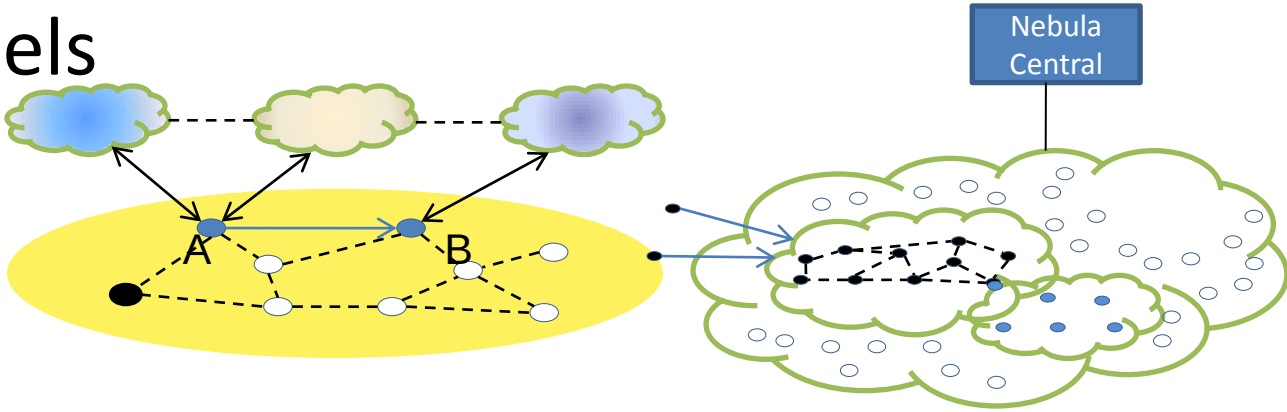
Living on the Edge: Scheduling Edge Resources Across the Cloud

Jon Weissman

Department of CS&E
University of Minnesota

Outline

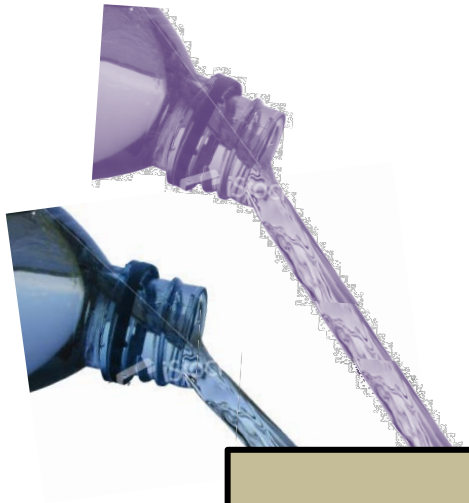
- Motivation
- New cloud models
  - Proxy Cloud
  - Nebula Cloud
- Summary



The “Standard” Cloud

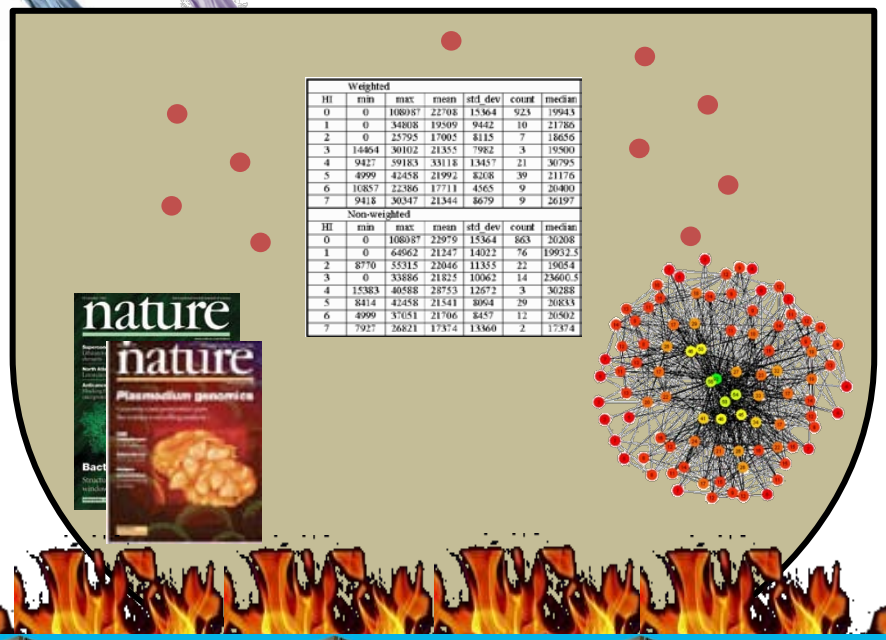
Data in

Results out



“No limits”
- Storage
- Computing

on-demand scaling
transparency



Weighted						
HI	min	max	mean	std dev	count	median
0	0	108087	22708	15364	923	19943
1	0	34808	19509	9442	10	21786
2	0	25795	17005	8115	7	18656
3	14464	30102	21355	7982	3	19500
4	9427	59183	33118	13457	21	30795
5	4999	42458	21992	8208	39	21176
6	10857	22386	17711	4565	9	20400
7	9418	30347	21344	8679	9	26197
Non-weighted						
HI	min	max	mean	std dev	count	median
0	0	108087	22979	15364	863	20208
1	0	64962	21247	14022	76	19932.5
2	8770	55315	22046	11355	22	19054
3	0	33886	21825	10062	14	23600.5
4	15383	40588	28753	12672	3	30288
5	8414	42458	21541	8064	29	20833
6	4999	37051	21706	8457	12	20502
7	7927	26821	17374	13360	2	17374

Appealing Features

- Consolidation
- Economies of scale
- Strong locality
  - (data <-> computing) => great for analytics
- Novel sharing platform
  - data/state and applications => gaming, web 2.0
- Packaging
  - Pour all tools/code/data into a VM
- Abstraction: connect, pour, collect

Cloud Landscape 50K Feet

- Data clouds
  - S3, SkySurvey, GoogleHealth
- Compute clouds
  - EC-2, IronScale
- Service clouds
  - Gmail, Gmaps, Google-earth
- Specialization
  - Non-functional: security, reliability, SLAs, cost

Confluence

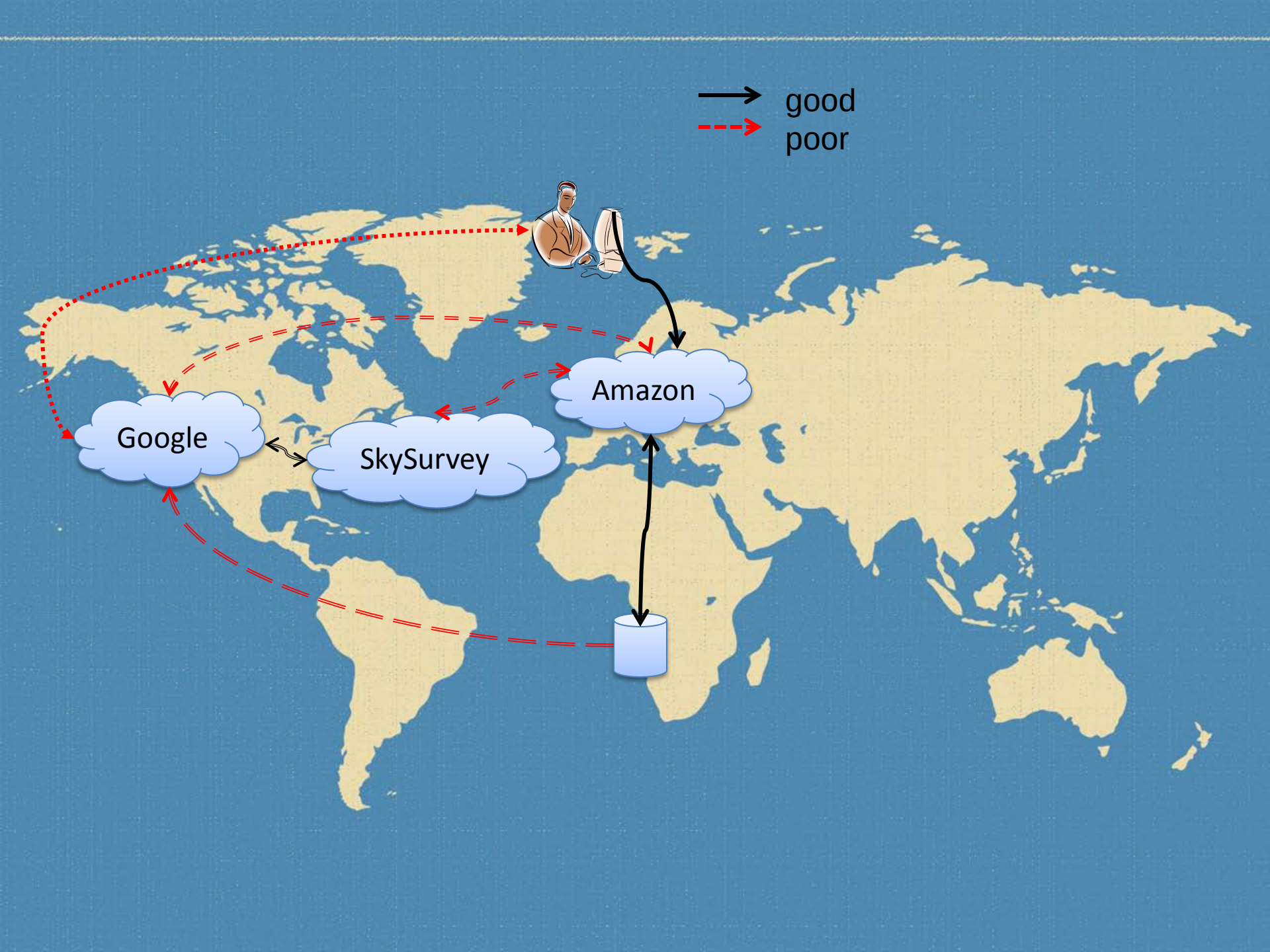
- Cloud diversity and specialization =>
- (1) no single cloud model will rule
- (2) new distributed models are attractive
- (3) emerging applications will utilize multiple clouds - “multi-cloud” applications

