Hooking your solver to GAMS

The COIN-OR/GAMSlinks project

Michael Bussieck Stefan Vigerske

INFORMS Annual Meeting, Washington D.C.
12.10.2008

Overview

1. Introduction
2. Why to interface a solver with GAMS?
   - Branch-Cut-Heuristic Facility
   - Quality assurance
   - Benchmarking
3. Hooking up your solver to GAMS
   - Write your own GAMS interface
4. Current Developments

Background

Goals of the COIN-OR / GAMSlinks project:

- easy access to open source solvers (mainly from COIN-OR) via GAMS
  - ⇒ broadening the audience of COIN-OR
  - ⇒ broadening the audience of GAMS
- help developers to connect their solvers to GAMS
- provide access to GAMS benchmarking and quality assurance tools

`www.coin-or.org`

`www.gams.com`

`https://projects.coin-or.org/GAMSlinks`

Timeline

2004 Michael Bussieck: links to GLPK and COIN-OR/CBC (code is public)

2004 GAMS 21.4: GAMS ships GLPK and CBC (incl. CLP) within the GAMS distribution (free of charge)

2006 Steven Dirkse: link to COIN-OR/Ipopt (not public)

Timeline

2004 Michael Bussieck: links to GLPK and COIN-OR/CBC (code is public)

2004 GAMS 21.4: GAMS ships GLPK and CBC (incl. CLP) within the GAMS distribution (free of charge)

2006 Steven Dirkse: link to COIN-OR/Ipopt (not public)

Jan. '07 start of COIN-OR/GAMSlinks project, Ipopt link becomes public

Feb. '07 GAMS 22.4: GAMS ships GLPK, CBC, and Ipopt

Timeline

2004 Michael Bussieck: links to GLPK and COIN-OR/CBC (code is public)

2004 GAMS 21.4: GAMS ships GLPK and CBC (incl. CLP) within the GAMS distribution (free of charge)

2006 Steven Dirkse: link to COIN-OR/Ipopt (not public)

Jan. '07 start of COIN-OR/GAMSlinks project, Ipopt link becomes public

Feb. '07 GAMS 22.4: GAMS ships GLPK, CBC, and Ipopt

Apr. '07 link to COIN-OR/Bonmin

Jun. '07 GAMS 22.5: GAMS ships GLPK, CBC, Ipopt, and Bonmin

Timeline

2004 Michael Bussieck: links to GLPK and COIN-OR/CBC (code is public)

2004 GAMS 21.4: GAMS ships GLPK and CBC (incl. CLP) within the GAMS distribution (free of charge)

2006 Steven Dirkse: link to COIN-OR/Ipopt (not public)

Jan. '07 start of COIN-OR/GAMSlinks project, Ipopt link becomes public

Feb. '07 GAMS 22.4: GAMS ships GLPK, CBC, and Ipopt

Apr. '07 link to COIN-OR/Bonmin

Jun. '07 GAMS 22.5: GAMS ships GLPK, CBC, Ipopt, and Bonmin

Aug. '07 link to general OSI solver (rudimentary links to Symphony, DyLP, Volume)

Dec. '07 link to Optimization Services (OSiL, OSrL)

Jan. '08 support of GAMS Branch-Cut-Heuristic (BCH) facility for CBC

Timeline

2004 Michael Bussieck: links to GLPK and COIN-OR/CBC (code is public)

2004 GAMS 21.4: GAMS ships GLPK and CBC (incl. CLP) within the GAMS distribution (free of charge)

2006 Steven Dirkse: link to COIN-OR/Ipopt (not public)

Jan. '07 start of COIN-OR/GAMSlinks project, Ipopt link becomes public

Feb. '07 GAMS 22.4: GAMS ships GLPK, CBC, and Ipopt

Apr. '07 link to COIN-OR/Bonmin

Jun. '07 GAMS 22.5: GAMS ships GLPK, CBC, Ipopt, and Bonmin

Aug. '07 link to general OSI solver (rudimentary links to Symphony, DyLP, Volume)

Dec. '07 link to Optimization Services (OSiL, OSrL)

Jan. '08 support of GAMS Branch-Cut-Heuristic (BCH) facility for CBC

Feb. '08 link to SCIP MIP solver (using CLP as LP subsolver)

May '08 GAMS 22.7: GAMS ships GLPK, CBC, Ipopt, Bonmin, and SCIP

May '08 support of GAMS BCH for SCIP and Bonmin

Timeline

2004 Michael Bussieck: links to GLPK and COIN-OR/CBC (code is public)

2004 GAMS 21.4: GAMS ships GLPK and CBC (incl. CLP) within the GAMS distribution (free of charge)

2006 Steven Dirkse: link to COIN-OR/Ipopt (not public)

Jan. '07 start of COIN-OR/GAMSlinks project, Ipopt link becomes public

Feb. '07 GAMS 22.4: GAMS ships GLPK, CBC, and Ipopt

Apr. '07 link to COIN-OR/Bonmin

Jun. '07 GAMS 22.5: GAMS ships GLPK, CBC, Ipopt, and Bonmin

Aug. '07 link to general OSI solver (rudimentary links to Symphony, DyLP, Volume)

Dec. '07 link to Optimization Services (OSiL, OSrL)

Jan. '08 support of GAMS Branch-Cut-Heuristic (BCH) facility for CBC

Feb. '08 link to SCIP MIP solver (using CLP as LP subsolver)

May '08 GAMS 22.7: GAMS ships GLPK, CBC, Ipopt, Bonmin, and SCIP

May '08 support of GAMS BCH for SCIP and Bonmin

now 6 supported platforms: Linux (Intel 32+64 bit), Solaris (Intel 64bit), MacOS (Intel PowerPC), Windows (32+64 bit)

Overview

1 Introduction

2 Why to interface a solver with GAMS?

Branch-Cut-Heuristic Facility

Quality assurance

Benchmarking

3 Hooking up your solver to GAMS

Write your own GAMS interface

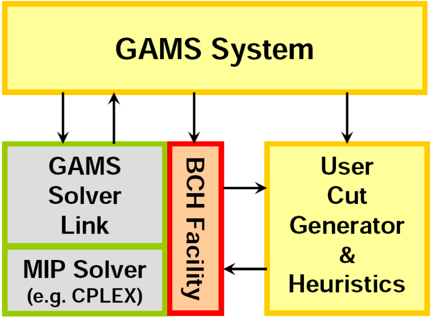
4 Current Developments

Branch-Cut-Heuristic Facility

BCH Facility: pass solver callbacks back into GAMS model space

http://www.gams.com/docs/bch.htm

- represent cut generator and heuristic in terms of original GAMS formulation
- independent of specific MIP solver
- can use any other solver in GAMS for computations



