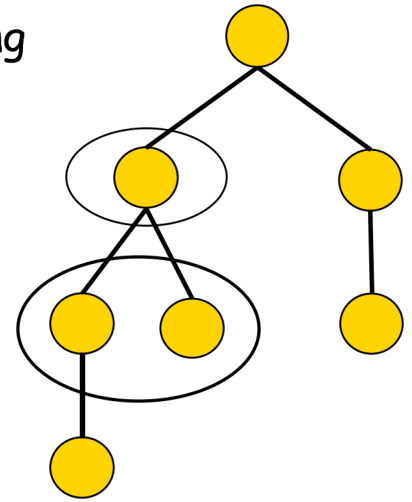
Metric Examples

- Optimization goal:
  - "Minimize total bandwidth consumption while keeping dissemination latencies under 100ms."
- In XPORT:
  - MIN(SUM(SUM(children, in_data)),brokers)
  - while SUM(ancestors, link_latency) < 100ms.
- Total bandwidth consumption

  Level 1: bandwidth consumption= SUM (in_data of all children)

  Level 2: system cost = SUM (bandwidth consumption)

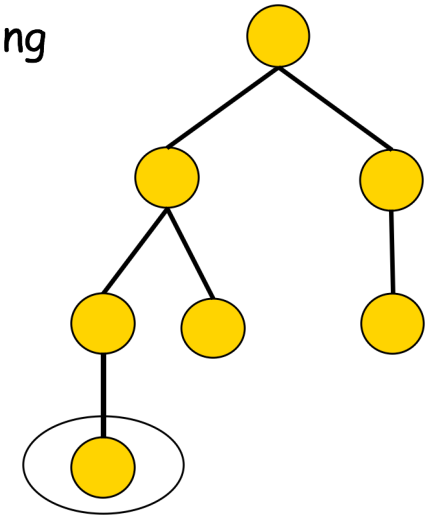
Aggregation over children



Metric Examples

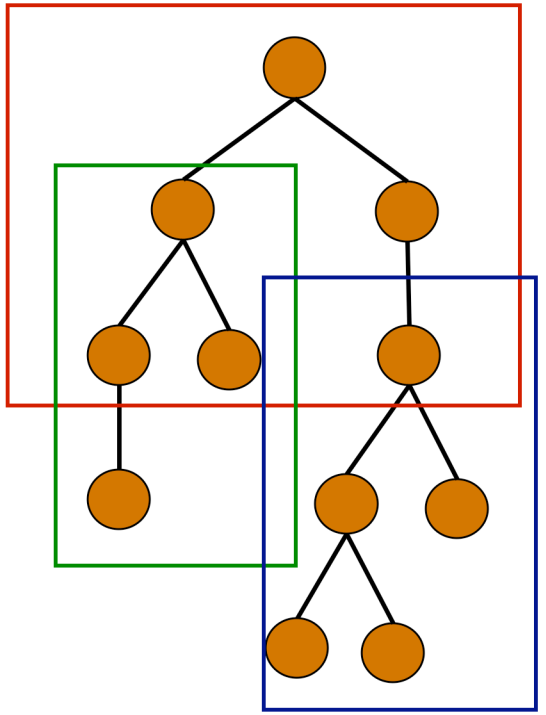
- Optimization goal:
  - "Minimize total bandwidth consumption while keeping dissemination latencies under 100ms."
- In XPORT:
  - MIN(SUM(SUM(children, in_data)),brokers)
  - while SUM(ancestors, link_latency) < 100ms.
- Total bandwidth consumption
  - Level 1: bandwidth consumption= SUM (in_data of all children)
  - Level 2: system cost = SUM (bandwidth consumption)
- Path latency
  - Level 1: path latency = SUM (link_latency on path)

Aggregation over path



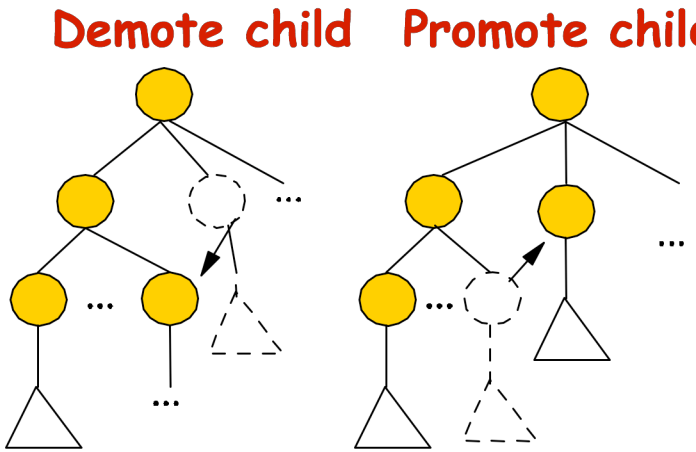
Optimization Approach

- Goal: minimize system cost
  - **Should be independent of specified metric**
- Reconfigure network structure to improve cost
  - Use transformations rules
  - Use cost model to quantify effect of each rule
- Divide network to **optimization units**
  - Apply transformation rules within the scope of this unit
  - Reduces the scope of the affected nodes



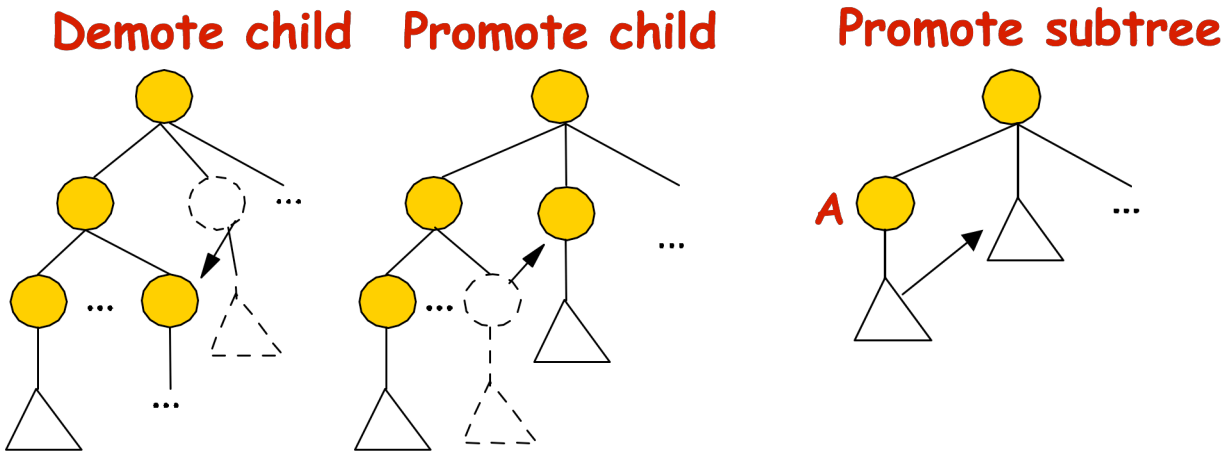
Network Transformations

- XPORT provides two primitive transformations
  - child demotion & child promotion
  - Cost model that quantifies the benefit
- Applications define composite transformations
  - e.g. subtree promotion, subtree demotion
  - Faster improvement, avoid local optimum
  - XPORT identifies automatically the cost function to quantify the composite ones



