Verified Linear Programming and Extensions

Christian Keil

Department of Applied Mathematics
Waseda University, Tokyo

EASIAM 2008

Outline

Introduction

Comparison of verified LP software

Extensions

- Mixed inter linear programming
- Conic programming

Summary

Why verified linear programming?

- Rounding errors in floating point arithmetic cause suboptimal or infeasible results
- Ordóñez and Freund, 2003:
  71% of netlib LP problems are ill-posed

Ben-Tal and Nemirovski, 2000

In real-world applications of Linear Programming one cannot ignore the possibility that a small uncertainty in the data (intrinsic for most real-world LP programs) can make the usual optimal solution of the problem completely meaningless from a practical viewpoint.

A linear program (LP)

Definition (Linear Program (LP))

min	$c^T x$	objective function
subject to	$Ax \leq a$	linear...
	$Bx = b$	...constraints
	$\underline{x} \leq x \leq \overline{x}$	simple bounds

- Simple bounds may be infinite

Outline

Introduction

Comparison of verified LP software

Extensions
- Mixed inter linear programming
- Conic programming

Summary

Verified LP software

Constraint programming

RealPaver (Granvilliers)

Verified LP software

Constraint programming

RealPaver (Granvilliers)

Rational arithmetic

exlp (Kiyomi), perPlex (Koch),
QSopt_ex (Applegate et al.)

Verified LP software

Constraint programming

RealPaver (Granvilliers)

Rational arithmetic

exlp (Kiyomi), perPlex (Koch),
QSopt_ex (Applegate et al.)

Global optimization

GlobSol (Kearfott), ICOS (Lebbah)

Verified LP software

Constraint programming
: RealPaver (Granvilliers)

Rational arithmetic
: exlp (Kiyomi), perPlex (Koch), QSopt_ex (Applegate et al.)

Global optimization
: GlobSol (Kearfott), ICOS (Lebbah)

Verified linear programming
: Lurupa (Keil)

Are we comparing apples and oranges?

What is solving?

- Return a rigorous result for the LP
- Type of result (exact, enclosure of optimal, near optimal point) often secondary from application point of view
- Lower bound (RealPaver) important for rigorous branch-and-bound schemes

Packages for different tasks with different outputs – fair?

- All can solve LP $\Rightarrow$ look at performance
- Test whether exploiting structure is necessary
- LP is an easy test,
  general problems of same size are much harder

A set of test problems

- 103 real-world problems from netlib and Meszaros's collection
  - Various applications
  - Difficult or interesting at time of submission
  - Less than 1500 variables
- Timeout depends on problem:
  100 times fastest solving time

Introducing performance profiles

Suppose two solvers on 5 problems

$t_{A,*}$	$t_{B,*}$
3s	3s
1s	10s
2s	4s
2s	20s
∞	100s

$t_{s,p}$ is the runtime for solver s on problem p

Introducing performance profiles

Suppose two solvers on 5 problems

$t_{A,*}$	$t_{B,*}$
3s	3s
1s	10s
2s	4s
2s	20s
∞	100s

$\longrightarrow$

$r_{A,*}$	$r_{B,*}$
1	1
1	10
1	2
1	10
∞	1

Definition (Runtime ratio)

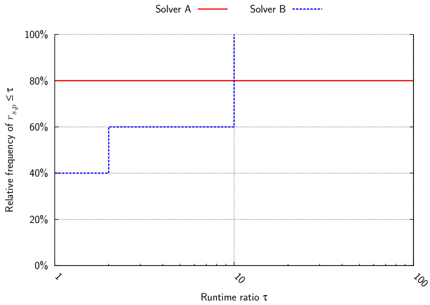
Runtime ratio r for a solver s on a problem p is

$$r_{s,p} := \frac{t_{s,p}}{\min_s\{t_{s,p}\}}$$

Introducing performance profiles

Suppose two solvers on 5 problems

$t_{A,*}$	$t_{B,*}$		$r_{A,*}$	$r_{B,*}$
3s	3s		1	1
1s	10s		1	10
2s	4s	$\longrightarrow$	1	2
2s	20s		1	10
∞	100s		∞	1