Availability:

CBC	cutting planes and heuristics
SCIP	cutting planes, heuristics, and incumbent report callback
BONMIN B&B	heuristics
BONMIN Hyb+OA	cutting planes and heuristics

Example: Trim Loss Minimization

Reference: I. Harjunkoski, T. Westerlund, R. Porn, H. Skrifvars. Different Transformations for Solving Non-Convex Trim Loss Problems by MINLP. EJOR 105 (1998), 594–603. `http://www.gams.com/modlib/libhtml/bchtlbas.htm`

- The task is to cut out some paper products of different sizes from a large raw paper roll, in order to meet a customer's order.
- nonconvex MINLP (bilinear terms)
- Heuristic:

	best solution	bound
Bonmin alone	–	87.6
Bonmin w/ user heu.	112.1	87.6
SBB alone	–	88.4
SBB w/ user heu.	107.5	88.1

Timelimit: 30 minutes

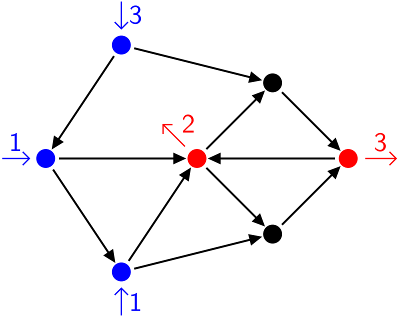
Example: Difficult Network Problem

Single-commodity, uncapacitated, fixed-charge network flow problem:

$$
\begin{aligned}
\min \quad & \sum_{(i,j)\in A} f_{ij} y_{ij} + c_{ij} x_{ij} & \\
\text{s.t.} \quad & \sum_{(j,i)\in\delta^-(i)} x_{ij} - \sum_{(i,j)\in\delta^+(i)} x_{ij} = b_i, & i \in V \\
& 0 \le x_{ij} \le M y_{ij}, \quad y_{ij} \in \{0,1\}, & (i,j) \in A
\end{aligned}
$$

Reference: F. Ortega, L. Wolsey, A branch-and-cut algorithm for the single-commodity, uncapacitated, fixed-charge network flow problem. Networks 41 (2003), No. 3, 143-158

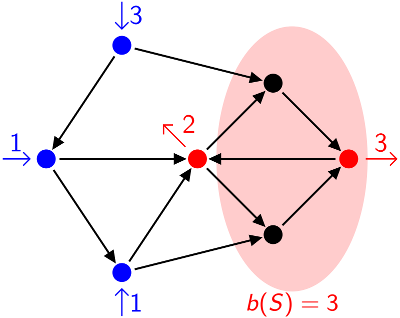
www.gams.com/modlib/libhtml/bchfcnet.htm



Simple Dicut Inequalities

Dicut: For $S \subset V$ with $b(S) > 0$:

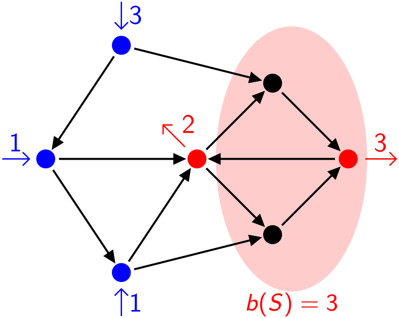
$$\sum_{(i,j)\in\delta^-(S)} y_{ij} \geq 1$$



Simple Dicut Inequalities

Dicut: For $S \subset V$ with $b(S) > 0$:

$$\sum_{(i,j)\in\delta^-(S)} y_{ij} \geq 1$$



Separation problem: find a good set S

$$\begin{aligned} \min \sum_{(i,j)\in A} & \bar{y}_{ij} z_j (1 - z_i) \\ \text{s.t.} \sum_{i\in V} & b_i z_i > 0 \\ & z_i \in \{0,1\}, \quad i \in V \end{aligned}$$

⇒ nonconvex quadratic binary program

⇒ let's use GAMS MIQCP solver

Example: Berlin52 Instance

Instance: Berlin52 (SteinLib): 52 nodes (1 source, 15 sinks), 1326 edges

Note: There are more efficient ways to solve Steiner tree problems...

Solver: SCIP 1.0

	no BCH		
# cuts	0		
# nodes	1470		
Time [s]	1000		
Gap	209.42%		

(Pentium IV 3GHz, 1GB, Linux 2.6.11, gcc 3.3.5)

Example: Berlin52 Instance

Instance: Berlin52 (SteinLib): 52 nodes (1 source, 15 sinks), 1326 edges

Note: There are more efficient ways to solve Steiner tree problems...

Solver: SCIP 1.0

Cuts: at most 20 cuts per round

	no BCH	with BCH using CPLEX	
# cuts	0	316	
# nodes	1470	1	
Time [s]	1000	1000	
Gap	209.42%	81.67%	

(Pentium IV 3GHz, 1GB, Linux 2.6.11, gcc 3.3.5, CPLEX 11.0.0)

Example: Berlin52 Instance

Instance: Berlin52 (SteinLib): 52 nodes (1 source, 15 sinks), 1326 edges

Note: There are more efficient ways to solve Steiner tree problems...

Solver: SCIP 1.0

Cuts: at most 20 cuts per round (stop BARON when 20 solutions have been found)

	no BCH	with BCH using CPLEX	with BCH using BARON
# cuts	0	316	610
# nodes	1470	1	1
Time [s]	1000	1000	268
Gap	209.42%	81.67%	0%

(Pentium IV 3GHz, 1GB, Linux 2.6.11, gcc 3.3.5, CPLEX 11.0.0, BARON 8.1.1)

Example: Berlin52 Instance

Instance: Berlin52 (SteinLib): 52 nodes (1 source, 15 sinks), 1326 edges

Note: There are more efficient ways to solve Steiner tree problems...

