Using Cyberinfrastructure for Computational Operations Research

Jeff Linderoth
ISE Department
Lehigh University
jtl3@lehigh.edu

IFORS Triennial
Honolulu, Hawaii July 14, 2005

Support for this work provided in part by NSF Grant SCI-0330607

It's a BAD Title

Tutorial: Using Cyberinfrastructure for Computational Operations Research

1. Tutorial
   - Implies that you know what you are talking about
2. Operations Research
   - I really only know about Optimization.[1]
   - OR disciplines such as Simulation, Statistics, Stochastic Models, etc., won't really be touched on here
3. Cyberinfrastructure
   - Will be biased (heavily) towards the infrastructure I know most about

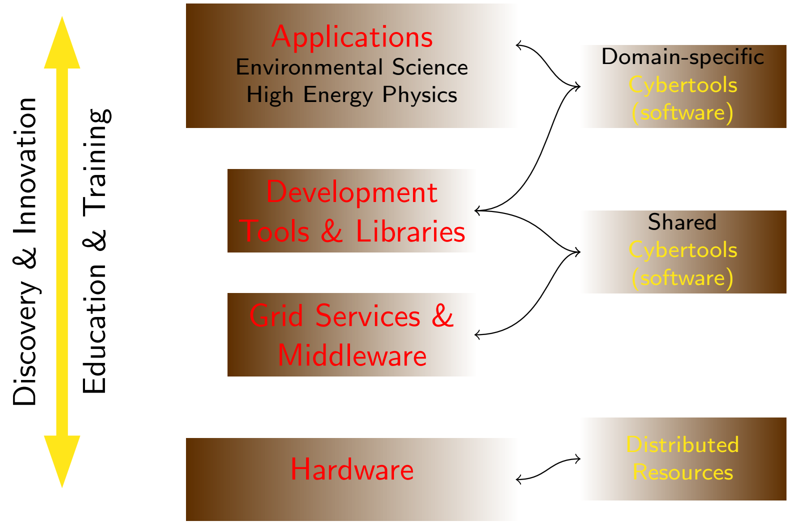
[1] And even that is debatable

Perhaps the Only Correct Slide of the Talk

Cyberinfrastructure describes the melding of tools and capabilities—hardware, middleware, software applications, algorithms, and networking—that are now transforming research and education, and are likely to do so for decades to come. There is a gathering avalanche of demand for cyberinfrastructure to suit a wide variety of needs in the open science community.

Dr. Arden L. Bement, Jr., Director, NSF

Integrated Cyberinfrastructure System



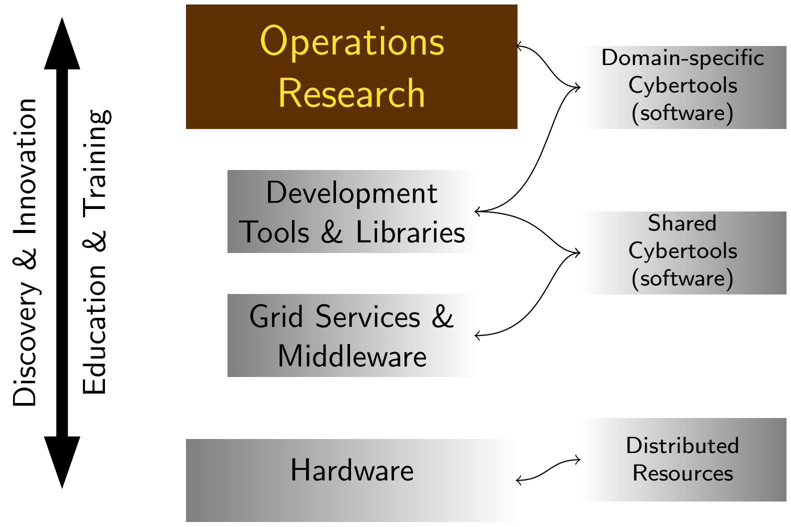
The Atkins Report

`www.cise.nsf.gov/sci/reports/atkins.pdf`

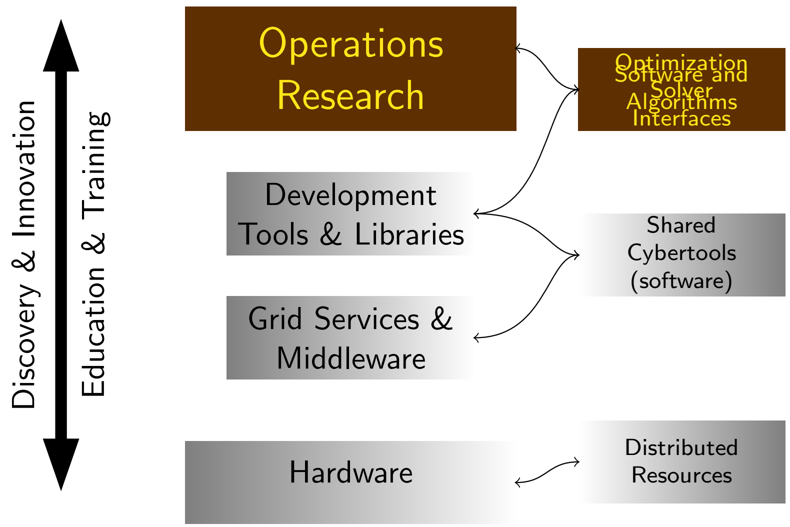
"A new age has dawned in scientific and engineering research, pushed by continuing progress in computing, information, and communication technology, and pulled by the expanding complexity, scope, and scale of today's challenges. The capacity of this technology has crossed thresholds that now make possible a comprehensive cyberinfrastructure on which to build new types of scientific and engineering knowledge environments"

- NSF CI-OR workshop:
  - `https://engineering.purdue.edu/PRECISE`
- NSF CI-Team Proposals:
  - `http://www.nsf.gov/crssprgm/ci-team/`

