THE SAS MILP SOLVER: CURRENT STATUS AND FUTURE DEVELOPMENTS

INFORMS Annual Meeting 2016, Nashville

Philipp M. Christophel
Menal Güzelsoy
Amar Narisetty
Imre Pólik

AGENDA

1. Introduction
2. Improvements in SAS/OR 14.2
   - Precedence detection
   - Modulo 2 solver
3. Computational results
4. Future Developments

ABOUT THE SAS MILP SOLVER

- SAS/OR: LP, QP, NLP, MILP, modeling language, network algorithms, constraint programming, derivative-free optimization, discrete event simulation
- Accessible through the PROC OPTMODEL modeling language and directly by calling PROC OPTMILP
- Algorithms
  - LP relaxation-based branch and cut
  - Decomposition
- Developed since 2004, available since 2006
- Latest version: SAS/OR 14.2

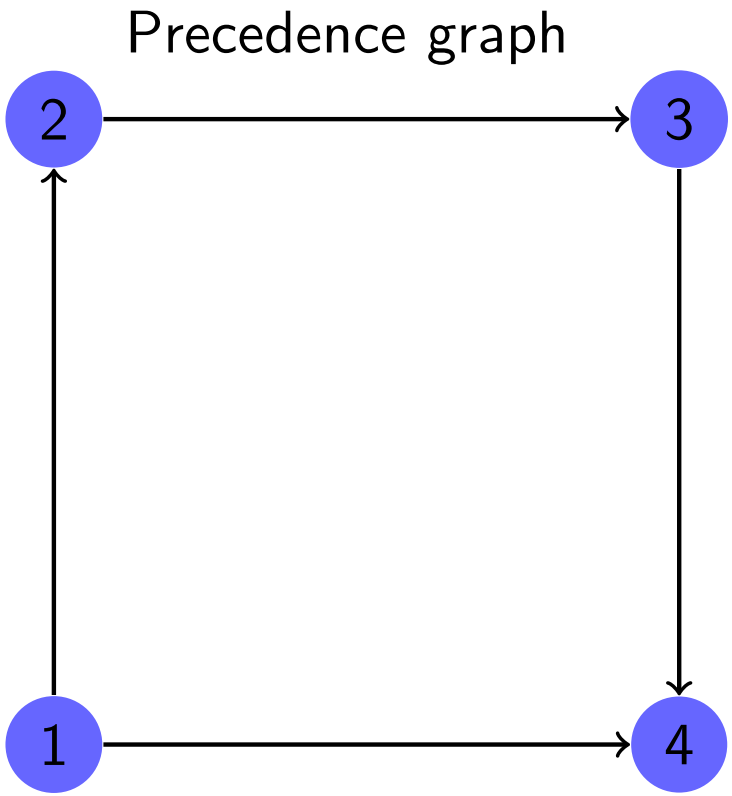
OVERVIEW

- Presolver
  - New techniques: two-row tightening, row aggregation, swap substitutions, free columns substitutions, mod 2 reductions, precedence constraint, number theory based tightenings and substitutions
- Cuts
  - New flow cover, flow path, and MIR cuts
- Simplex
  - Crash basis improvements
  - Perturbation improvements
- Other
  - More dual solution tightening
  - (Orbital) Probing in node presolver
  - New pre-LP heuristics

PRECEDENCE DETECTION

- Given constraints of the form $x \leq y$ or $x \leq 1 - y$ identify the redundant ones
- Precedence graph ($\mathcal{G}$): each variable and its complement gets a node
- Arcs indicate the precedence relationship
- Transitive closure ($\bar{\mathcal{G}}$): adding all the implied arcs to the graph
- Redundancy detection: $uv \in \mathcal{G}$ is redundant iff $\exists w \in \mathcal{G}, w \neq u, v : uw \in \bar{\mathcal{G}}, wv \in \bar{\mathcal{G}}$
- Need to be careful about back edges and self links