Solver: SCIP 1.0

Cuts: at most 20 cuts per round (stop BARON when 20 solutions have been found)

	no BCH	with BCH using CPLEX	with BCH using BARON
# cuts	0	316	610
# nodes	1470	1	1
Time [s]	1000	1000	268
Gap	209.42%	81.67%	0%

(Pentium IV 3GHz, 1GB, Linux 2.6.11, gcc 3.3.5, CPLEX 11.0.0, BARON 8.1.1)

Overhead of BCH ≈ wall clock time – time in SCIP – time in BARON
126s (47%) 268s 124s 18s

Computational Experience

FCNetLib: `http://www.gamsworld.org/performance/fcnetlib`

- 83 instances from Wolsey's web page (`http://www.core.ucl.ac.be/wolsey/ufcn.htm`)
- translated into GAMS models with 83 data files (by Alexey Koptsevich)
- rerun BCH benchmarks originally done by Alexey Koptsevich (2004)

Detailed Results: `http://www.coin-or.org/GAMSlinks/benchmarks`

Computational Experience

FCNetLib: `http://www.gamsworld.org/performance/fcnetlib`

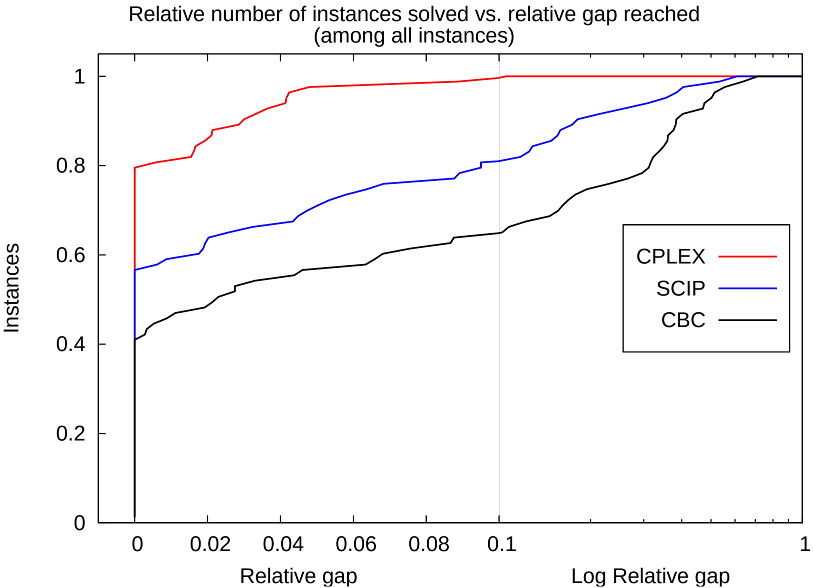
- 83 instances from Wolsey's web page (`http://www.core.ucl.ac.be/wolsey/ufcn.htm`)
- translated into GAMS models with 83 data files (by Alexey Koptsevich)
- rerun BCH benchmarks originally done by Alexey Koptsevich (2004)

Detailed Results: `http://www.coin-or.org/GAMSlinks/benchmarks`

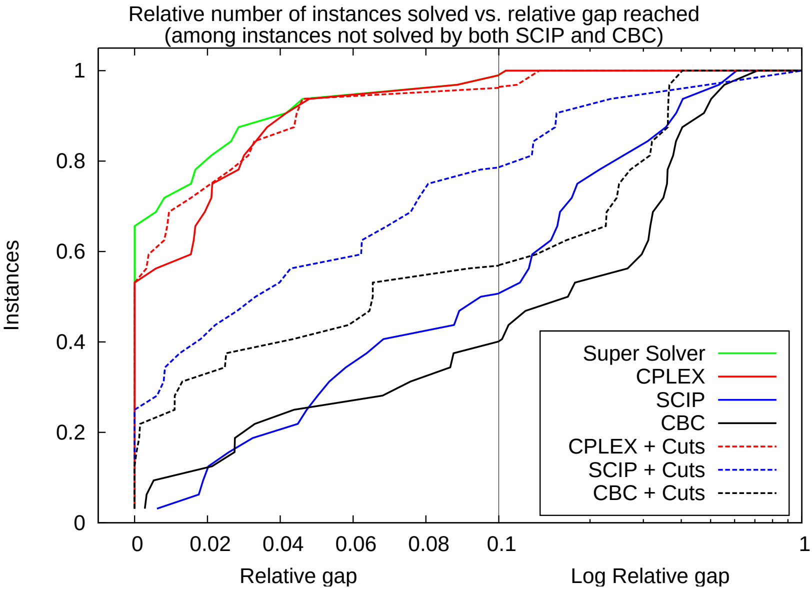
Computational Environment:

- Intel Core2 Duo T7500, 2 GB, Linux 2.4.16, GCC 4.2.1
- GAMS 22.6, CPLEX 11.0.0, CBC 2.1, SCIP 1.0
- Timelimit: 30 minutes

Results: no dicuts, all instances



Results: with dicuts, only difficult instances (32)



Overview

1. Introduction
2. Why to interface a solver with GAMS?
   - Branch-Cut-Heuristic Facility
   - Quality assurance
   - Benchmarking
3. Hooking up your solver to GAMS
   - Write your own GAMS interface
4. Current Developments

Testing a solver with the GAMS model libraries

Testlib Library:

- developed for quality control and testing

Testing a solver with the GAMS model libraries

Testlib Library:

- developed for quality control and testing
- models designed to check solver correctness, special functions, ...
- E.g. `lp03.gms`: examine behavior of LP solver on model with many free variables and when it restarts from an optimal basis

Testing a solver with the GAMS model libraries

Testlib Library:

- developed for quality control and testing
- models designed to check solver correctness, special functions, ...
- E.g. `lp03.gms`: examine behavior of LP solver on model with many free variables and when it restarts from an optimal basis
- `quality.gms`: driver for quality tests of all sorts
- Number of failed tests of GAMS / COIN-OR solvers on testsuites:

	LP (12)	MIP (5)	QCP (4)	NLP (2)
BONMIN	0	0	0	0
CBC/CLP	2	0	–	–
GLPK	0	0	–	–
IPOPT	0	–	0	0
SCIP	0	2	0	0

(date: 09.10.2008)

Testing a solver with the GAMS model libraries

Testlib Library:

- developed for quality control and testing
- models designed to check solver correctness, special functions, ...
- E.g. `lp03.gms`: examine behavior of LP solver on model with many free variables and when it restarts from an optimal basis
- `quality.gms`: driver for quality tests of all sorts

Model Library:

- a collection of small to medium-size models from various applications
- `gmstest.gms`, `slvtest.gms`: runs solvers on models and checks results

$\Rightarrow$ $\approx$ 420 solver runs with a GAMS in demo mode on COIN-OR solvers

Testing a solver with the GAMS model libraries

Testlib Library:

- developed for quality control and testing
- models designed to check solver correctness, special functions, ...
- E.g. `lp03.gms`: examine behavior of LP solver on model with many free variables and when it restarts from an optimal basis
- `quality.gms`: driver for quality tests of all sorts

Model Library:

- a collection of small to medium-size models from various applications
- `gmstest.gms`, `slvtest.gms`: runs solvers on models and checks results

⇒ ≈ 420 solver runs with a GAMS in demo mode on COIN-OR solvers

Both test libraries are part of the tests in the COIN-OR nightly builds.

