Performance Analysis of Grid-Enabled GAMS

Michael R. Bussieck
MBussieck@gams.com
GAMS Software GmbH
http://www.gams.de

Steven P. Dirkse
sDirkse@gams.com
GAMS Development Corp.
http://www.gams.com

Armin Pruessner
aPruessner@gams.com
GAMS Development Corp.
http://www.gams.com

INFORMS 2006
Pittsburgh November 6, 2006

Welcome/Agenda

- GAMS Grid Computing Facility
- Performance World / PAVER
- Computational Experiments
- Results & Summary

Agenda

- GAMS Grid Computing Facility
- Performance World / PAVER
- Computational Experiments
- Results & Summary

What's New???

- Improvements on all frontiers
  - Connectivity Tools
    - Databases
    - Spreadsheets
    - Specialized Visualization Tools (e.g. VEDA)
  - Productivity Tools
    - IDE Improvements
    - Charting Engine
  - Interfaces
    - Using GAMS from Application Environments
  - Solver Interfacing
    - Branch-and-Cut-and-Heuristic (BCH) Facility
    - Grid Computing

What is Grid Computing?

A pool of connected computers managed and available as a common computing resource

- Effective sharing of CPU power
- Massively parallel task execution
- Scheduler handles management tasks
- E.g. Condor, Sun N6 Grid Engine, Globus
- Can be rented or owned in common
- Licensing & security issues

Typical Application for GAMS & Grid

```
mymodel.solvelink=3;
loop(scenario,
  demand=d(scenario); cost=scost(scenario);
  solve mymodel min obj using minlp;
  report(scenario) = var.l; ;

Repeat
  loop(scenario$h(scenario),
    if(handlestatus(h(scenario))=2,
      mymodel.handle=h(scenario); h(scenario)=0;
      execute_loadhandle mymodel;
      report(scenario)=var.l ));
  if(card(h), execute 'sleep 1');
until card(h)=0 or timeelapsed > 100;
```

Agenda

- GAMS Grid Computing Facility
- Performance World / PAVER
- Computational Experiments
- Results & Summary

Performance World

Performance World

Welcome to the Performance World!

Performance World is a forum for discussion and dissemination of information and tools about all aspects of performance testing of solvers for mathematical programming problems. This world has been established in response to user demands for independent and reproducible performance results.

Overall performance highly depends on problem formulation, solver, and tuning parameters. Our performance tools are designed to serve the different needs of our user community. One user may be interested in finding the most reliable way to solve a proprietary or classified model. On the other hand, an academic researcher may be interested in testing a new algorithm against a set of existing test problems and competing approaches. The main features are:

- Uniform access to a comprehensive set of established and new test problems
- Automation tools for collecting performance measurements
- Tools for analyzing and visualizing test results

Editorial Board

PerformanceLib

Performance Tools

Performance List

Related Links

Search

Contact

What's New:

- Try our online PAVER Server for automated performance analysis and visualization, batch file creation and model translation
- New tools for analyzing non-convex or discrete models
- MINLP type models from the MINLP World have been added to the PerformanceLib

Performance Libraries

- Performance tests require public test libraries
  - Creating models for this is not feasible
  - Shared test libraries allow reproducible results

- PerformLIB contains multiple MIP libraries
  - FCNETLIB
  - *MIPLIB*
  - *MittelmannMIP*
  - *NEOS*