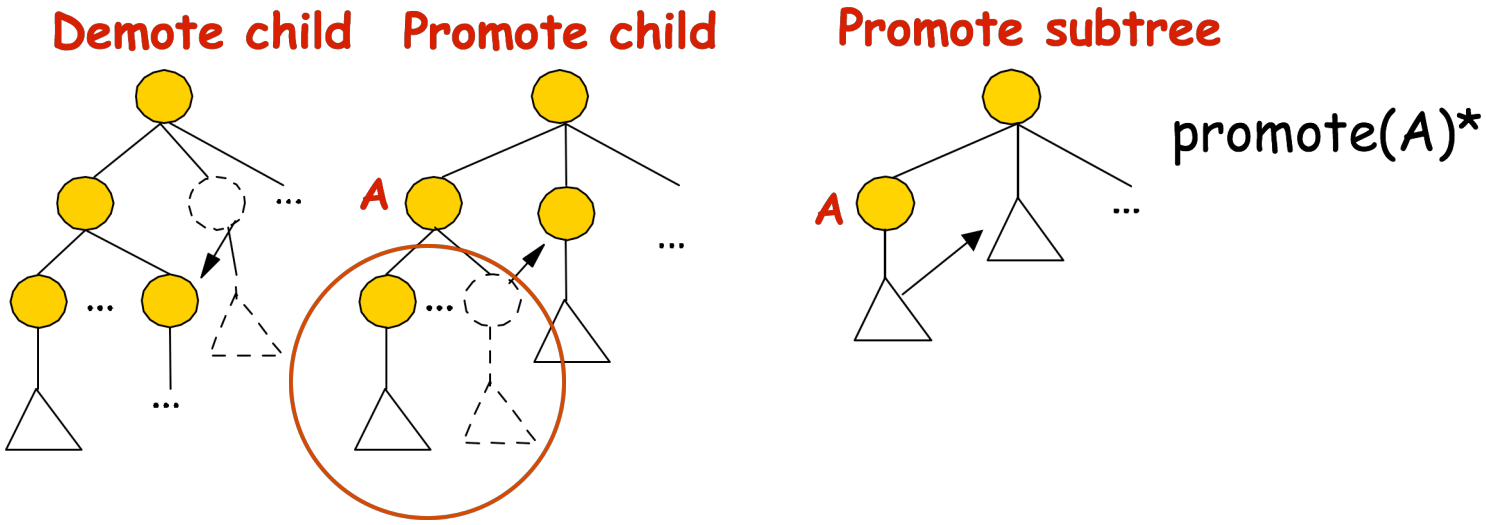
Network Transformations

- XPORT provides two primitive transformations
  - child demotion & child promotion
  - Cost model that quantifies the benefit
- Applications define composite transformations
  - e.g. subtree promotion, subtree demotion
  - Faster improvement, avoid local optimum
  - XPORT identifies automatically the cost function to quantify the composite ones

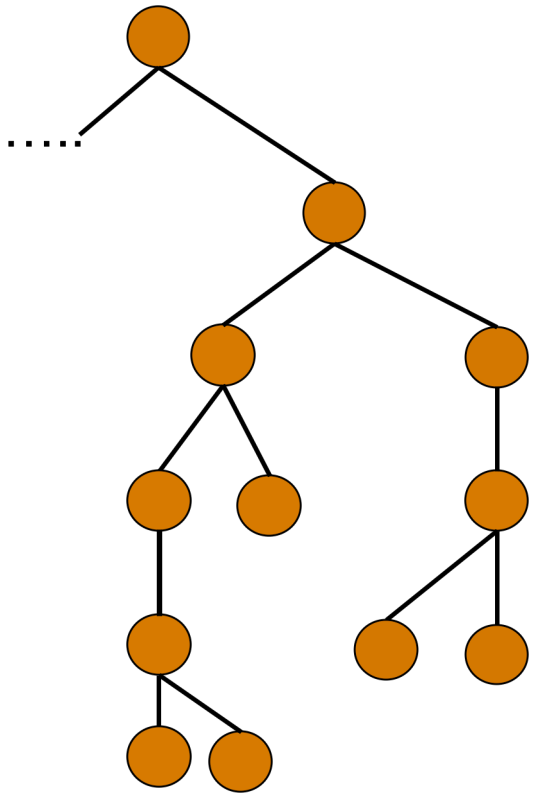


Network Transformations

- XPORT provides two primitive transformations
  - child demotion & child promotion
  - Cost model that quantifies the benefit
- Applications define composite transformations
  - e.g. subtree promotion, subtree demotion
  - Faster improvement, avoid local optimum
  - XPORT identifies automatically the cost function to quantify the composite ones

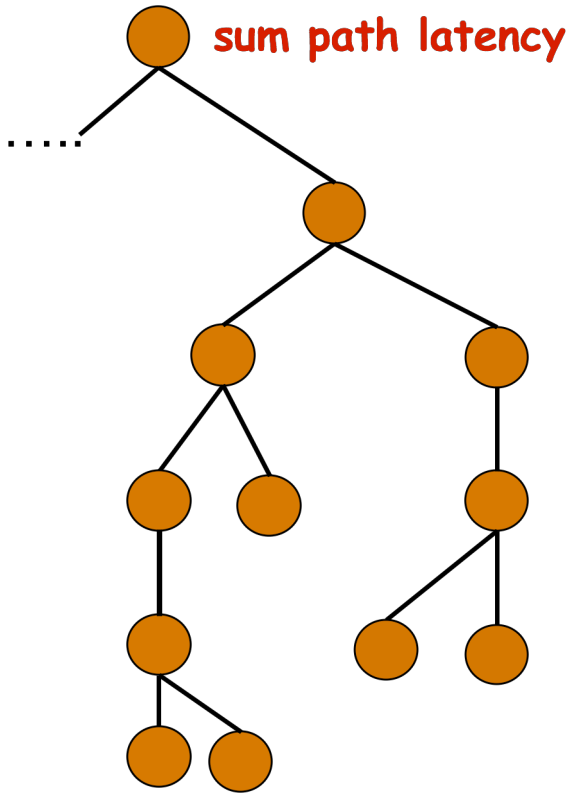


Quantify Cost Improvement I



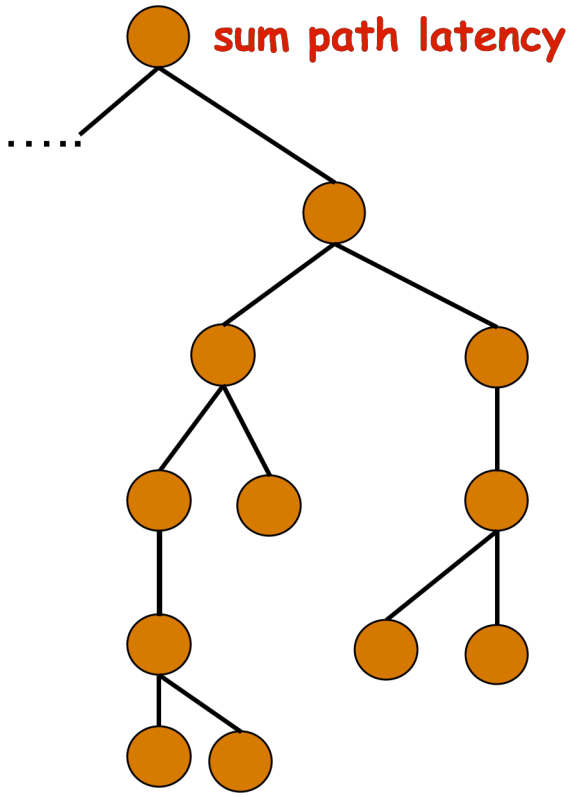
Quantify Cost Improvement I

- Exhaustive approach for cost benefit
  - Calculate new cost of all nodes and compare!