Overview

1 Introduction

2 Why to interface a solver with GAMS?

- Branch-Cut-Heuristic Facility
- Quality assurance
- Benchmarking

3 Hooking up your solver to GAMS

- Write your own GAMS interface

4 Current Developments

The GAMS World (www.gamsworld.org)

A collection of many worlds:

- CONE World: Conic Optimization
- GLOBAL World: Global Optimization of NLPs
- MINLP World: Mixed Integer Nonlinear Programming
- MPEC World: Mathematical Programs with Equilibrium Constraints
- MPSGE World: Mathematical Programming System for General Equilibrium
- Translation: translate GAMS models into other languages
- Performance World: performance testing of solvers

The GAMS World (www.gamsworld.org)

A collection of many worlds:

- CONE World: Conic Optimization
- GLOBAL World: Global Optimization of NLPs
- MINLP World: Mixed Integer Nonlinear Programming
- MPEC World: Mathematical Programs with Equilibrium Constraints
- MPSGE World: Mathematical Programming System for General Equilibrium
- Translation: translate GAMS models into other languages
- Performance World: performance testing of solvers
    - PerformanceLib: Libraries of test problems, e.g., LINLib, FCNetLib
    - Performance Tools: simplifying performance data collection, measurement, postprocessing, and visualization
    - PAVER: Server for Automated Performance Analysis & Visualization

The GAMS World (www.gamsworld.org)

A collection of many worlds:

- CONE World: Conic Optimization
- GLOBAL World: Global Optimization of NLPs
- MINLP World: Mixed Integer Nonlinear Programming
- MPEC World: Mathematical Programs with Equilibrium Constraints
- MPSGE World: Mathematical Programming System for General Equilibrium
- Translation: translate GAMS models into other languages
- Performance World: performance testing of solvers
  - PerformanceLib: Libraries of test problems, e.g., LINLib, FCNetLib
  - Performance Tools: simplifying performance data collection, measurement, postprocessing, and visualization
  - PAVER: Server for Automated Performance Analysis & Visualization

Detail for benchmarks from next slides at
`http://www.coin-or.org/GAMSlinks/benchmarks`

MIP Benchmarks

Benchmark Setting:

- all MIPs from LINLib (incl. miplib3) ⇒ 125 models
- Competitors:
  - CPLEX 11.1.1
  - CBC 2.2
  - SCIP 1.1 (with CLP 1.8)
  - SCIP 1.0 (with CLP 1.8)
  - GLPK 4.30
- timelimit: 1 hour
- gap tolerance: 0%

MIP Benchmarks

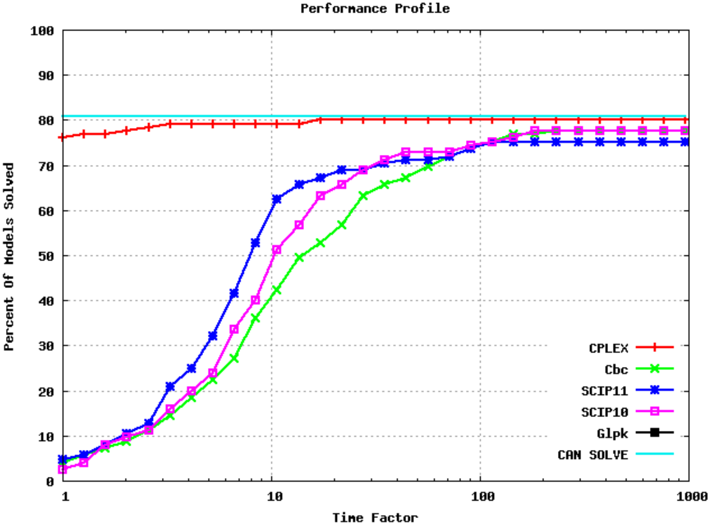
Benchmark Setting:

- all MIPs from LINLib (incl. miplib3) ⇒ 125 models
- Competitors:
  - CPLEX 11.1.1
  - CBC 2.2
  - SCIP 1.1 (with CLP 1.8)
  - SCIP 1.0 (with CLP 1.8)
  - GLPK 4.30
- timelimit: 1 hour
- gap tolerance: 0%

Results:

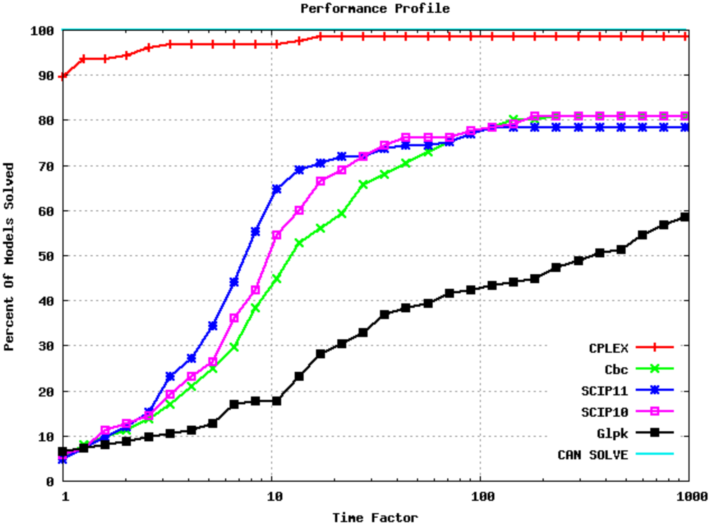
	CPLEX	CBC	SCIP 1.1	SCIP 1.0	GLPK
solved to optimality	80.0%	77.6%	75.2%	77.6%	48.8%
integer feasible solution found	19.2%	22.4%	24.8%	20.8%	48.0%
no solution – timelimit exceeded				1	3
no solution – failed	1			1	1

MIPs from LINLib - Performance profile



- solved = gap closed to 0% (and found best solution among all solvers)
- GLPK not included because no report of dual bound