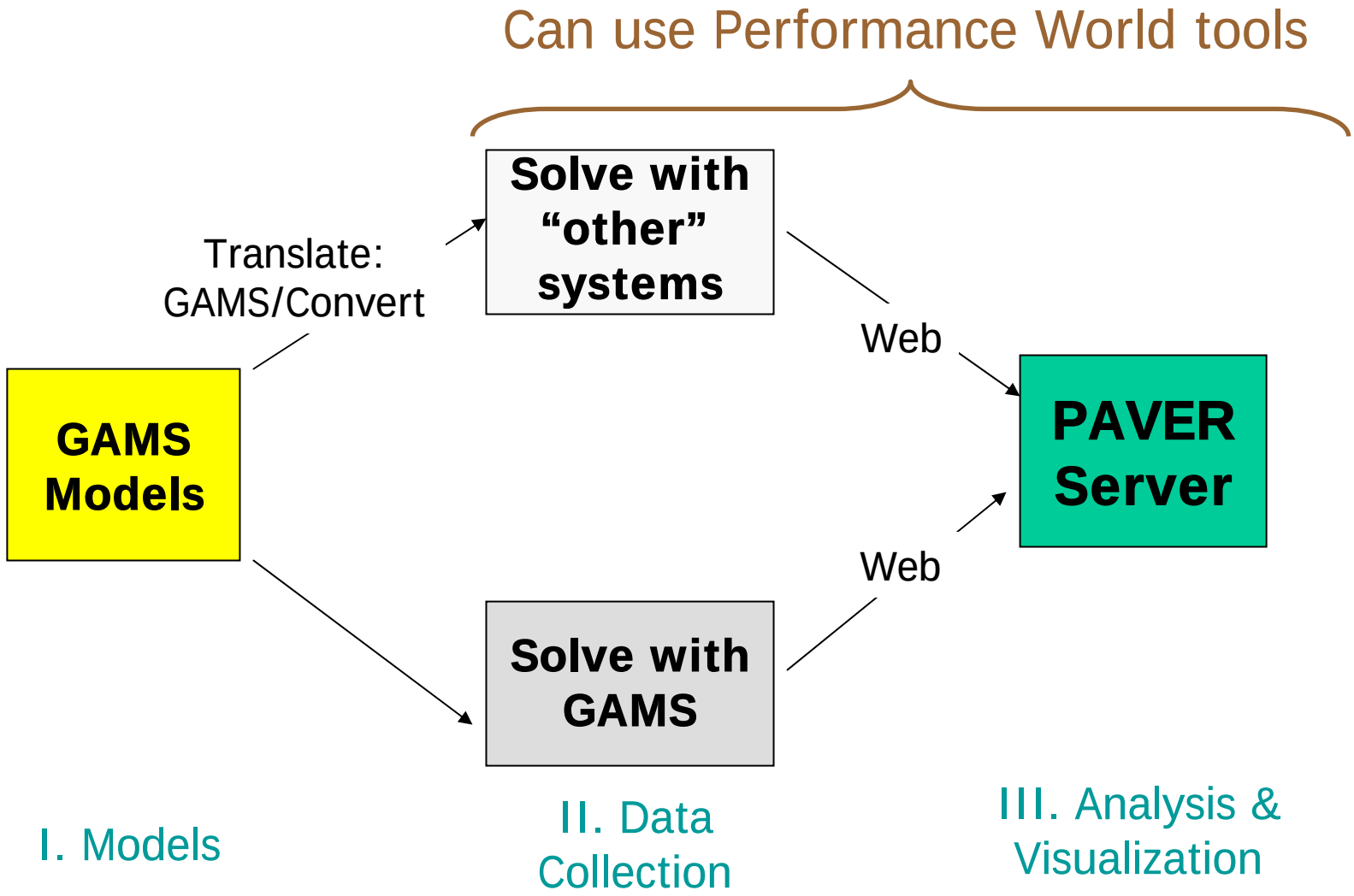
Tools: Performance Analysis

- Different objectives:
  - Solver robustness and correctness
  - Solver efficiency
  - Quality of solution (nonconvex and discrete models)

→Tools are GAMS independent

→Results in HTML format: platform independent

Open Testing Architecture



PAVER Server

- PAVER server (**P**erformance **A**nalysis and **V**isualization for **E**ffortless **R**eproducibility)