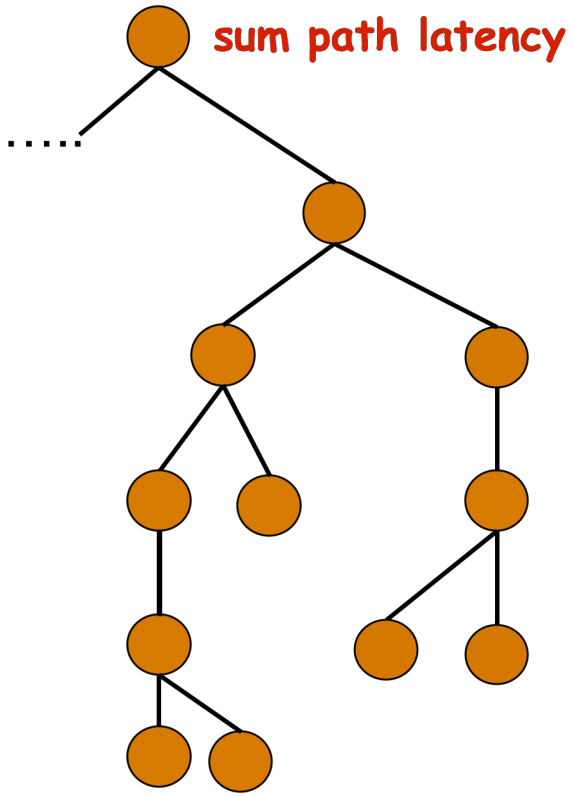
Quantify Cost Improvement I

- Exhaustive approach for cost benefit
  - Calculate new cost of all nodes and compare!
- XPORT's approach
  - Identify affected nodes of each transformation



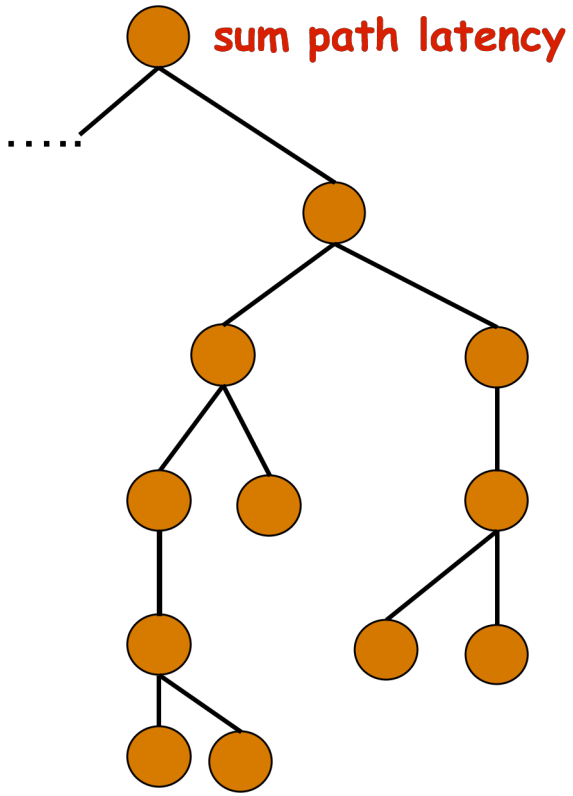
Quantify Cost Improvement I

- Exhaustive approach for cost benefit
  - Calculate new cost of all nodes and compare!
- XPORT's approach
  - Identify affected nodes of each transformation
  - Define:



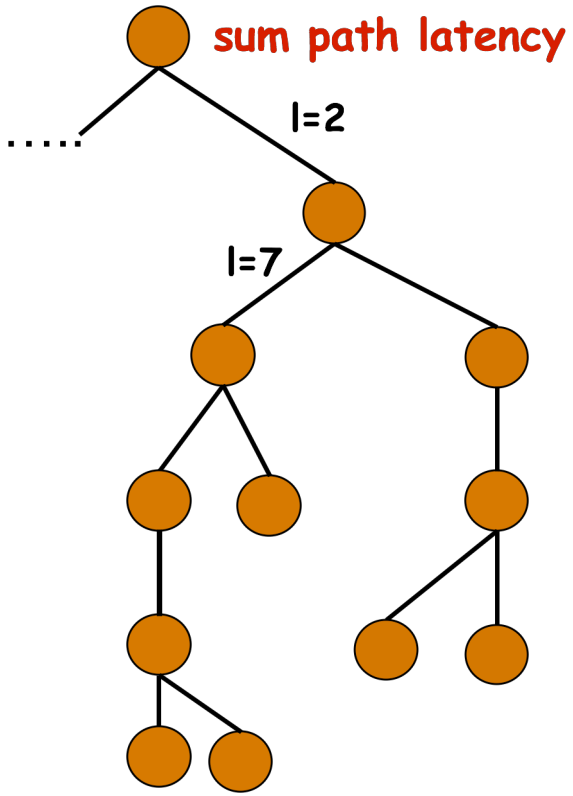
Quantify Cost Improvement I

- Exhaustive approach for cost benefit
  - Calculate new cost of all nodes and compare!
- XPORT's approach
  - Identify affected nodes of each transformation
  - Define:
    - Dependents of node: nodes affected by its cost



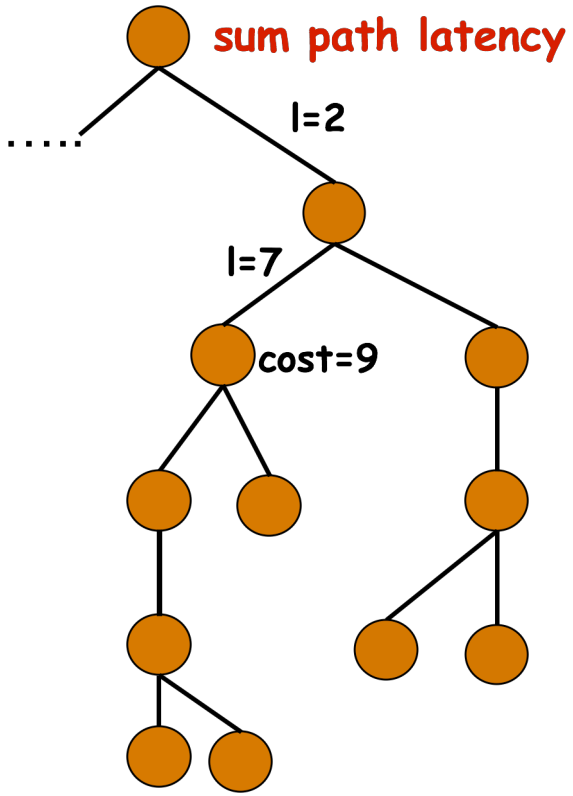
Quantify Cost Improvement I

- Exhaustive approach for cost benefit
  - Calculate new cost of all nodes and compare!
- XPORT's approach
  - Identify affected nodes of each transformation
  - Define:
    - Dependents of node: nodes affected by its cost



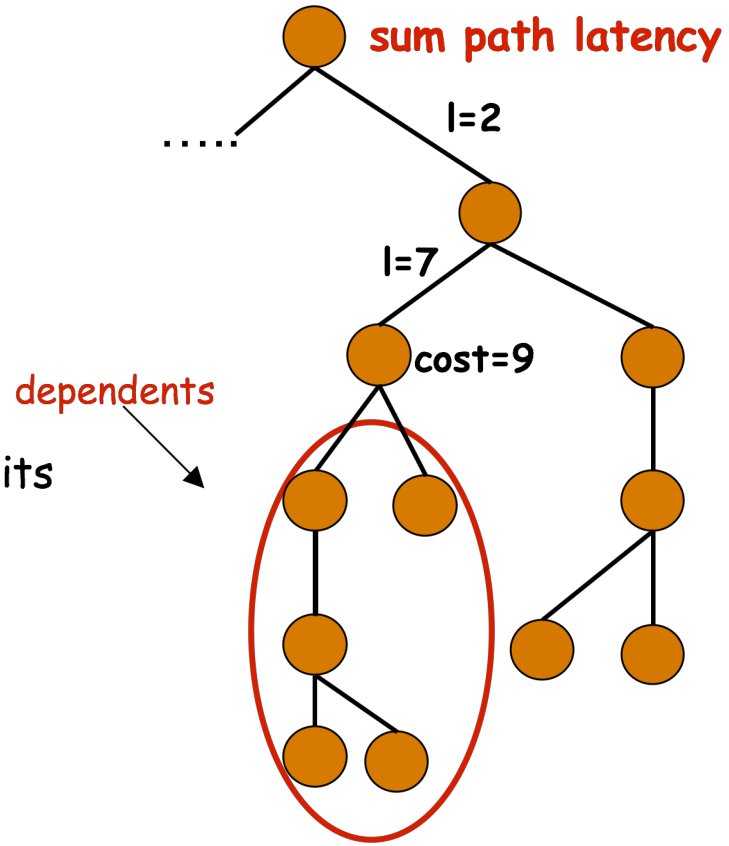
Quantify Cost Improvement I

- Exhaustive approach for cost benefit
  - Calculate new cost of all nodes and compare!
- XPORT's approach
  - Identify affected nodes of each transformation
  - Define:
    - Dependents of node: nodes affected by its cost



