Many-task Computing: Bridging the Gap between High-Throughput Computing and High-Performance Computing

Ioan Raicu
Distributed Systems Laboratory
Computer Science Department
University of Chicago

Department of Computer Science
Colorado State University
March 2nd, 2009
Acknowledgements

• Committee Members
  – Ian Foster (advisor)
  – Rick Stevens
  – Alex Szalay

• University of Chicago
  – Computer Science Department
  – Computational Institute

• Argonne National Laboratory
  – Math and Computer Science Division
  – Argonne Leadership Computing Facility

• NASA
  – Ames Research Center
  – Jerry C. Yan

• Over 60 collaborators
- Lead by Dr. Ian Foster
- Research Areas:
  - Distributed systems
  - Grid middleware
  - Grid applications
  - Designing, implementing, and evaluating systems, protocols, and applications
  - Data-intensive scientific computing
- People:
  - 1 faculty (Dr. Ian Foster)
  - 12 students
  - 2 research staff
  - 13 alumni
People:
- Director: Ian Foster
- 70 faculty and scientists
- 30 full-time professional staff
- 14 graduate students

Focus
- Deep Supercomputing
- Data Intensive Computing
- Next Generation Cybertools

Many high-impact projects
- Open Science Grid
- TeraGrid
- Globus
- National Microbial Pathogen Research Center
- Social Informatics Data Grid
- Chicago Biomedical Consortium

http://www.cl.uchicago.edu/index.php
Math and Computer Science Div.
Argonne National Laboratory


- **People:**
  - Associate Director: Ian Foster
  - Over 180 staff, researchers, scientists, developers

- **Research Areas**
  - Algorithms, Software, and Applications
  - Parallel Tools
  - Distributed Systems Research
  - Collaborative and Virtual Environments
  - Computational Science

- **Many high-impact projects**
  - TeraGrid (largest national cyber-infrastructure project)
  - Globus Toolkit (defacto Grid Computing middleware)
  - MPI (synonymous with HPC and Supercomputing)
  - PVFS (scalable and high performance parallel file system)
- People:
  - Director: Pete Beckman
  - Over 40 staff, researchers, scientists, developers
- Leadership class computing resources to scientific community
- Hosts the IBM Blue Gene/P supercomputer
  - 500TF system
  - #5 in Top500
Distributed Resources

- UChicago CS (50+ machines over the UChicago campus)
- UChicago TeraPort (274-cores)
- UC/ANL Cluster (316 processors)
- PlanetLab (912 nodes at 470 sites all over the world)
- UChicago PADS (7TF, 512-cores)
- ANL SiCortex 5832 (6TF, 5832-cores)
- Open Science Grid (43K-cores across 80 institutions in the US)
- IBM Blue Gene/P Supercomputer at ANL (~500TF, 160K-cores)
- TeraGrid (161K-cores across 11 institutions and 22 systems over the US)
  - Includes a Sun Constellation supercomputer (~500TF, 62K-cores)
Clusters, Grids, Supercomputers

[Distributed Systems]

Supercomputers

Grids

Clusters

Clouds

Web 2.0

Application Oriented

Services Oriented

Scale

[GCE08] “Cloud Computing and Grid Computing 360-Degree Compared”
Cluster Computing: PADS

Computer clusters using commodity processors, network interconnects, and operating systems.
Grids tend to be composed of multiple clusters, and are typically loosely coupled, heterogeneous, and geographically dispersed.

Tommy Minyard, TACC
Highly-tuned computer clusters using commodity processors combined with custom network interconnects and customized operating system.
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

HPC: High-Performance Computing
- Synonymous with supercomputing
- Tightly-coupled applications
- Implemented using Message Passing Interface (MPI)
- Large of amounts of computing for short periods of time
- Usually requires low latency interconnects
- Measured in FLOPS
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
Problem Space

Input Data Size

Hi

Med

Low

Number of Tasks

1

1K

1M

MapReduce/MTC
(Data Analysis, Mining)

MTC
(Big Data and Many Tasks)

HPC
(Heroic MPI Tasks)