MIPs from LINLib - Performance profile



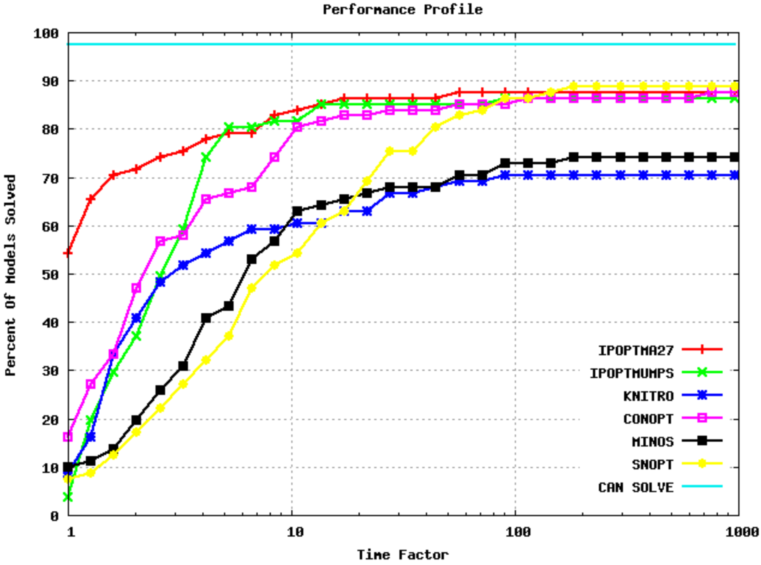
- solved = found best solution among all solvers
- performance profile visualizes only quality of primal bound

NLP Benchmarks

Benchmark Setting:

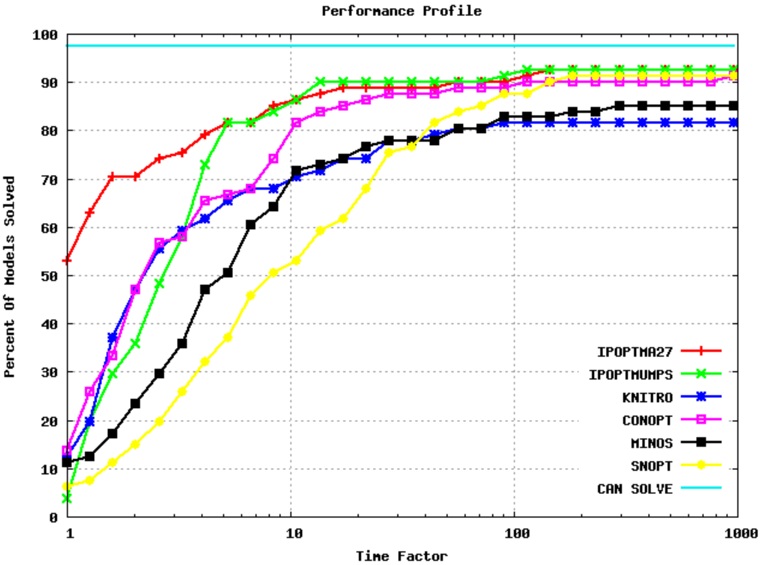
- 81 models with at least 1000 nonlinear nonzeros from the GAMS GlobalLib
- Competitors:
  - IPOPT 3.5 with MA27
  - IPOPT 3.5 with MUMPS 4.8.2
  - KNITRO 5.1.2
  - CONOPT 3.14S
  - MINOS 5.51
  - SNOPT 7.2-4
- timelimit: 1 hour
- relative tolerance: 10^{-8}

81 NLPs from GlobalLib - Performance profile



- solved = found best solution among all solvers

81 NLPs from GlobalLib - Performance profile



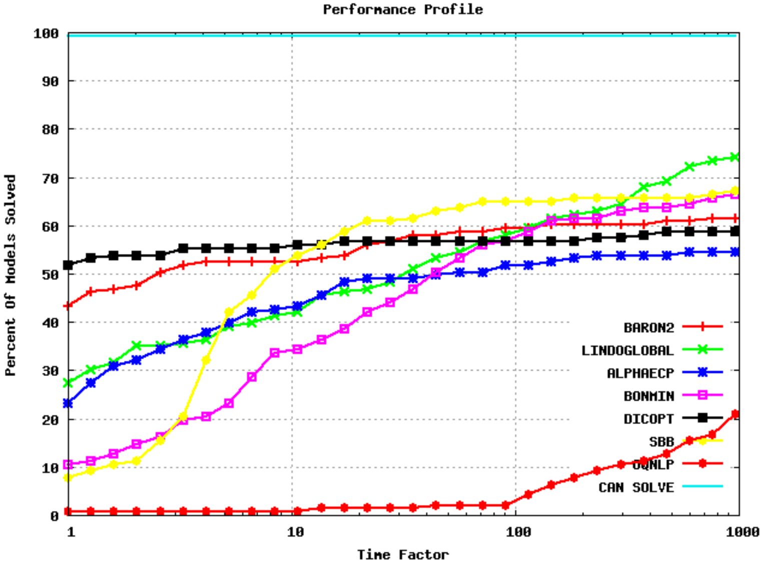
- solved = found a feasible solution

MINLP Benchmarks

Benchmark Setting:

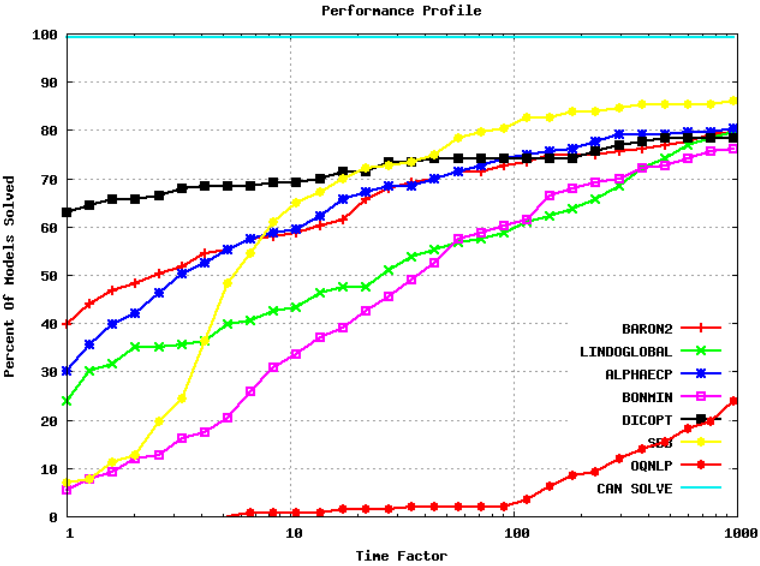
- 143 models from the GAMS MINLPLib
- Competitors:
  - BARON 8.1.5
  - LINDOGLOBAL 5.0.1.292
  - AlphaECP 1.63
  - BONMIN 0.99
  - DICOPT 2x-C
  - SBB
  - OQNLP
- timelimit: 1 hour
- gap tolerance: 1%