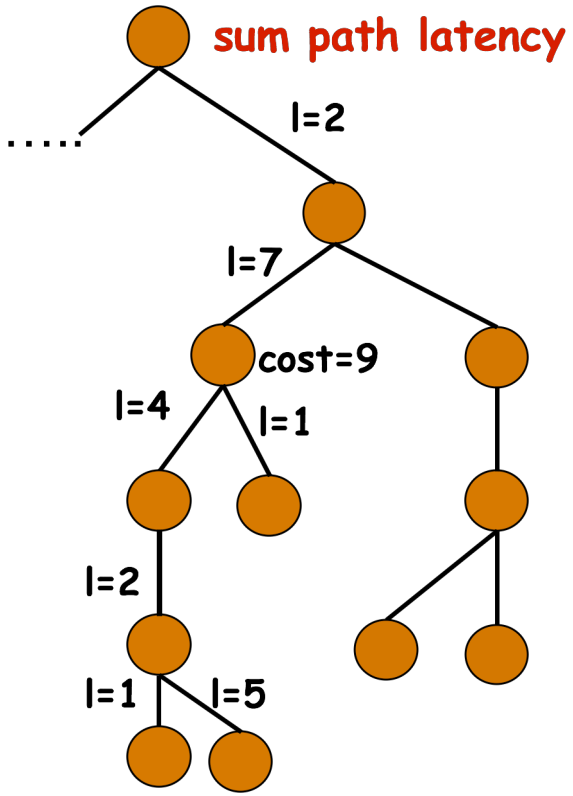
Quantify Cost Improvement I

- Exhaustive approach for cost benefit
  - Calculate new cost of all nodes and compare!
- XPORT's approach
  - Identify affected nodes of each transformation
  - Define:
    - Dependents of node: nodes affected by its cost



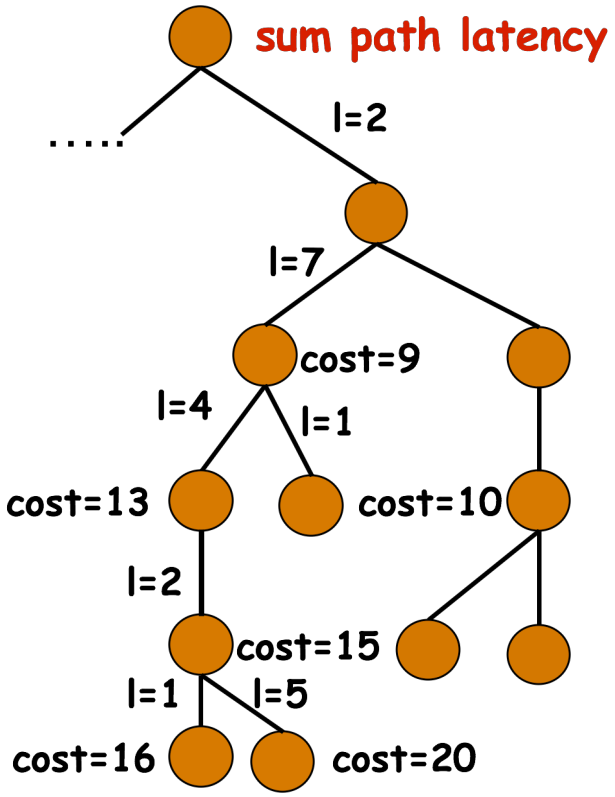
Quantify Cost Improvement I

- Exhaustive approach for cost benefit
  - Calculate new cost of all nodes and compare!
- XPORT's approach
  - Identify affected nodes of each transformation
  - Define:
    - Dependents of node: nodes affected by its cost



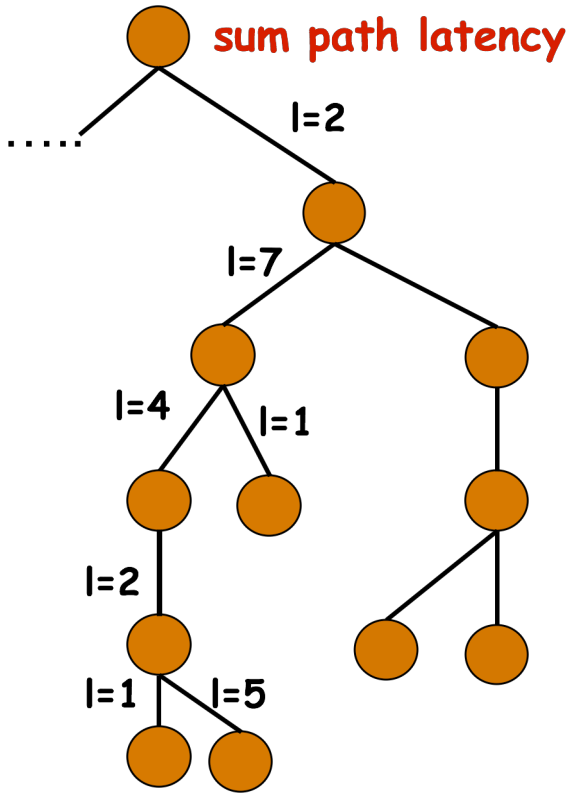
Quantify Cost Improvement I

- Exhaustive approach for cost benefit
  - Calculate new cost of all nodes and compare!
- XPORT's approach
  - Identify affected nodes of each transformation
  - Define:
    - Dependents of node: nodes affected by its cost



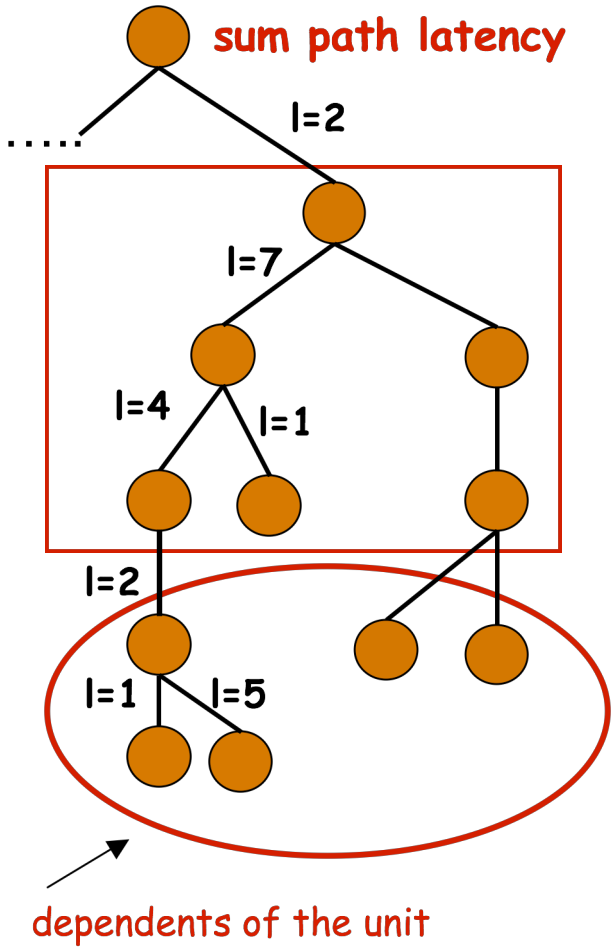
Quantify Cost Improvement I

- Exhaustive approach for cost benefit
  - Calculate new cost of all nodes and compare!
- XPORT's approach
  - Identify affected nodes of each transformation
  - Define:
    - Dependents of node: nodes affected by its cost
    - Dependents of optimization unit
      - Nodes affected outside the unit