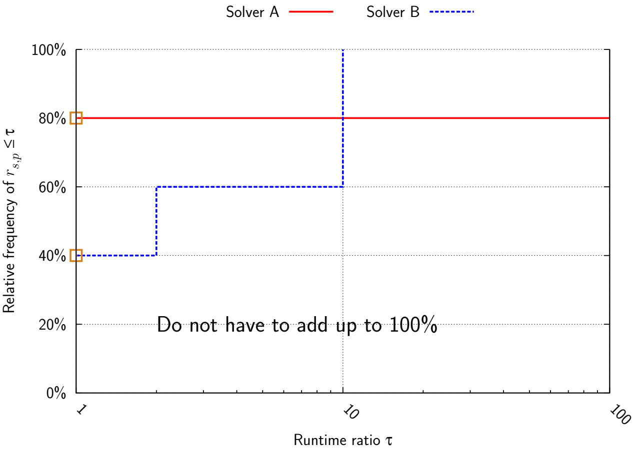
Definition (Performance profile)

Cumulative distribution function of runtime ratios

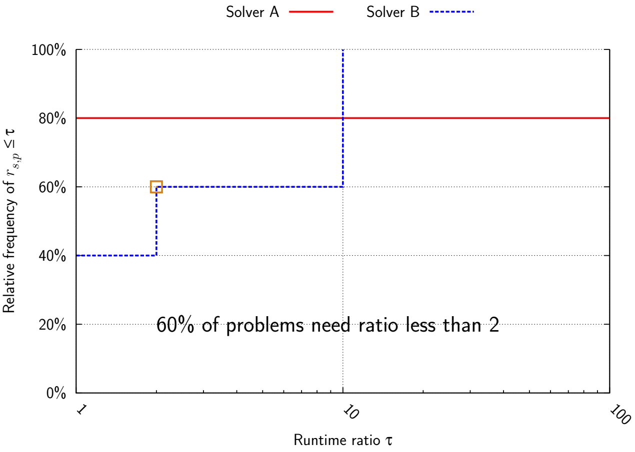
$$\rho_s(\tau) := \frac{|\{p \mid r_{s,p} \leq \tau\}|}{|\{p\}|}$$