HTC/MTC
(Many Loosely Coupled Tasks)
Challenges for MTC

1. Slow job dispatch rates
2. Long queue times
3. Poor shared/parallel file system scaling

<table>
<thead>
<tr>
<th>System</th>
<th>Comments</th>
<th>Throughput (tasks/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Condor (v6.7.2) - Production</td>
<td>Dual Xeon 2.4GHz, 4GB</td>
<td>0.49</td>
</tr>
<tr>
<td>PBS (v2.1.8) - Production</td>
<td>Dual Xeon 2.4GHz, 4GB</td>
<td>0.45</td>
</tr>
<tr>
<td>Condor (v6.7.2) - Production</td>
<td>Quad Xeon 3 GHz, 4GB</td>
<td>2</td>
</tr>
<tr>
<td>Condor (v6.8.2) - Production</td>
<td></td>
<td>0.42</td>
</tr>
<tr>
<td>Condor (v6.9.3) - Development</td>
<td>Quad Xeon 3 GHz, 4GB</td>
<td>11</td>
</tr>
<tr>
<td>Condor-J2 - Experimental</td>
<td></td>
<td>22</td>
</tr>
</tbody>
</table>

[UC07] “Harnessing Grid Resources with Data-Centric Task Farms”
Projected Growth Trends

Number of Cores vs. Manufacturing Process

Pat Helland, Microsoft, The Irresistible Forces Meet the Movable Objects, November 9th, 2007

Top500 Projected Development,
Growing Storage/Compute Gap

- **Local Disk:**
  - 2002-2004: ANL/UC TG Site (70GB SCSI)
  - Today: PADS (RAID-0, 6 drives 750GB SATA)

- **Cluster:**
  - 2002-2004: ANL/UC TG Site (GPFS, 8 servers, 1Gb/s each)
  - Today: PADS (GPFS, SAN)

- **Supercomputer:**
  - 2002-2004: IBM Blue Gene/L (GPFS)
  - Today: IBM Blue Gene/P (GPFS)
• [JS09] “Middleware Support for Many-Task Computing”, under preparation
• [HPDC09] “The Quest for Scalable Support of Data Intensive Workloads in Distributed Systems”, under review
• [DIDC09] “Towards Data Intensive Many-Task Computing”, under review
• [SC08] “Towards Loosely-Coupled Programming on Petascale Systems”
• [MTAGS08 Workshop] Workshop on Many-Task Computing on Grids and Supercomputers
• [MTAGS08] “Many-Task Computing for Grids and Supercomputers”
• [GCE08] “Cloud Computing and Grid Computing 360-Degree Compared”
• [SWF08] “Scientific Workflow Systems for 21st Century e-Science, New Bottle or New Wine?”
• [DADC08] “Accelerating Large-scale Data Exploration through Data Diffusion”
• [TG08] “Data Intensive Scalable Computing on TeraGrid: A Comparison of MapReduce and Swift”
• [GlobusWorld08] “Managing and Executing Loosely Coupled Large Scale Applications on Clusters, Grids, and Supercomputers”
• [NOVA08] “Realizing Fast, Scalable and Reliable Scientific Computations in Grid Environments”
• [UC07] “Harnessing Grid Resources with Data-Centric Task Farms”
• [Globus07] “Falkon: A Proposal for Project Globus Incubation”
• [SC07] “Falkon: a Fast and Light-weight task executIO framework”
• [MSES07] “A Data Diffusion Approach to Large Scale Scientific Exploration”
• [SWF07] “Swift: Fast, Reliable, Loosely Coupled Parallel Computation”
• [TG07] “Dynamic Resource Provisioning in Grid Environments”
• [NASA06-08] “Harnessing Grid Resources to Enable the Dynamic Analysis of Large Astronomy Datasets”
• [SC06] “Harnessing Grid Resources to Enable the Dynamic Analysis of Large Astronomy Datasets”
• [TG06] “AstroPortal: A Science Gateway for Large-scale Astronomy Data Analysis”
• [NSF06] “The Importance of Data Locality in Distributed Computing Applications”
Techniques to Support MTC