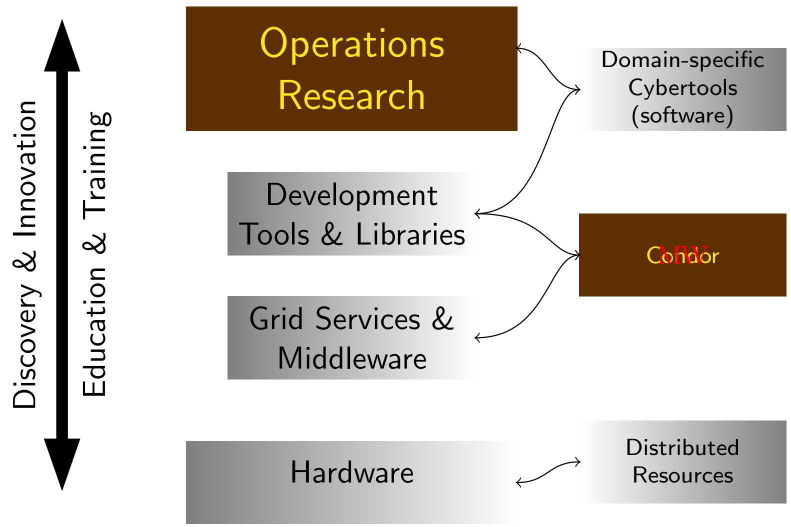
Integrated Cyberinfrastructure System



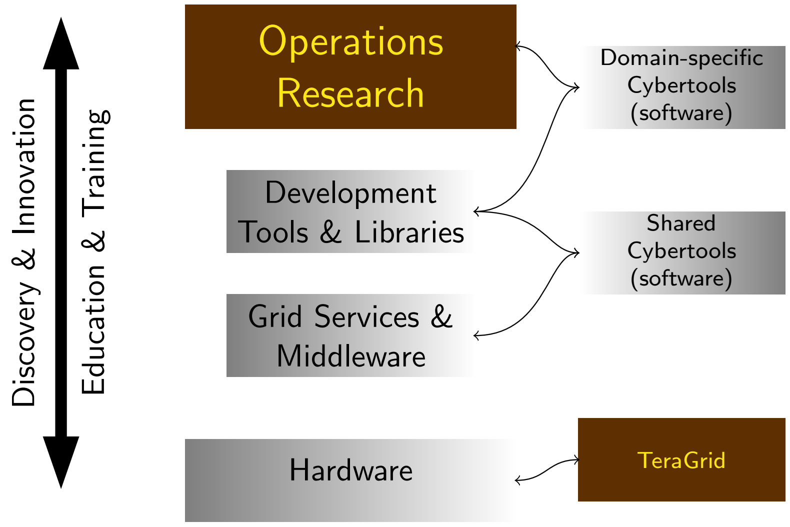
Integrated Cyberinfrastructure System



Integrated Cyberinfrastructure System



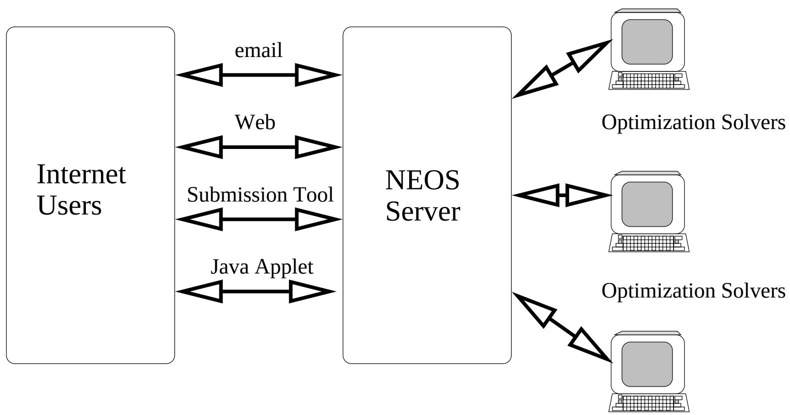
Integrated Cyberinfrastructure System



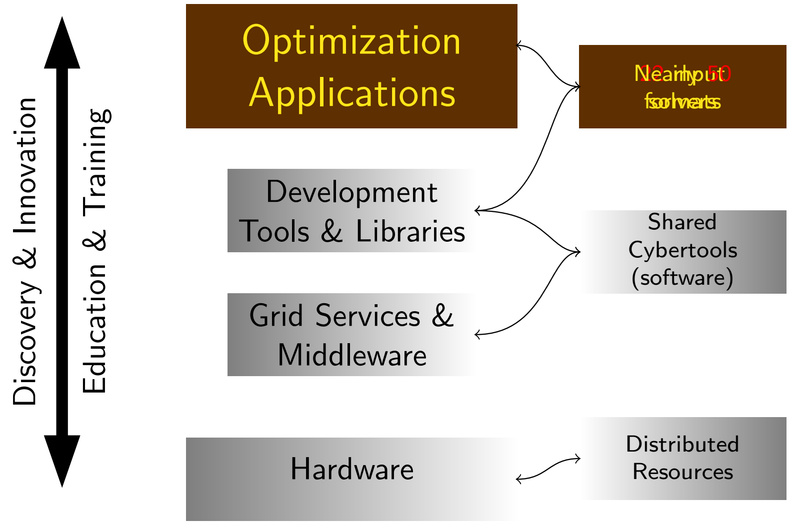
The First(?) CI-OR

- NEOS—Network Enabled Optimization System.
- www-neos.mcs.anl.gov
- An easy interface to allow users to solve their numerical optimization problems with remote resources.
- Problem can be specified in 22 different formats.
  - AMPL, Gams are most popular.
- Started in 1994. email interface to Server, based on netlib
- September 1995, Version 1.
- February 2002, Version 4. Kestrel interface.
- April, 2005. Version 5. Python, XML.

