Many-Task Computing

Ioan Raicu
Computer Science Department
Illinois Institute of Technology

CS 595
Hot Topics in Distributed Systems: Data-Intensive Computing
September 8th, 2010

HTC: High-Throughput Computing

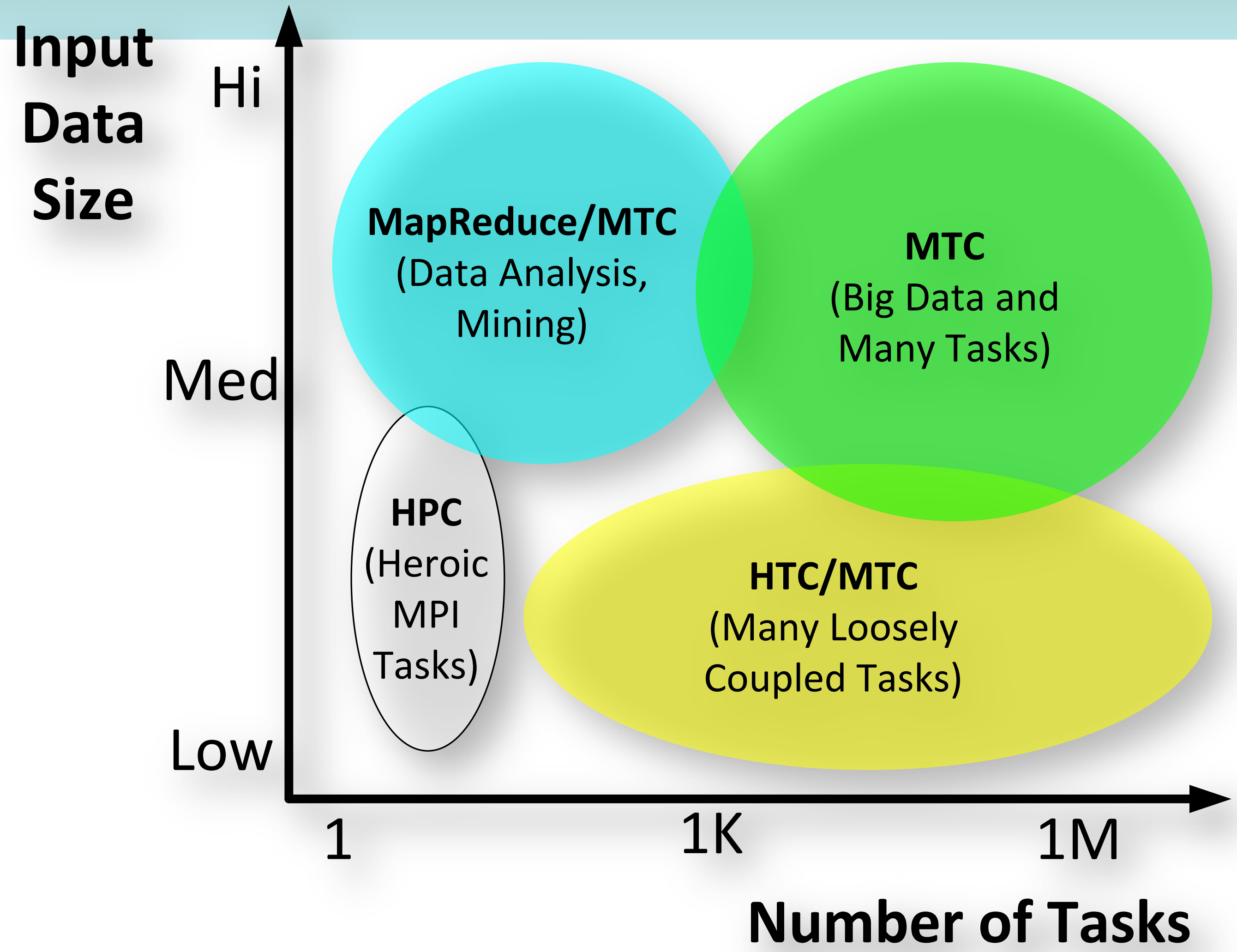
- Typically applied in clusters and grids
- Loosely-coupled applications with sequential jobs
- Large amounts of computing for long periods of times
- Measured in operations per month or years

MTC: Many-Task Computing

- Bridge the gap between HPC and HTC
- Applied in clusters, grids, and supercomputers
- Loosely coupled apps with HPC orientations
- Many activities coupled by file system ops
- Many resources over short time periods
  - Large number of tasks, large quantity of computing, and large volumes of data

[MTAGS08 Workshop] Workshop on Many-Task Computing on Grids and Supercomputers 2008
[SC08] "Towards Loosely-Coupled Programming on Petascale Systems"
[MTAGS08] "Many-Task Computing for Grids and Supercomputers"

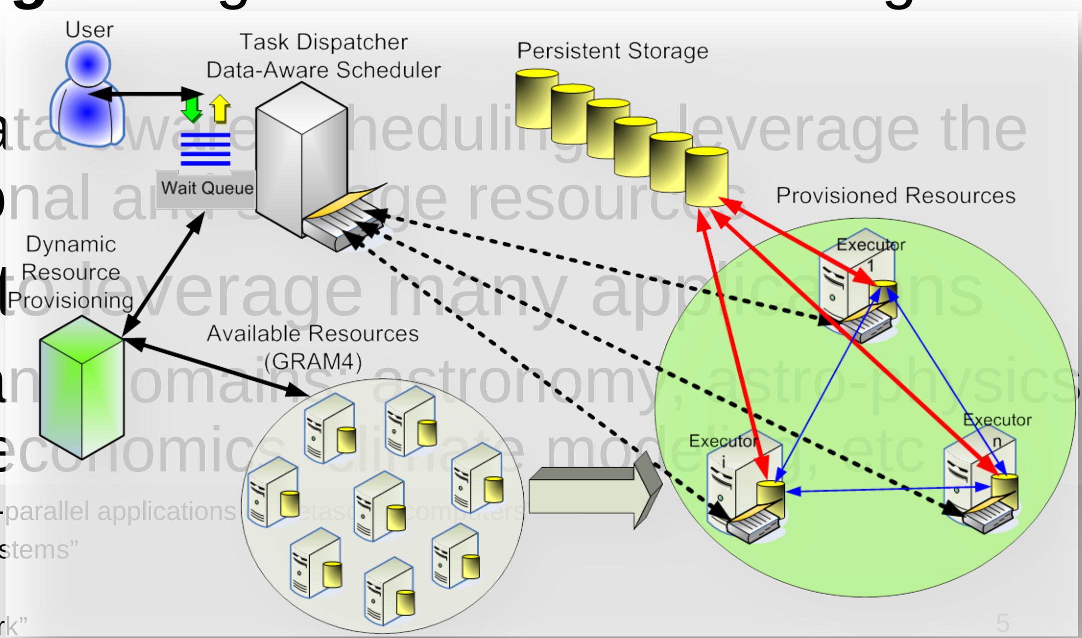
Problem Space



[MTAGS08] “Many-Task Computing for Grids and Supercomputers”