143 MINLPs from MINLPLib - Performance profile



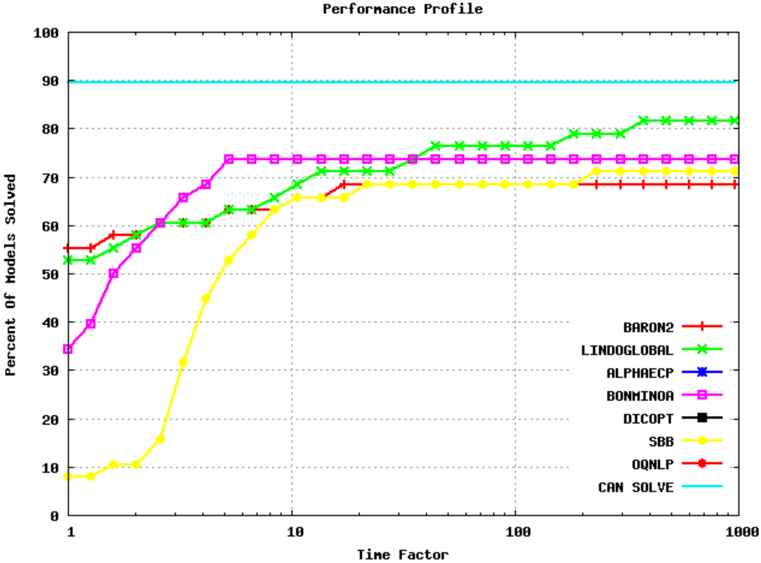
- solved = found best solution among all solvers

143 MINLPs from MINLPLib - Performance profile



- solved = found a feasible solution

38 convex MINLPs from MINLPLib - Performance profile



- solved = gap closed to 0% (and found best solution among all solvers)

Overview

1. Introduction

2. Why to interface a solver with GAMS?
   - Branch-Cut-Heuristic Facility
   - Quality assurance
   - Benchmarking

3. Hooking up your solver to GAMS
   - Write your own GAMS interface

4. Current Developments

GAMS I/O libraries

precompiled GAMS I/O libraries via `GAMSlinks/ThirdParty/GAMSIO/get.GAMSIO`

supported architectures:

- Linux on 32- and 64-bit x86-CPUs with GNU or Intel compilers
- Solaris on 64-bit x86-CPUs with GNU compiler
- MacOS on x86-CPUs with GNU or Intel compiler
- Windows 64-bit x86 with Intel Compiler
- (Windows 32-bit x86 with old MS and Compaq Compilers)

GAMS I/O libraries

precompiled GAMS I/O libraries via `GAMSlinks/ThirdParty/GAMSIO/get.GAMSIO`

supported architectures:

- Linux on 32- and 64-bit x86-CPUs with GNU or Intel compilers
- Solaris on 64-bit x86-CPUs with GNU compiler
- MacOS on x86-CPUs with GNU or Intel compiler
- Windows 64-bit x86 with Intel Compiler
- (Windows 32-bit x86 with old MS and Compaq Compilers)

several choices for accessing GAMS models:

- IOLib: used for CBC, GLPK, OSI
- SMAG: used for Ipopt, Bonmin, SCIP, OS

GAMS I/O libraries

precompiled GAMS I/O libraries via `GAMSlinks/ThirdParty/GAMSIO/get.GAMSIO`

supported architectures:

- Linux on 32- and 64-bit x86-CPUs with GNU or Intel compilers
- Solaris on 64-bit x86-CPUs with GNU compiler
- MacOS on x86-CPUs with GNU or Intel compiler
- Windows 64-bit x86 with Intel Compiler
- (Windows 32-bit x86 with old MS and Compaq Compilers)

several choices for accessing GAMS models:

- IOLib: used for CBC, GLPK, OSI
- SMAG: used for Ipopt, Bonmin, SCIP, OS
- GMO: will replace IOLib and SMAG
  - distributed with newer GAMS systems
  - interfaces to C, C++, C#, Delphi, Fortran90, Java, VisualBasic
  - easier to link against than IOLib or SMAG
  - more than just an I/O library

Classes in the GAMSlinks project

GamsModel: (IOLib)

- hides IOLib functions, allows easy access to a linear mixed-integer model
- provides an OSI - compatible problem representation
- writes GAMS solution file given OSI object or primal/dual values, basis status, status codes, ...

Classes in the GAMSlinks project

GamsModel: (IOLib)

- hides IOLib functions, allows easy access to a linear mixed-integer model
- provides an OSI - compatible problem representation
- writes GAMS solution file given OSI object or primal/dual values, basis status, status codes, ...

GamsDictionary: (IOLib, SMAG)

- access to variable and equation names and texts

Classes in the GAMSlinks project

GamsModel: (IOLib)

- hides IOLib functions, allows easy access to a linear mixed-integer model
- provides an OSI - compatible problem representation
- writes GAMS solution file given OSI object or primal/dual values, basis status, status codes, ...

GamsDictionary: (IOLib, SMAG)

- access to variable and equation names and texts

GamsOptions: (IOLib, SMAG)

- easy access to GAMS option file reader

Classes in the GAMSlinks project

GamsModel: (IOLib)

- hides IOLib functions, allows easy access to a linear mixed-integer model
- provides an OSI - compatible problem representation
- writes GAMS solution file given OSI object or primal/dual values, basis status, status codes, ...

GamsDictionary: (IOLib, SMAG)

- access to variable and equation names and texts

GamsOptions: (IOLib, SMAG)

- easy access to GAMS option file reader

GamsMessageHandler, SmagJournal: (IOLib, SMAG)

- redirects output into GAMS output channels (logfile, statusfile)

Classes in the GAMSlinks project

GamsModel: (IOLib)

- hides IOLib functions, allows easy access to a linear mixed-integer model
- provides an OSI - compatible problem representation
- writes GAMS solution file given OSI object or primal/dual values, basis status, status codes, ...

GamsDictionary: (IOLib, SMAG)

- access to variable and equation names and texts

GamsOptions: (IOLib, SMAG)

- easy access to GAMS option file reader

GamsMessageHandler, SmagJournal: (IOLib, SMAG)

- redirects output into GAMS output channels (logfile, statusfile)

GamsBCH: (IOLib, SMAG)

- access to GAMS Branch-Cut-Heuristic facility (cut and heuristic callbacks)

Classes in the GAMSlinks project

GamsModel: (IOLib)

- hides IOLib functions, allows easy access to a linear mixed-integer model
- provides an OSI - compatible problem representation
- writes GAMS solution file given OSI object or primal/dual values, basis status, status codes, ...