  www.gamsworld.org/performance/paver

- Online server to facilitate performance testing and analysis/visualization

- Results sent via e-mail in HTML format
  – System independent

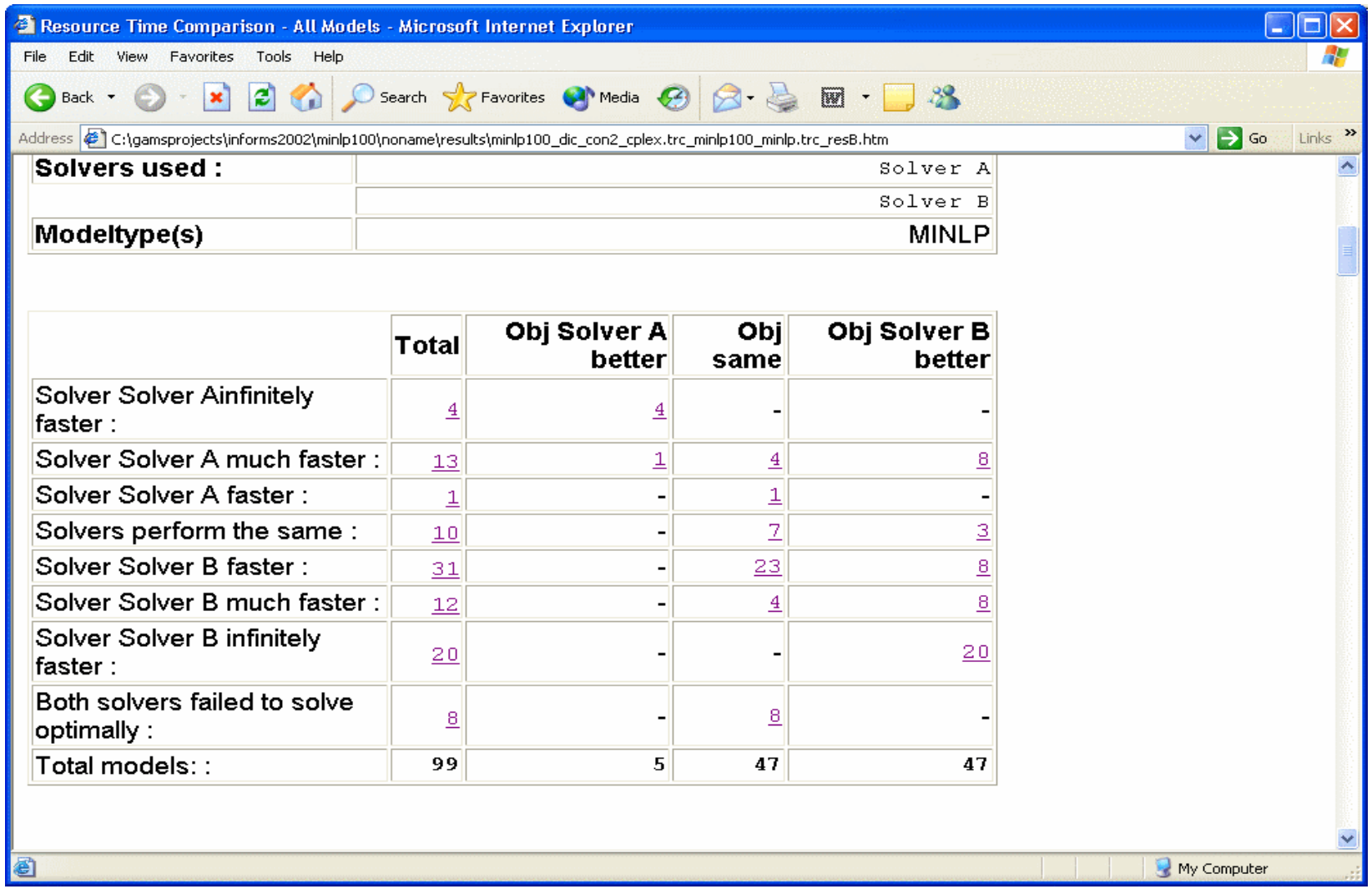
Tools: Efficiency

Resource Time Utility:

- Cross comparison of solver resource times of two solvers
- Ratios of resource times