Falkon

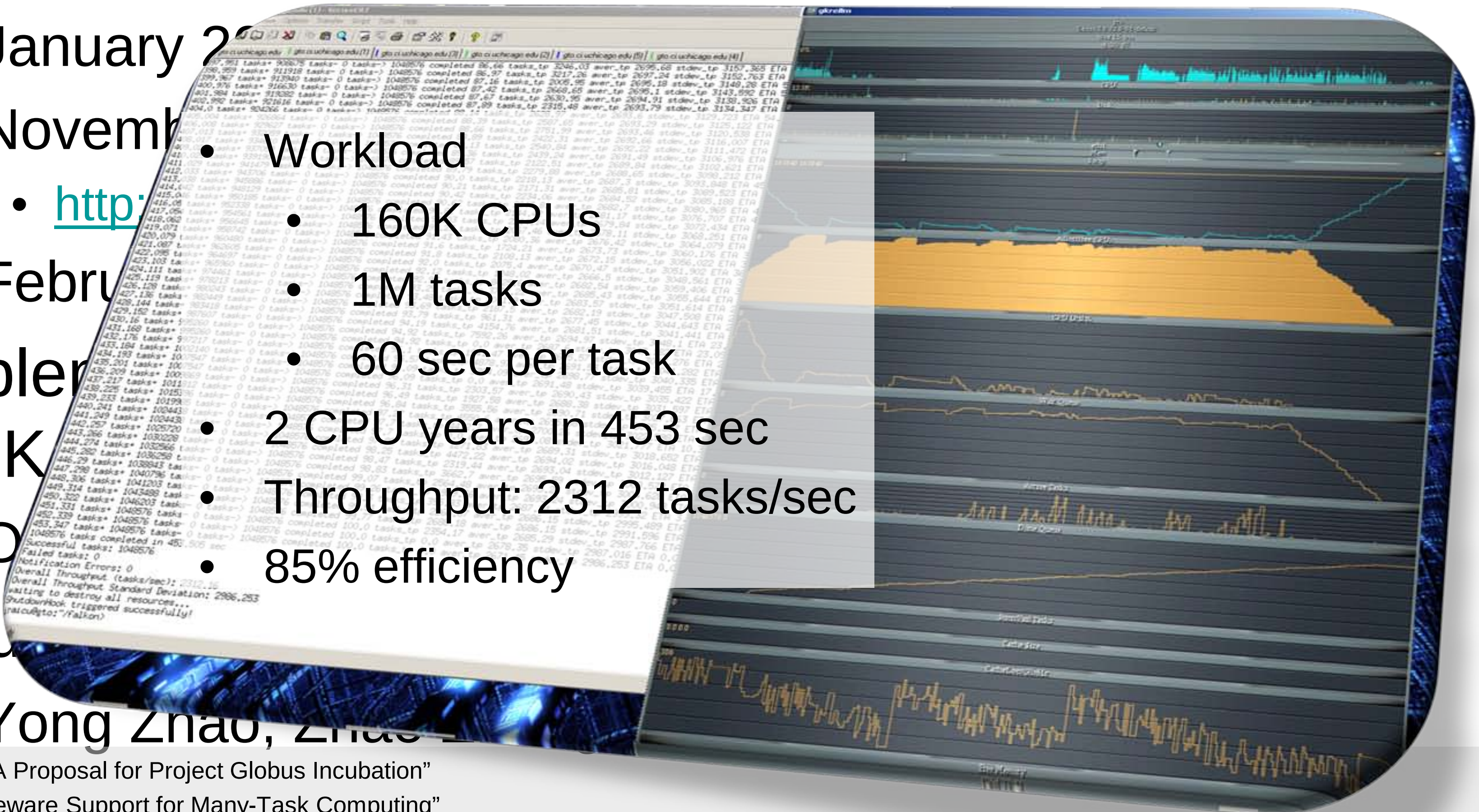
- ***Goal:*** enable the ***rapid and efficient*** execution of many independent jobs on large compute clusters
- Combines three components:
  - a ***streamlined task dispatcher***
  - ***resource provisioning*** through multi-level scheduling techniques
  - ***data diffusion*** and data-aware scheduling to leverage the co-located computational and storage resources
- Integration into Swift to leverage many applications
  - Applications cover many domains: astronomy, astro-physics, medicine, chemistry, economics, climate modeling, etc



[SciDAC09] "Extreme-scale scripting: Opportunities for large task-parallel applications on petascale computers"
[SC08] "Towards Loosely-Coupled Programming on Petascale Systems"
[Globus07] "Falkon: A Proposal for Project Globus Incubation"
[SC07] "Falkon: a Fast and Light-weight tasK executiON framework"
[SWF07] "Swift: Fast, Reliable, Loosely Coupled Parallel Computation"

Falkon Project

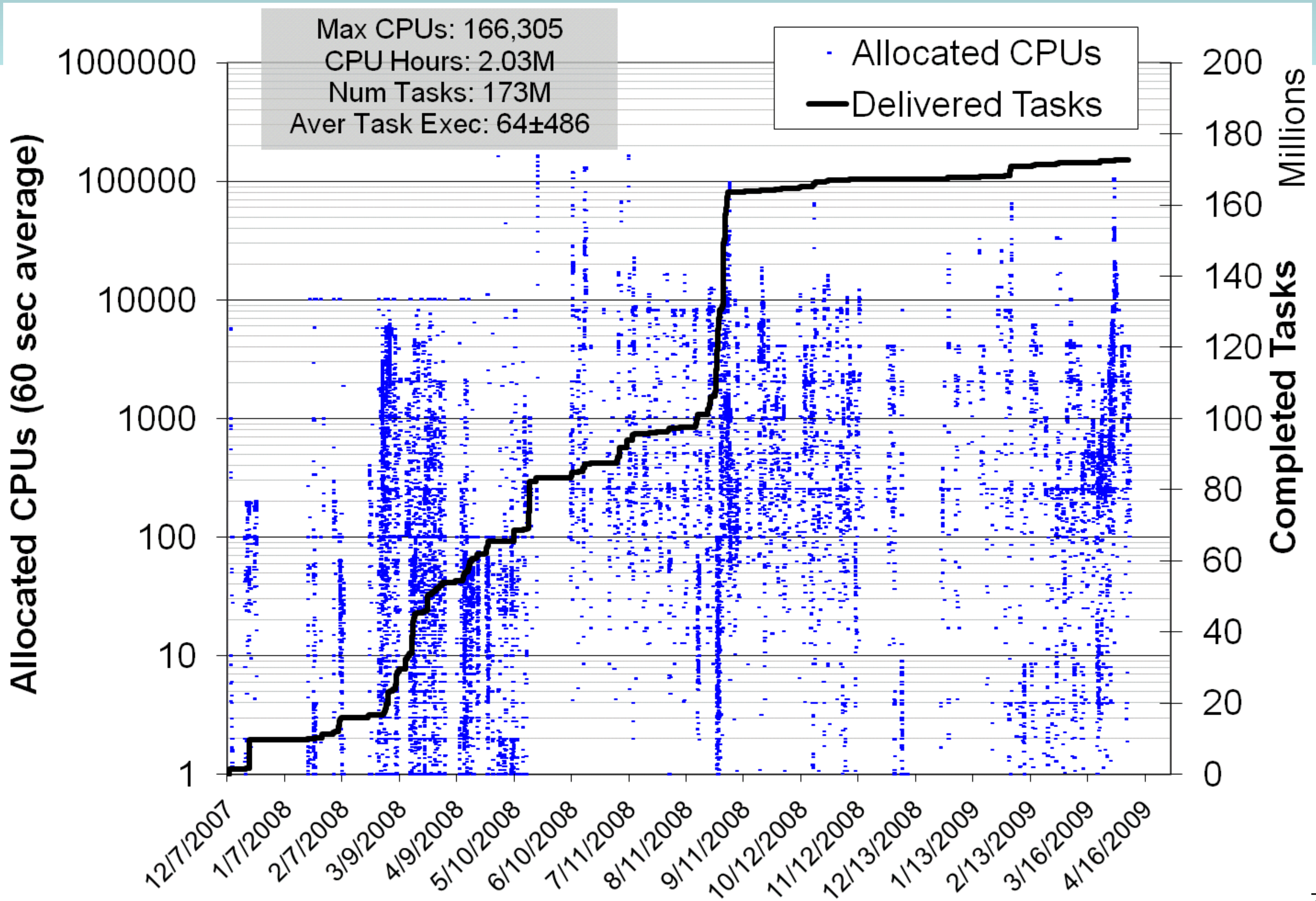
- Falkon is a real system
  - Late 2005: Initial prototype, AstroPortal
  - January 2
  - Novemb
    - http:
  - Febru
- Implen (~1K
  - O
- Sou
  - Yong Zhao, Zhao