GamsDictionary: (IOLib, SMAG)

- access to variable and equation names and texts

GamsOptions: (IOLib, SMAG)

- easy access to GAMS option file reader

GamsMessageHandler, SmagJournal: (IOLib, SMAG)

- redirects output into GAMS output channels (logfile, statusfile)

GamsBCH: (IOLib, SMAG)

- access to GAMS Branch-Cut-Heuristic facility (cut and heuristic callbacks)

GamsGDX (in development): (IOLib, SMAG)

- reads and writes files in GAMS Data Exchange format

Gams Modeling Object: Instance → Solver

Access a model and write solution file via GMO:

gmoLoadInfoGms	Load Gams control file
gmoLoadDataGms	Load model data defined by control file
gmoM	Number of equations
gmoN	Number of variables
gmoGetVarLower	Get variable lower bounds
gmoGetMatrixRow	Get constraint matrix in row order
gmoEvalFunc	Evaluate the constraint i
gmoEvalGrad	Update nonlinear gradients of constraint
gmoEvalHessLag	Evaluate Hessian of Lagrangian
gmoSetSolution	Set solution values
gmoUnloadSolutionGms	Unload solution to the Gams solution file

...

Gams Modeling Object: Solver → Instance

Create or modify a model via GMO:

`gmoInitData`	`Initializes GMO data`
`gmoAddRow`	`Add a row`
`gmoAddCol`	`Add a column`
`gmoSetAltRHS`	`Set alternative RHS`
`gmoSetAltVarBounds`	`Set alternative variable bounds`

```
...
```

Gams Modeling Object: Solver → Instance

Create or modify a model via GMO:

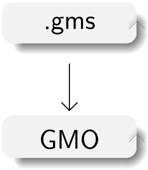
gmoInitData	Initializes GMO data
gmoAddRow	Add a row
gmoAddCol	Add a column
gmoSetAltRHS	Set alternative RHS
gmoSetAltVarBounds	Set alternative variable bounds

...

Solve a GMO instance with a GAMS solver:

conReadyAPI	Define the conopt API and load the option object
conCallSolver	Call conopt
cpxReadyAPI	Define the cplex API and load the option object
cpxModifyProblem	Update Cplex object with rhs, obj, bounds, jac.
cpxCallSolver	Call cplex

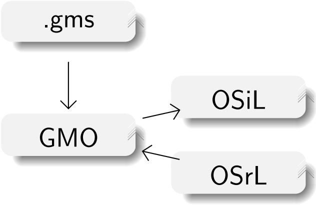
GAMS / GMO / Optimization Services (OS)



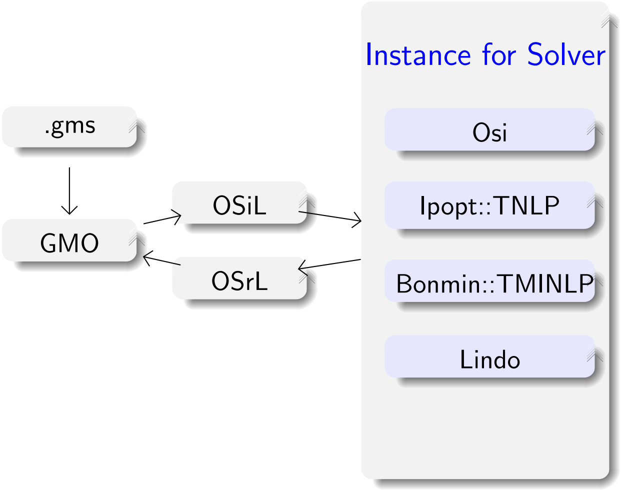
.gms

GMO

GAMS / GMO / Optimization Services (OS)



GAMS / GMO / Optimization Services (OS)