PRECEDENCE DETECTION



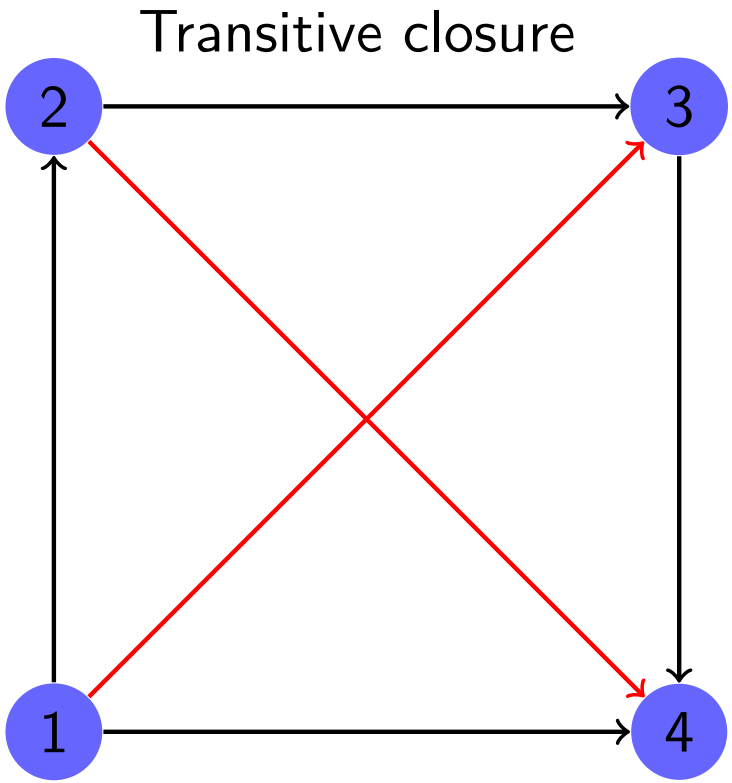
$$x_1 \leq x_2$$

$$x_2 \leq x_3$$

$$x_3 \leq x_4$$

$$x_1 \leq x_4$$

PRECEDENCE DETECTION



$$x_1 \leq x_2$$

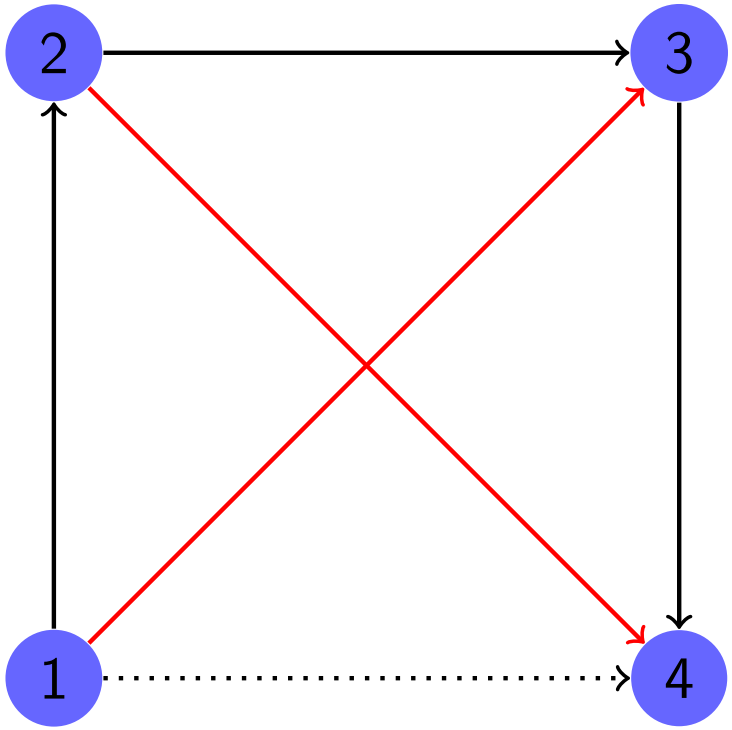
$$x_2 \leq x_3$$

$$x_3 \leq x_4$$

$$x_1 \leq x_4$$

PRECEDENCE DETECTION

(14) is redundant



$$x_1 \leq x_2$$

$$x_2 \leq x_3$$

$$x_3 \leq x_4$$

$$\cancel{x_1 \leq x_4}$$

PRECEDENCE DETECTION

- Already had transitive closure implemented as part of SAS/OR
- Very fast
- The closure can be used to speed up probing
- Very effective on the `opm2` class of instances

Instance	Rows	Rows left
`opm2_z7_s2`	26675	15699
`opm2_z10_s2`	146299	65358
`opm2_z11_s8`	205459	88265
`opm2_z12_s7`	297768	124278
`opm2_z12_s14`	297299	124194