Multi-Cloud Applications

- Distributed data mining
  - Ex: weather data + commodity prices
- Scientific workflows
  - Ex: life science: GenBank<->BLAST<->PubMed, …
- Mashups
  - Ex: GoogleEarth + CDC pandemic data
- Multi-cloud parallel frameworks
  - Ex: MapReduce across multiple data centers

Cloud Obstacles

- Clouds are centralized but the world is distributed
- What is distributed?
  - clouds, data, users
- Distribution leads to performance, reliability, and security bottlenecks


good
poor
Amazon
Google
SkySurvey

Idea

- Make the cloud more “distributed”
  - “move” it closer to data
  - “move” it closer to end-users
  - “move” it closer to other clouds

- How?
  - exploit the rich collection of edge computers
  - volunteers (P2P, @home)
  - scheduling and resource selection are key

Two Projects

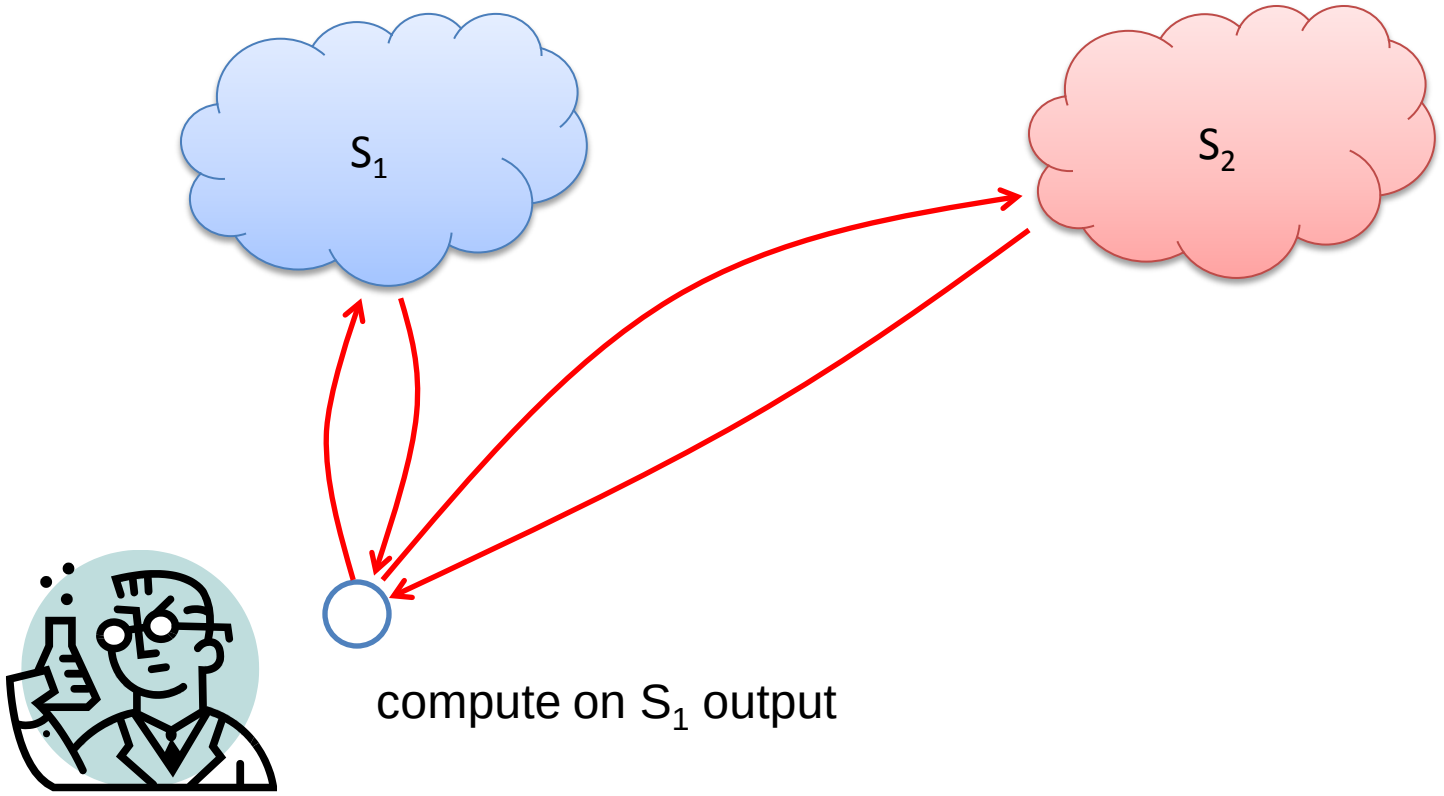
- Augment the Cloud with a proxy network
- Re-factor the Cloud to a more distributed platform

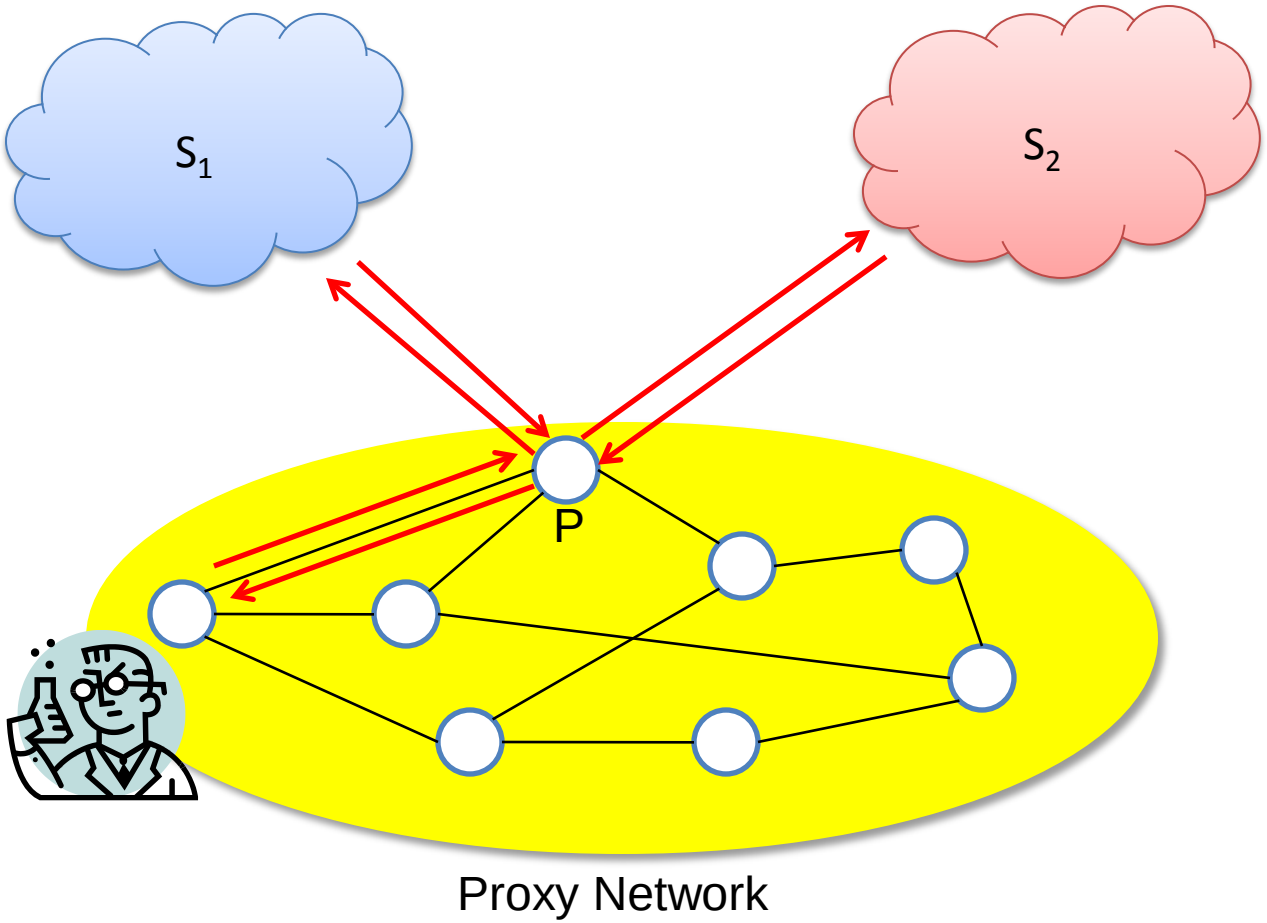
Proxy Network

Proxy Network

- Promote
  - cloud <-> end-user locality
  - cloud <-> cloud locality
- Via edge nodes