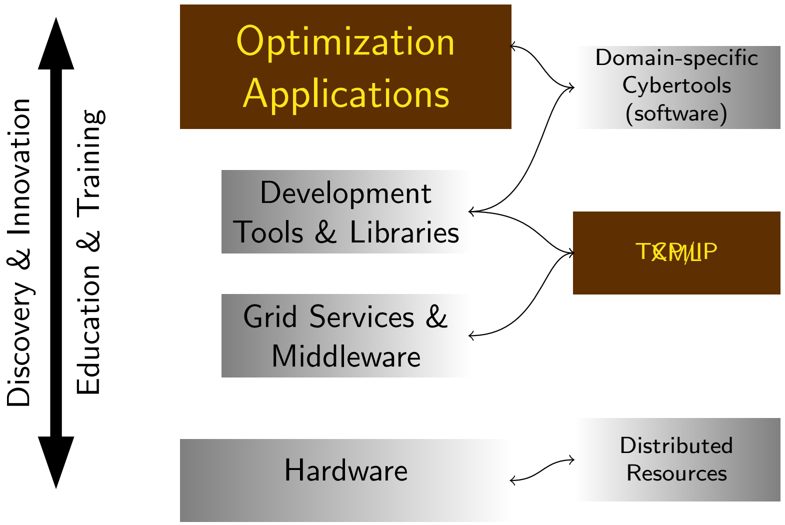
The NEOS System



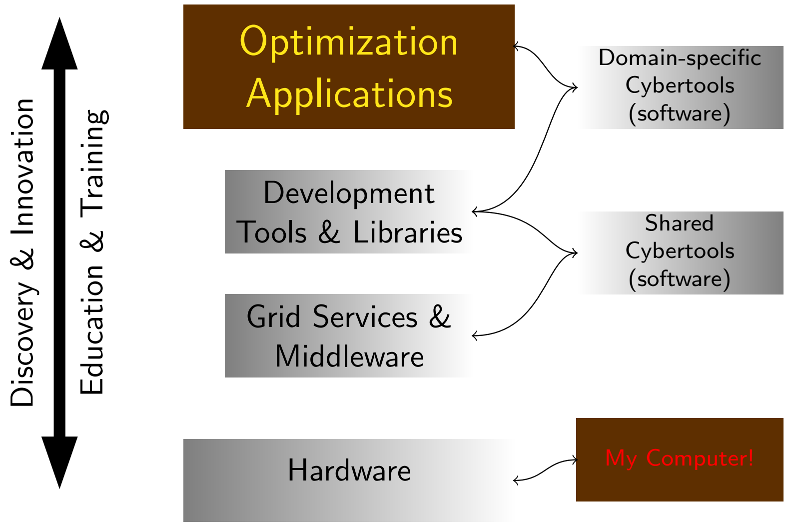
NEOS as CI



NEOS as CI



NEOS as CI



It Keeps Going

- NEOS is still going strong.
  - >12K jobs/month!
- When I show NEOS to people I usually get two reactions
  1. Wow!
  2. Is it free? (Why?)
- Should services like NEOS be free?
  - People do useful things that they otherwise wouldn't do.
  - Infrastructure often is free (think highway system) and makes a wide range of productive activities possible

An Extremely Limited Sample of NEOS Users

- We are using NEOS services for duty-scheduling for ground handling activities in a regional airport environment.
- We used NEOS to solve nonlinear optimization problems associated with models of physical properties in chemistry.
- I am dealing with ultimate limit-state analysis of large dams by means of a non-standard approach; this requires solving problems of linear and non-linear program-ming. The NEOS server is an extraordinary tool to perform parametric tests on small models, in order to choose the best suited solver.

An Extremely Limited Sample of NEOS Users

- I have been able to build and solve a prototype combinatorial auction MIP model using AMPL and NEOS in a fraction of the time it would have required me to do this had I needed to requisition a solver and install it locally.
- I am using the LOQO solver and code written in AMPL to perform numerical optimization of a spinor Bose-Einstein condensate.
- I have been working on a system for protein structure prediction. I had need to incorporate a nonlinear solver to handle packing of sidechain atoms in the protein.

WEBOPT

- `www.webopt.org`
- "A European NEOS"
- Building subscription-based, web-enabled optimization support for specific industries
  - Asset-Liability Management
  - Logistics and Supply Chain
  - Energy distribution
  - Agriculture and Environmental Planning
- Ask Gautam Mitra for more details!

COIN-OR `www.coin-or.org`

COmputational INfrastructure for Operations Research

- A consortium of researchers in industry and academia dedicated to improving the state of computational research in OR
- An initiative promoting the development and use of inter-operable open-source software for OR
- A non-profit organization known as the COIN-OR foundation

- A library of (inter-operable) software tools for optimization
- A development platform for open source projects in the OR community

COIN-OR Library

Robin Lougee-Heimer will be HAPPY to tell you more

- OSI: Open Solver Interface
- CGL: Cut Generation Library
- BCP: Branch Cut and Price
- VOL: VOLume algorithm for linear programming
- CLP: Coin Linear Programming Toolkit
- CBC: Coin Branch and Cut
- IPOPT: Interior Point OPTimizer for NLP
- NLPAPI: NonLinear Programming API
- DFO: Derivative Free Optimizer
- OTS: Open Tabu Search
- SMI: Stochastic Modeling Interface

Emerging OR Cyberinfrastructure

- LPFML
  - Bob Fourer, Leo Lopes, Kipp Martin
  - A W3C Schema for representing linear programming problem instances in XML
- NaGML
  - Gordon Bradley
  - Family of XML languages for network and graph data files
- Optimization Services
  - Bob Fourer, Jun Ma, Kipp Martin. `http://www.optimizationservices.org`
  - A framework for next generation of distributed optimization systems, relying on established web standards (XML) and services (SOAP)
- Session ~~RA-14~~, RB-14

General Cybertools

- NMI: NSF Middleware Initiative
  - Software, online services, best practices, architecture documents, policies
- Probably "best known" for its GRIDS Center Software Suite
  - Software for security, data management, execution management, monitoring
  - Globus, Condor, MPICH-G2, . . .
- `http://www.nsf-middleware.org`

Condor

THE UNIVERSITY of WISCONSIN MADISON

Peter Keller
Miron Livny
Erik Paulsen
Rajesh Raman
Marvin Solomon
Todd Tannenbaum
Doug Thain
Derek Wright

`http://www.cs.wisc.edu/condor`

What is Condor?