- Workload
  - 160K CPUs
  - 1M tasks
  - 60 sec per task
- 2 CPU years in 453 sec
- Throughput: 2312 tasks/sec
- 85% efficiency

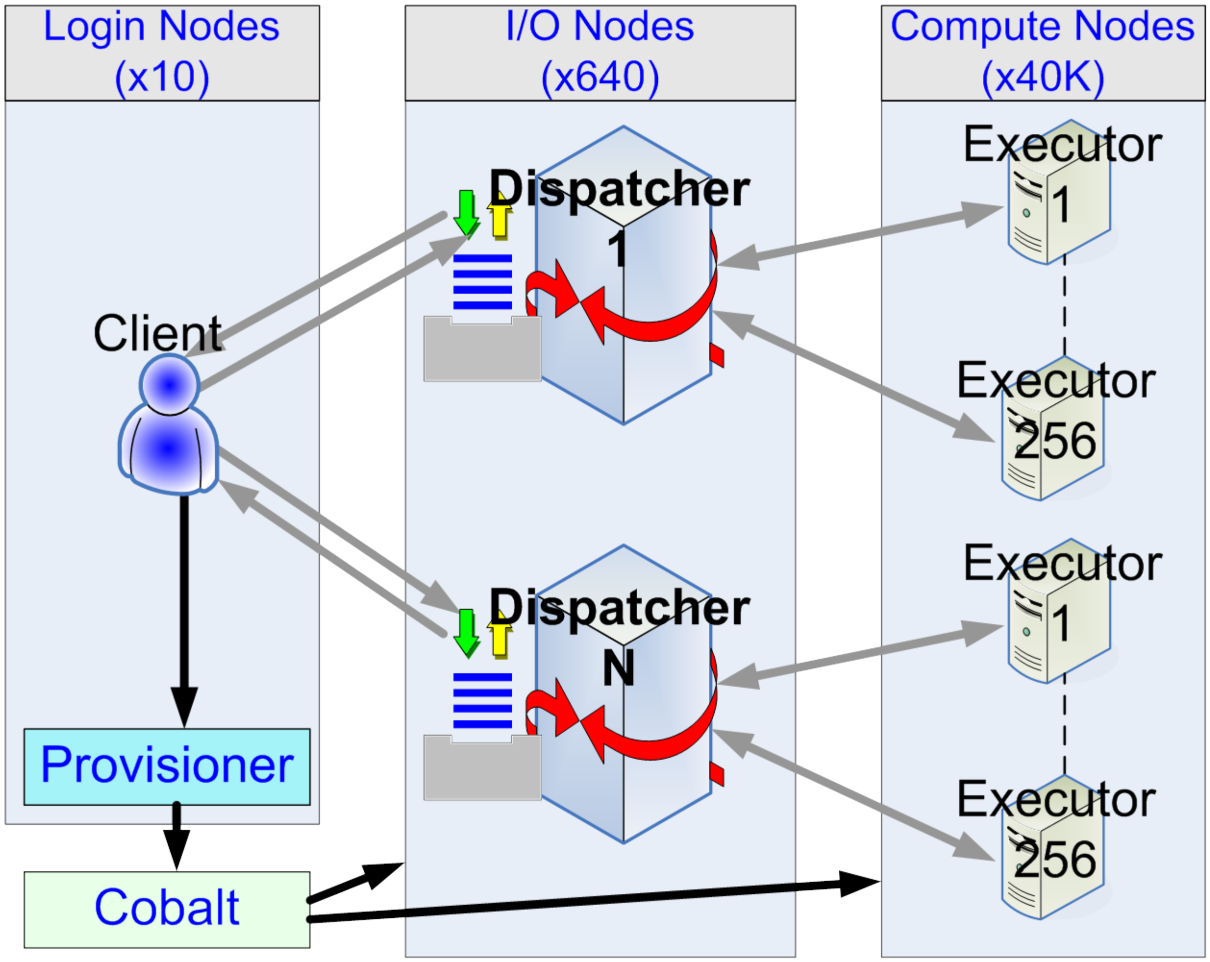
[Globus07] "Falkon: A Proposal for Project Globus Incubation"
[CLUSTER10] "Middleware Support for Many-Task Computing"

Falkon Activity History (16 months)



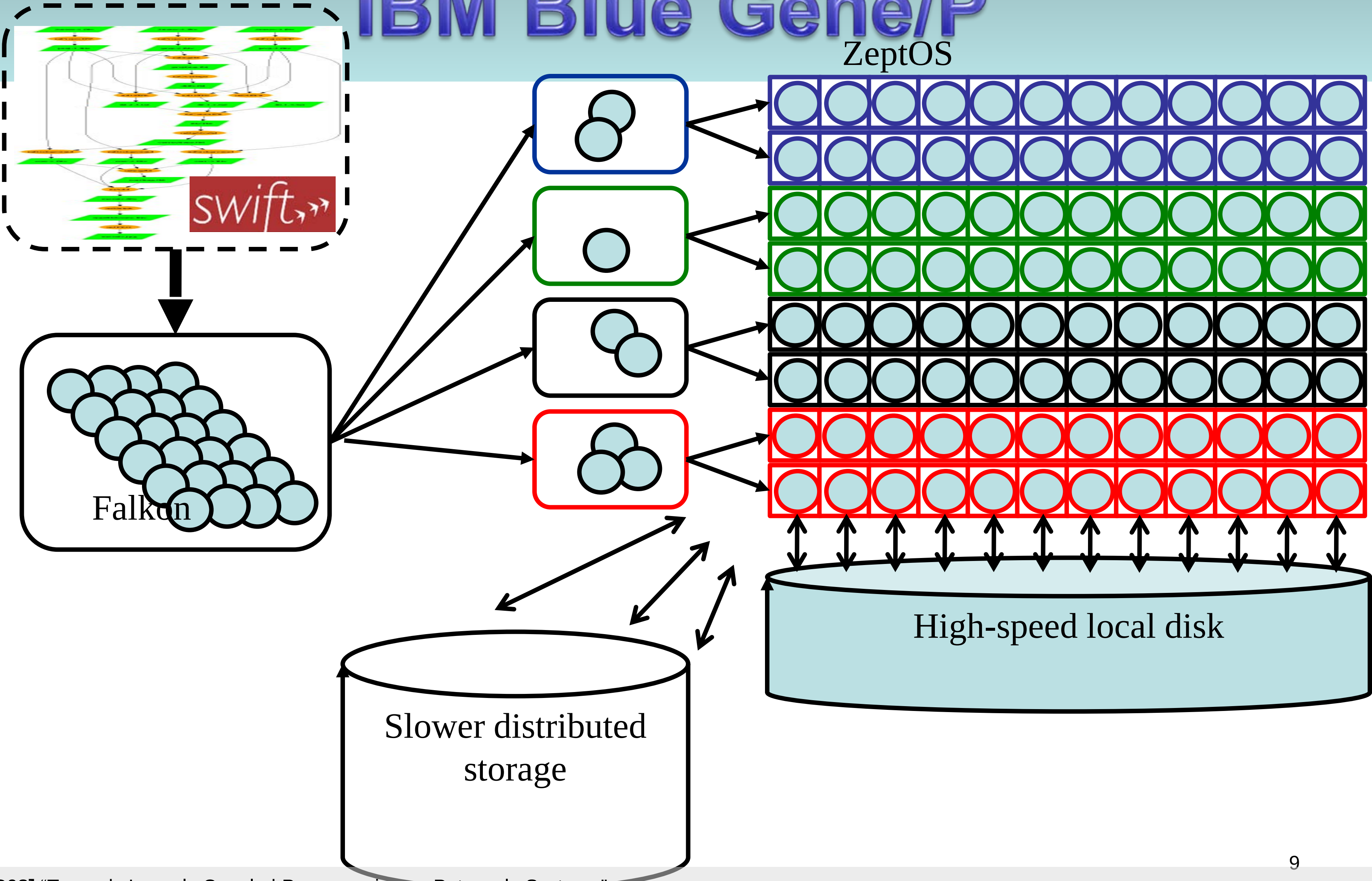
[TPDS09] “Middleware Support for Many-Task Computing”, under preparation