Locality Bottleneck



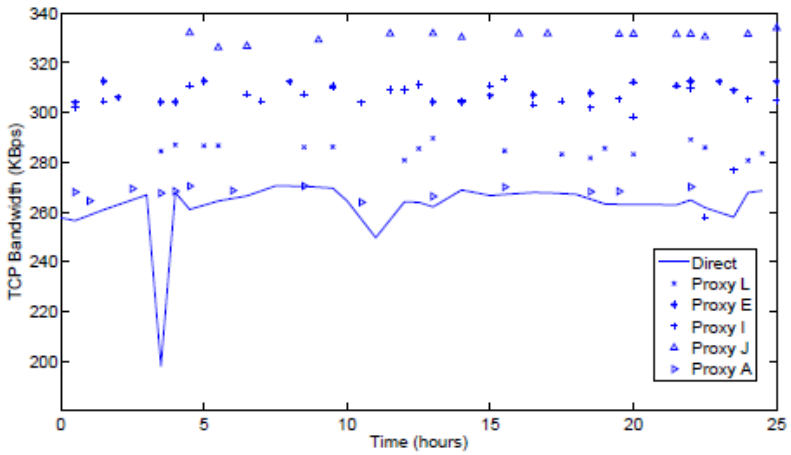


Proxy Network

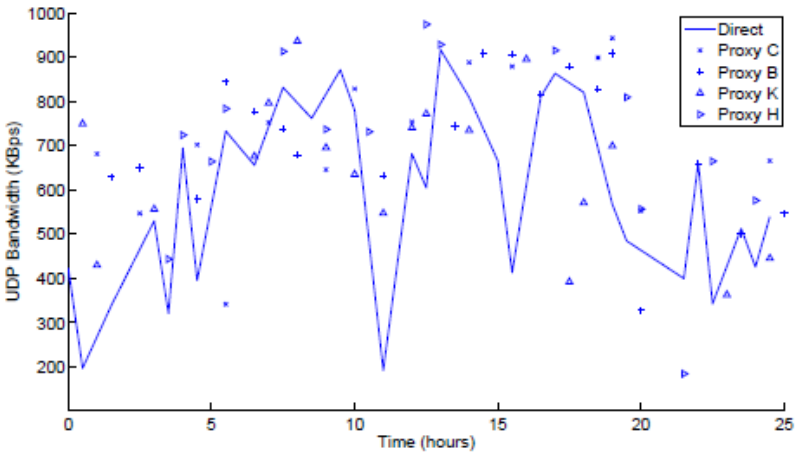
The Power of Proxies: Roles

- Cloud service interaction
  - Proxy as a client
- Routing
  - Proxy routes data to other proxies
- Computing
  - Proxy executes data operators: compress, filter, merge, mine, …
- Caching
  - Proxy caches data (from cloud, computations, …)

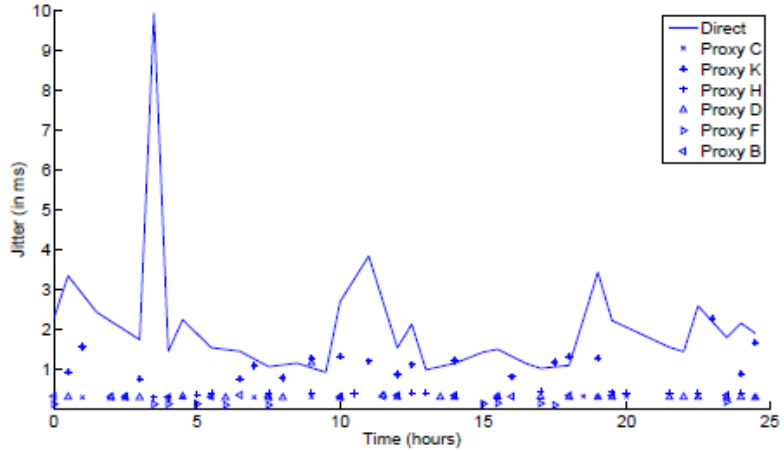
Networking Benefits



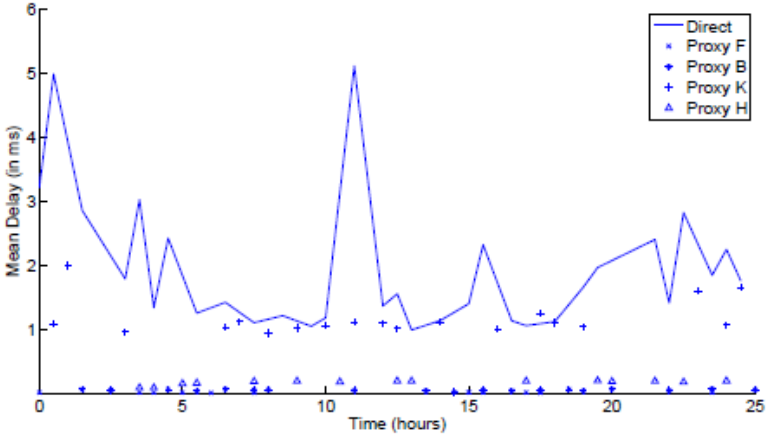
(a) TCP bandwidth (KBps) (Higher is better)



(b) UDP bandwidth (KBps) (Higher is better)



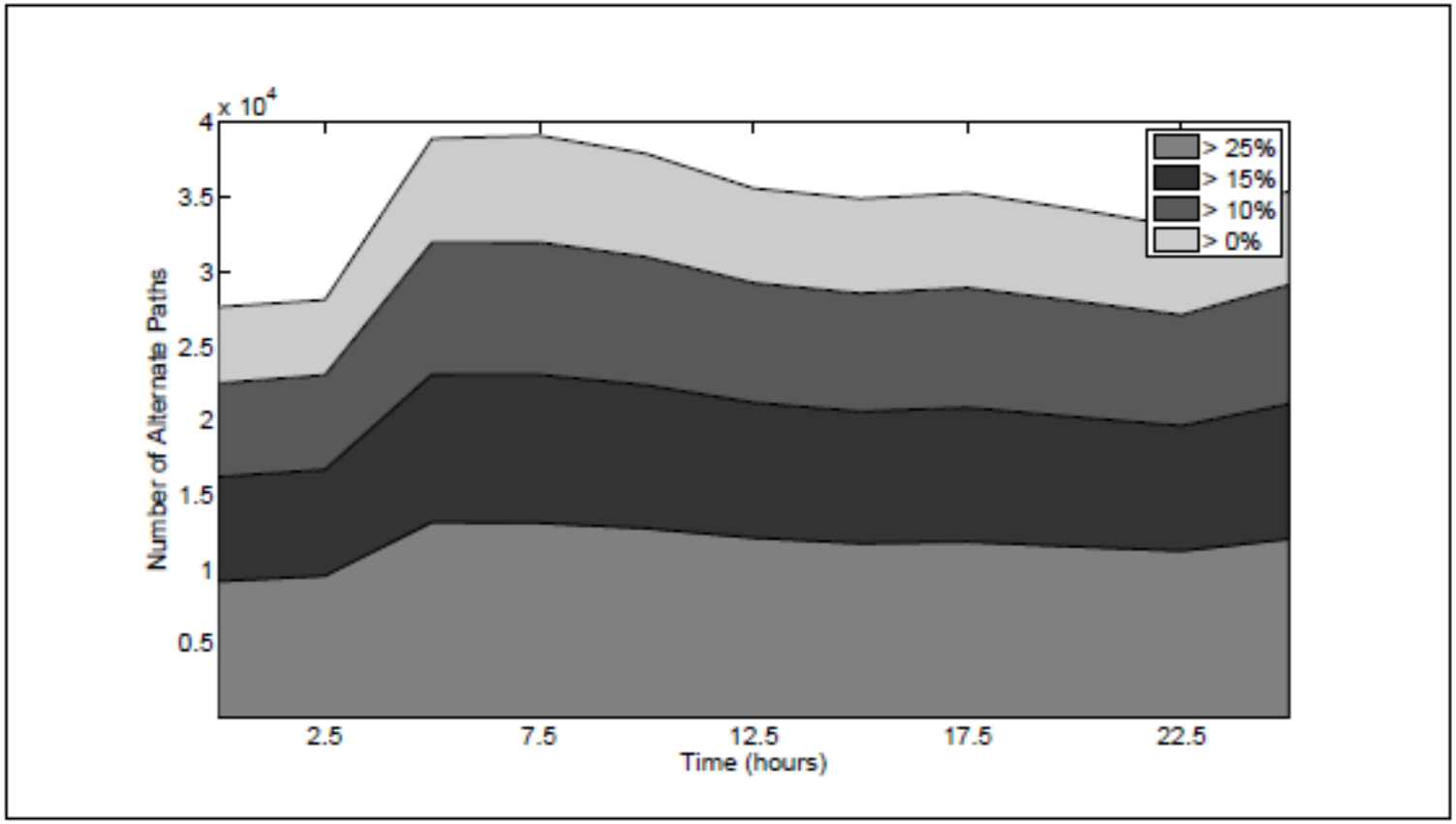
(c) UDP delay (ms) (Lower is better)



(d) UDP jitter (ms) (Lower is better)