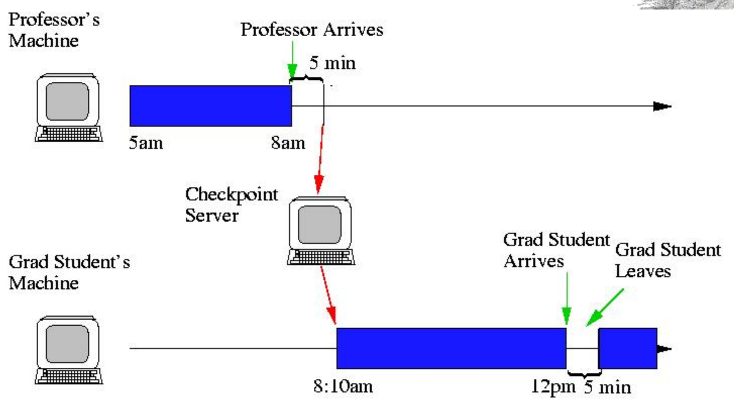
- Manages collections of "distributively owned" workstations
  - User need not have an account or access to the machine
  - Workstation owner specifies conditions under which jobs are allowed to run
  - All jobs are scheduled and "fairly" allocated among the pool
- How does it do this?
  - Scheduling/Matchmaking
  - Jobs can be checkpointed and migrated
  - Remote system calls provide the originating machines environment

Matchmaking

MyType = Job
TargetType = Machine
Owner = ralphs
Cmd = cplex
Args = seymour.d10.mps
HasCplex = TRUE
Memory ≥ 64
Rank = KFlops
Arch = x86_64
OpSys = LINUX

MyType = Machine
TargetType = Job
Name = nova9
HasCplex = TRUE
Arch = x86_64
OpSys = LINUX
Memory = 256
KFlops = 53997
RebootedDaily = TRUE

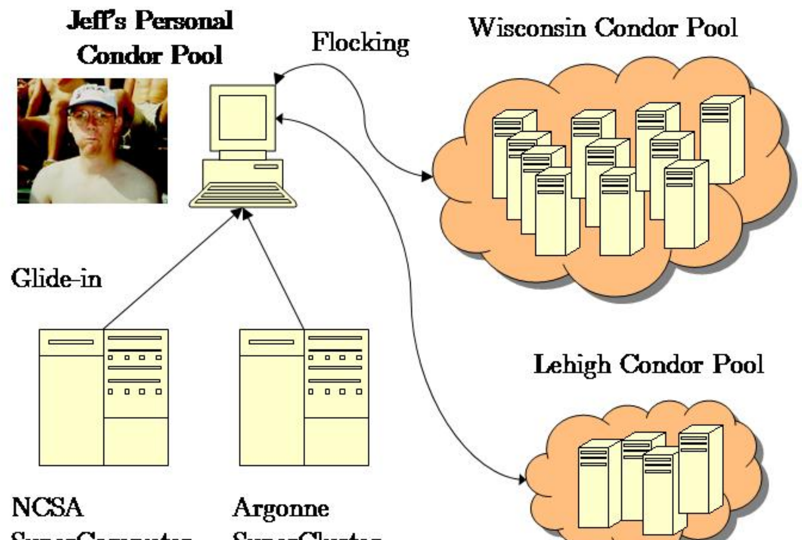
Checkpointing/Migration



Other Condor Features

- Pecking Order
  - Users are assigned priorities based on the number of CPU cycles they have recently used
  - If someone with higher priority wants a machine, your job will be booted off
- Flocking
  - Condor jobs can negotiate to run in other Condor pools.
- Glide-in
  - Globus provides a “front-end” to many traditional supercomputing sites.
  - Submit a Globus job which creates a temporary Condor pool on the supercomputer, on which users jobs may run.

Personal Condor—A Computational Grid



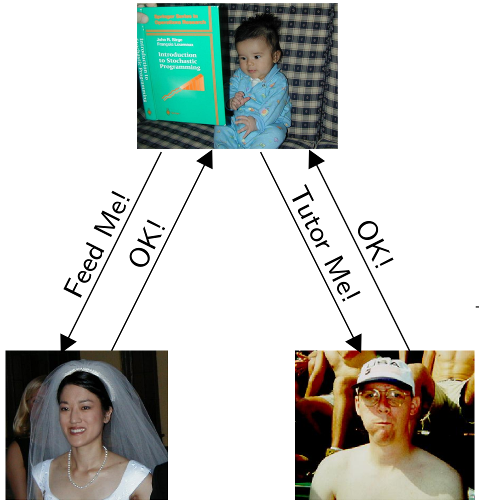
Grid-Enabling Algorithms

- Condor and "glide-in" gives us the infrastructure from which to build a grid (the spare CPU cycles),
- We still need a mechanism for controlling on optimization algorithm on a computational grid
- No guarantee about how long a processor will be available.
- No guarantee about when new processors will become available

---

- To make parallel algorithms dynamically adjustable and fault-tolerant, we could (should?) use the master-worker paradigm
- What is the master-worker paradigm, you ask?

Master-Worker!



- Master assigns tasks to the workers
- Workers perform tasks, and report results back to master
- Workers do not communicate (except through the master)

- Simple!
- Fault-tolerant
- Dynamic

MW

THE UNIVERSITY of WISCONSIN MADISON
- Wen-Han Goh
- Sanjeev Kulkarni
- Greg Thain
- Mike Yoder

ARGONNE NATIONAL LABORATORY
- Jean-Pierre Goux
- Jeff Linderoth

---

http://www.cs.wisc.edu/condor/mw

MW

- There are three abstraction is the master-worker paradigm: Master, Worker, and Task.
- MW is a software package that encapsulates these abstractions
  - C++ abstract classes
  - User writes 10 functions
  - The MWized code will transparently adapt to the dynamic and heterogeneous environment
- MW also has abstract layer to resource management and communications packages
  - Condor/PVM, Condor/Files
  - Condor/Unix Sockets
  - Single processor

But wait there's more!

- User-defined checkpointing of master
- Task Scheduling
  - MW assigns first task to first idle worker
  - Lists of tasks and workers can be arbitrarily ordered and reordered
  - User can set task rescheduling policies
- User-defined benchmarking
  - A (user defined) task is sent to each worker upon initialization
  - By accumulating normalized task CPU time, MW computes a performance statistic that is comparable between runs.

MW Classes

- MWMaster
  - `get_userinfo()`
  - `setup_initial_tasks()`
  - `pack_worker_init_data()`
  - `act_on_completed_task()`
- MWTask
  - `(un)pack_work`
  - `(un)pack_result`
- MWWorker
  - `unpack_worker_init_data()`
  - `execute_task()`

- MW is being equipped with a **black box** task and worker class—The `execute_task()` method can be to simply execute a program.

