Logistical Multicast for Data Distribution

Jason Zurawski

Martin Swany

Micah Beck

Ying Ding

University of Delaware

The University of Tennessee

Talk Progression

- Goals / Motivation
- Logistical Networking Overview
- Measurement Gathering / Analysis
- Experimental Methods
- Results
- Conclusion
- Future Plans

Goals

- Illustrate the problem posed by information locality within the Grid.
- Existing software in conjunction with intelligent modifications can strengthen an old idea.
- Low tech solutions provide an adequate performance metric.

Motivation

- The Grid paradigm allows for dynamic access to resources
- Data staging is often overlooked
  - Time can be significant when resources are spread out
- Multicast speeds up data distribution
  - Most resources participate in the process
  - Overall performance relies on heuristics

Multicast Benefits

- Reduction of transfer distance / time
  - Long end to end transfers broken into shorter hops
  - Greater degree of parallel transfers
- Bandwidth requirements spread throughout network
  - Individual links have less chance of being saturated
- New path potentially faster than direct link
  - Speed of transfer dictated by the slowest link

Multicast Framework

- Require more than IP Multicast
- Functionality to manage storage, distribution of data
  - Reservation system to reserve space
  - Long term storage potential
  - Access from arbitrary locations through credentials
- Leverage storage / processing utility within the network fabric
  - Transform computers into data staging points

Logistical Networking

- Similar to 'logistics' seen in transportation industry
  - Move and stage cargo according to demand, cost
- Comes in many flavors
  - Internet Backplane Protocol (IBP)
  - Logistical Session Layer (LSL)
- Characteristics:
  - 'Depots' act as warehouses for en-route data
  - Toolkits to transfer, store, and mange the process

The Internet Backplane Protocol

- Treat the Internet as computer backplane
  - Address internal 'resources' directly
- TCP is end-to-end; ignores structure of network 'core'
  - IBP allows resources 'in the middle' to be addressed
  - Aides applications in transfer and storage duties
- Located worldwide
  - Most located on PlanetLab

IBP Role

- Utilize Depots and Multicast primitive
  - Vehicle for experimental evaluation
- Devise schedules that ensure fast, efficient transfers
  - Defining the role of forwarding is the crux of this work
  - Schedules are input of Multicast
- Gather feedback
  - Transfer performance can be monitored and reported

Performance Measurements

- Top level network view reveals performance
  - Network measurements can give point-to-point results
- Active probes discern current conditions
  - Measure latency, bandwidth, etc.
- Two basic approaches:
  - Live probes
    - On-demand requests between points
  - Historic Measurements
    - Past performance; finite time spans
- Prediction?

Historic Performance Measurements

- 'iperf' results from PlanetLab
  - Readily available, reasonably complete
- Why Bother?
  - Information is 'good enough'
  - Minimal collection / analysis = faster schedule creation
  - Ideal for dynamic applications; depends on sensitivity
- Could we do better?
  - Live measurements are hard to scale
  - Small probe cannot reveal underlying network behavior

iperf Results

- The iperf results need time to become 'complete'
  - To avoid flooding and thus producing false positives
  - Performing ~360,000 tests (~600 machines) takes time
- Adequate schedules start with several weeks of data
  - Want to get understanding of unladen network performance
  - Collect max value; gives 'best' picture of network
- Further processing is required

Preprocessing

- Measurements represented as matrix
- The accrued N*N matrix can be sparse
  - Hosts may be unavailable as algorithm proceeds
- Use heuristics to supplement existing data:
  - Exchange measurements between sites (use IP netmask)
    - Fill gaps with 'neighbors' observations
  - Elevate bandwidth between 'similar' hosts
    - Close machines = high bandwidth

Measurement Example

- Initial iperf data:

	1.A	*1.B*	*1.C*	*2.C*	*1.D*	*1.E*	*2.E*
1.A	10.43	1.44	1.78	1.79	2.37	1.52	1.45
1.B	0.06	5.44	0.07	0.07	*	0.06	0.05
1.C	*	3.46	3.57	*	5.53	3.73	*
2.C	6.6	1.43	7.65	2.34	*	*	3.7
1.D	*	1.36	1.79	1.71	2.34	1.52	1.51
1.E	*	9.91	*	3.57	*	2.34	*
2.E	*	*	3.66	*	3.15	6.06	*

- Results after processing:

	1.A	*1.B*	*1.C*	*2.C*	*1.D*	*1.E*	*2.E*
1.A	999	1.44	1.78	1.79	2.37	1.52	1.45
1.B	0.06	999	0.07	0.07	*	0.06	0.05
1.C	6.6	3.46	999	999	5.53	3.73	3.7
2.C	6.6	3.46	999	999	5.53	3.73	3.7
1.D	*	1.36	1.79	1.71	999	1.52	1.51
1.E	*	9.91	3.66	3.57	3.15	999	999
2.E	*	9.91	3.66	3.57	3.15	999	999

Schedule Creation

- After processing a distribution schedule is created
  - Describes roles of nodes; forwarding or receiving
- A tree structure defines the Multicast
- Edge cost is inverse of “achievable bandwidth”
  - i.e. 1/bandwidth
  - Transforms initial measurement into transfer latency
  - Cost of entire path is defined by slowest link

Schedule Creation

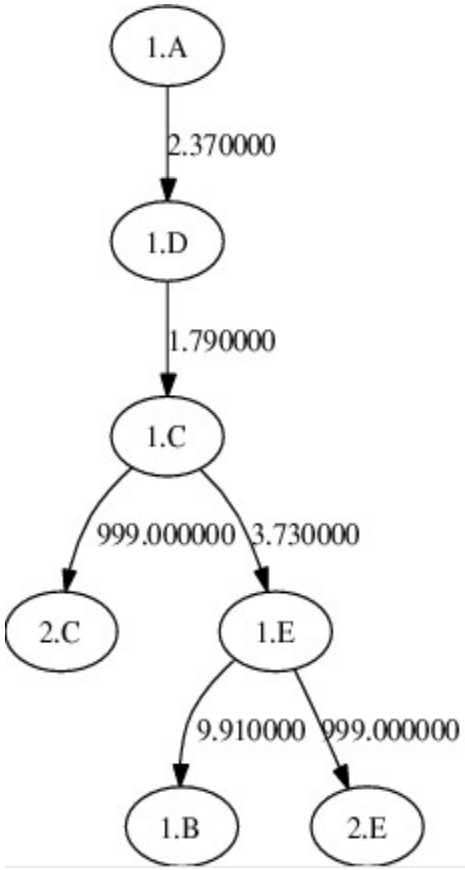
- To find 'best' of slowest links, borrow an AI technique
  - Adversarial approaches (Minimax) ensure best possible move at each stage
  - Minimize impact of poor connections
- Minimax generates a Minimum Spanning Tree (MST)
  - Tree is input to Multicast
- Similar to Prim's Algorithm and other greedy tree building algorithms

Minimax Example

- Using this processed data:

	1.A	*1.B*	*1.C*	*2.C*	*1.D*	*1.E*	*2.E*
1.A	999	1.44	1.78	1.79	2.37	1.52	1.45
1.B	0.06	999	0.07	0.07	*	0.06	0.05
1.C	6.6	3.46	999	999	5.53	3.73	3.7
2.C	6.6	3.46	999	999	5.53	3.73	3.7
1.D	*	1.36	1.79	1.71	999	1.52	1.51
1.E	*	9.91	3.66	3.57	3.15	999	999
2.E	*	9.91	3.66	3.57	3.15	999	999

- Generates this tree:



Minimax Algorithm

- Small differences in measurements may exist
  - Add a value to establish acceptance 'threshold'
  - i.e. values within ϵ are the same
- Slight modifications to basic algorithm:
  - Measurements with slight variations are considered similar
    - Defined as 'epsilon' (ϵ)
  - i.e. values within ϵ are the same
  - We always know the root of the tree
    - Poor performance by root results in forwarding to suitable nodes

Experiment Details

- Experimental set includes ~50 hosts on PlanetLab
- Measurement data collected over 3 week time span
- A control broadcast along with a set of Logistical Multicast operations were repeated numerous times
  - Data size was varied between 1 and 16 megabytes
  - ϵ was varied between 0 and .3