Distributed Falkon Architecture



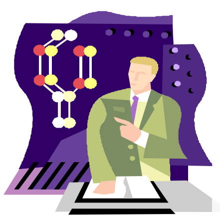
[SC08] "Towards Loosely-Coupled Programming on Petascale Systems"

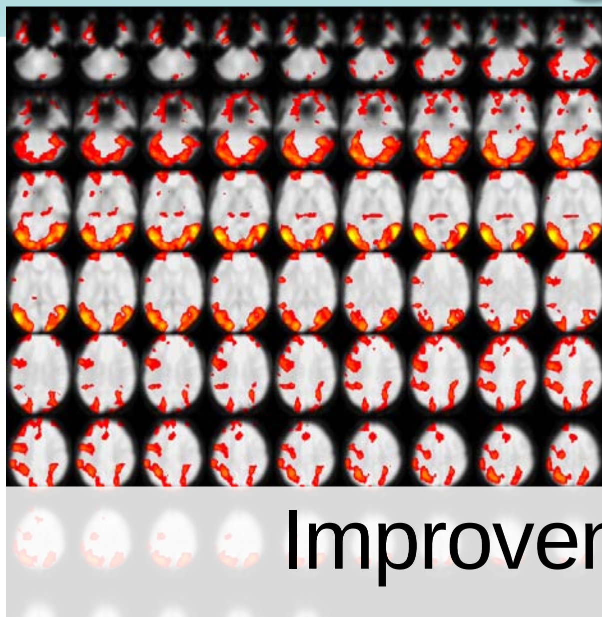
Managing 160K CPUs
IBM Blue Gene/P

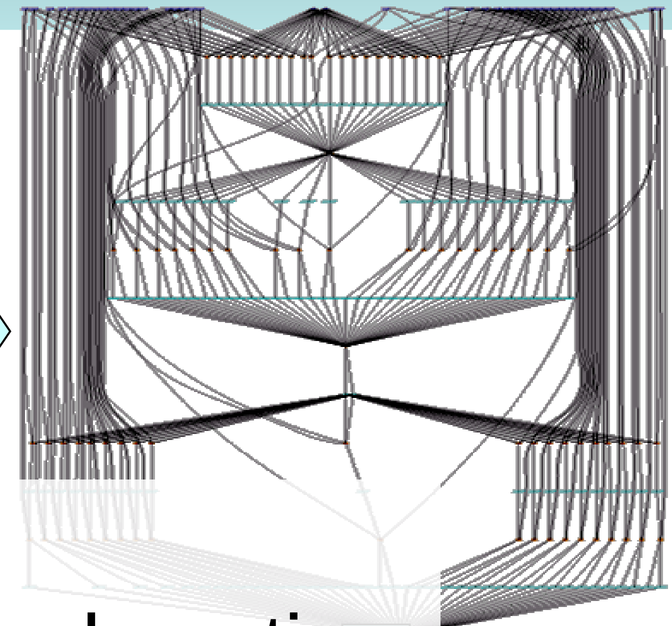


[SC08] "Towards Loosely-Coupled Programming on Petascale Systems"

Applications
Medical Imaging: fMRI

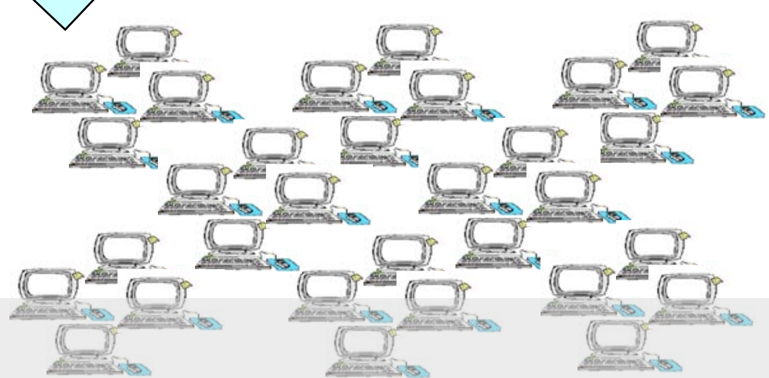






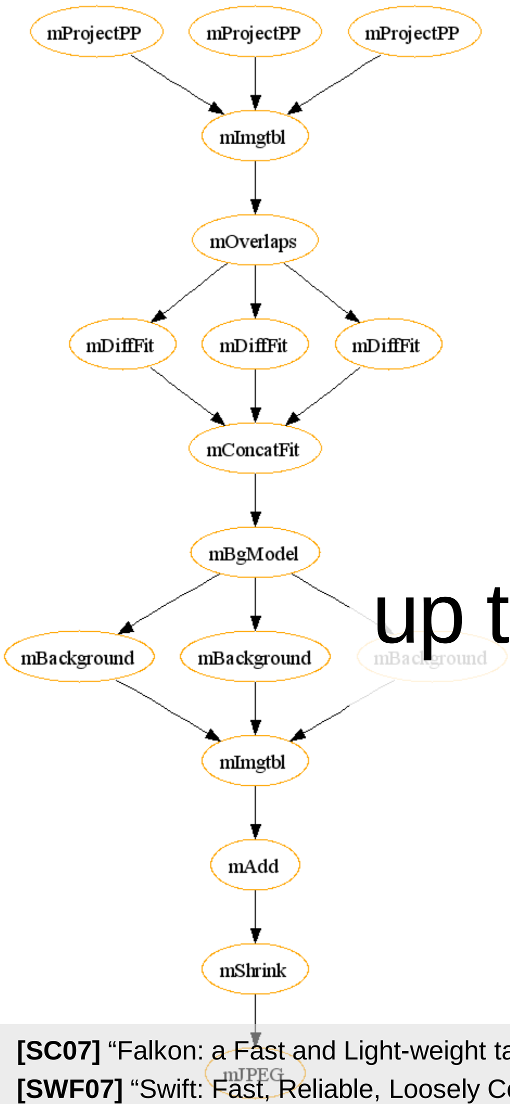
Improvement:
up to 90% lower end-to-end run time

- Wide range of analyses
  - Testing, interactive analysis, production runs
  - Data mining
  - Parameter studies



[SC07] "Falkon: a Fast and Light-weight tasK executiON framework"
[SWF07] "Swift: Fast, Reliable, Loosely Coupled Parallel Computation"