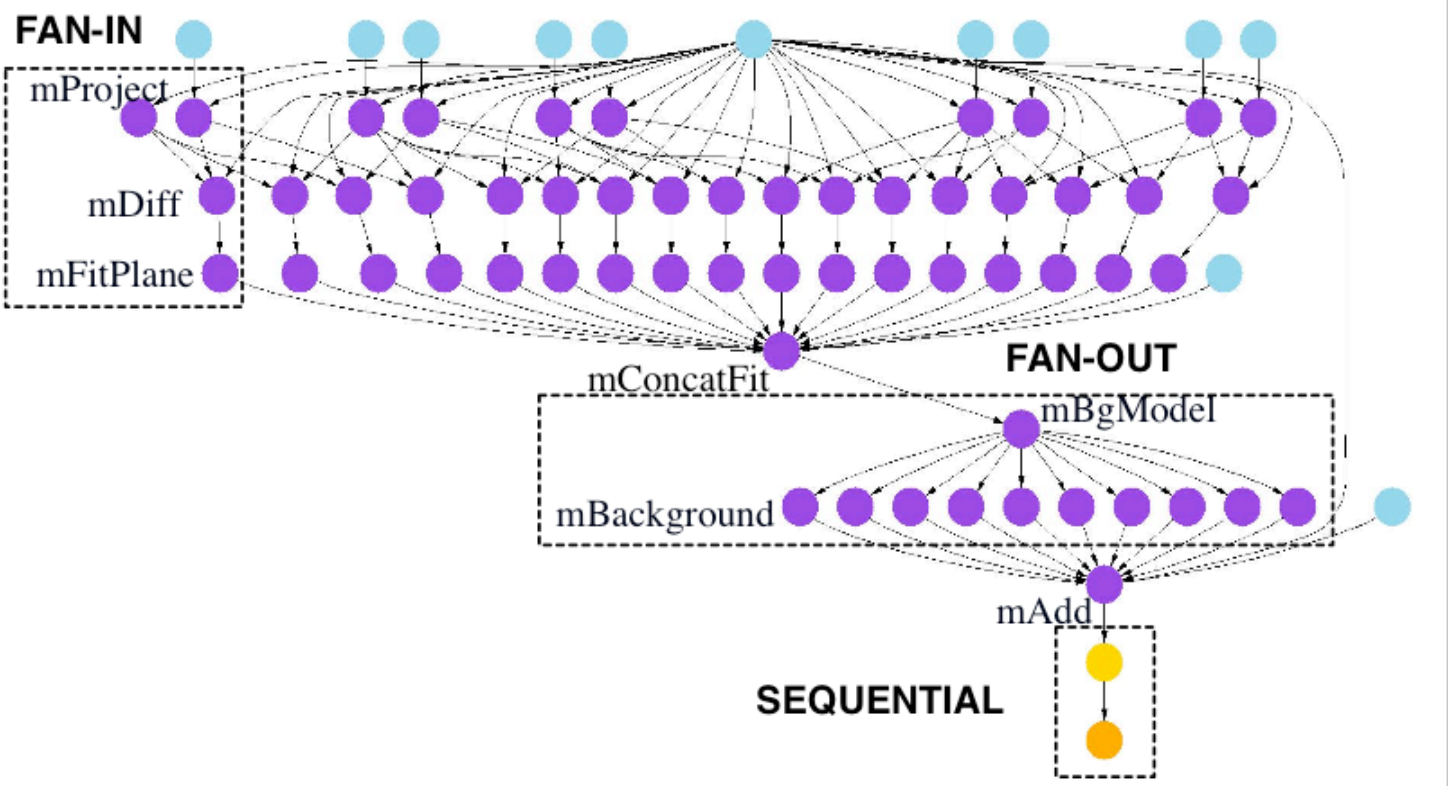
Network dashboard: proxy.cs.umn.edu

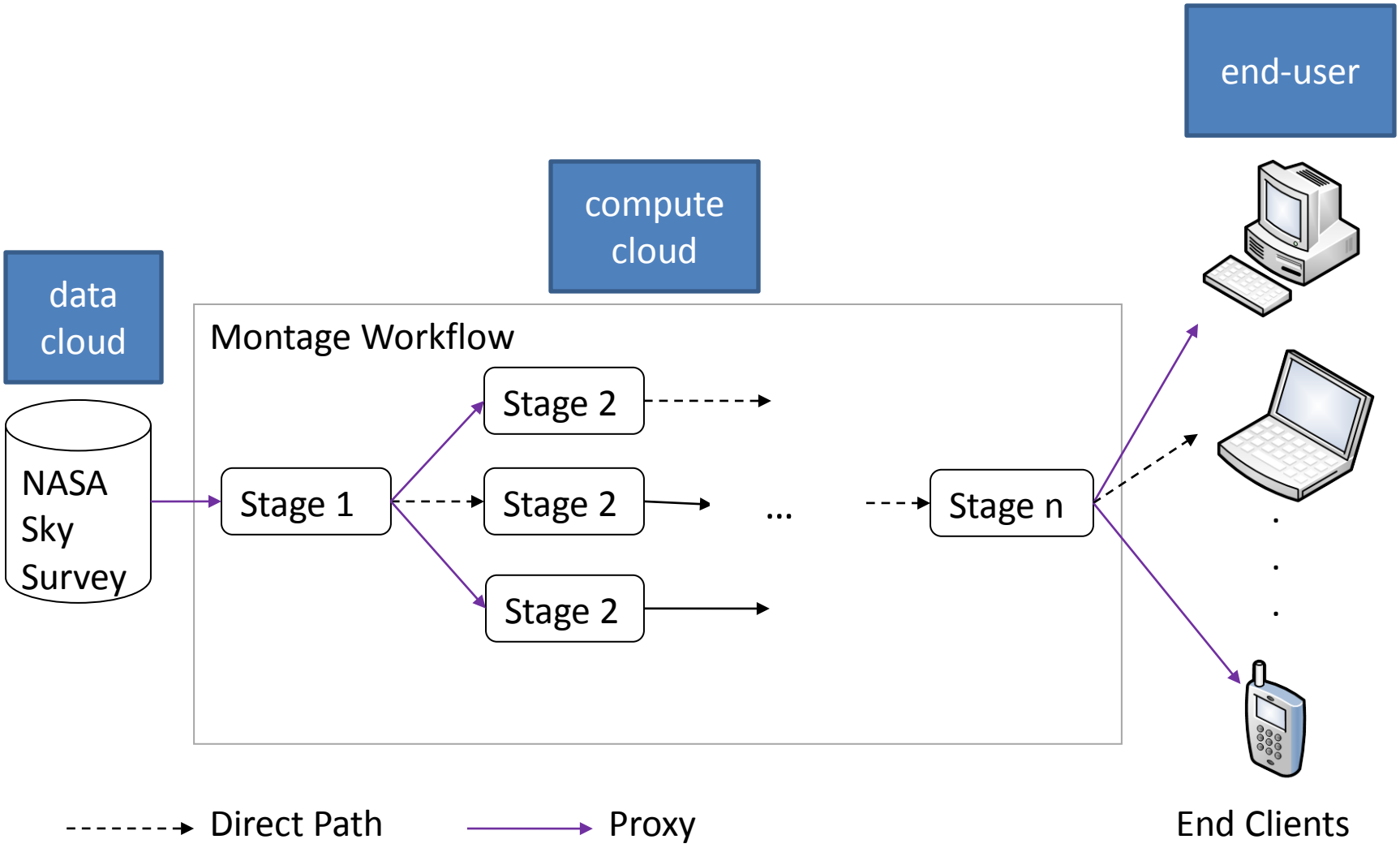
Many better data paths …



Over 70% of paths can be accelerated

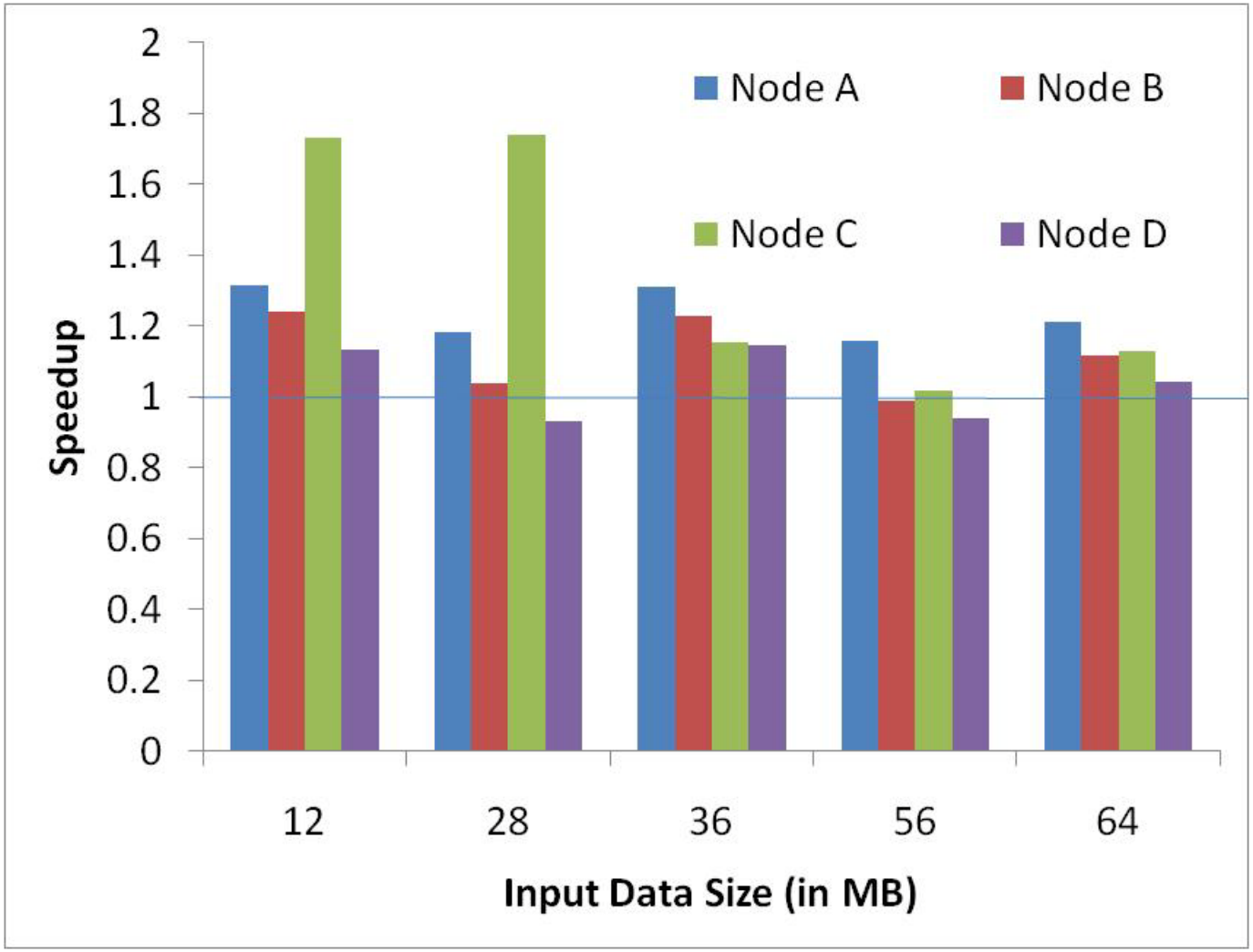
Example: Montage



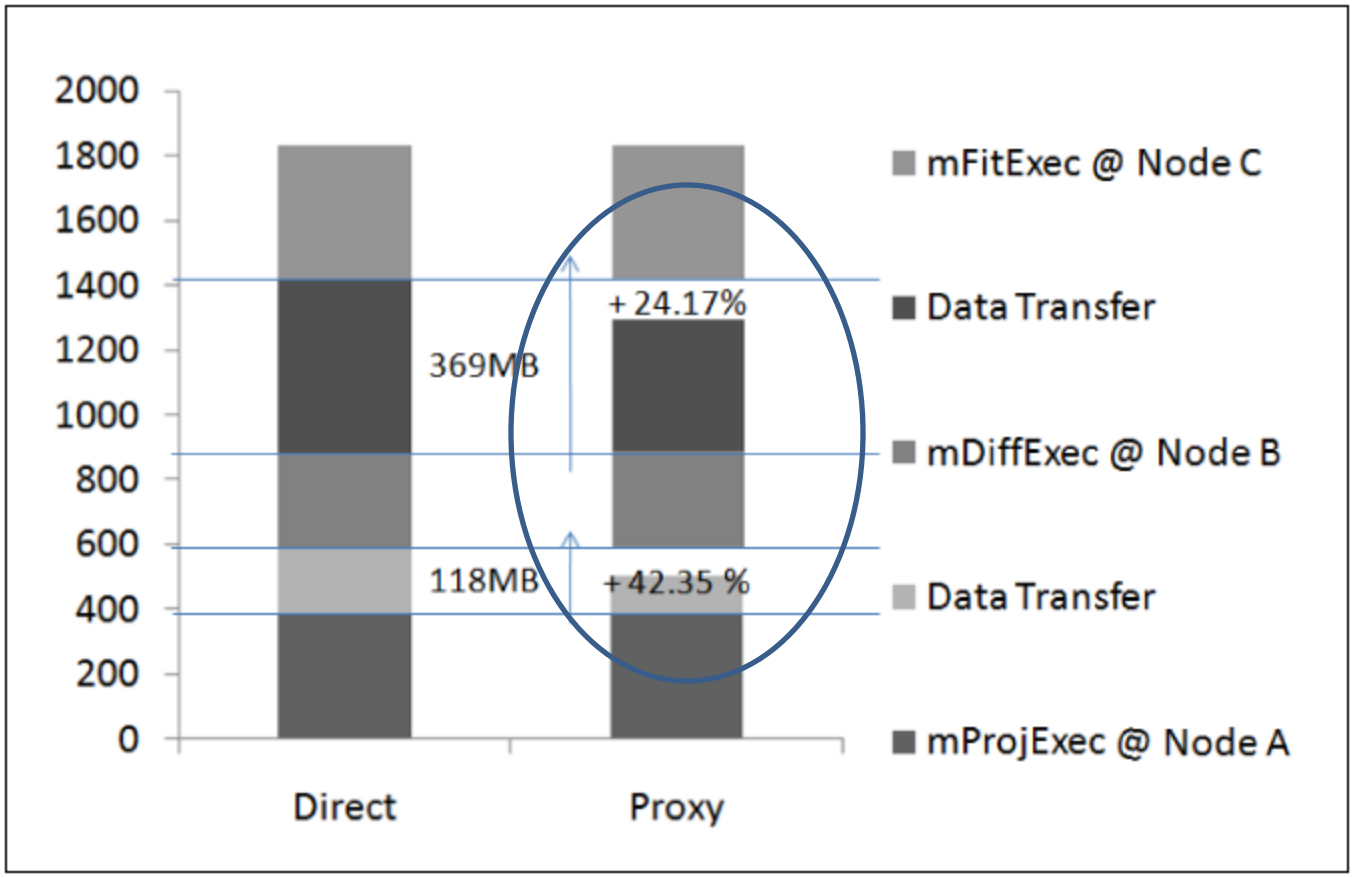