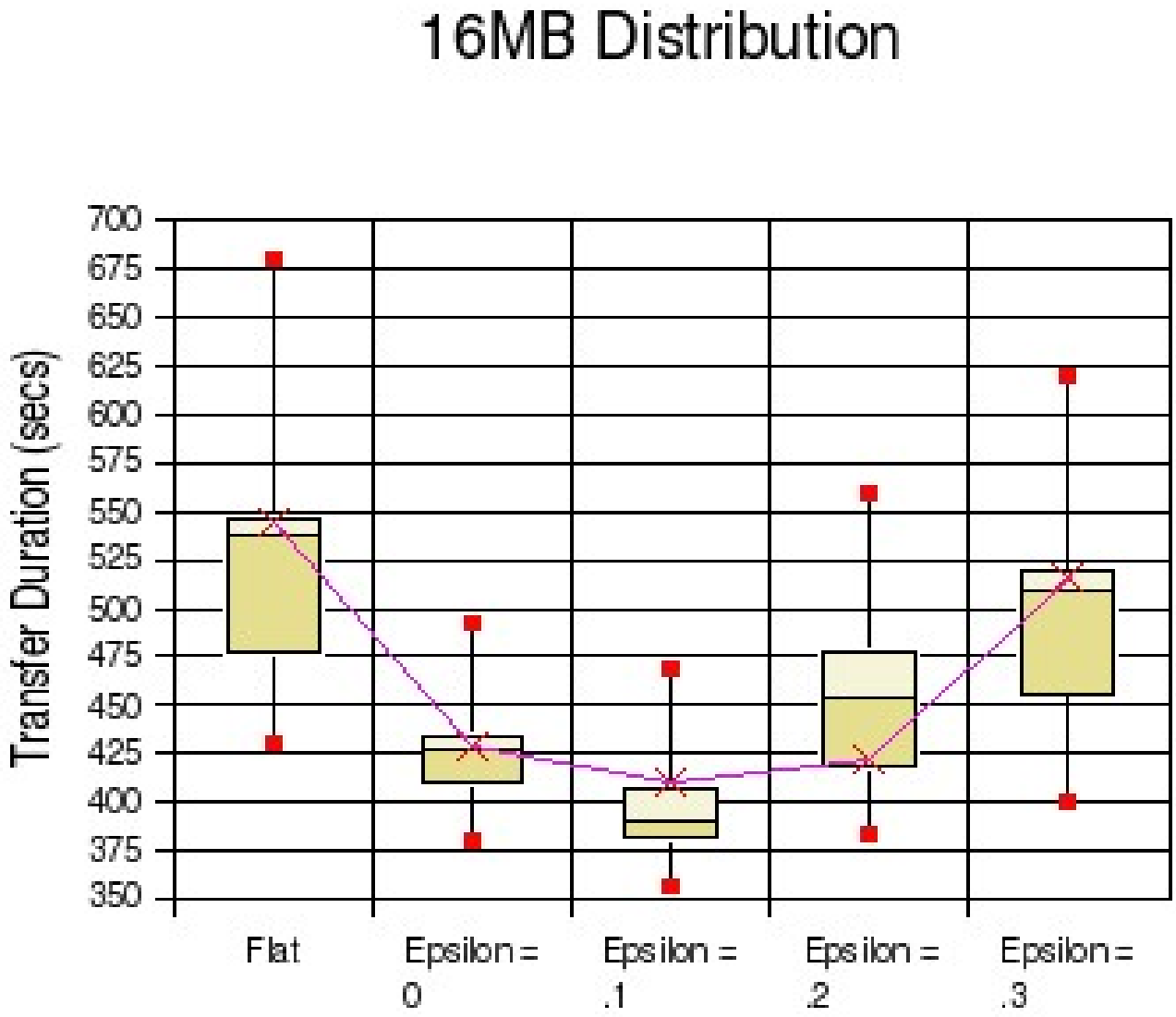
Experiment Details

- Host sets remained consistent over tests
  - Nodes verified for 'liveness'
  - Unavailable nodes replaced with proxies from same subnet
- Statistics gathered per run:
  - Total Transfer Time
  - Node Branching
  - Tree Depth