MWApplications

- MWFATCOP (Chen, Ferris, Linderoth) – A branch and cut code for linear integer programming
- MWMINLP (Goux, Leyffer, Nocedal) – A branch and bound code for nonlinear integer programming
- MWQPBB (Linderoth) – A branch-and-bound code for nonconvex quadratic programming
- MWAND (Linderoth, Shen) – A (preliminary) nested-decomposition code for multistage stochastic linear programming
- MWATR (Linderoth, Shapiro, Wright) – A (trust-region) cutting plane code for linear stochastic programming and verification of solution quality
- MWQAP (Anstreicher, Brixius, Goux, Linderoth) – A branch and bound code for solving the quadratic assignment problem

Stochastic Programming Collaborators

Alex Shapiro
ISyE
Georgia Tech

Steve Wright
Computer Science Department
University of Wisconsin-Madison

Stochastic Programming

A Stochastic Program

$$\min_{x \in X} \{ f(x) \stackrel{\text{def}}{=} \mathbb{E}_P F(x, \omega) \}$$

- $F(x,\omega)$ is a real-valued function of two vector variables.
- $x \in \mathbb{R}^n$ and $\omega \in \mathbb{R}^d$
- ω is a random vector having probability distribution P
- $X \subseteq \mathbb{R}^n$

A Two-Stage Stochastic LP

$$\min_{x \geq 0, Ax = b} c^T x + \mathcal{Q}(x)$$

- $\mathcal{Q}(x) \stackrel{\text{def}}{=} \mathbb{E}_P[Q(x, \xi(\omega))]$
- $Q(x, \xi(\omega))$ is the optimal value of the recourse problem

$$\min q^T y$$

$$\begin{aligned} Wy &= h(\omega) - T(\omega)x \\ y &\geq 0 \end{aligned}$$

Two-Stage Stochastic Linear Programming

- We assume that the P has finite support, so $\xi(\omega)$ has a finite number of possible realizations (scenarios):

$$\mathcal{Q}(x) = \sum_{i=1}^{N} p_i Q(x, \xi_i)$$

- For a partition of the N scenarios into chunks $\mathcal{N}_1, \mathcal{N}_2, \ldots \mathcal{N}_t$, let $\mathcal{Q}_{[j]}(x)$ be the contribution of the jth chunk to $\mathcal{Q}(x)$:

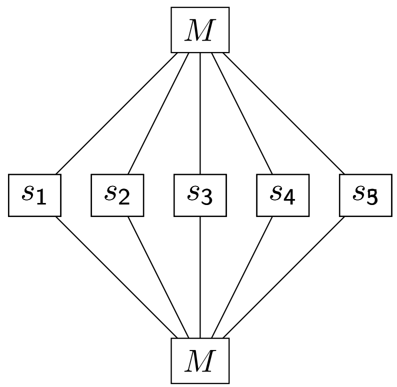
$$\mathcal{Q}_{[j]}(x) \stackrel{\text{def}}{=} \sum_{i \in \mathcal{N}_j} p_i Q(x, \xi_i)$$

- $\mathcal{Q}(x) = \sum_{j=1}^{t} \mathcal{Q}_{[j]}$

Important (and well-known) Facts

- $Q(x, \xi_i)$, $\mathcal{Q}_{[\cdot]}(x)$, and $\mathcal{Q}(x)$ are piecewise linear convex functions of x.
- If π_i is an optimal dual solution to the linear program corresponding to $Q(\hat{x}, \xi_i)$, then $-T_i^T \pi_i \in \partial Q(\hat{x}, \xi_i)$
  - $g_j(\hat{x}) \stackrel{\text{def}}{=} \sum_{i \in \mathcal{N}_j} -p_i T_i^T \pi_i \in \partial \mathcal{Q}_{[j]}(\hat{x})$.
- Represent $\mathcal{Q}_{[j]}(x)$ by an artificial variable θ_j and find supporting planes for θ_j
  - $\theta_j \geq g_j(x^k)^T x + (\mathcal{Q}_{[j]}(x^k) - g_j^T x^k)$ (*)
- Evaluation of $\mathcal{Q}(\hat{x})$ is separable
- We can solve linear programs corresponding to each $Q(\hat{x}, \xi_i)$ independently – in parallel!

Multicut L-shaped method



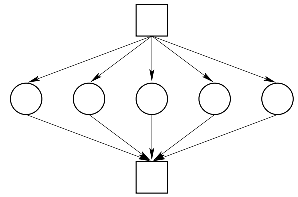
1. Solve the **master problem** M with the current θ_j-approximations to $\mathcal{Q}_{[j]}(x)$ for x^k.
2. Solve the **subproblems**, (s_j) evaluating $\mathcal{Q}_{[j]}(x^k)$ and obtaining a subgradient $g_j(x^k)$. Add inequalities (*) to the master problem
3. k = k+1. Goto 1.

MWImplementation

- Work
  - One or more scenario chunks $\mathcal{N}_{j_1}, \ldots \mathcal{N}_{j_C}$ and point $(\hat{x})$
- Result
  - A subgradient of each of the $\mathcal{Q}_{[j_k]}(\hat{x})$.
- `act_on_completed_task`
  - Add subgradient inequalities to master problem
  - Solve master problem if all workers have reported their results for the iteration

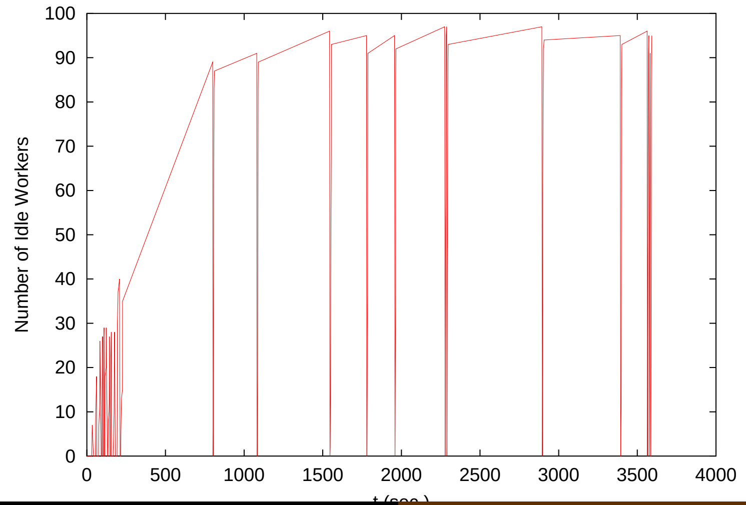
Headaches!