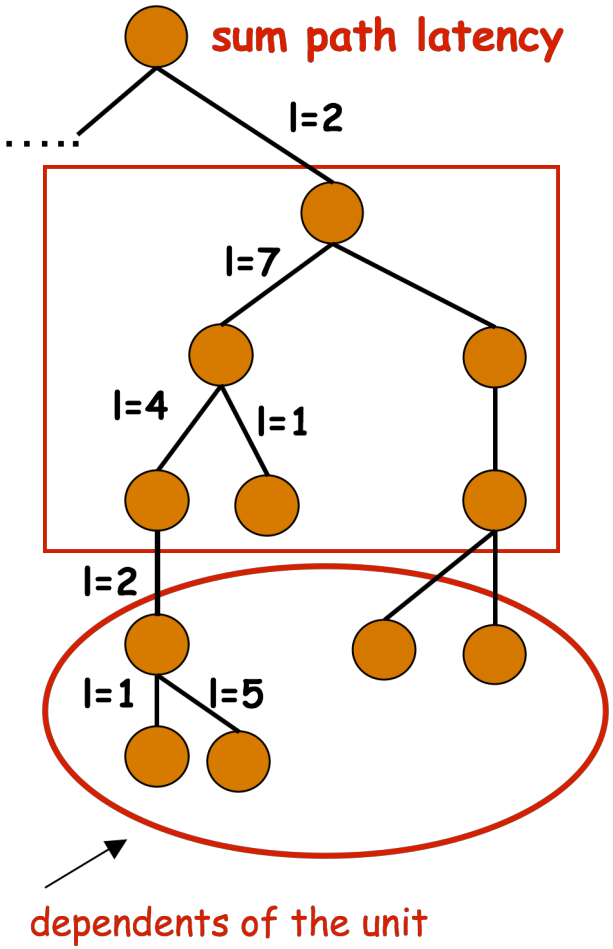
Quantify Cost Improvement I

- Exhaustive approach for cost benefit
  - Calculate new cost of all nodes and compare!
- XPORT's approach
  - Identify affected nodes of each transformation
  - Define:
    - Dependents of node: nodes affected by its cost
    - Dependents of optimization unit
      - Nodes affected outside the unit



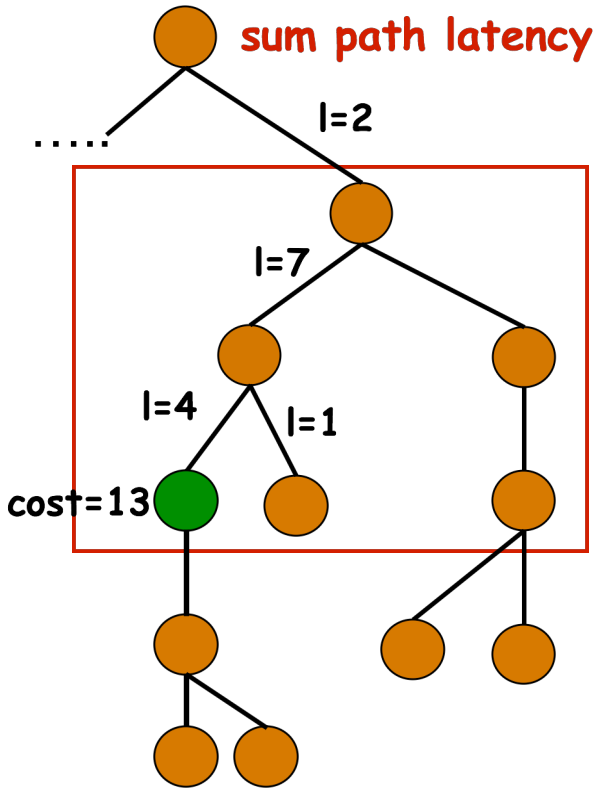
Quantify Cost Improvement I

- Exhaustive approach for cost benefit
  - Calculate new cost of all nodes and compare!
- XPORT's approach
  - Identify affected nodes of each transformation
  - Define:
    - Dependents of node: nodes affected by its cost
    - Dependents of optimization unit
      - Nodes affected outside the unit
  - Cost benefit is an aggregation of new cost of nodes in the unit and the dependents' cost
  - Maintain state for dependents
    - Calculate changes of the dependents' cost
    - Minimize communication outside the unit



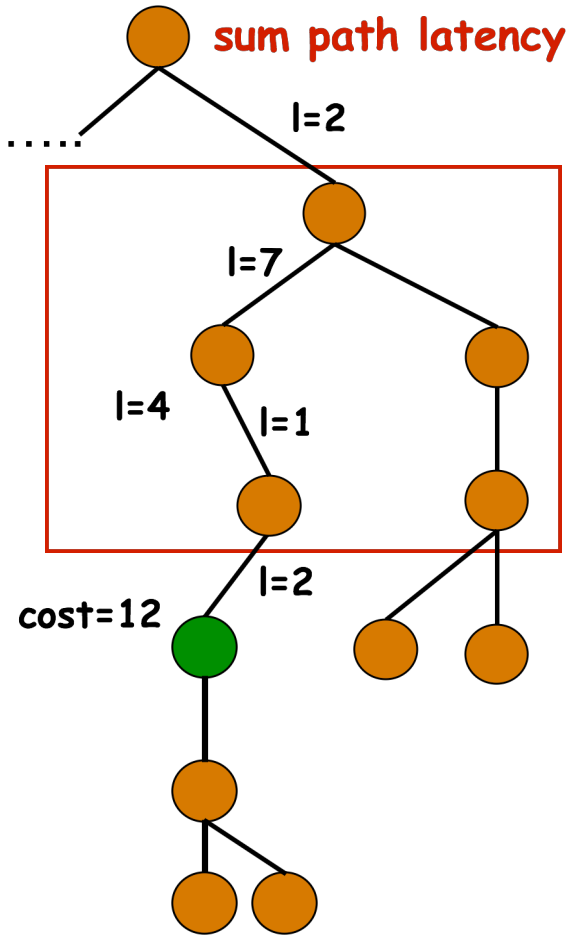
Quantify Cost Improvement II

- Example: sum of path latency
  - State: Dependence set size
    - all dependents are affected equally