Accelerated at 3 different points

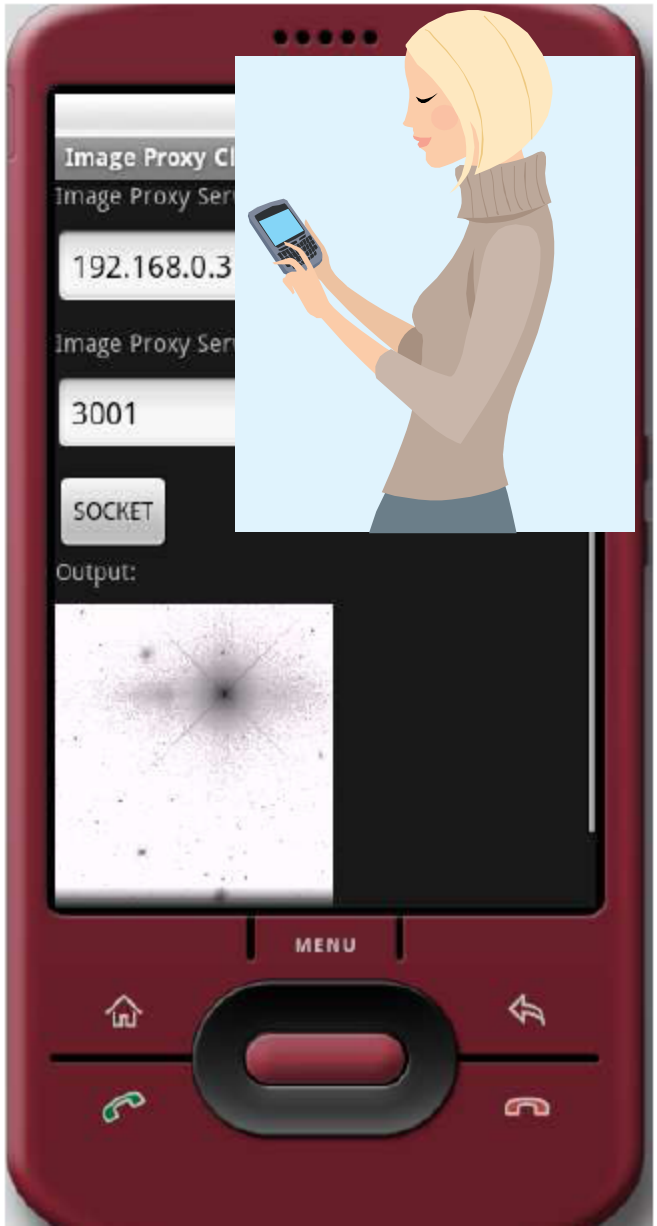
From SkySurvey to compute nodes



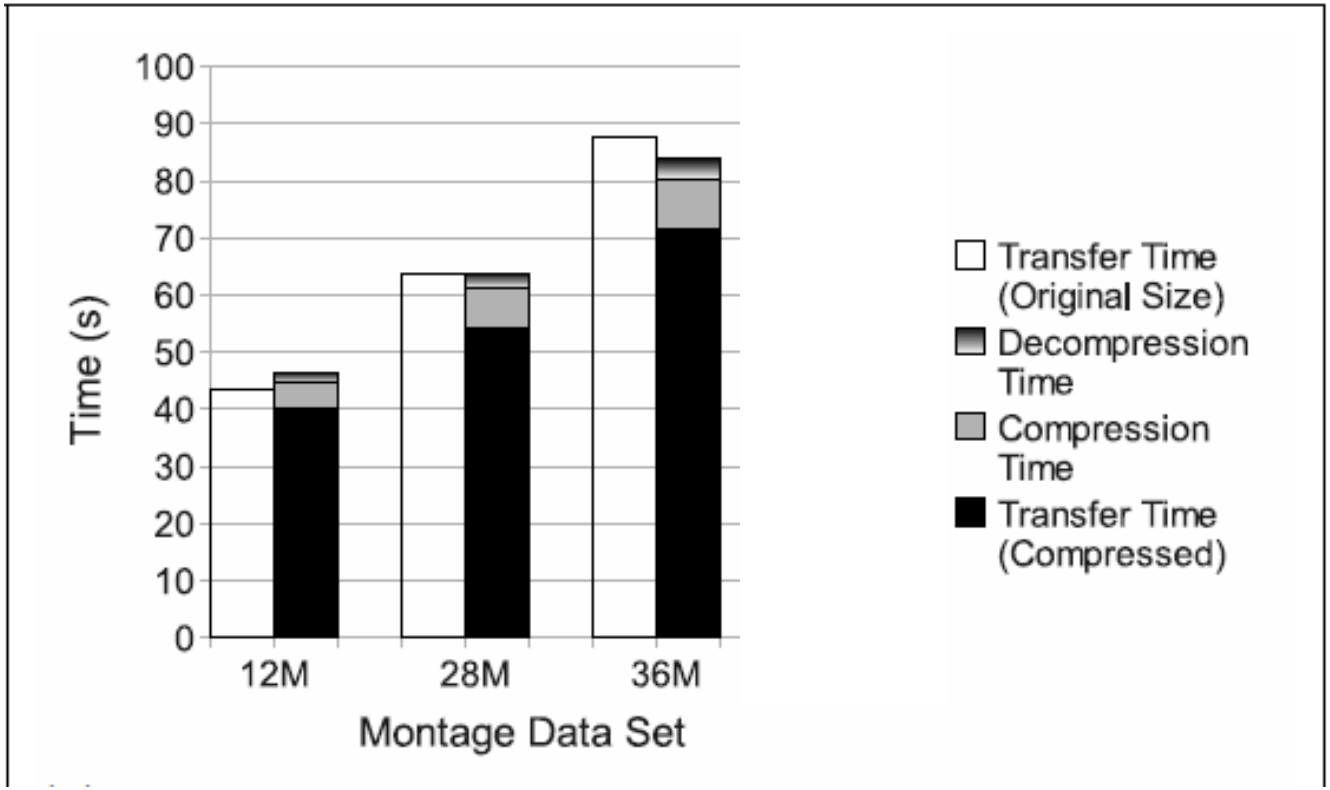
Inside Montage …



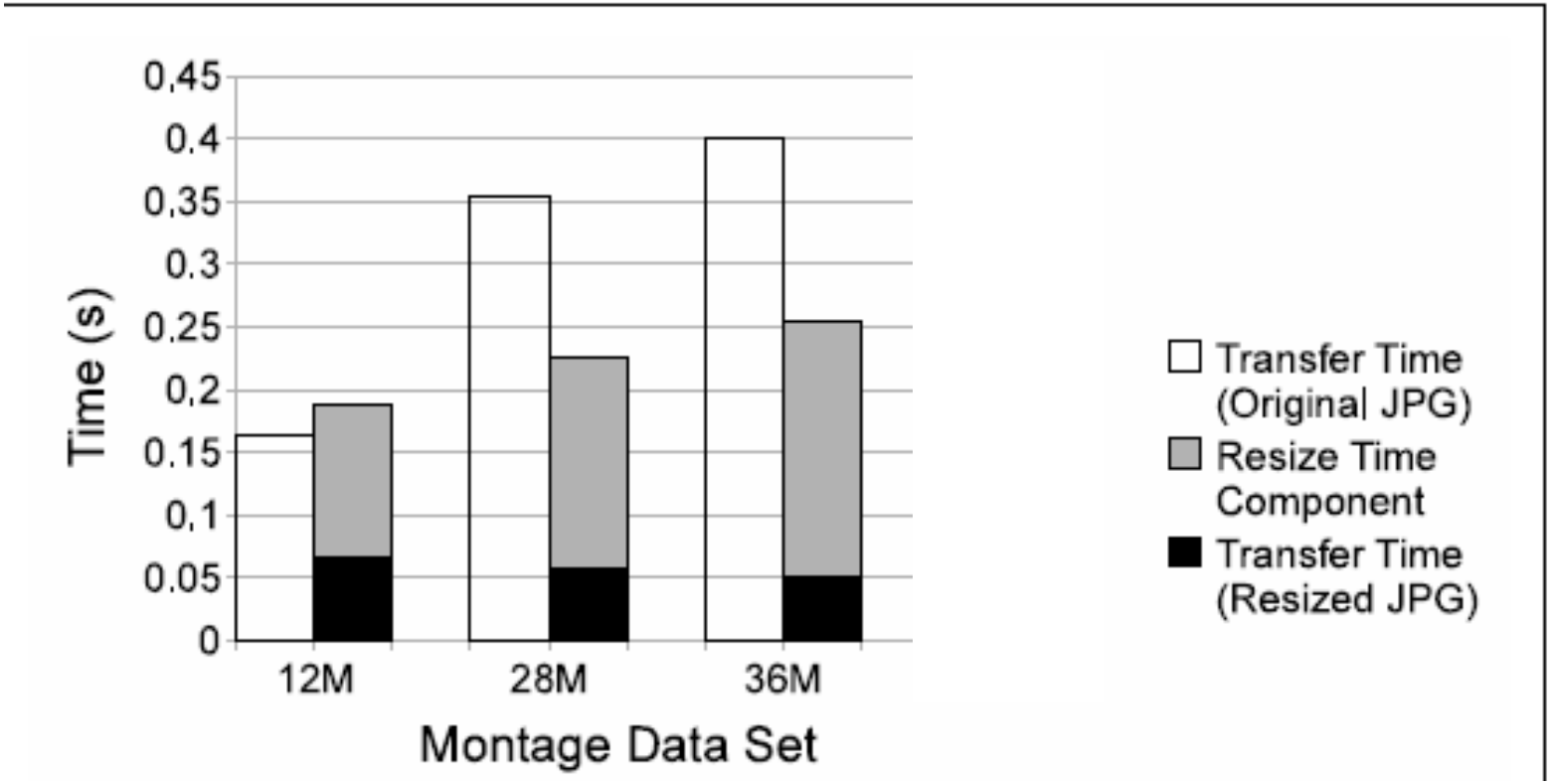
From Montage to the end-user



Desktop User



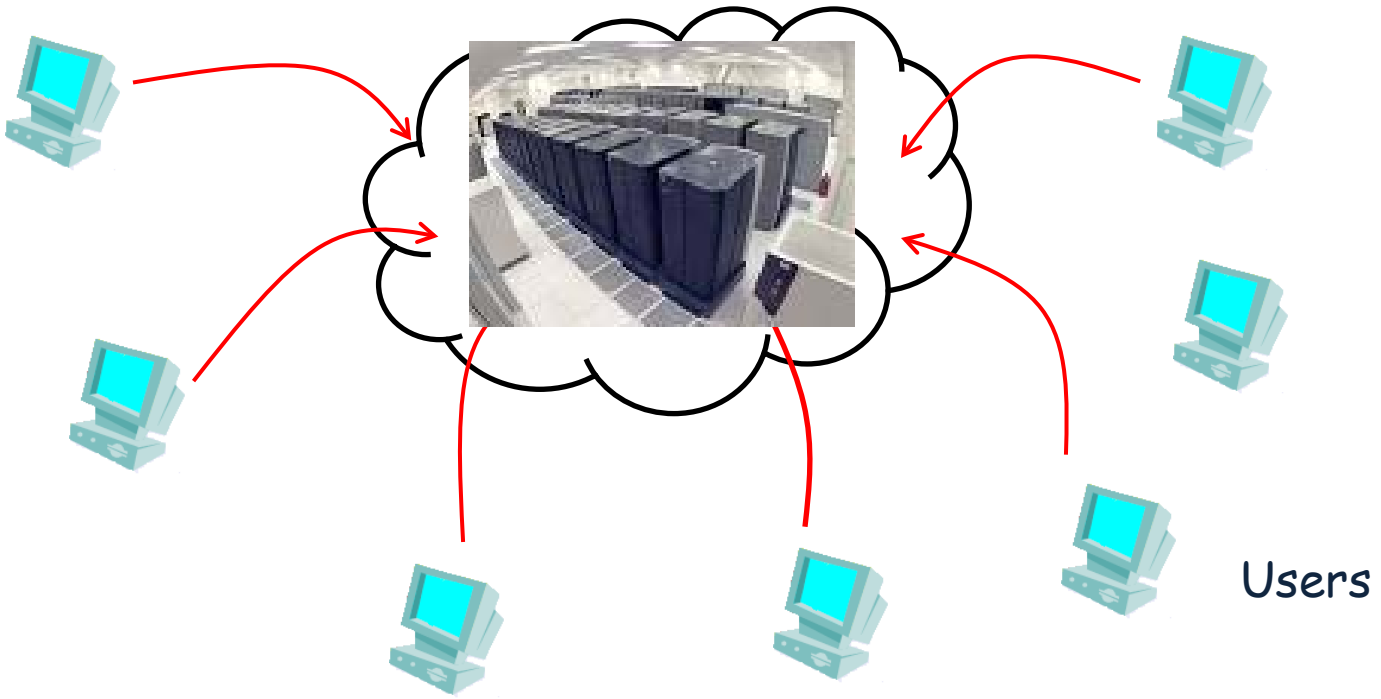
Mobile user


0,45
0,4