- Solving the master problem is a "synchronization point" of the algorithm
  - Amdahl's Law: Parallel efficiency is limited by the amount of synchronization.



- This synchronization problem is MUCH worse in Computational Grid computing environments!

Worker Usage—Number of Idle Workers



Stamp Out Synchronicity!

- On a Grid, different processors act at different speeds,
- Many may wait idle for the "slowpoke"
- Even worse, grid computing tools can fail to inform the user that their worker has failed!

Asynchronicity is key!

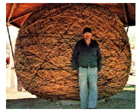
Asynchronous methods are preferred for traditional parallel computing environments. They are nearly required for Grid Computing environments!

ATR – An Asynchronous Trust Region Method

- Keep a “basket” $\mathcal{B}$ of trial points for which we are evaluating the objective function
- Make decision on whether or accept new iterate x^{k+1} after entire $\mathcal{Q}(x^k)$ is computed
- Populate the basket quickly by initially solving the master problem after only α% of the scenario LPs have been solved
- Greatly reduces the synchronicity requirements
- Might be doing some “unnecessary” work – the candidate points might be better if you waited for complete information from the preceding iterations

What is CI
Domain Specific Cybertools
Shared Cybertools
Stochastic Programming
Quadratic Assignment Problem
Using Distributed Resources
NEOS
WebOpt
Coin-OR

The World’s Largest LP



- Storm – A stochastic cargo-flight scheduling problem (Mulvey and Ruszczyński)
- We aim to solve an instance with 10,000,000 scenarios
- $x \in \Re^{121}, y_k \in \Re^{1259}$
- The deterministic equivalent LP is of size

$$A \in \Re^{985{,}032{,}889 \times 12{,}590{,}000{,}121}$$

Jeff Linderoth Cyber-OR

What is CI
Domain Specific Cybertools
Shared Cybertools
Stochastic Programming
Quadratic Assignment Problem
Using Distributed Resources
NEOS
WebOpt
Coin-OR

The Super Storm Computer

Number	Type	Location
184	Intel/Linux	Argonne
254	Intel/Linux	New Mexico
36	Intel/Linux	NCSA
265	Intel/Linux	Wisconsin
88	Intel/Solaris	Wisconsin
239	Sun/Solaris	Wisconsin
124	Intel/Linux	Georgia Tech
90	Intel/Solaris	Georgia Tech
13	Sun/Solaris	Georgia Tech
9	Intel/Linux	Columbia U.
10	Sun/Solaris	Columbia U.
33	Intel/Linux	Italy (INFN)
1345		

A Sad Fact of Life

Very few optimization software vendors want to give me 1000’s of licenses

Cyberinfrastructure to the rescue

We must have access to quality, open components from which to build our algorithms.

Hooray for COIN!!!

Jeff Linderoth Cyber-OR

What is CI
Domain Specific Cybertools
Shared Cybertools
Stochastic Programming
Quadratic Assignment Problem
Using Distributed Resources
NEOS
WebOpt
Coin-OR

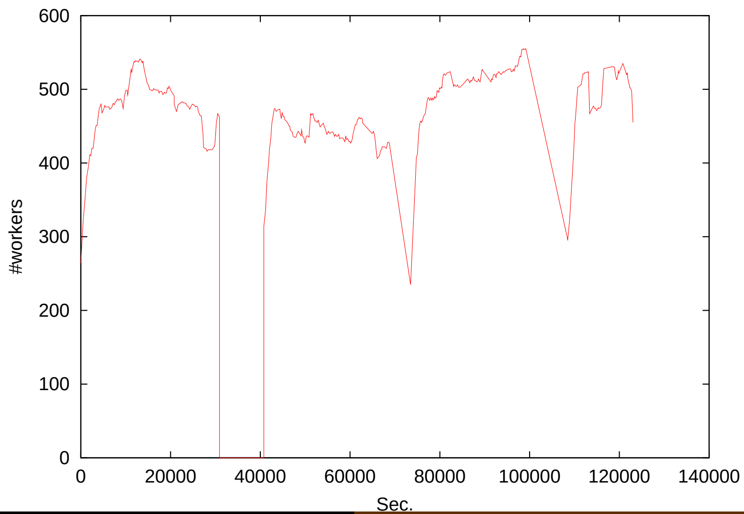
TA-DA!!!!!

Storm is solved

Wall clock time	31:53:37
CPU time	1.03 Years
Avg. # machines	433
Max # machines	556
Parallel Efficiency	67%
Master iterations	199
CPU Time solving the master problem	1:54:37
Maximum number of rows in master problem	39647

Jeff Linderoth Cyber-OR

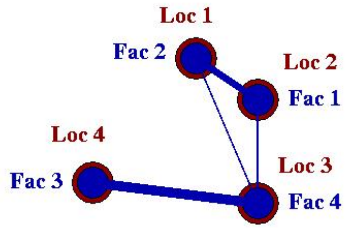
Number of Workers



The Quadratic Assignment Problem

Mathematical Formulation

$$\min_{\pi \in \Pi} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\pi(i)\pi(j)} + \sum_{i=1}^{n} c_{i\pi(i)}$$



- Assign facilities to locations
- QAP is NP-"Super"-Hard
- Branch and Bound is the method of choice, but very few tight, computable, bounds exist

QAP Collaborators

Kurt Anstreicher
University of Iowa

Nate Brixius
Micro$oft

Jean-Pierre Goux
Argonne, Northwestern, and Artelys

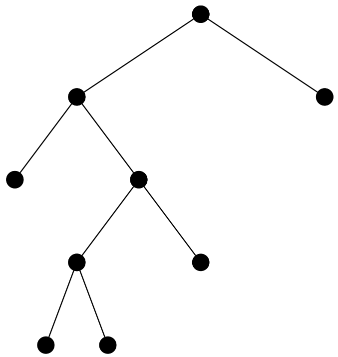
Branch and Bound

- Kurt Anstreicher and Nate Brixius showed that the solution to the following problem provides a lower bound to the solution of QAP:

$$\begin{array}{ll} \min & f(X) \equiv \textbf{vec}(X)^T Q \, \textbf{vec} X + C \bullet X \\ \text{such that} & Xe = X^T e = e, \qquad X \geq 0. \end{array}$$