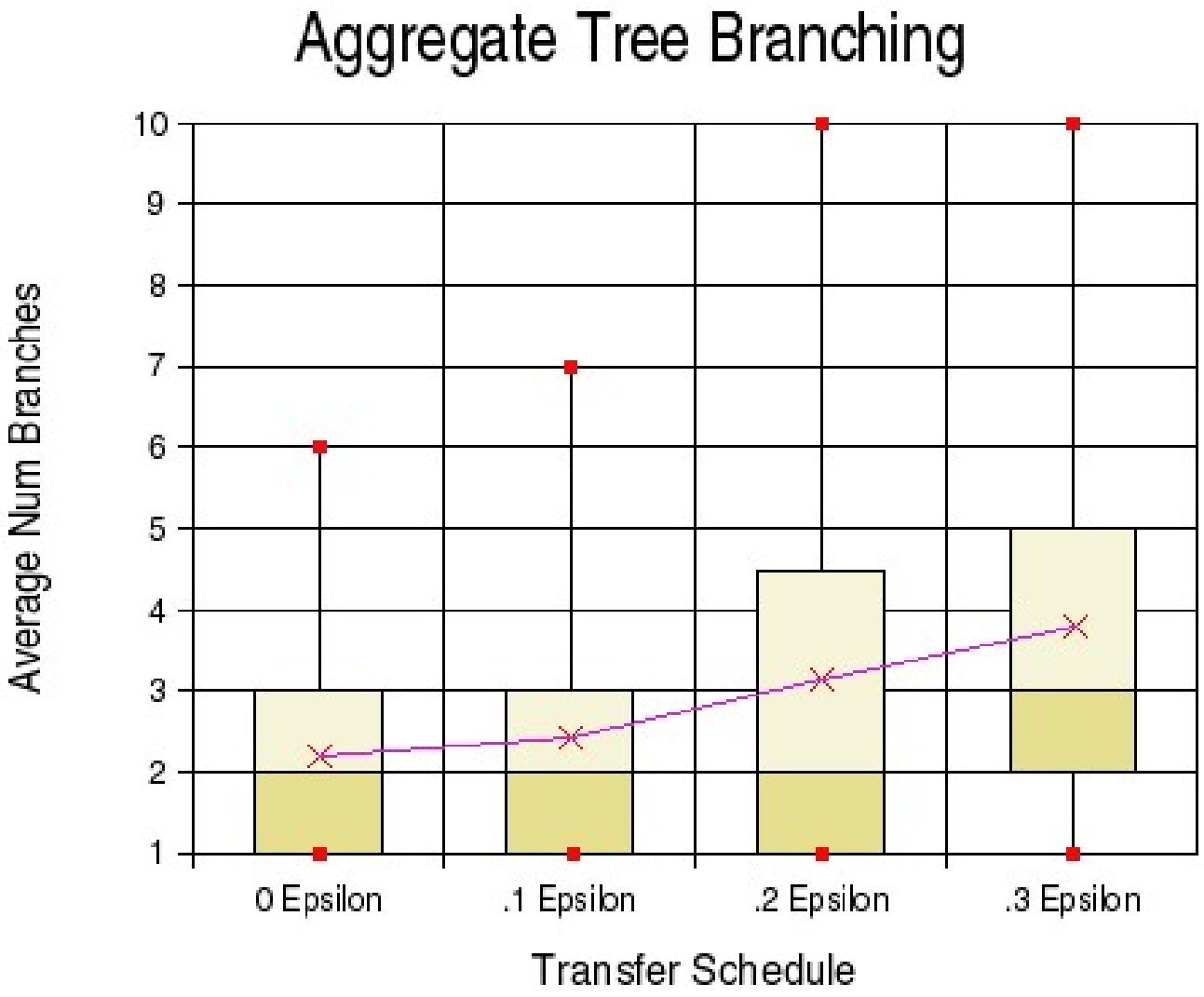
Experimental Results

- Broadcast performed the worst (naturally)
- Multicast was variable; depends on ϵ
- As ϵ increased, tree branching increased
  - ϵ expands window of 'equivalence'; allows broader range of 'acceptable' measurements
- As ϵ increased, tree depth decreased
  - Adding children to nodes decreases overall tree depth