0,35
0,3
0,25
0,2
0,15
0,1
0,05
0
Time (s)
12M
28M
36M
Montage Data Set
Transfer Time (Original JPG)
Resize Time Component
Transfer Time (Resized JPG)

Nebulas

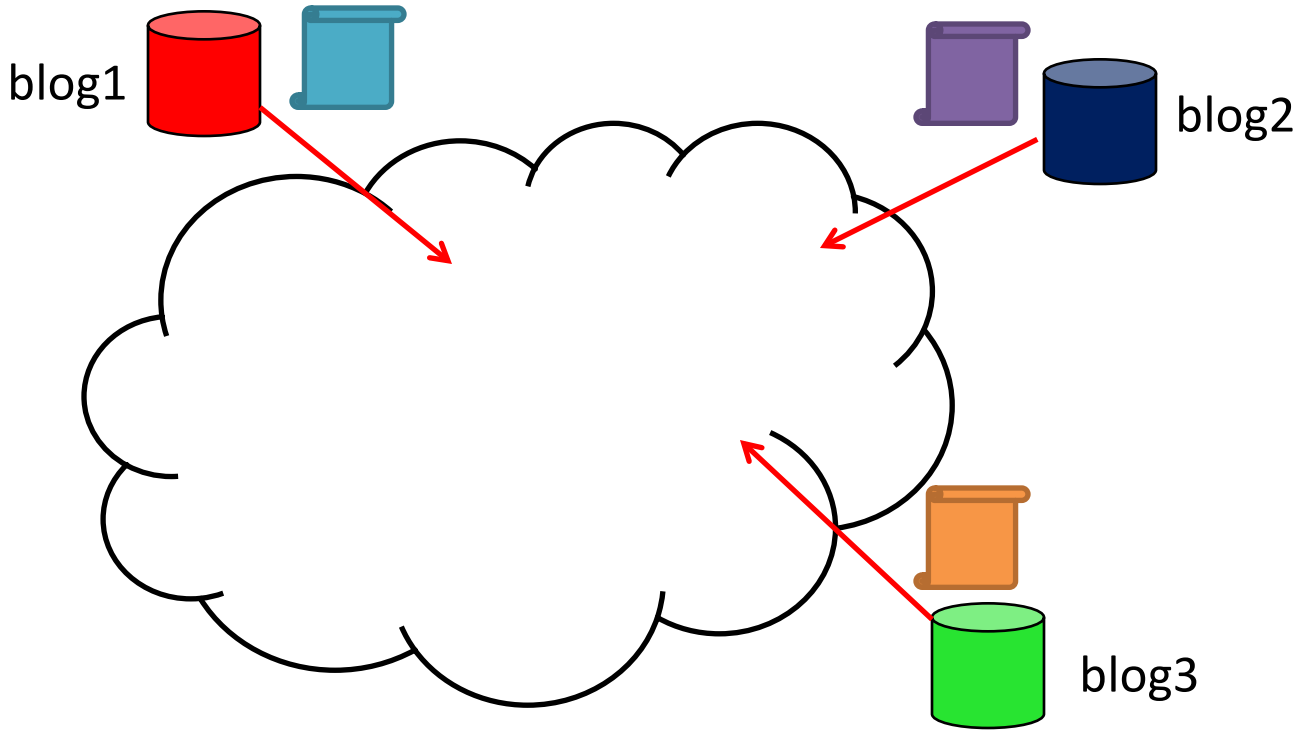
Current Cloud Model

Question: Are there applications that are not well suited to this cloud model?