- Streamlined task dispatching
- Dynamic resource provisioning
  - Multi-level scheduling
  - Resources are acquired/released in response to demand
- Data diffusion
  - Data diffuses from archival storage to transient resources
  - Resource “caching” allows faster responses to subsequent requests
  - Co-locate data and computations to optimize performance

[DADC08] “Accelerating Large-scale Data Exploration through Data Diffusion”
[UC07] “Harnessing Grid Resources with Data-Centric Task Farms”
[TG07] “Dynamic Resource Provisioning in Grid Environments”
• Abstract model
  – Models the efficiency and speedup of entire workloads
  – Captures techniques to support MTC
    • Streamlined task dispatching
    • Dynamic resource provisioning
    • Data diffusion
• Middleware to support MTC
  – Falkon: a fast a light-weight execution framework
  – Reference Implementation of the abstract model

[SC07] “Falkon: a Fast and Light-weight task execution framework”
**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

---


[SC07] “Falkon: a Fast and Light-weight task execution framework”

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

- Login Nodes (x10)
- I/O Nodes (x640)
- Compute Nodes (x40K)

Client
Provisioner
Cobalt

Dispatcher 1
Dispatcher N

Executor 1
Executor 256
Executor 256
Executor 1
Executor 256

Managing 160K CPUs
IBM Blue Gene/P

ZeptOS

High-speed local disk

Slower distributed storage

Data Diffusion

- Resource acquired in response to demand
- Data diffuse from archival storage to newly acquired transient resources
- Resource “caching” allows faster responses to subsequent requests
- Resources are released when demand drops
- Optimizes performance by co-scheduling data and computations
- Decrease dependency of a shared/parallel file systems
- Critical to support data intensive MTC

[DADC08] “Accelerating Large-scale Data Exploration through Data Diffusion”
Dispatch Throughput

<table>
<thead>
<tr>
<th>System</th>
<th>Comments</th>
<th>Throughput (tasks/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Condor (v6.7.2) - Production</td>
<td>Dual Xeon 2.4GHz, 4GB</td>
<td>0.49</td>
</tr>
<tr>
<td>PBS (v2.1.8) - Production</td>
<td>Dual Xeon 2.4GHz, 4GB</td>
<td>0.45</td>
</tr>
<tr>
<td>Condor (v6.7.2) - Production</td>
<td>Quad Xeon 3 GHz, 4GB</td>
<td>2</td>
</tr>
<tr>
<td>Condor (v6.8.2) - Production</td>
<td></td>
<td>0.42</td>
</tr>
<tr>
<td>Condor (v6.9.3) - Development</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>Condor-J2 - Experimental</td>
<td>Quad Xeon 3 GHz, 4GB</td>
<td>22</td>
</tr>
</tbody>
</table>

Efficiency

Number of Processors

Resource Provisioning Overheads
IBM Blue Gene/P

- Initializing Falkon Resource
- Starting GT4 and Falkon Service
- Booting Partitions

Time (sec)
0 120 240 360 480 600 720 840 960 1080 1200 1320

Number of Processors
256 512 1024 2048 4096 8192 16384 32768 65536 131072 163840
• End-to-end execution time:
  – 1260 sec in ideal case
  – 4904 sec \rightarrow 1276 sec
• Average task queue time:
  – 42.2 sec in ideal case
  – 611 sec \rightarrow 43.5 sec
• Trade-off:
  – Resource Utilization for Execution Efficiency

[SC07] “Falkon: a Fast and Light-weight task execution framework”
Scheduling Policies

- **FA**: first-available
  - simple load balancing
- **MCH**: max-cache-hit
  - maximize cache hits
- **MCU**: max-compute-util
  - maximize processor utilization
- **GCC**: good-cache-compute
  - maximize both cache hit and processor utilization at the same time

[**DADC08**] “Accelerating Large-scale Data Exploration through Data Diffusion”
[**HPDC09**] “The Quest for Scalable Support of Data Intensive Applications in Distributed Systems”, under review
[**DIDC09**] “Towards Data Intensive Many-Task Computing”, under review
Data-Aware Scheduler Profiling

- 3GHz dual CPUs
- ANL/UC TG with 128 processors
- Scheduling window: 2500 tasks
- Dataset
  - 100K files
  - 1 byte each
- Tasks
  - Read 1 file
  - Write 1 file