- $Q \equiv (B \otimes A) - (I \otimes S) - (T \otimes I)$
- S and T are obtained from the spectral decompositions of A and B
- There are more details
- This is a convex quadratic programming problem relaxation

Tree-Based Computations



- Feasible solution $\Rightarrow$ upper bound
- Relaxed problem $\Rightarrow$ lower bound

Branch-and-Bound

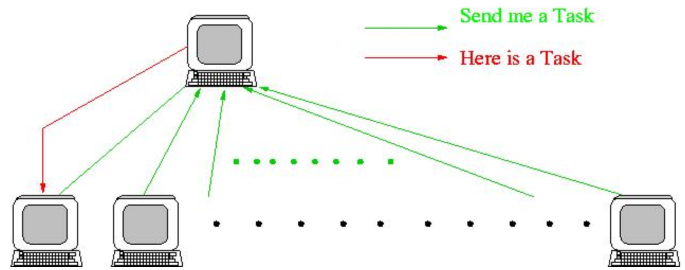
1. Is solution to relaxed problem feasible?

   Yes? YAHOO!

   No? Break problem into smaller pieces. Goto 1.

MW Implementation

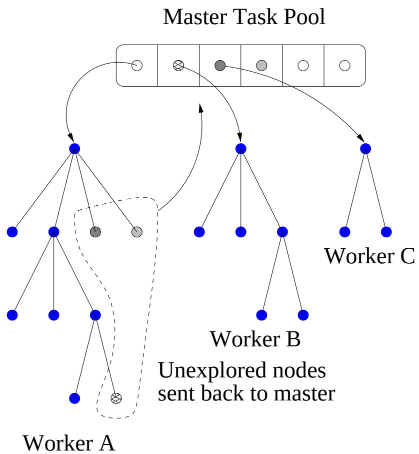
- Fitting the B & B algorithm into the master-worker paradigm is not ground-breaking research
- We must avoid "contention" at the master



All The Queueing Theory I Know

- We can reduce contention in two ways
  1. Increase the service rate
  2. Reduce the arrival rate
- A parallel depth-first oriented strategy achieves these goals.
  - Available worker is given "deepest" node by master
  - Worker examines the subtree rooted at this node in a depth-first fashion for t seconds.

Parallel Depth-First Search



- Other "standard" search strategies fail completely!
  - Too much memory required at master
  - Too many nodes passed back to master
- Don't try this at home!
  - If you don't have a good upper bound with which to fathom, this can fail miserably!

Truth in Advertising

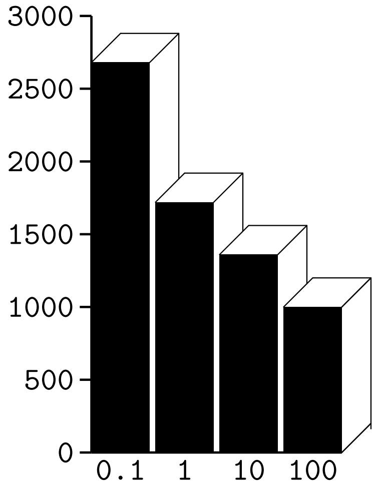
- The parallel depth-first search strategy is awful too!
- Define the parallel efficiency:

$$\eta = \frac{\Sigma(\text{Time workers spend executing tasks})}{\Sigma(\text{Time workers are available})}$$

- In our initial implementation, $\eta = 0.41$
- Since there is very little synchronization required in the algorithm, this number is shockingly low!

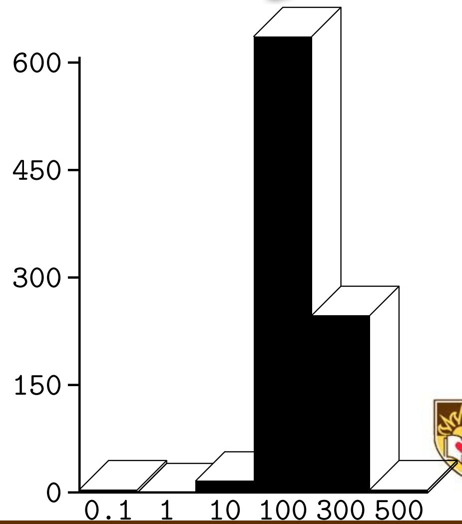
Deducing the Problem

- We may want the workers to examine a subtree for t seconds, but that doesn't mean that there are t seconds of work!
- A histogram of task times:



Elementary, Dear Watson

- Make sure that workers only pass back nodes that will have enough "meat"
  - Order children so that "easy" ones are first in the DFS stack
  - Allow additional time for workers to pop up the DFS stack, finishing off remaining easy nodes.
- η improved to 0.9



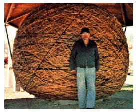
The Holy Grail

- (NUG30) ($n = 30$) has been the "holy-grail" of computational QAP research for $>$ 30 years
- In 2000, Anstreicher, Brixius, Goux, & Linderoth set out to solve this problem
- Using an old idea of Knuth, we estimated the CPU time required to solve NUG30 to be 5-10 years on a fast workstation
- We'd better get a pretty power computing platform!

Our Computational Grid

Number	Type	Location
414	Intel/Linux	Argonne
96	SGI/Irix	Argonne
1024	SGI/Irix	NCSA
16	Intel/Linux	NCSA
45	SGI/Irix	NCSA
246	Intel/Linux	Wisconsin
146	Intel/Solaris	Wisconsin
133	Sun/Solaris	Wisconsin
190	Intel/Linux	Georgia Tech
94	Intel/Solaris	Georgia Tech
54	Intel/Linux	Italy (INFN)
25	Intel/Linux	New Mexico
5	Intel/Linux	Columbia U.
10	Sun/Solaris	Columbia U.
12	Sun/Solaris	Northwestern
2510		

NUG30 is solved!