→ Can use online using PAVER

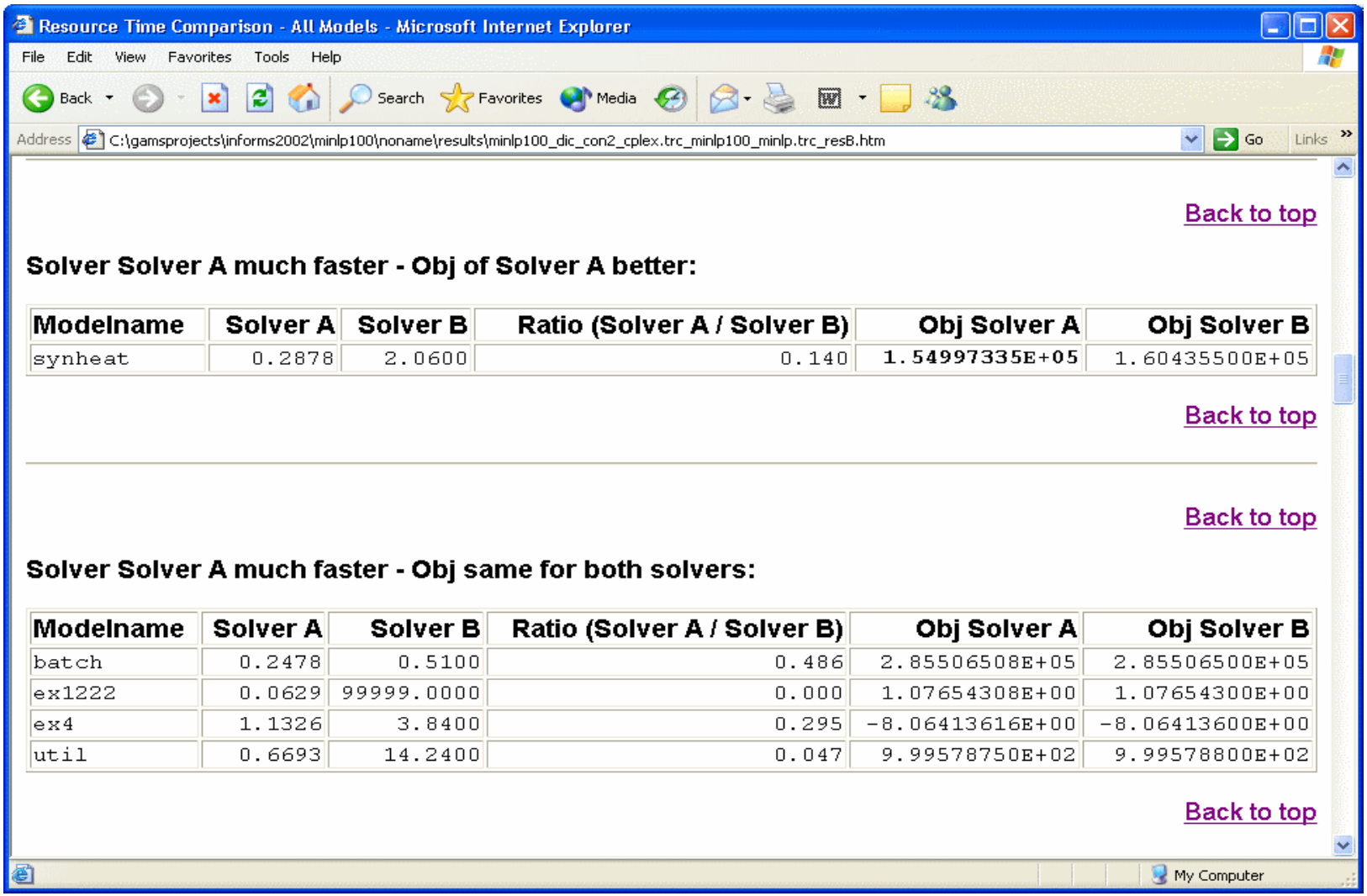
PAVER: Solver Resource Time



Solvers used :	Solver A
	Solver B
Modeltype(s)	MINLP

	Total	Obj Solver A better	Obj same	Obj Solver B better
Solver Solver Ainfinitely faster :	4	4	-	-
Solver Solver A much faster :	13	1	4	8
Solver Solver A faster :	1	-	1	-
Solvers perform the same :	10	-	7	3
Solver Solver B faster :	31	-	23	8
Solver Solver B much faster :	12	-	4	8
Solver Solver B infinitely faster :	20	-	-	20
Both solvers failed to solve optimally :	8	-	8	-
Total models: :	99	5	47	47

PAVER: Resource Time (cont.)