[3GHz dual CPUs • ANL/UC TG with 128 processors • Scheduling window: 2500 tasks • Dataset • 100K files • 1 byte each • Tasks • Read 1 file • Write 1 file]

Synthetic Workloads

- Monotonically Increasing Workload
  - Emphasizes increasing loads
- Sine-Wave Workload
  - Emphasizes varying loads
- All-Pairs Workload
  - Compare to best case model of active storage
- Image Stacking Workload (Astronomy)
  - Evaluate data diffusion on a real large-scale data-intensive application from astronomy domain
**Data Diffusion**

**Monotonically Increasing Workload**

[HPDC09] “The Quest for Scalable Support of Data Intensive Applications in Distributed Systems”, under review

Throughput:
- Average: 14Gb/s vs 4Gb/s
- Peak: 81Gb/s vs. 6Gb/s

Response Time ➔
- 3 sec vs 1569 sec ➔ 506X

[HPDC09] “The Quest for Scalable Support of Data Intensive Applications in Distributed Systems”, under review
Data Diffusion
Sine-Wave Workload

- GPFS $\rightarrow$ 5.7 hrs, $\sim$8Gb/s, 1138 CPU hrs
- GCC+SRP $\rightarrow$ 1.8 hrs, $\sim$25Gb/s, 361 CPU hrs
- GCC+DRP $\rightarrow$ 1.86 hrs, $\sim$24Gb/s, 253 CPU hrs

[HPDC09] “The Quest for Scalable Support of Data Intensive Applications in Distributed Systems”, under review
• Pull vs. Push
  – Data Diffusion
    • Pulls task working set
    • Incremental spanning forest
  – Active Storage:
    • Pushes workload working set to all nodes
    • Static spanning tree

Christopher Moretti, Douglas Thain, University of Notre Dame

[HPDC09] “The Quest for Scalable Support of Data Intensive Applications in Distributed Systems”, under review
Data Diffusion vs. Active Storage
All-Pairs Workload

- Best to use active storage if
  - Slow data source
  - Workload working set fits on local node storage
- Best to use data diffusion if
  - Medium to fast data source
  - Task working set $\ll$ workload working set
  - Task working set fits on local node storage
- If task working set does not fit on local node storage
  - Use parallel file system (i.e. GPFS, Lustre, PVFS, etc)

[HPDC09] "The Quest for Scalable Support of Data Intensive Applications in Distributed Systems", under review
[DIDC09] "Towards Data Intensive Many-Task Computing", under review
Applications
Swift Architecture

Specification
- Abstract computation
- SwiftScript Compiler
  - Virtual Data Catalog
  - SwiftScript

Scheduling
- Execution Engine
  - (Karajan w/ Swift Runtime)
  - Swift runtime callouts
- Status reporting
- Provenance collector

Execution
- Virtual Node(s)
  - File1
  - File2
  - File3
  - App F1
  - App F2
  - Provenance data

Provisioning
- Falkon Resource Provisioner
- Amazon EC2

References:
[NOVA08] “Realizing Fast, Scalable and Reliable Scientific Computations in Grid Environments”
[SWF07] “Swift: Fast, Reliable, Loosely Coupled Parallel Computation”
Applications
Medical Imaging: fMRI

- 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”
• GRAM vs. Falkon: 85%~90% lower run time
• GRAM/Clustering vs. Falkon: 40%~74% lower run time

[SC07] “Falkon: a Fast and Light-weight tasK executiON framework”
[SWF07] “Swift: Fast, Reliable, Loosely Coupled Parallel Computation”
Applications
Astronomy: Montage

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
Astronomy: Montage

- GRAM/Clustering vs. Falkon: 57% lower application run time
- MPI* vs. Falkon: 4% higher application run time
- * MPI should be lower bound

[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

[NOVA08] “Realizing Fast, Scalable and Reliable Scientific Computations in Grid Environments”
Applications
Molecular Dynamics: MolDyn