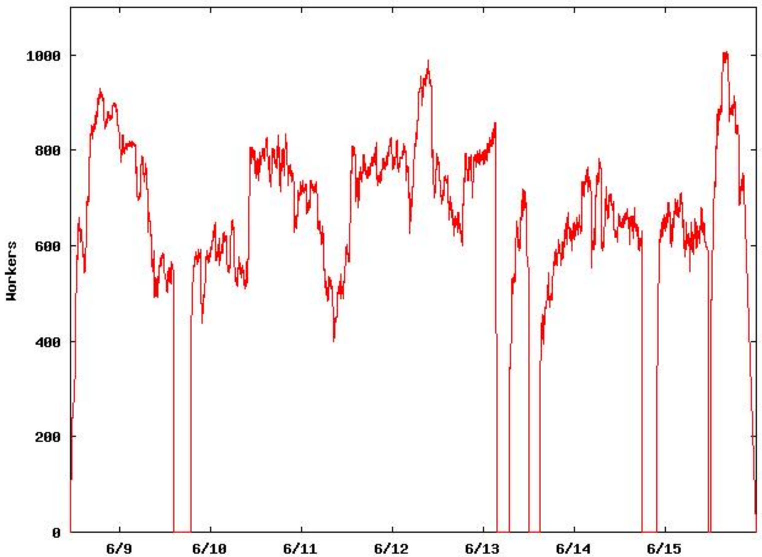


14, 5, 28, 24, 1, 3, 16, 15, 10, 9, 21, 2, 4, 29, 25, 22, 13, 26, 17, 30, 6, 20, 19, 8, 18, 7, 27, 12, 11, 23

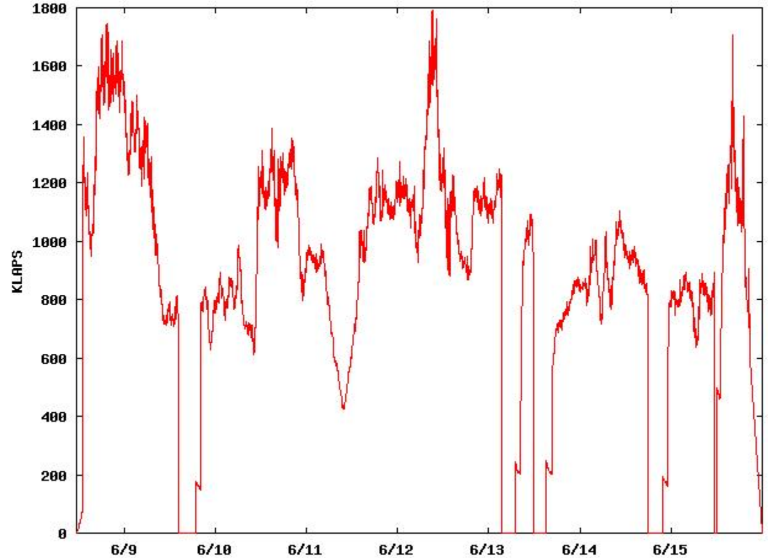
"My father used 3.46×10^8 CPU seconds, and all I got was this lousy permutation"

Wall Clock Time:	6:22:04:31
Avg. # Machines:	653
CPU Time:	≈ 11 years
Nodes:	11,892,208,412
LAPs:	574,254,156,532
Parallel Efficiency:	92%

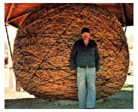
Workers



KLAPS



Even More Wasted CPU Time

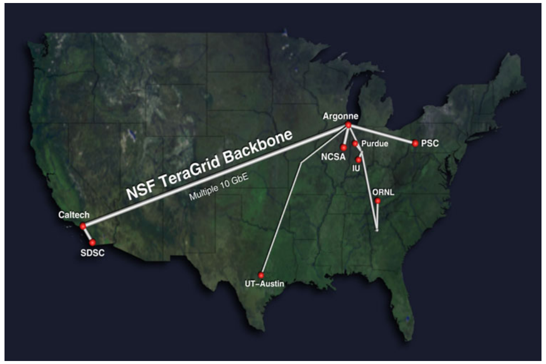


	KRA30B	KRA32	THO30
Wall Clock Time (Days)	3.79	12.3	17.2
Avg. # Machines	462	576	661
Max. # Machines	780	1079	1307
CPU Time (Years)	4.32	15.2	24.7
Nodes	5.14×10^9	16.7×10^9	34.3×10^9
LAPs	188×10^9	681×10^9	1.13×10^{12}
Parallel Efficiency:	92%	87%	89%

Getting Started with CI

Where do I get my 1000 Processors?

- The Teragrid: http://www.teragrid.org



The Teragrid

- Over 15 TeraFLOPS!
- Dozens of Petabytes of online and archival storage
- 30Gbps backbone

Site	#	Type
SDSC	608	Itanium, Power-4
NCSA	2798	Itanium, Altix
UC/ANL	316	Itanium, Xeon
CACR	104	Itanium
PSC	5248	Alpha
Purdue	1280	Pentium, Power-3
TACC	974	Pentium, Ultra-Sparc
	11328	

Tutorial?

Cyberinfrastructure

Computer hardware, software, standards, and interfaces that engender a broad range of productive activities

OR Cyberinfrastructure

- COIN-OR: http://www.coin-or.org
- NEOS: http://www-neos.mcs.anl.gov

Shared Cyberinfrastructure

- NMI: http://www.nsf-middleware.org
- Condor: http://www.cs.wisc.edu/condor
- MW: http://www.cs.wisc.edu/condor/mw

Putting it all together

Distributed Resources

- The Teragrid: `http://www.teragrid.org`

The Upshot

- You can put all of these components together to solve BIG problems
- We still need to use our OR expertise to engineer the algorithms for the computational platform

Using CI for Optimization?

- Applications well-suited for (this generation of) CI:
  - Compute-intensive rather than data intensive
  - Asynchronous
  - Where increase in processors will lead to larger problems being solved.

1. Stochastic Programming. Robust optimization.
   - Algorithm complexity does not increase exponentially in problem size.
2. Pattern Search Methods
   - Expensive "black box" function evaluations farmed out to processors on the grid.
3. Expand the scope of problems that can be solved in very difficult classes:
   - Global optimization, Mixed integer nonlinear programming, Stochastic Integer Programming.

We want YOU

to use the emerging cyberinfrastructure

With CI, the Science of Better can do Better Science!

- Slides will be posted at COR@L: Center for Optimization Reseach @ Lehigh: `http://coral.ie.lehigh.edu`
- `mailto:jtl3@lehigh.edu`