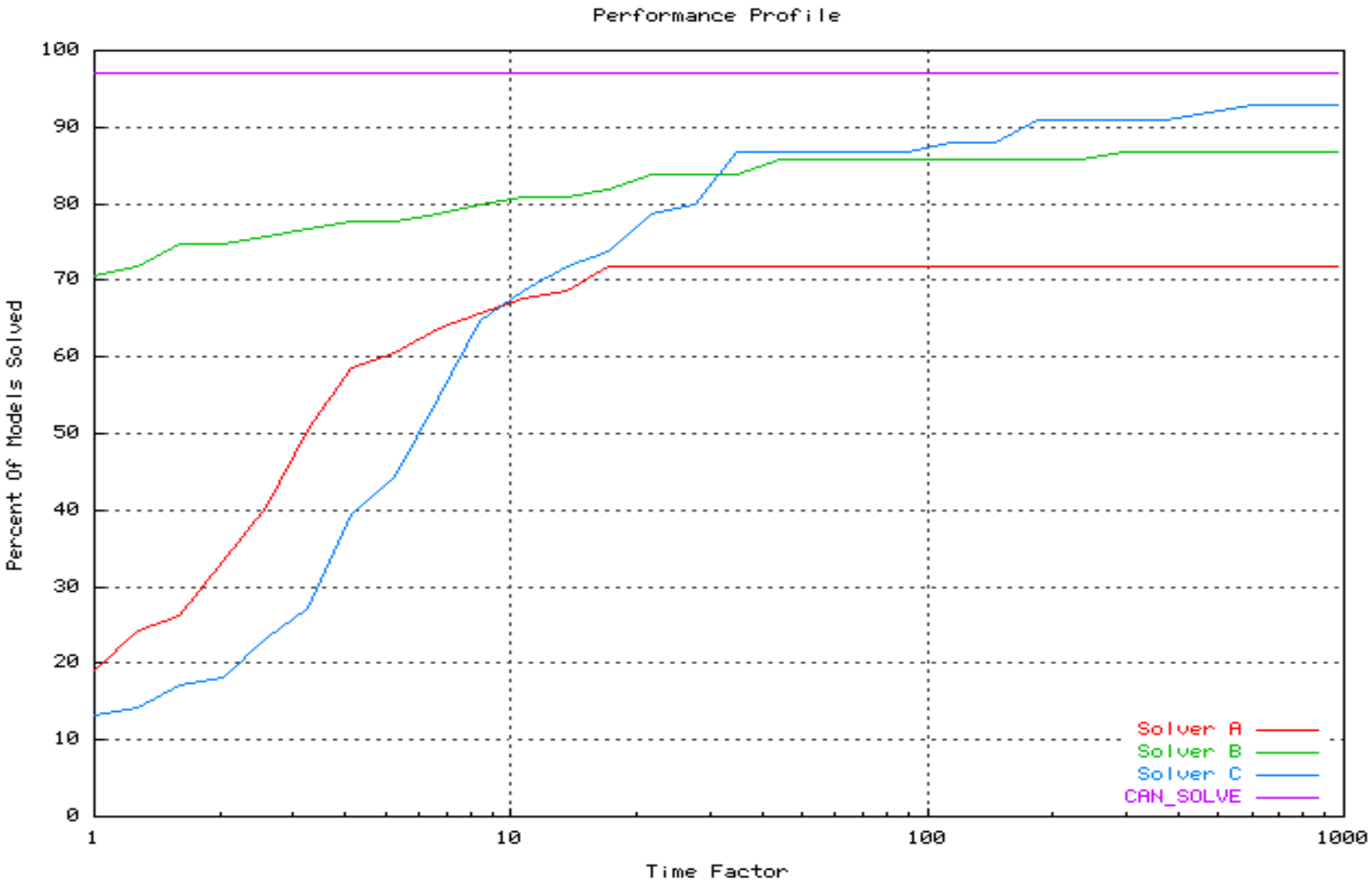


Tools: Visualization

Performance Profiles (Dolan and More, 2002):

- Cumulative distribution function for a performance metric
- Performance metric: ratio of current solver time over best time of all solvers
- Intuitively: probability of success if given τ times fastest time (τ=ratio)

Profiles (best resource time)



Agenda

- GAMS Grid Computing Facility
- Performance World / PAVER
- Computational Experiments
- Results & Summary

Parallel MIP

- MIP/B&C Algorithm ideal to parallelize
  - Multi-threaded (SMP, shared memory) B&B
    - E.g. CPLEX, XPRESS
  - Master/Worker Paradigm (process nodes in parallel)
    - Software: FATCOP/Condor, BCP/PVM, PICO/MPI
  - A-priori subdivision into *n* independent problems
    - Seymour problem solved that way
    - Open Pit Mining (openpit in GAMS Model library)
      - Partitioning integer variables to subdivide model into 4096 sub-problems
      - Experiments (Ferris) at UW using Condor Pool

Problems with a-priori Partitioning

- 99% of sub-problems very easy to solve
- 1% (almost) as difficult as the original problem

- How can we find *n* sub-problems with similar (but reduced) level of difficulty?
  - B&C Code keeps a list of *open/unexplored* nodes
  - Problem-bounds of these open nodes represent partitioning of the original problem

Node	Nodes Left	Objective	IInf	Best Integer	Best Node	Cuts/ ItCnt	Gap
0	0	29.6862	64		29.6862	165	
100	37	17.0000	14		25.0000	2230	
200	70	21.8429	22		24.0000	4022	

- GAMS/CPLEX Option **`dumptree n`** creates *n* bound files

Putting it all together

```
Generate n=64 sub-problems using GAMS/CPLEX with
  dumptree 64;

loop(n,
  load nth bound file;
  generate and submit nth sub-problem
);

Repeat
  loop(n$(not collected),
    if (n finished,
      load nth-solution and mark n as collected));
  sleep some time;
Until all collected;
```