Applications
Astronomy: Montage

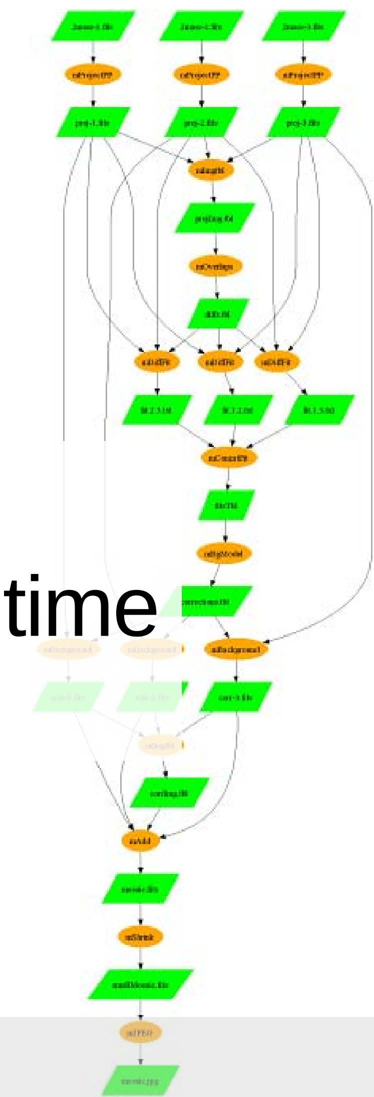


Improvement:

up to 57% lower end-to-end run time

Within 4% of MPI

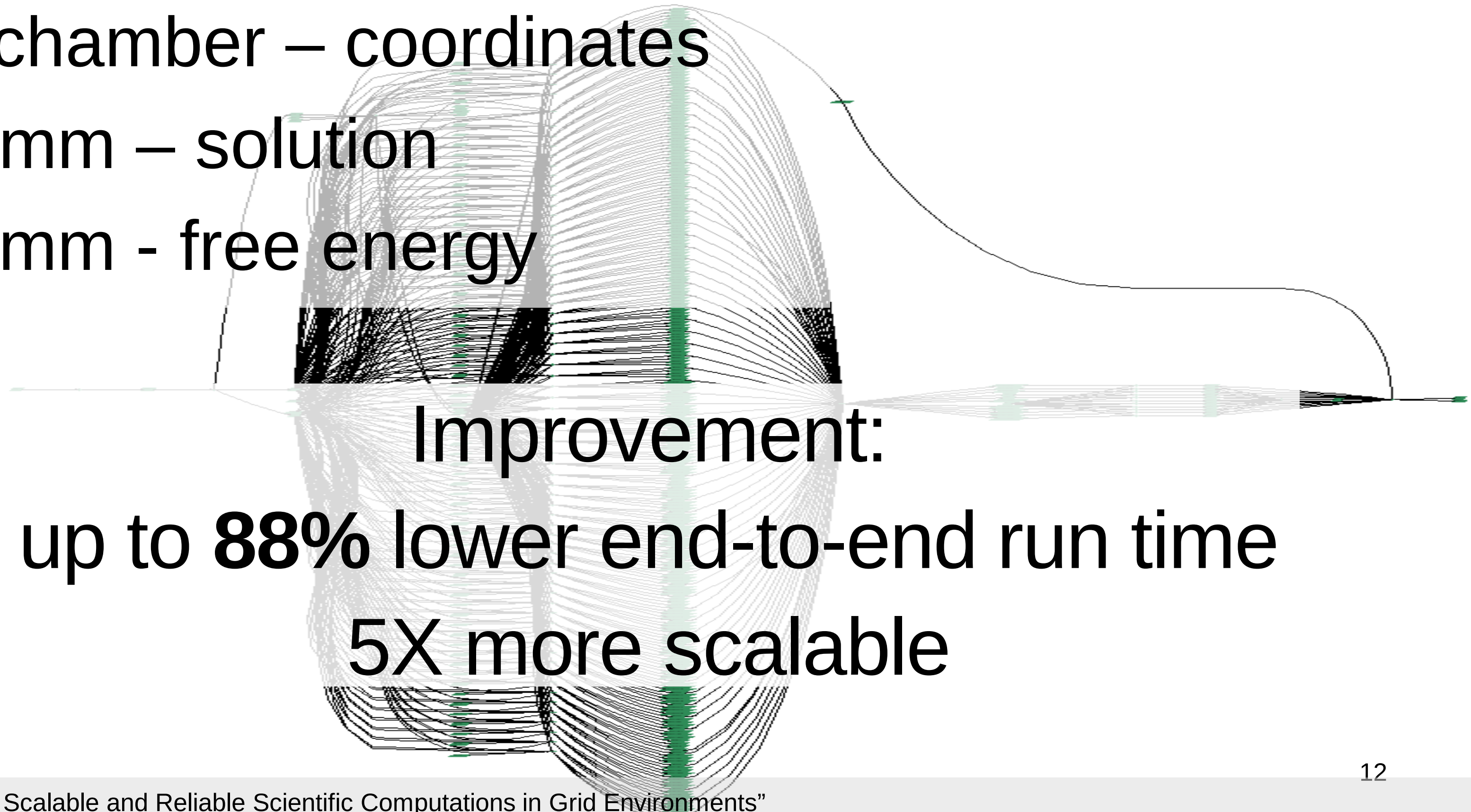
B. Berriman, J. Good (Caltech)
J. Jacob, D. Katz (JPL)



[SC07] "Falkon: a Fast and Light-weight tasK executiON framework"
[SWF07] "Swift: Fast, Reliable, Loosely Coupled Parallel Computation"

Applications
Molecular Dynamics: MolDyn

- Determination of free energies in aqueous solution
  - Antechamber – coordinates
  - Charmm – solution
  - Charmm - free energy



Improvement:

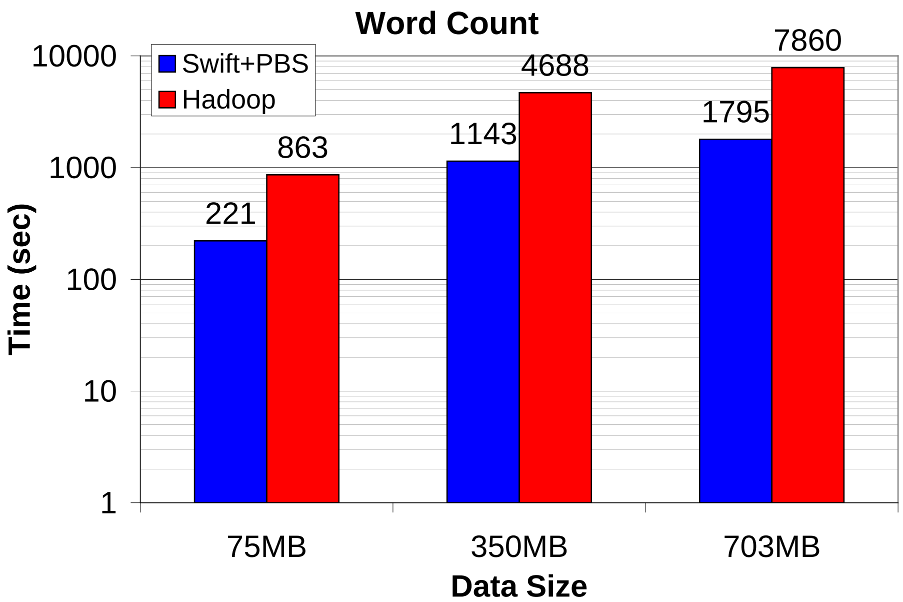
up to **88%** lower end-to-end run time

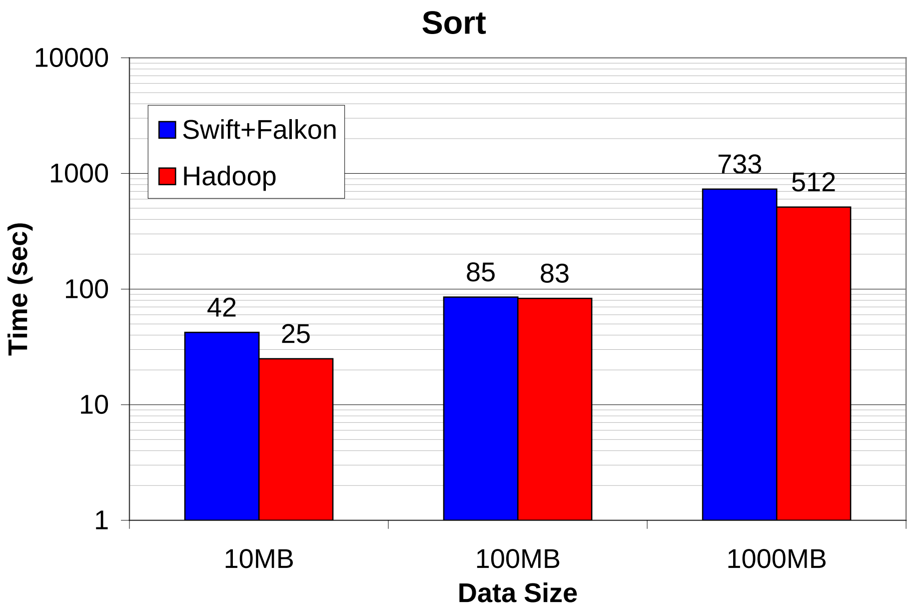
5X more scalable

[NOVA08] "Realizing Fast, Scalable and Reliable Scientific Computations in Grid Environments"

Applications
Word Count and Sort

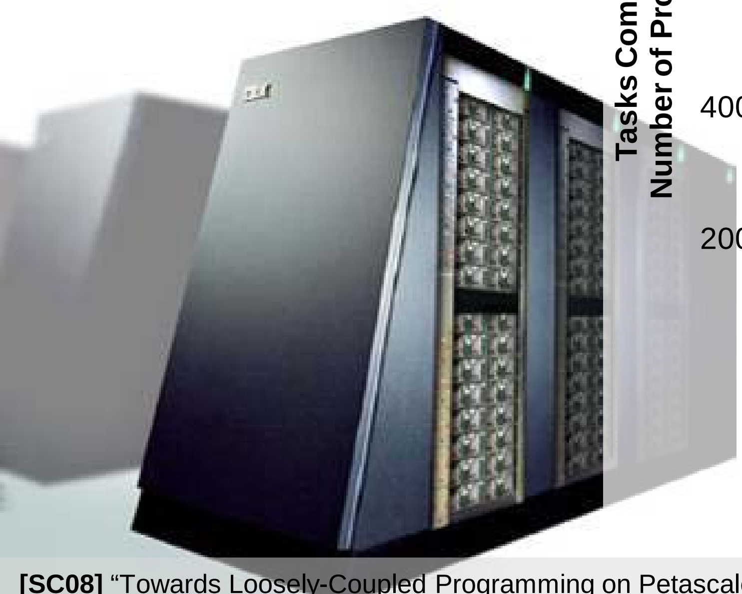
- Classic benchmarks for MapReduce
  - Word Count
  - Sort
- Swift and Falkon performs similar or better than Hadoop (on 32 processors)