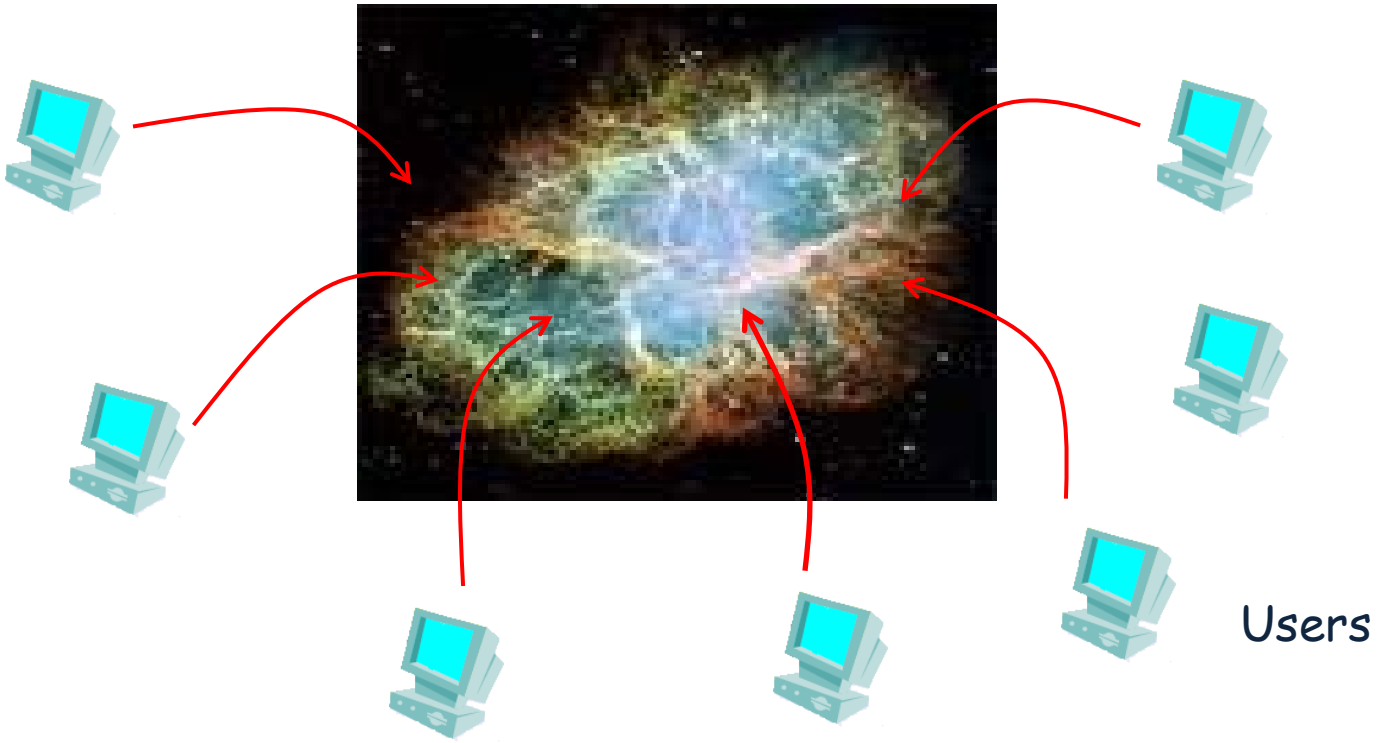
Dispersed-Data-Intensive Services

- Data is geographically distributed
  - Costly, inefficient to move to central location



Nebula

- Decentralized, less-managed cloud
  - Dispersed storage/compute resources
  - Low user cost



How is Nebula different from @home?

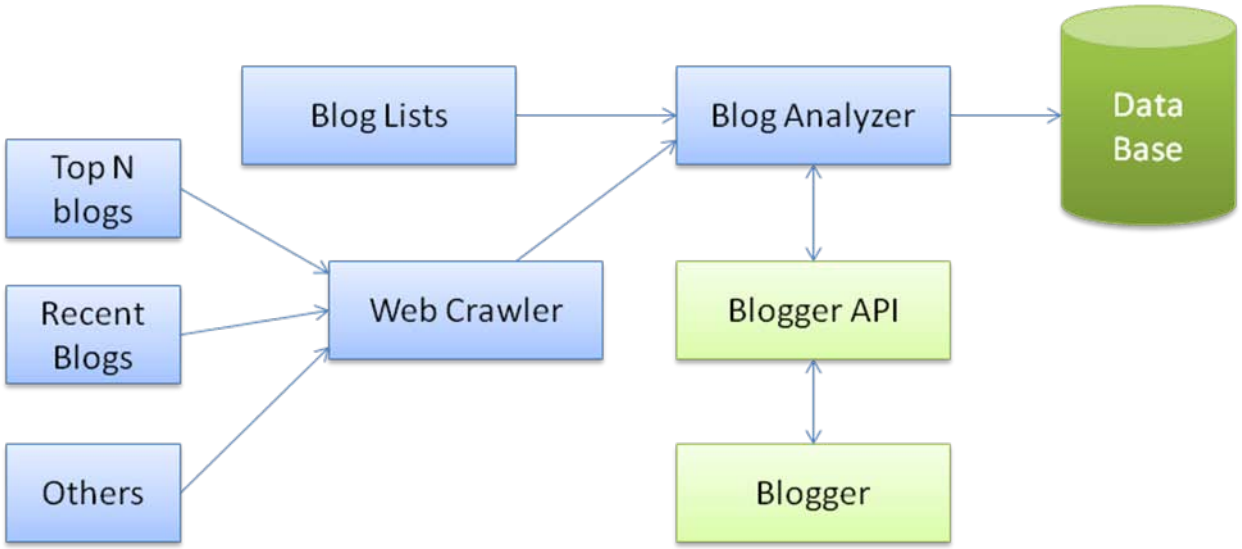
- Cloud-oriented services impose new requirements

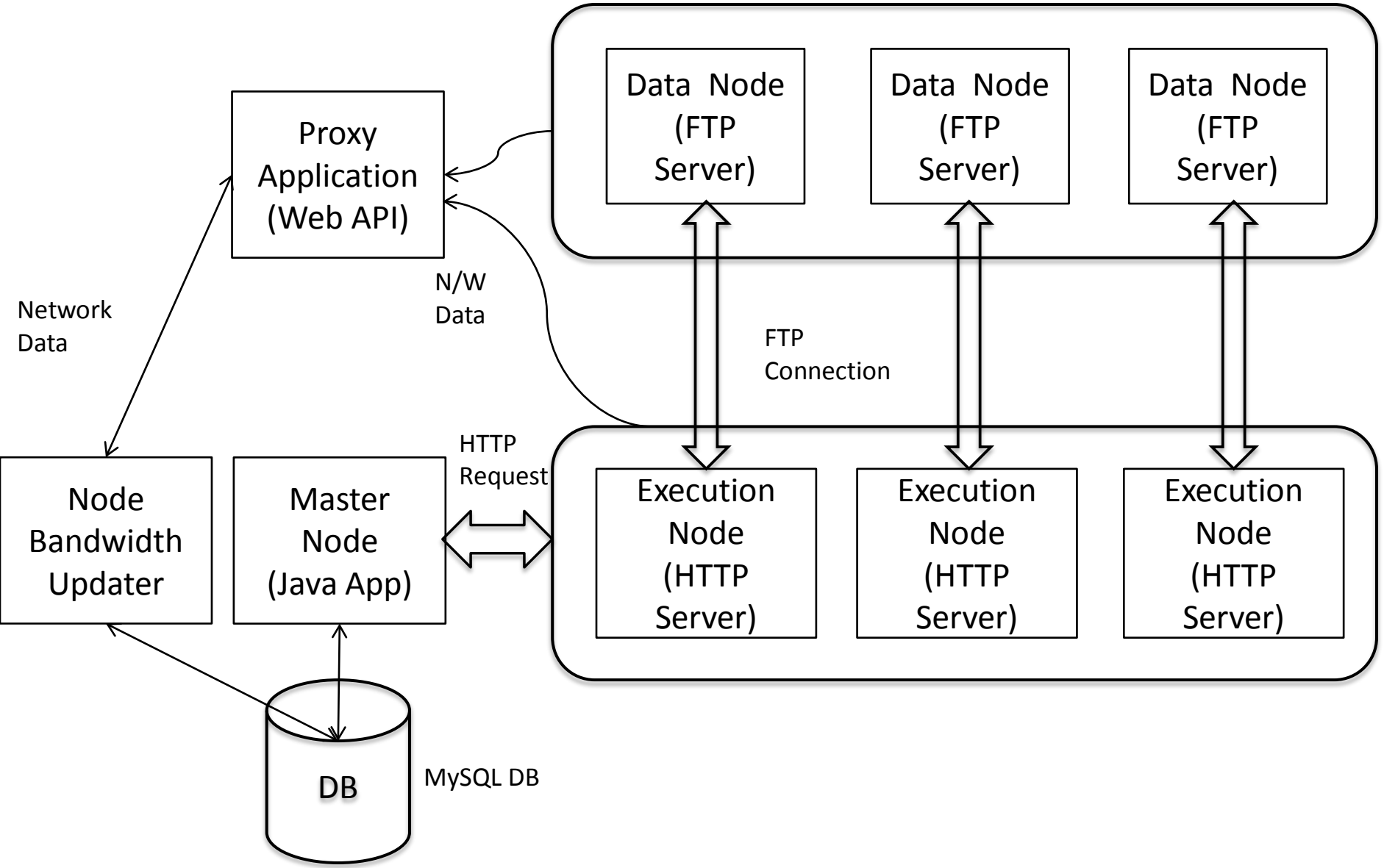
Requirement	Nebula	@home
Collective performance	High	None
Locality/Context-awareness	High	Low
Statefulness	High/medium	Low