Communication & Strategy

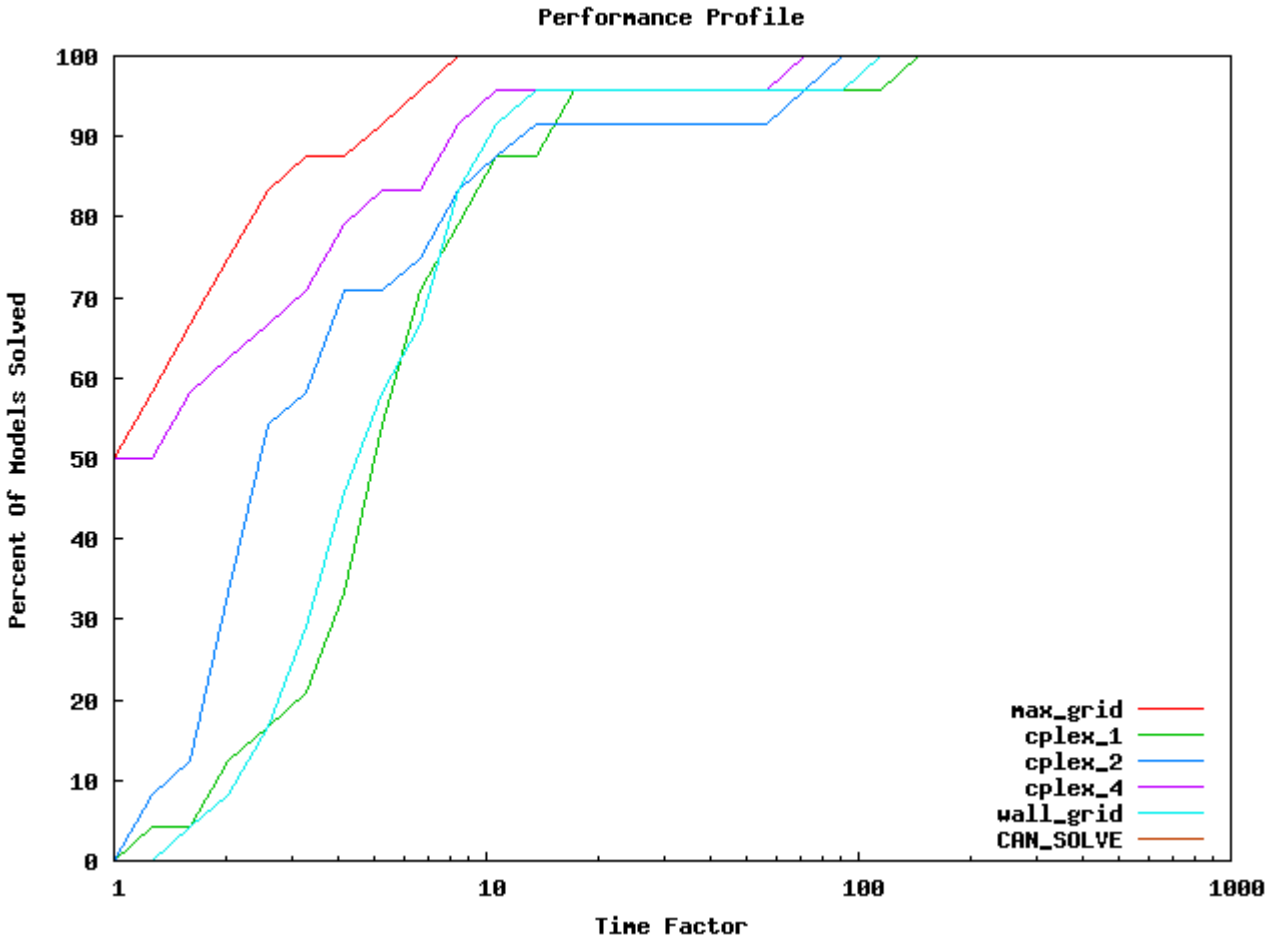
- An incumbent solution allows one to prune nodes with larger LP solution value in all sub-problems.
- Ergo, communicate a newly found incumbent to all sub-problems
  - Sub-problems not started: Start with a **`cutoff`**
  - Running sub-problems: Update the **`cutoff`** with a GAMS/CPLEX option file that is read while running

- Strategy:
  - Have one machine working on good solutions (e.g. CPLEX **`mipemphasis 1`** or **`4`**) using original problem
  - Sub-problems emphasize on best-bound (e.g. CPLEX **`mipemphasis 3`**)