How to read performance profiles

$r_{A,*}$	$r_{B,*}$
1	1
1	10
1	2
1	10
∞	1

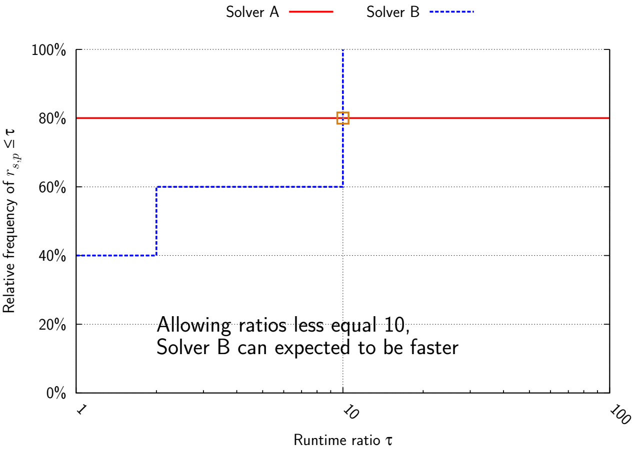


$r_{A,*}$	$r_{B,*}$
1	1
1	10
1	2
1	10
∞	1



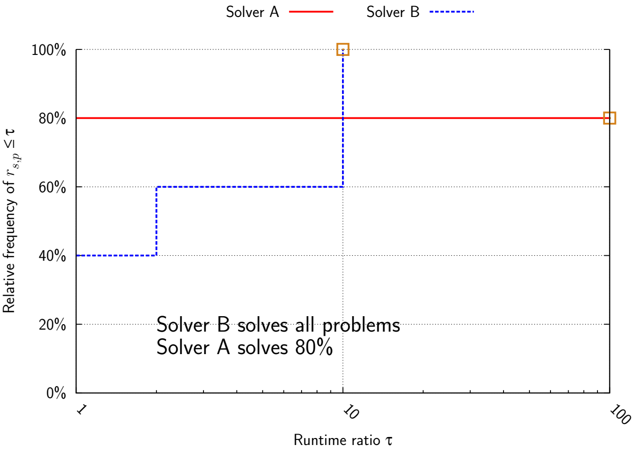
How to read performance profiles

$r_{A,*}$	$r_{B,*}$
1	1
1	10
1	2
1	10
∞	1



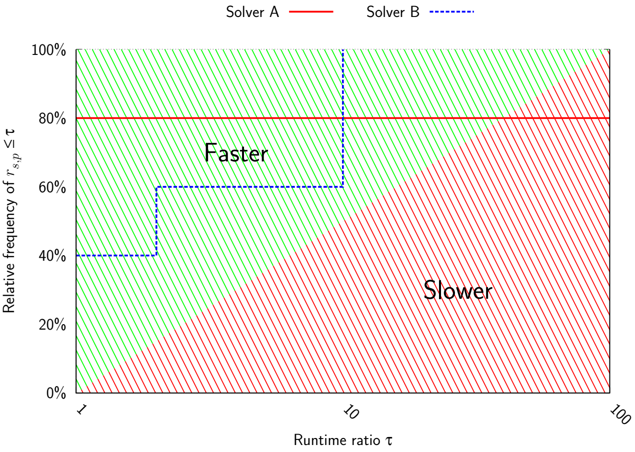
How to read performance profiles

$r_{A,*}$	$r_{B,*}$
1	1
1	10
1	2
1	10
∞	1

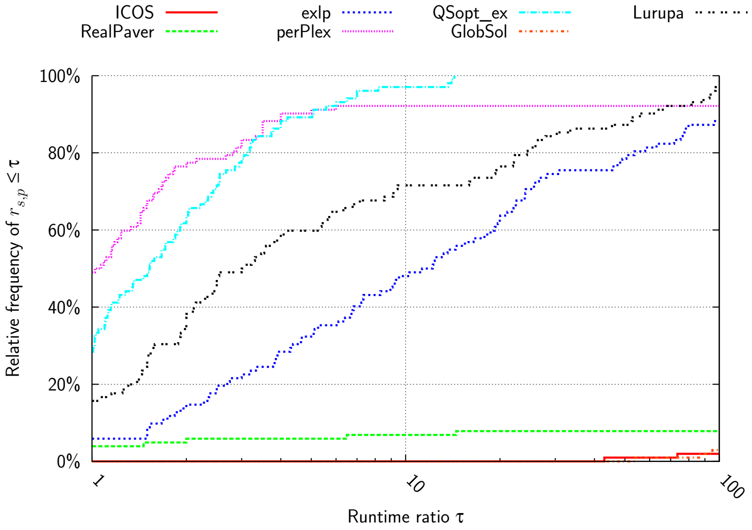


How to read performance profiles

$r_{A,*}$	$r_{B,*}$
1	1
1	10
1	2
1	10
∞	1



Performance profiles for real-world problems



Outline

Introduction

Comparison of verified LP software

Extensions
- Mixed inter linear programming
- Conic programming

Summary

A mixed integer linear program (MILP)

Definition (Mixed Integer Linear Program (MILP))

min	$c^T x$	objective function
subject to	$Ax \leq a$	linear...
	$Bx = b$	...constraints
	$\underline{x} \leq x \leq \overline{x}$	simple bounds
	$x_{\mathcal{Z}} \in \mathbb{Z}$	integrality constraints

- Simple bounds may be infinite

An innocuous test problem?