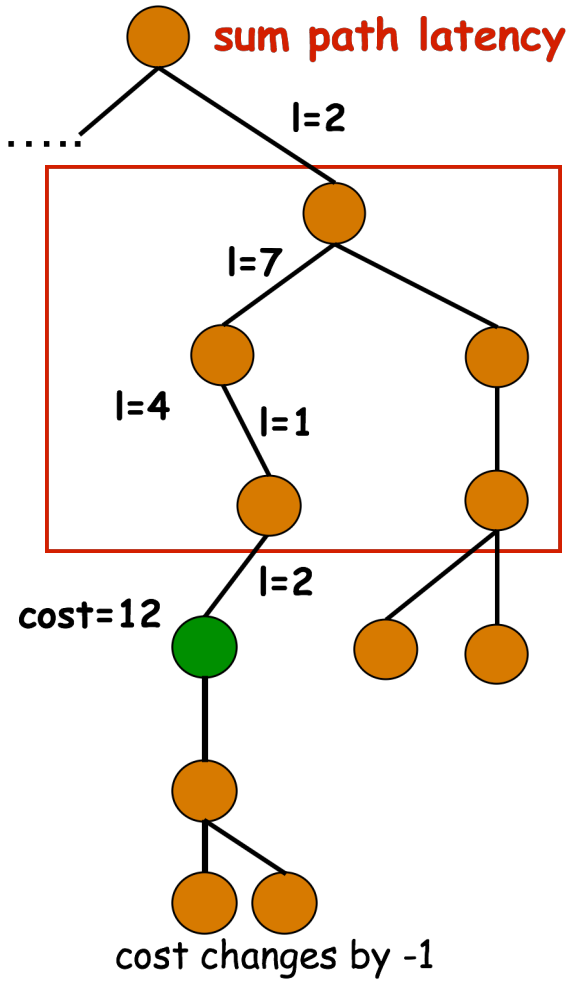
Quantify Cost Improvement II

- Example: sum of path latency
  - State: Dependence set size
    - all dependents are affected equally



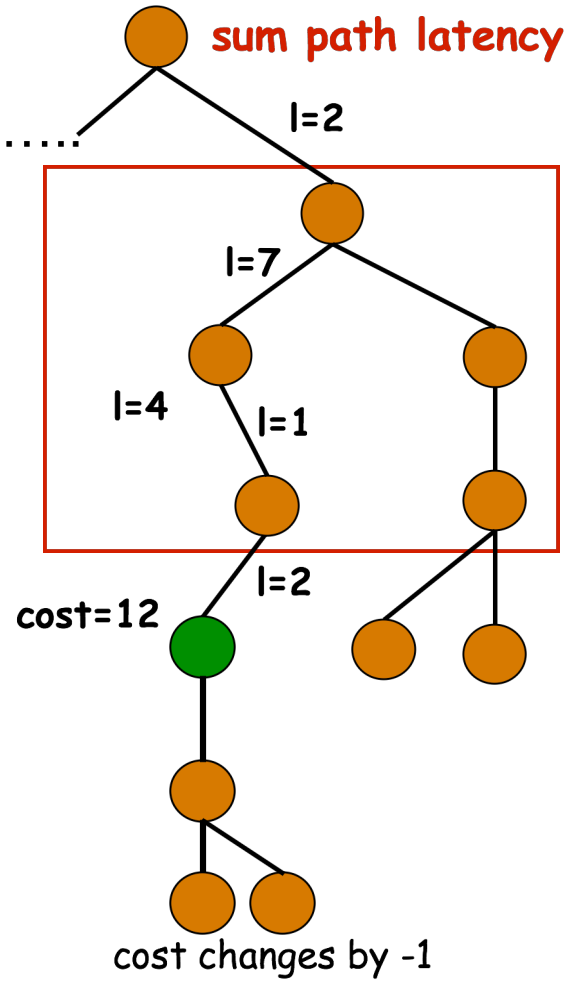
Quantify Cost Improvement II

- Example: sum of path latency
  - State: Dependence set size
    - all dependents are affected equally



Quantify Cost Improvement II

- Example: sum of path latency
  - State: Dependence set size
    - all dependents are affected equally
- Based on function semantics automatically identify:
  - State required
  - Metadata exchanged for optimization purposes
  - O(1) for most aggregation functions



Run-time optimization protocols

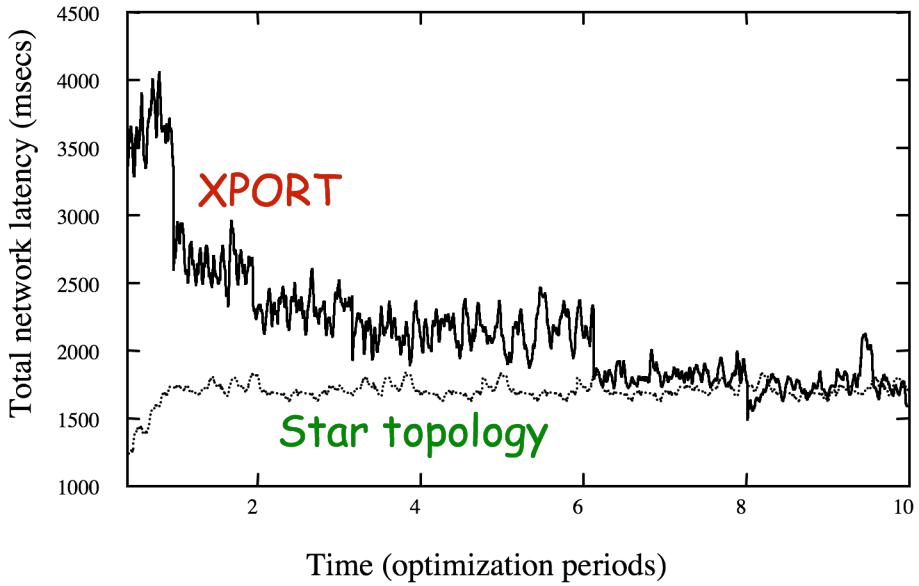
- Bottleneck-based
  - Optimize global cost
    - max CPU load in network
  - Optimize critical units
    - Units with max CPU load
  - Apply best transformation in the system
  - Guarantee improvement
  - Reactive optimization
    - Work on units with most loaded node
- Opportunistic
  - Optimize cost of unit
    - max CPU load in unit
  - Optimize all units
    - Independent from CPU load
  - Apply best transformation in the unit
  - No guarantees
  - Proactive optimization
    - Could prevent nodes from increasing their CPU load

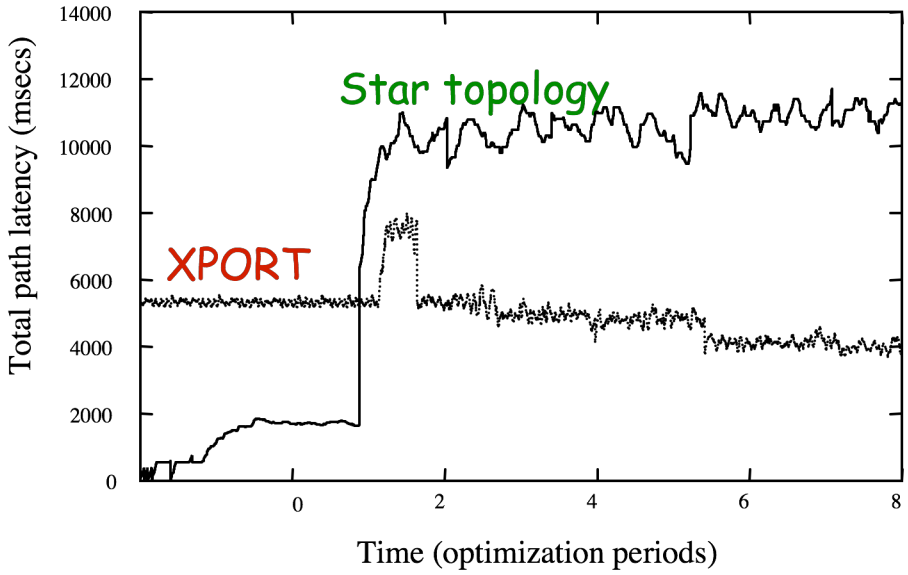
Experimental results

- RSS Feed dissemination application
  - Clients connect to a proxy and send their requests
  - Root periodically polls RSS sources
- Experiments:
  - 100 clients, 700 RSS Feeds
  - 20 -100 nodes @PlanetLab & LAN Emulation