Test environment

- Start with MIP models from PerformLib
  - MIPLIB, MittelmannMIP, NEOS
  - Single-thread CPLEX solve with 3hr limit to pick models
- Accessing a large grid was non-trivial - benchmarking runs require priority on a grid or time measurements are inaccurate.
- We substituted a “mini-grid”, a 4-CPU machine
- Each problem was submitted as a batch of 64 sub-problems
- For each problem, measure:
  - Wall time: start-to-finish time, including model generation
  - Max time: maximum solution time of any sub-problem

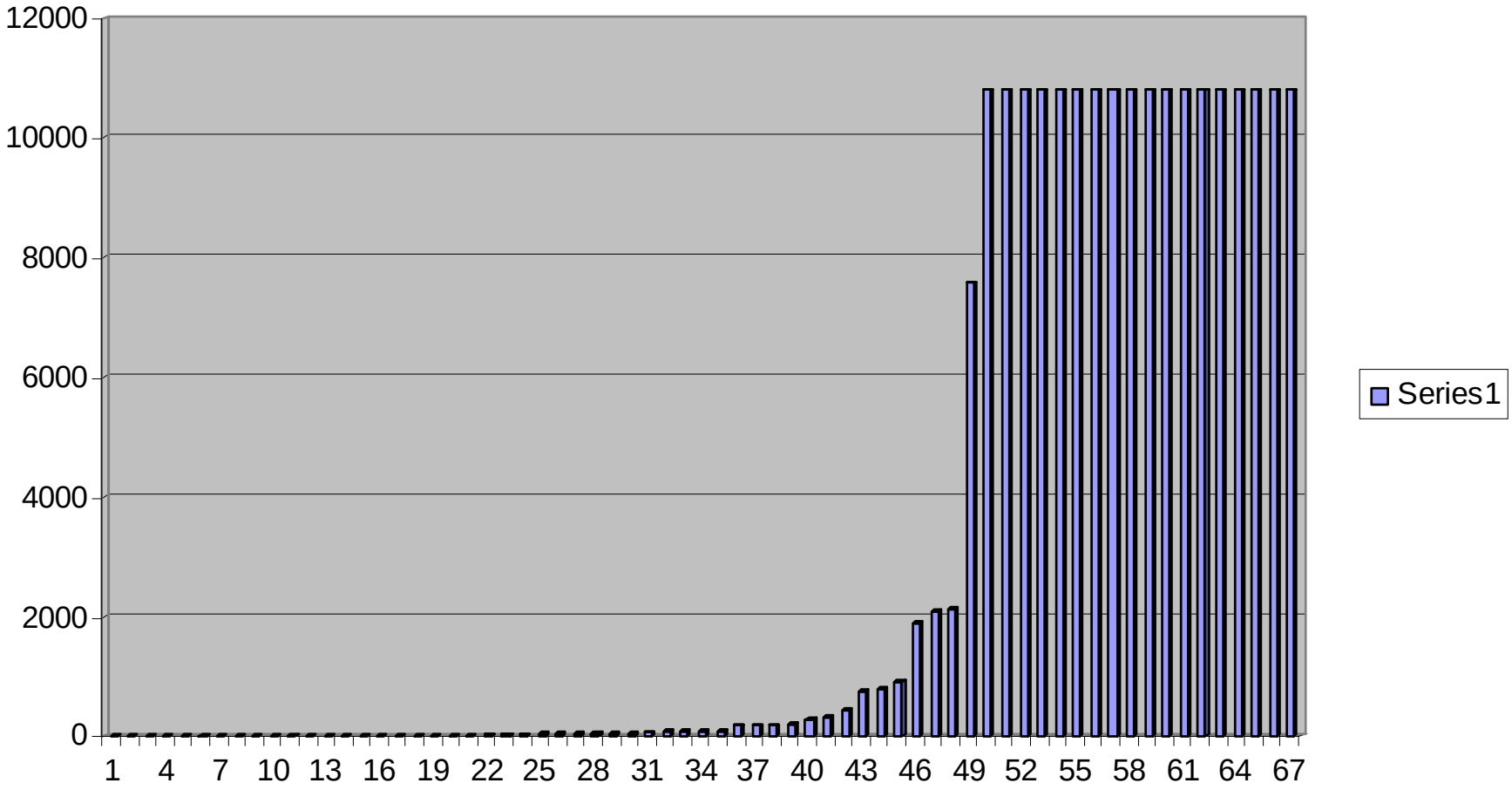
Performance profile, 24 models (proven optimal)