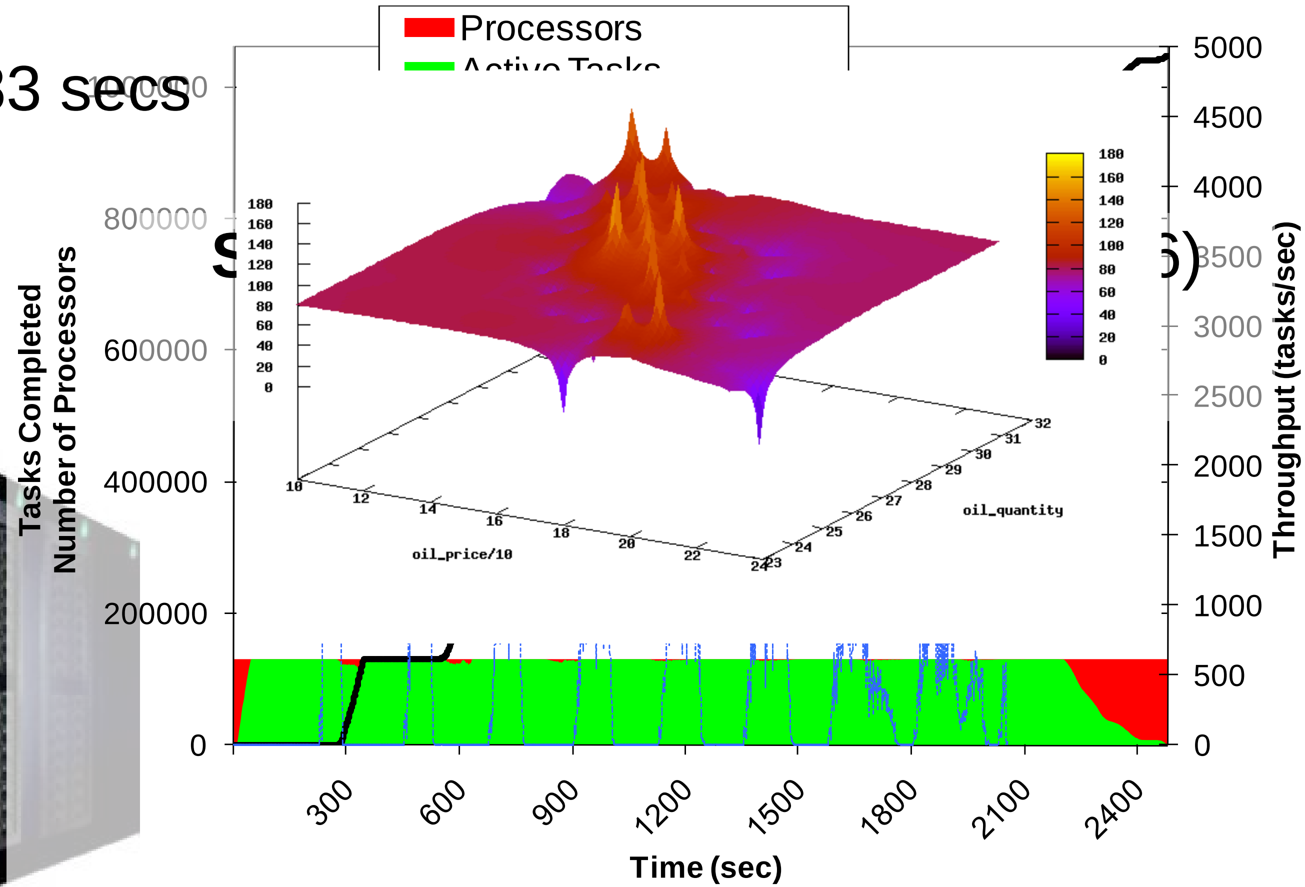




Applications
Economic Modeling: MARS

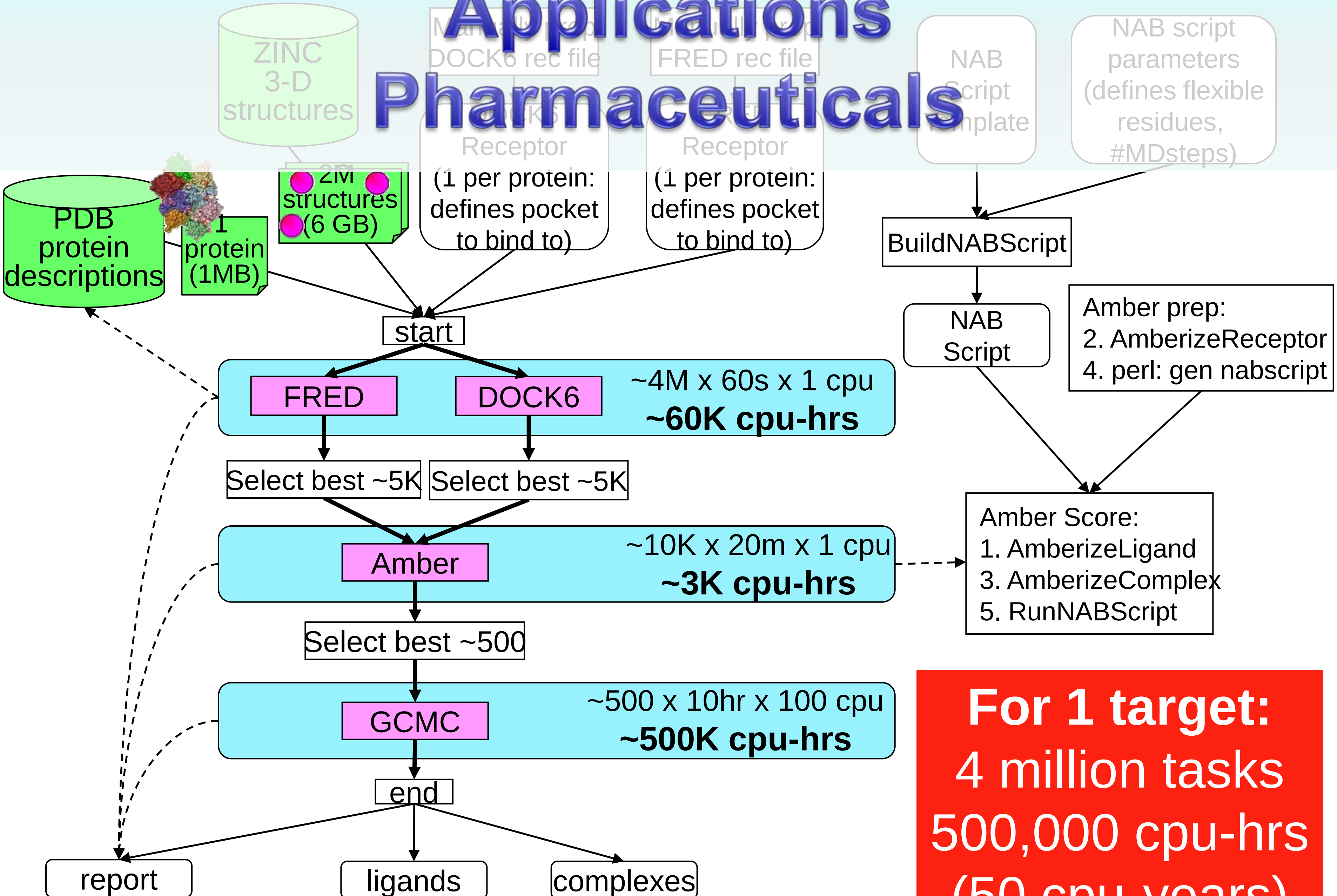
- CPU Cores: 130816
- Tasks: 1048576
- Elapsed time: 2483 secs
- CPU Years: 9.3





[SC08] "Towards Loosely-Coupled Programming on Petascale Systems"

Applications Pharmaceuticals



For 1 target:
4 million tasks
500,000 cpu-hrs
(50 cpu-years)

Applications
Pharmaceuticals: DOCK

CPU cores: 118784

Tasks: 934803

Elapsed time: 2.01 hours

Compute time: 21.43 CPU years

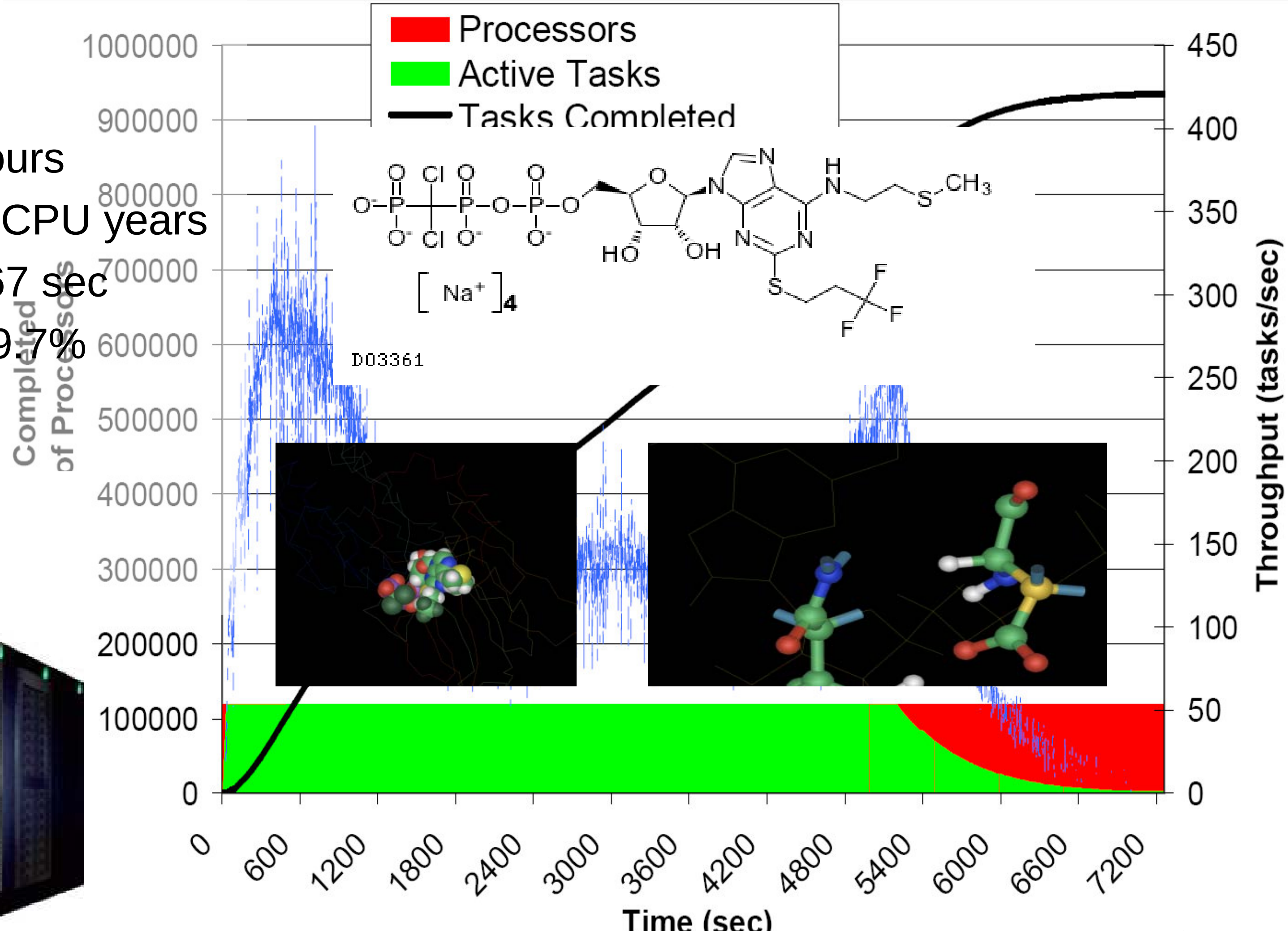
Average task time: 667 sec

Relative Efficiency: 99.7%

(from 16 to 32 racks)

Utilization:

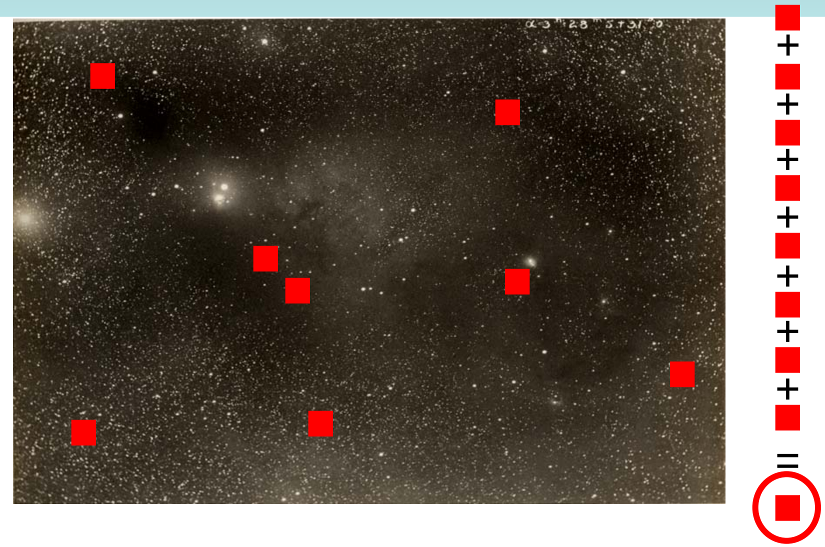
- Sustained: 99.6%
- Overall: 78.3%

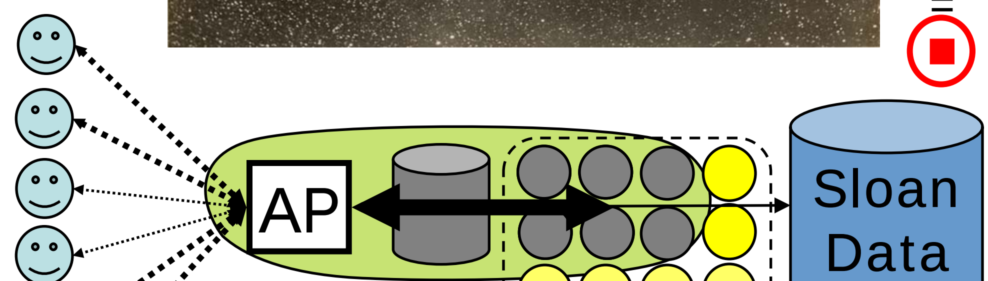


[SC08] “Towards Loosely-Coupled Programming on Petascale Systems”

Applications
Astronomy: AstroPortal

- Purpose
  - On-demand “stacks” of random locations within ~10TB dataset
- Challenge
  - Processing Costs:
    - O(100ms) per object
  - Data Intensive:
    - 40MB:1sec
  - Rapid access to 10-10K “random” files
  - Time-varying load



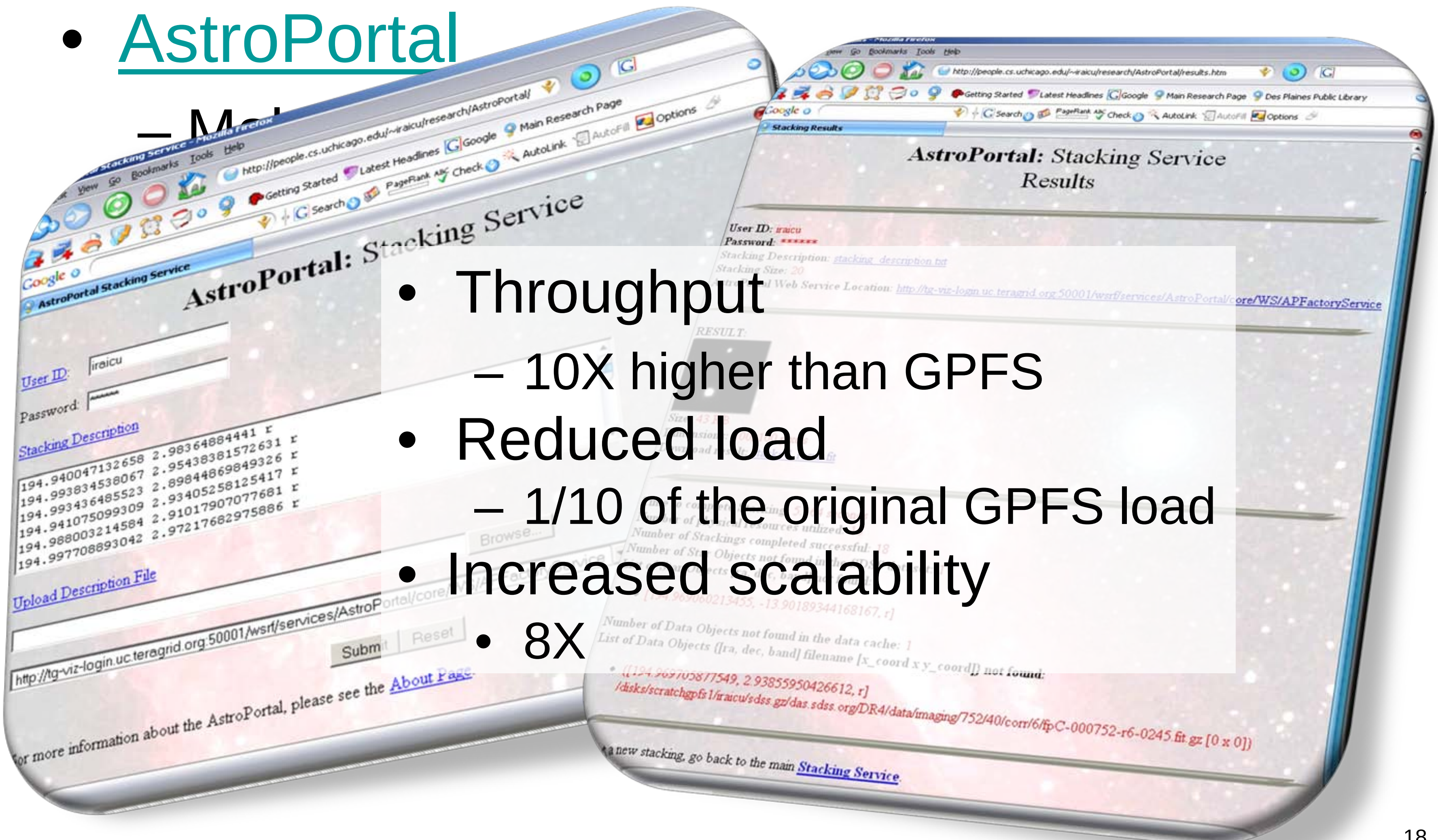


Locality	Number of Objects	Number of Files
1	111700	111700
1.38	154345	111699
2	97999	49000
3	88857	29620
4	76575	19145
5	60590	12120
10	46480	4650
20	40460	2025
30	23695	790

[DADC08] “Accelerating Large-scale Data Exploration through Data Diffusion”
[TG06] “AstroPortal: A Science Gateway for Large-scale Astronomy Data Analysis”

Applications
Astronomy: AstroPortal

- AstroPortal



- Throughput
  - 10X higher than GPFS
- Reduced load
  - 1/10 of the original GPFS load
- Increased scalability
  - 8X

Conclusions

- There is more to HPC than tightly coupled MPI, and more to HTC than embarrassingly parallel long jobs
- Data locality is critical at large-scale

Mythbusting

- ~~Embarrassingly Hap~~pily parallel apps are trivial to run
  - Logistical problems can be tremendous
- Loosely coupled apps do not require “supercomputers”
  - Total computational requirements can be enormous
  - Individual tasks may be tightly coupled
  - Workloads frequently involve large amounts of I/O
  - Make use of idle resources from “supercomputers” via backfilling
  - Costs to run “supercomputers” per FLOP is among the best
- Loosely coupled apps do not require specialized system software
  - Their requirements on the job submission and storage systems can be extremely large
- Shared/parallel file systems are good for all applications
  - They don’t scale proportionally with the compute resources
  - Data intensive applications don’t perform and scale well
  - Growing compute/storage gap

Questions

?