Convergence & Adaptivity results

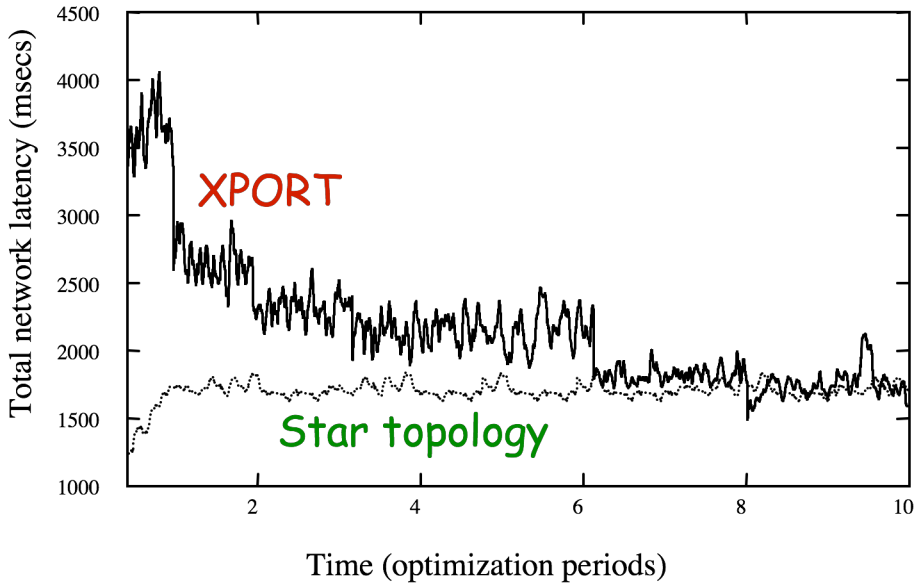
- Network latency
  - XPORT converges to optimal topology after 8 transformations
- Network + matching latency
  - Optimization adapts to increased cpu load

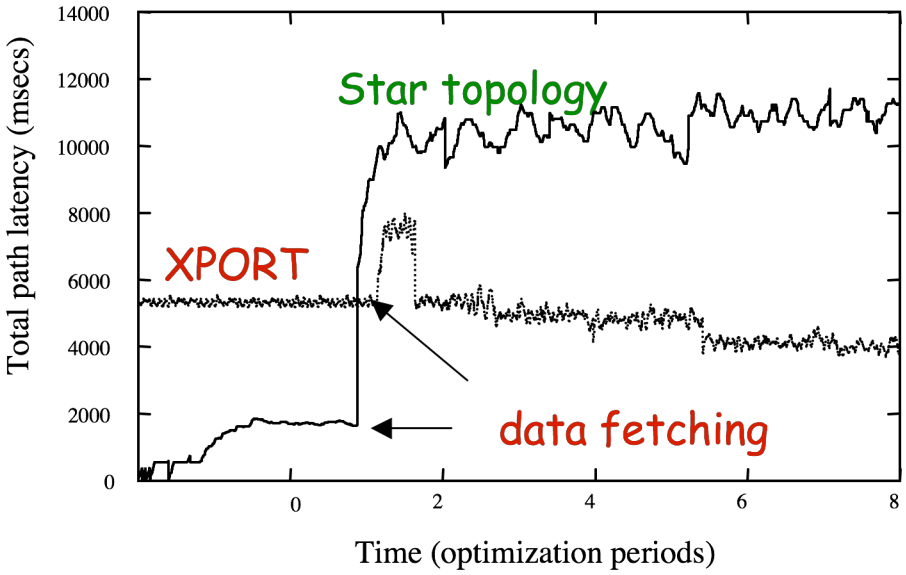




Convergence & Adaptivity results

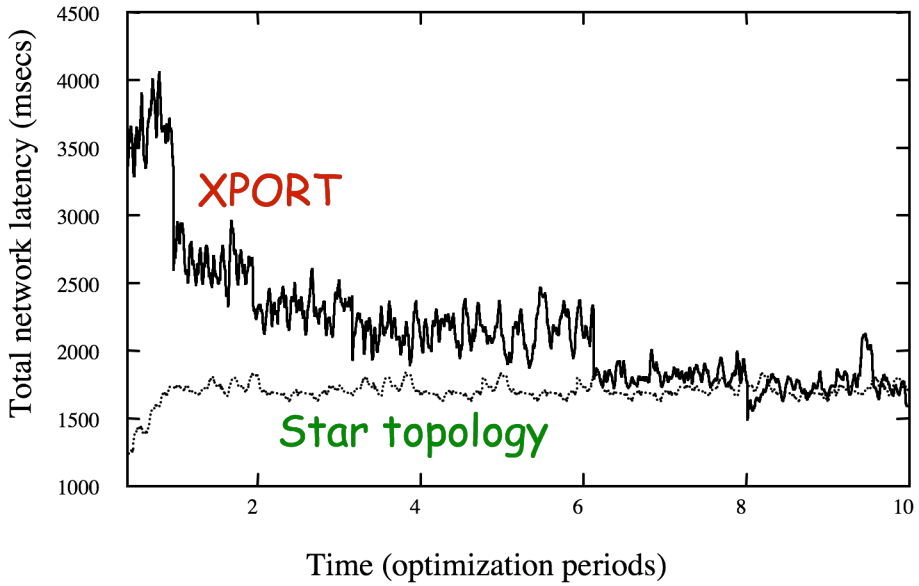
- Network latency
  - XPORT converges to optimal topology after 8 transformations
- Network + matching latency
  - Optimization adapts to increased cpu load