Example (Neumaier and Shcherbina, 2004)

$$
\begin{aligned}
\min\ & -x_{20} \\
\text{s. t.}\ & (s+1)x_1 - x_2 \geq s-1 \\
& -sx_{i-1} + (s+1)x_i - x_{i+1} \geq (-1)^i(s+1) && i: 2,\ldots,19 \\
& -sx_{18} - (3s-1)x_{19} + 3x_{20} \geq -(5s-7) \\
& 0 \leq x_i \leq 10 && i: 1,\ldots,13 \\
& 0 \leq x_i \leq 10^6 && i: 14,\ldots,20 \\
& \text{all } x \in \mathbb{Z}.
\end{aligned}
$$

- Integer variables and coefficients $\Rightarrow$ expect exact solution

Results of commercial state-of-the-art solver

Set $s = 6 \Rightarrow$ several state-of-the-art solver fail

- bonsaiG and Xpress find no solution
- CPLEX, GLPK, Xpress-MP/Integer, and MINLP even claim: integer infeasible
- $x = (1, 2, 1, 2, \ldots, 1, 2)^T$ feasible

Explanation of commercial-solver performance

Solving with branch-and-bound method

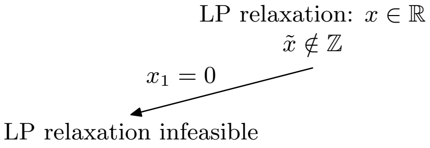
- Iteratively fixing non-integer variables to integer values

$$\text{LP relaxation: } x \in \mathbb{R}$$
$$\tilde{x} \notin \mathbb{Z}$$

Explanation of commercial-solver performance

Solving with branch-and-bound method

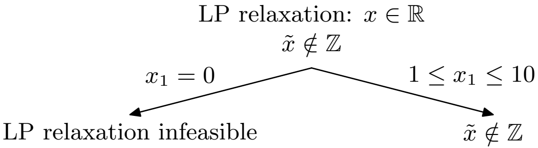
- Iteratively fixing non-integer variables to integer values



Explanation of commercial-solver performance

Solving with branch-and-bound method

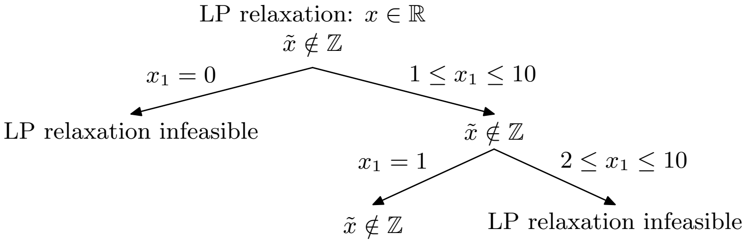
- Iteratively fixing non-integer variables to integer values



Explanation of commercial-solver performance

Solving with branch-and-bound method

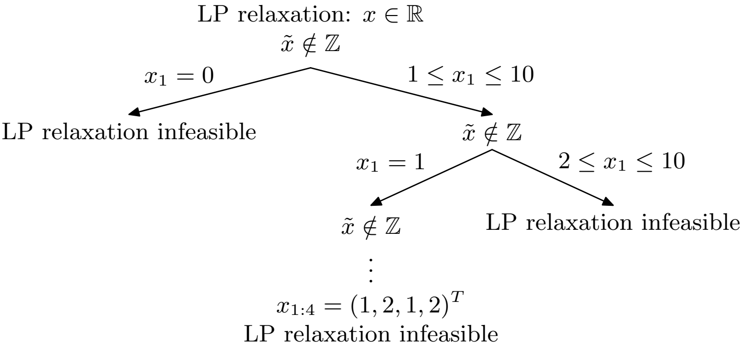
- Iteratively fixing non-integer variables to integer values



Explanation of commercial-solver performance

Solving with branch-and-bound method

- Iteratively fixing non-integer variables to integer values



MILP is judged infeasible

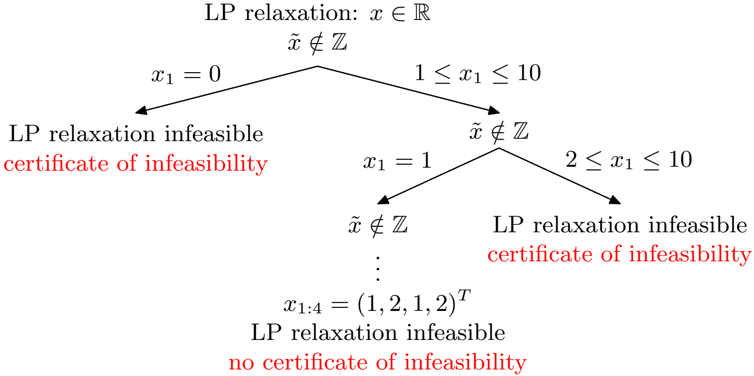
A verified MILP solver

Student project by Doubli (2008)