Common Service Characteristics

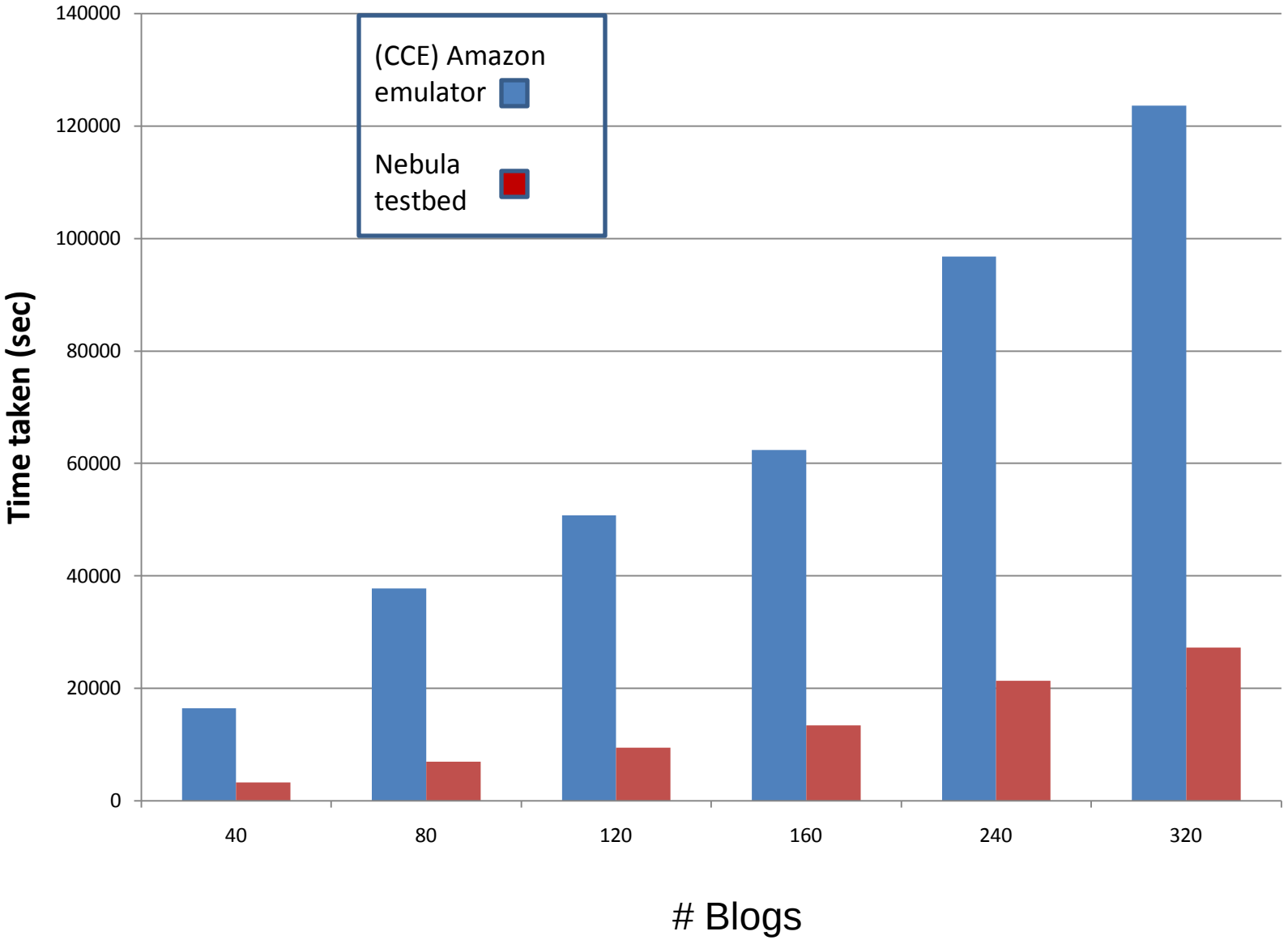
- Elastic resource consumption
  - Scale up/down based on demand
- Data/user distribution
  - Execution dependent on location of data/user
- Limited cost
  - May not want to pay for resources
- Weaker performance/robustness requirements
  - Some failures may be ok

Example: blog analysis

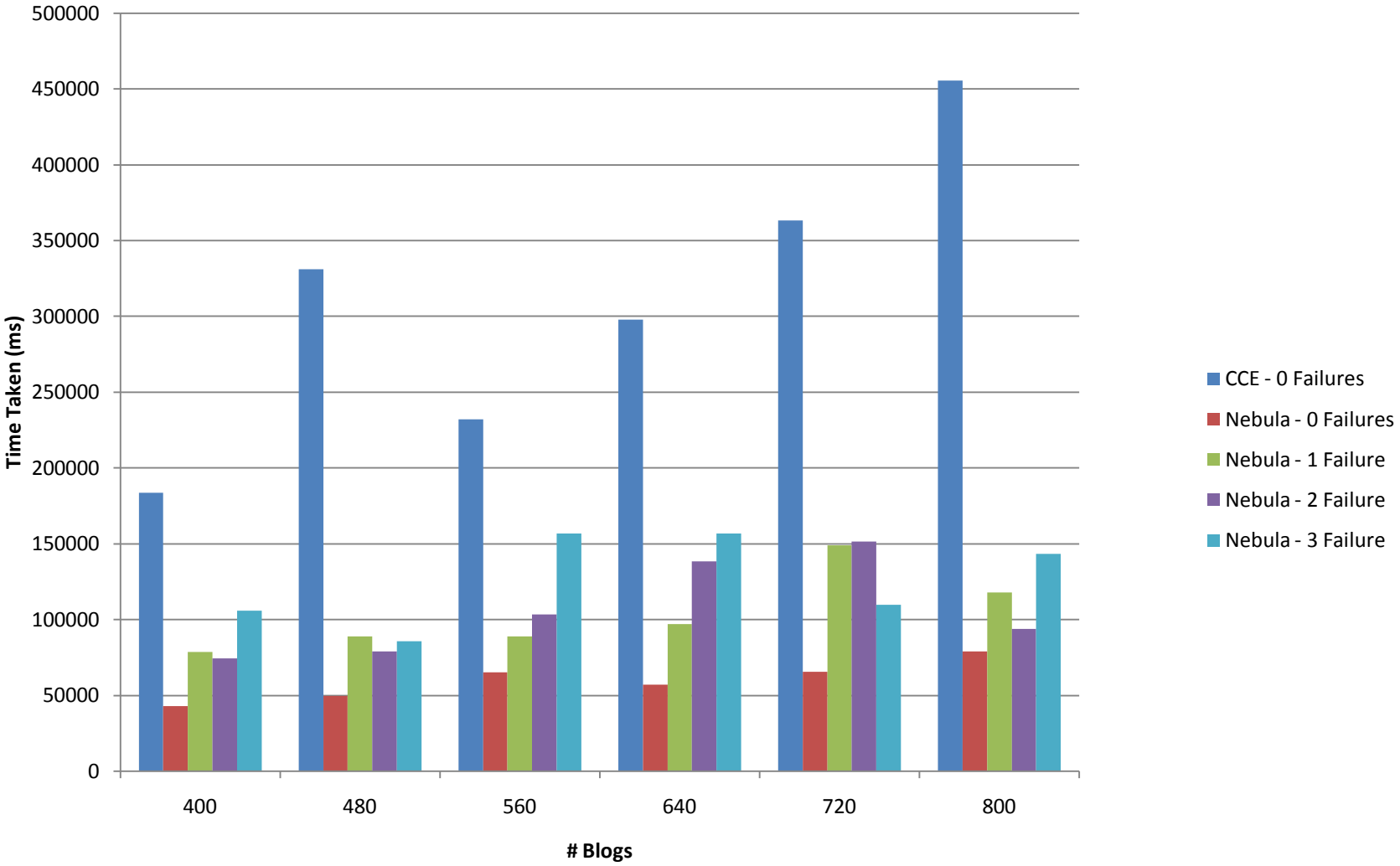



Data Node (FTP Server)
Data Node (FTP Server)
Data Node (FTP Server)
Proxy Application (Web API)
N/W Data
Network Data
FTP Connection
HTTP Request
Node Bandwidth Updater
Master Node (Java App)
Execution Node (HTTP Server)
Execution Node (HTTP Server)
Execution Node (HTTP Server)
DB
MySQL DB

Results


140000
120000
100000
80000
60000
40000
20000
0
Time taken (sec)
(CCE) Amazon emulator
Nebula testbed
40
80
120
160
240
320
Blogs

Results – failure



Summary

- Trends
  - Dynamic distributed data
  - Mobile users
  - Multi-cloud applications

- Our vision of the Cloud
  - locality of users, data, other clouds
  - Proxy solution: less disruptive
  - Nebulas: more radical
  - Complementary to conventional clouds