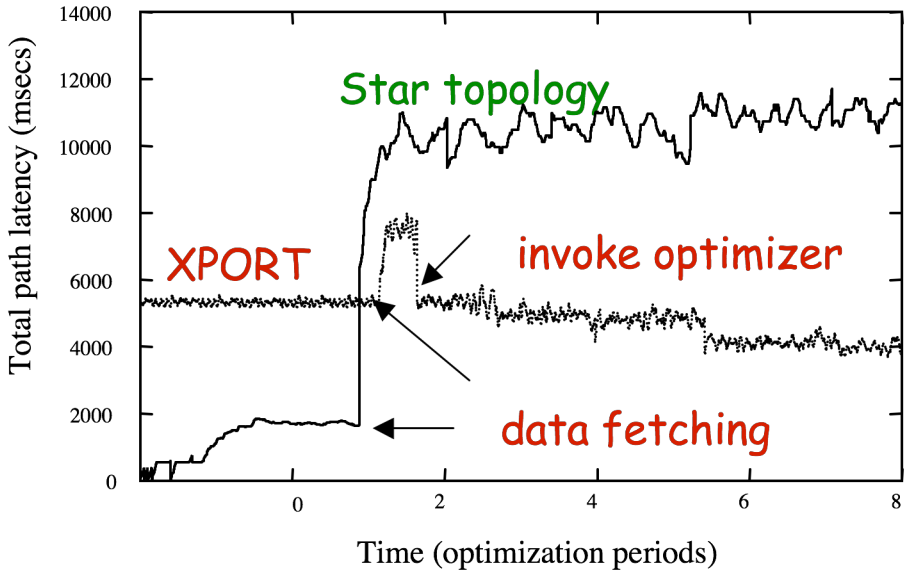




Convergence & Adaptivity results

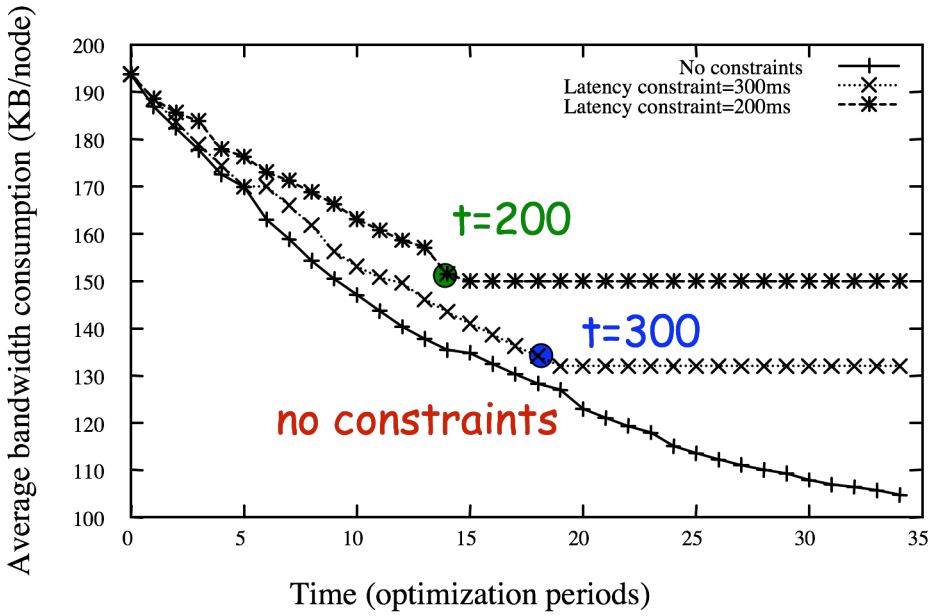
- Network latency
  - XPORT converges to optimal topology after 8 transformations
- Network + matching latency
  - Optimization adapts to increased cpu load

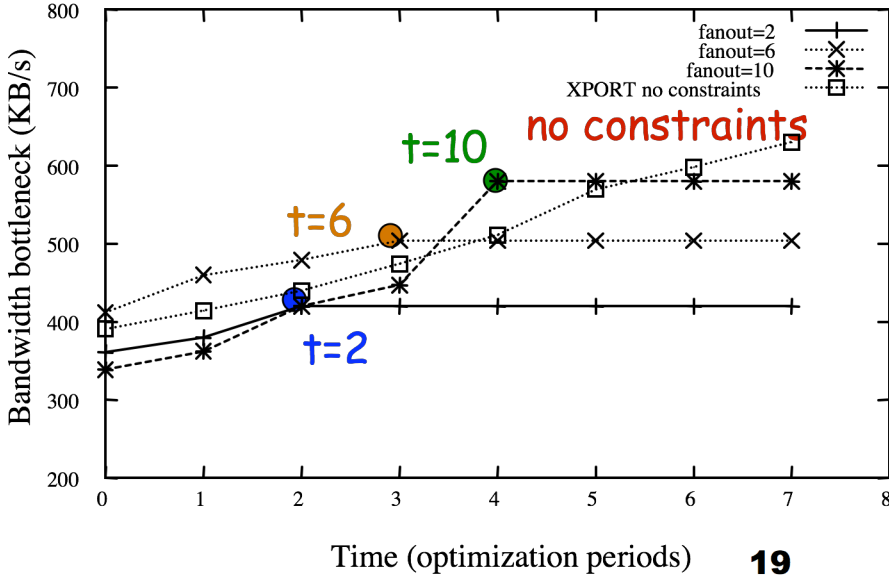




Constrained metrics

- Avg bandwidth consumption
  - aggregation (SUM) over children
- Constraint on path latency
  - 200ms -300ms
- Minimum bottleneck bandwidth
  - aggregation (MIN) over path
- Constraint on node fanout
  - 2-10 children

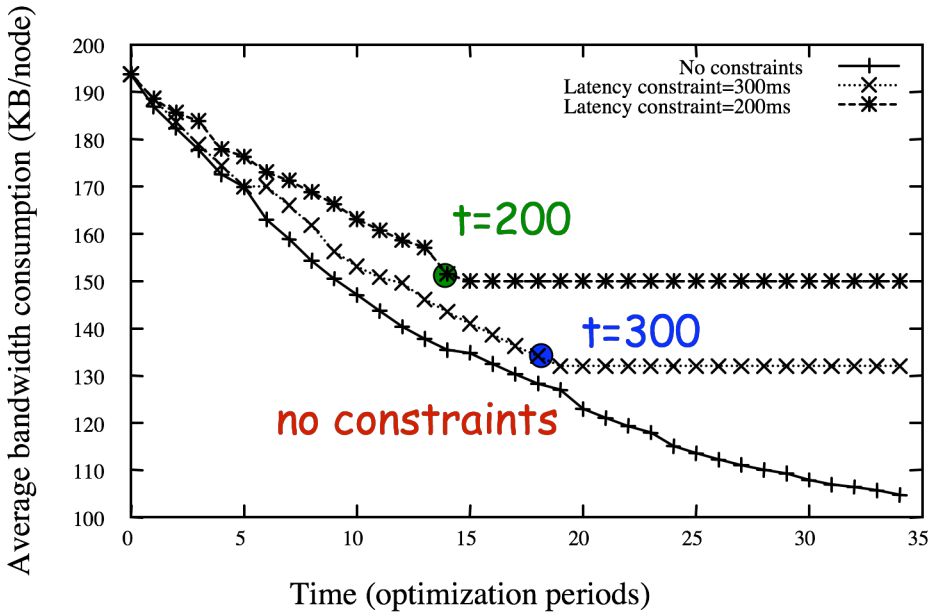


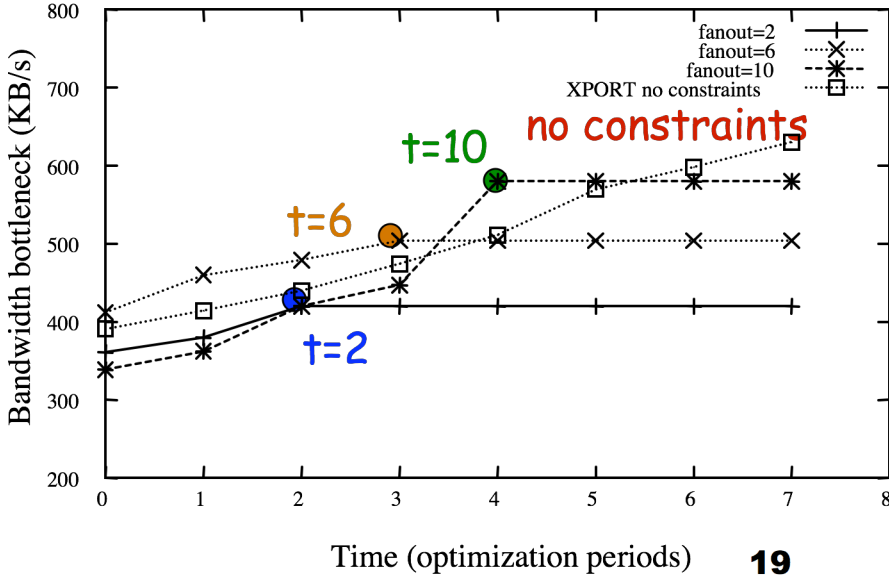


Constrained metrics

- Avg bandwidth consumption
  - aggregation (SUM) over children
- Constraint on path latency
  - 200ms -300ms
- Minimum bottleneck bandwidth
  - aggregation (MIN) over path
- Constraint on node fanout
  - 2-10 children

More relaxed constraints allow XPORT to perform better





Related work

- Extensibility in databases
  - System R [Astrahan'76] , Starburst [Schwarz'86] , GiST [Hellerstein'95]
- Networks
  - Click [Kohler'00], MACEDON [Rodriguez'04], P2 [Loo'05]
- Adaptive optimization for dissemination networks
  - [Banerjee'03], [Zhou'06]

Future Work

- Profile extensibility
  - Stateful subscriptions
- Collection & dissemination integration
- Support overlay meshes

Conclusions

- We explored extensibility in overlay routing trees
  - Profile-based data dissemination
- We designed and implemented XPORT
  - Application-aware dissemination-based infrastructure
  - Highly extensible and application-customizable