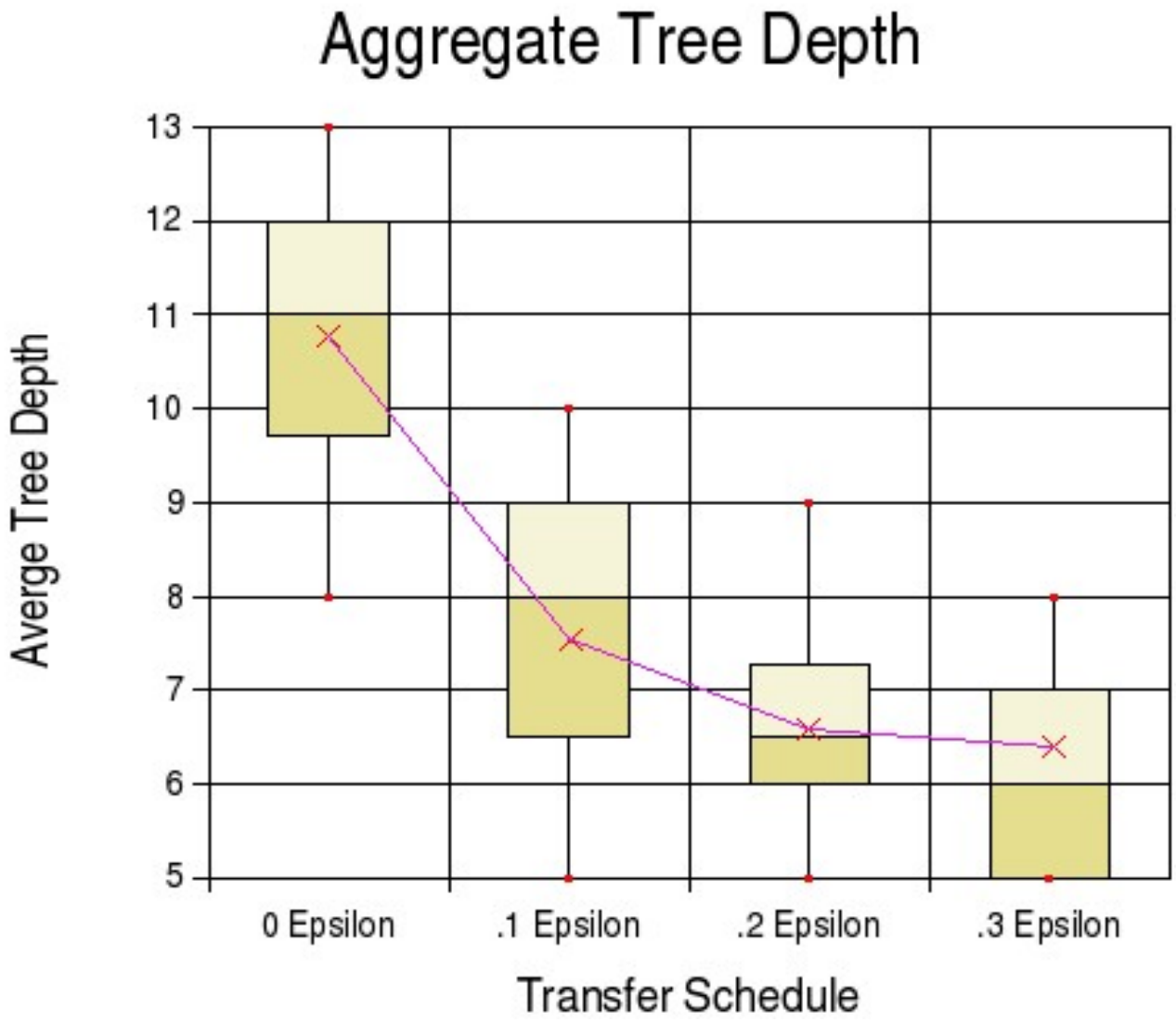
Experimental Results



Experimental Results



Experimental Results



Conclusion

- Data movement is important, and needs to be considered when designing Grid-ware.
- Efficient movement comes from observation of the network
- Historic measurements are 'good enough', and offer benefits over hard to scale live techniques
- Properly designed schedules outperform lesser informed alternatives

Future Plans

- Active / Passive Hybrid
  - Passive measurements (SNMP router data) could provide better understanding of network
  - Combinations could yield promising results
- Aggregation of other available metrics
  - Combinations of various measurements widen the benefit over any one technique
  - Repositories of information could speed the collection and generation process

Related Work

- Initial Logistical Multicast Work
  - M. Beck et al. "*An Exposed Approach to Reliable Multicast in Heterogeneous Logistical Networks*"
- Networking Scheduling using MSTs
  - M. Swany. "*Improving Throughput for Grid Applications with Network Logistics*"
  - M. Swany et al. "*Network Scheduling for Computational Grid Environments*"
- Application Level Multicast
  - S. Birrer, et al. "*FatNemo: Building a Resilient Multi-Source Multicast Fat-Tree*"
  - J. Jannotti, et al. "*Overcast: Reliable Multicasting with an Overlay Network*"