- 244 molecules → 20497 jobs
- 15091 seconds on 216 CPUs → 867.1 CPU hours
- Efficiency: 99.8%
- Speedup: 206.9x → 8.2x faster than GRAM/PBS
- 50 molecules w/ GRAM (4201 jobs) → 25.3 speedup

[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

From: [SC08] "Towards Loosely-Coupled Programming on Petascale Systems"
Applications
Pharmaceuticals

Start

PDB protein descriptions

ZINC 3D structures

DOCK6 receptor

FRED receptor

NAB script parameters

BuildNABScript

NAB Script

Amber prep:
1. AmberizeLigand
2. AmberizeReceptor
3. AmberizeComplex
4. perl: gen nabscript

Amber Score:
5. RunNABScript

Select best ~5K

Select best ~5K

Select best ~500

GCMC

end

report

ligands

complexes

FRED

DOCK6

~4M x 60s x 1 cpu

~60K cpu-hrs

~10K x 20m x 1 cpu

~3K cpu-hrs

~500 x 10hr x 100 cpu

~500K cpu-hrs

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

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

<table>
<thead>
<tr>
<th>Locality</th>
<th>Number of Objects</th>
<th>Number of Files</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>111700</td>
<td>111700</td>
</tr>
<tr>
<td>1.38</td>
<td>154345</td>
<td>111699</td>
</tr>
<tr>
<td>2</td>
<td>97999</td>
<td>49000</td>
</tr>
<tr>
<td>3</td>
<td>88857</td>
<td>29620</td>
</tr>
<tr>
<td>4</td>
<td>76575</td>
<td>19145</td>
</tr>
<tr>
<td>5</td>
<td>60590</td>
<td>12120</td>
</tr>
<tr>
<td>10</td>
<td>46480</td>
<td>4650</td>
</tr>
<tr>
<td>20</td>
<td>40460</td>
<td>2025</td>
</tr>
<tr>
<td>30</td>
<td>23695</td>
<td>790</td>
</tr>
</tbody>
</table>

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

- Aggregate throughput:
  - 39Gb/s
  - 10X higher than GPFS
- Reduced load on GPFS
  - 0.49Gb/s
  - 1/10 of the original load

High data locality
- Near perfect scalability

[DADC08] “Accelerating Large-scale Data Exploration through Data Diffusion”
Falkon Project

- Falkon is a real system
  - Late 2005: Initial prototype, AstroPortal
  - January 2007: Falkon v0
  - November 2007: Globus incubator project v0.1
    - [http://dev.globus.org/wiki/Incubator/Falkon](http://dev.globus.org/wiki/Incubator/Falkon)
  - February 2009: Globus incubator project v0.9

- Implemented in Java (~20K lines) and C (~1K lines)
  - Based on the Globus Toolkit 4.0
  - Open source: svn co [https://svn.globus.org/repos/falkon](https://svn.globus.org/repos/falkon)

- Source code contributors (beside myself)
  - Yong Zhao, Zhao Zhang, Ben Clifford, Mihael Hategan, Gabriela Turcu

- Ideas contributors
  - Ian Foster, Mike Wilde, Catalin Dumitrescu, Alex Szalay, Jim Gray, …
Falkon Activity History (10 months)

Max CPUs: 163K
CPU Hours: 1.4M
Num Tasks: 164M
Task Exec: 31 sec

Allocated CPUs
Delivered Tasks

Falkon Monitoring

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

Related Work: Local Resource Management

- [Litzkow88]: Condor
- [Bode00]: Portable Batch System (PBS)
- [Zhou92]: Load Sharing Facility (LSF)
- [Gentzsch01]: Sun Grid Engine (SGE)
- [Anderson04]: BOINC volunteer computing
- [Desai05]: Cobalt

Conclusion:
- Most supported HPC and/or HTC, but due to relatively heavy-weight implementations they are not suitable for MTC
- None addressed data intensive workloads with data-aware schedulers
Related Work: Resource Provisioning

- [Banga99, Stankovic99]: Multi-Level Scheduling
- [Appleby01]: Oceano - SLA Based Management of a Computing Utility
- [Frey02, Mehta06]: Condor glide-ins
- [Walker06]: MyCluster (based on Condor glide-ins)
- [Ramakrishnan06]: Grid Hosting with Adaptive Resource Control
- [Bresnahan06]: Provisioning of bandwidth
- [Singh06]: Simulations