- MILP solver based on miqp.m (Bemporad and Mignone)
- Lurupa to make branch-and-bound completely rigorous: bounds on optimal value, certificates of infeasibility

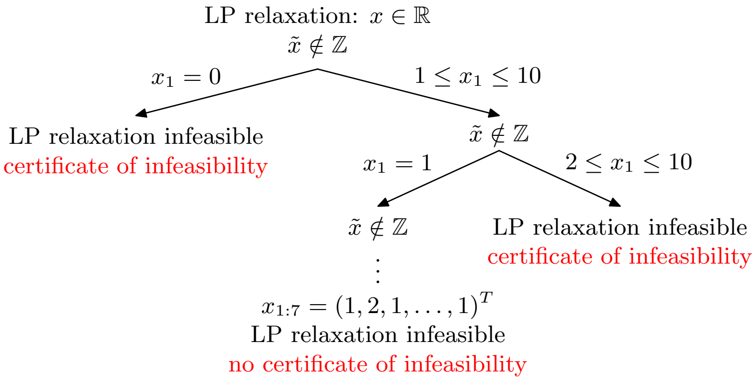
Result of verified MILP solver

Solving with completely rigorous branch-and-bound method



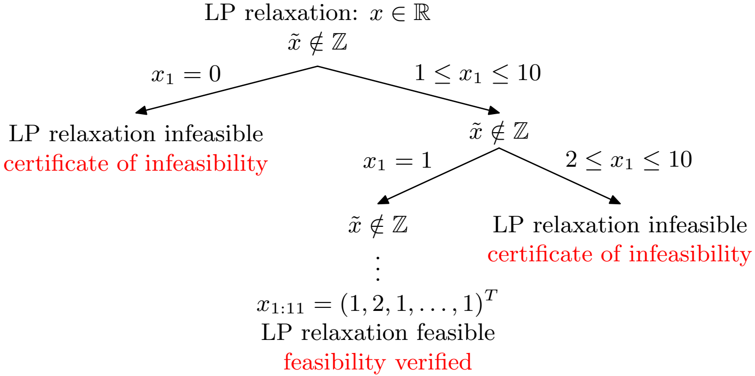
Result of verified MILP solver

Solving with completely rigorous branch-and-bound method



Result of verified MILP solver

Solving with completely rigorous branch-and-bound method



$x = (1, 2, 1, \ldots, 2)^T$ feasible and optimal

A conic program

Definition (Conic Program)

$$\begin{array}{rll} \min & c^T x & \text{objective function} \\ \text{subject to} & Ax \leq a & \text{linear}\ldots \\ & Bx = b & \ldots\text{constraints} \\ & x \in \mathcal{C} & \text{conic constraint} \end{array}$$

- Efficiently solvable under mild assumptions (Ben-Tal and Nemirovski)

Why conic programming

Rockafellar (1993)

In fact the great watershed in optimization isn't between linearity and nonlinearity, but convexity and nonconvexity.

- Conic program universal convex program (Nesterov and Nemirovski)

Verified conic programming

- Algorithms by Jansson:
  - Second-order cone programming (SOCP)
  - Semidefinite programming (SDP)
  - Conic programming in vector lattices
- Compute rigorous enclosures of primal and dual feasible points

Outline

Summary

Cheap rigorous error bounds available for

- Ill-conditioned and even ill-posed convex problems
- Convex relaxations in global optimization

- Neumaier and Shcherbina 2004, MILP
- Several publications by
  Institute for Reliable Computing
  Hamburg University of Technology
  `http://ti3.tu-harburg.de`

The end

Thank you for your

a- 

http://www.tu-harburg.de/~keil