Instance for Solver

.gms

GMO

OSiL

OSrL

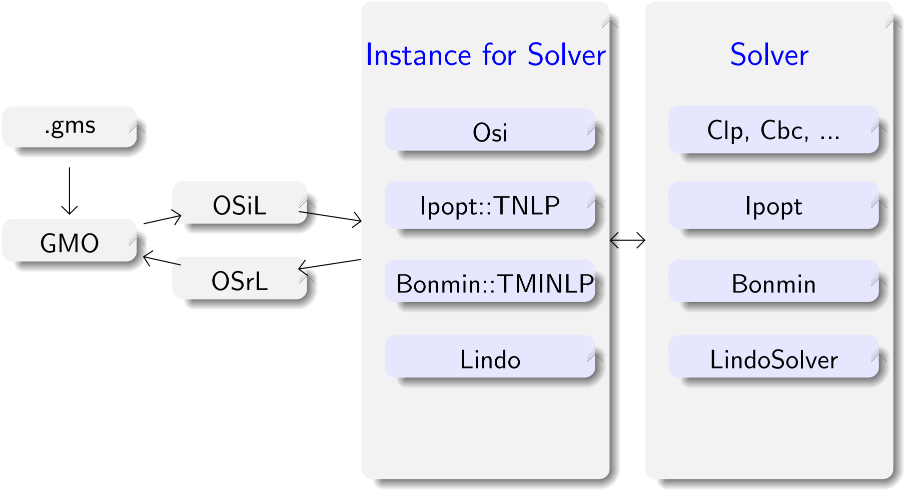
Osi

Ipopt::TNLP

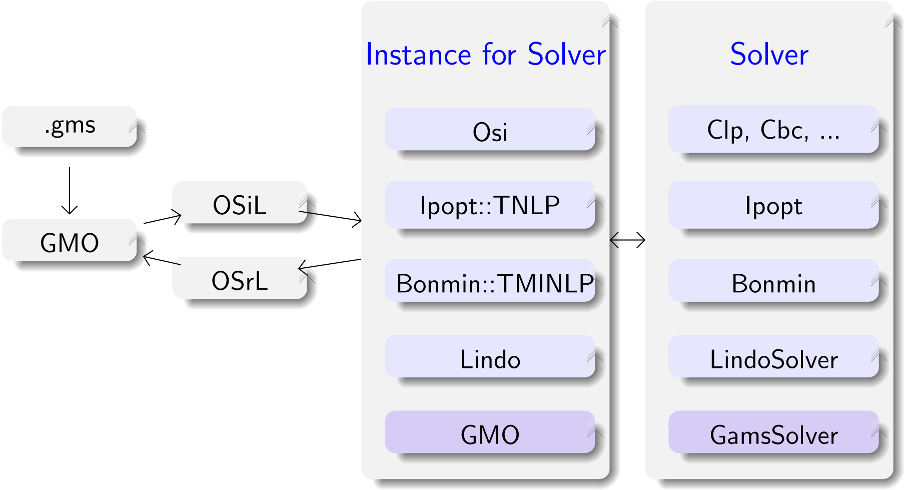
Bonmin::TMINLP

Lindo

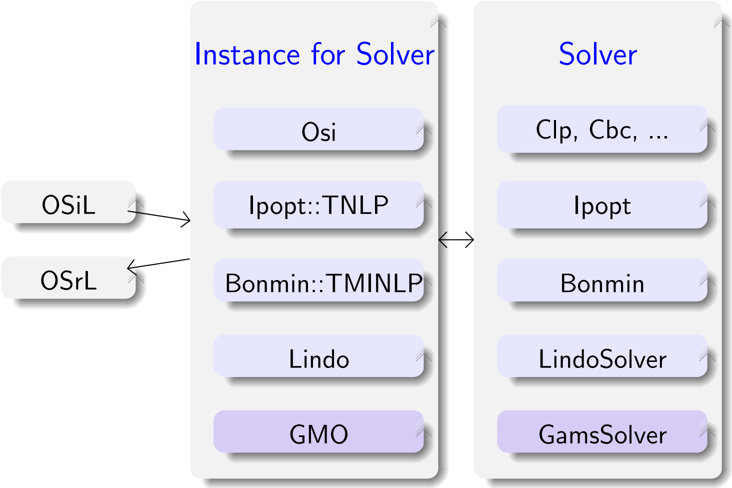
GAMS / GMO / Optimization Services (OS)



GAMS / GMO / Optimization Services (OS)



GAMS / GMO / Optimization Services (OS)



That's it.

Thanks!

PS: GAMS evaluation licences (2 months, all solvers) at

- `http://www.gams.com/evals/dc/l/gamslice.txt` (Linux)
- `http://www.gams.com/evals/dc/m/gamslice.txt` (MacOS X)
- `http://www.gams.com/evals/dc/w/gamslice.txt` (Windows)

Performance Profiles

E.D. Dolan and J.J. More, Mathematical Programming, 91, 2002:

- compare performance of solver $s \in \mathcal{S}$ on problem $p \in \mathcal{P}$ with best performance by any solver on problem p:

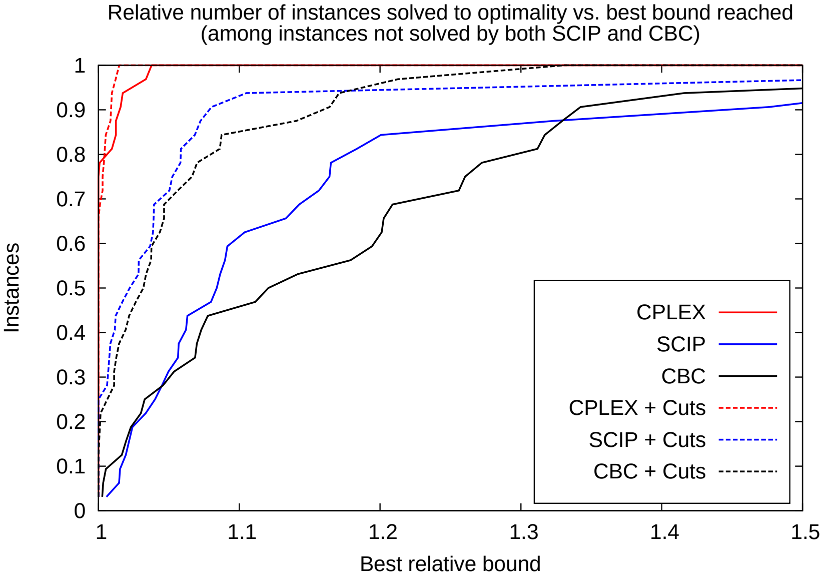
$$\rho(p, s) := \frac{t_{p,s}}{\min_{s' \in \mathcal{S}} t_{p,s'}}$$

- $t_{p,s}$ = time solver s spend on p, $\quad t_{p,s} = \infty$ if s did not *solve* p
- $P_s(\tau)$ = probability that performance ratio $\rho(p, s)$ within factor of τ of best possible ratio:

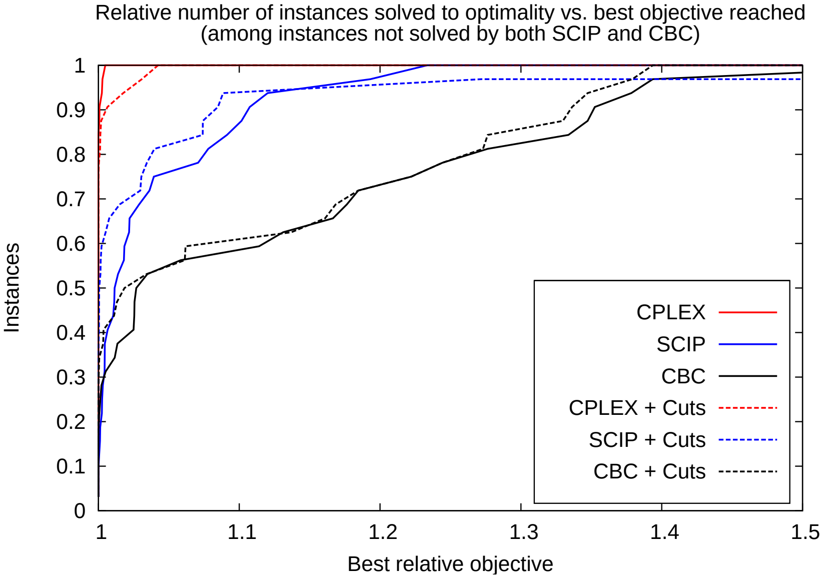
$$P_s(\tau) := \frac{|\{p \in \mathcal{P} : \rho(p, s) \leq \tau\}|}{|\mathcal{P}|}$$

- percentage of models that solver s will *solve* if for each model, s can have a maximum resource time of τ times the minimum time
- s solved p: found feasible point *or* found best solution among all solvers

Results: with dicuts, only difficult instances (32)



Results: with dicuts, only difficult instances (32)


Relative number of instances solved to optimality vs. best objective reached
(among instances not solved by both SCIP and CBC)
Instances
Best relative objective
CPLEX
SCIP
CBC
CPLEX + Cuts
SCIP + Cuts
CBC + Cuts