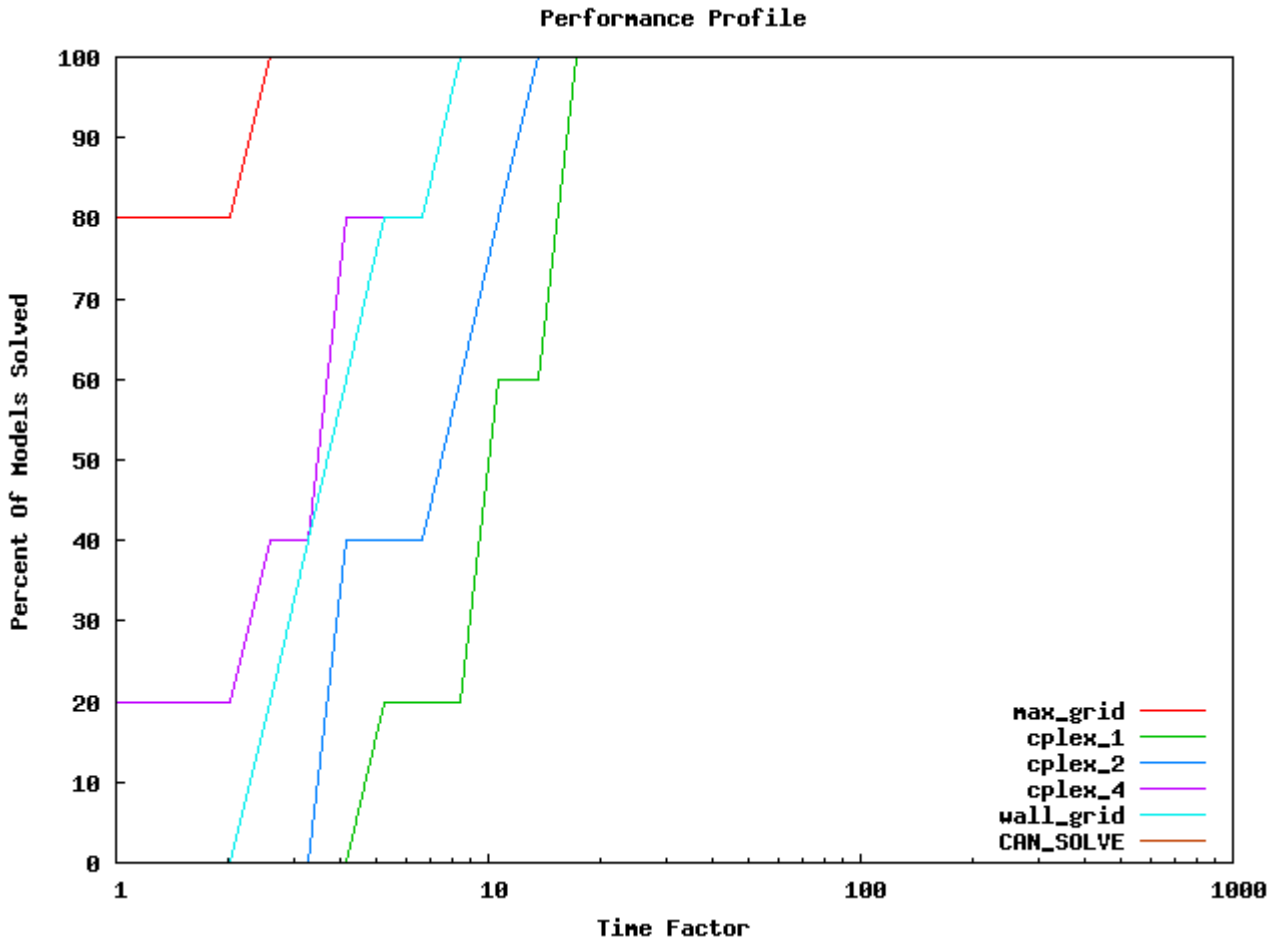
Some results

	Max	Wall	CPLEX-4 4 threads
bienst2	245	721	570
harp2	1000	1046	469
mas74	87	294	290
seymour1	48	349	340
tr12-30	480	2169	1662

Solution times, CPLEX-1, 3hr limit



Performance profile, 5 models (proven optimal)



Summary

- PAVER tools are still available and very useful
- GAMS Grid Facility useful for parallelizing MIP solves
  - Needs good division into subproblems – divide the work equally
  - Not so useful for easy problems
- Most MIP models in the test libraries seem to be easy or very difficult