Conclusion: None allowed for dynamic resizing of resource pool (independent of application logic) based on system load
Related Work: Data Management

- [Ghemawat03, Dean04]: MapReduce + GFS
- [Bialecki05]: Hadoop + HDFS
- [Gu06]: Sphere + Sector
- [Tatebe04]: Gfarm
- [Chervenak04]: RLS, DRS
- [Kosar06]: Stork

Conclusions
- None focused on the co-location of storage and generic black box computations with data-aware scheduling while operating in a dynamic elastic environment
- Swift + Falkon + Data Diffusion is arguably a more generic and powerful solution than MapReduce
• There is more to HPC than tightly coupled MPI, and more to HTC than embarrassingly parallel long jobs
  – MTC: Many-Task Computing
  – Addressed real challenges in resource management in large scale distributed systems to enable MTC
  – Covered many domains (via Swift and Falkon): astronomy, medicine, chemistry, molecular dynamics, economic modelling, and data analytics
Contributions

- Identified that data locality is crucial to the efficient use of large scale distributed systems for data-intensive applications ➔ Data Diffusion
  - Integrated streamlined task dispatching with data aware scheduling policies
  - Heuristics to maximize real world performance
  - Suitable for varying, data-intensive workloads
  - Proof of O(NM) Competitive Caching
Emarrassingly Happily 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 numbers of resources and machines that may be spread over backfilling
  - Costs to run “supercomputers” per FLOP is among the best

“Impossible only means that you haven't found the solution yet.”
Anonymous

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
• More information: [http://people.cs.uchicago.edu/~iraicu/](http://people.cs.uchicago.edu/~iraicu/)

• Related Projects:
  – Falkon: [http://dev.globus.org/wiki/Incubator/Falkon](http://dev.globus.org/wiki/Incubator/Falkon)
  – Swift: [http://www.ci.uchicago.edu/swift/index.php](http://www.ci.uchicago.edu/swift/index.php)

• Dissertation Committee:
  – Ian Foster, The University of Chicago & Argonne National Laboratory
  – Rick Stevens, The University of Chicago & Argonne National Laboratory
  – Alex Szalay, The Johns Hopkins University

• People contributing ideas, slides, source code, applications, results, etc
  – Ian Foster, Alex Szalay, Rick Stevens, Mike Wilde, Jim Gray, Catalin Dumitrescu, Yong Zhao, Zhao Zhang, Gabriela Turcu, Ben Clifford, Mihael Hategan, Allan Espinosa, Kamil Iskra, Pete Beckman, Philip Little, Christopher Moretti, Amitabh Chaudhary, Douglas Thain, Quan Pham, Atilla Balkir, Jing Tie, Veronika Nefedova, Sarah Kenny, Gregor von Laszewski, Tiberiu Stef-Praun, Julian Bunn, Andrew Binkowski, Glen Hocky, Donald Hanson, Matthew Cohoon, Fangfang Xia, Mike Kubal, …

• Funding:
  – **NASA**: Ames Research Center, Graduate Student Research Program
    • Jerry C. Yan, NASA GSRP Research Advisor
  – **NSF**: TeraGrid
Publications Central to Dissertation

(2006 – 2009)


11. Ioan Raicu, Yong Zhao, Ian Foster, Alex Szalay. “Accelerating Large-scale Data Exploration through Data Diffusion”, Int. Workshop on Data-Aware Distributed Computing 2008.


17. Ioan Raicu, Ian Foster. “Harnessing Grid Resources to Enable the Dynamic Analysis of Large Astronomy Datasets: Year 1 Status and Year 2 Proposal”, GSRP, Ames Research Center, NASA, February 2007 -- Award funded 10/1/07 - 9/30/08.


22. Ioan Raicu, Ian Foster. “Harnessing Grid Resources to Enable the Dynamic Analysis of Large Astronomy Datasets”, GSRP, Ames Research Center, NASA, February 2006 -- Award funded 10/1/06 - 9/30/07.