MODULO 2 SOLVER

- Take all integer equalities modulo 2
- Solve the system with Gaussian elimination
  - dense bitfield storage with 64bit words
  - all coefficients are 0-1
  - pivoting operation is bitwise XOR: very fast
- Detect infeasibility
- Obtain the parity of a variable: tighten or fix
- Identify linear relations between 2 binaries

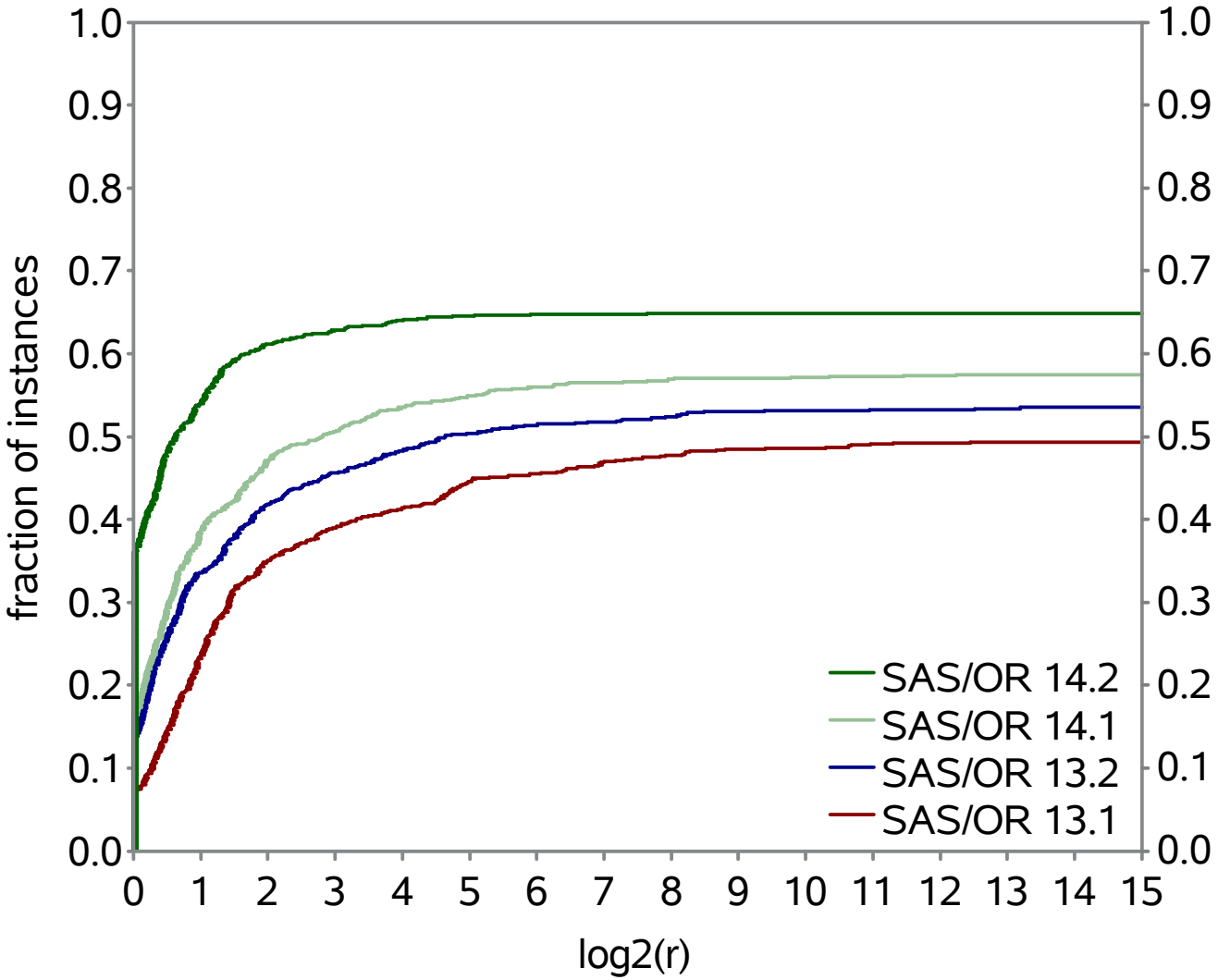
$$x + y \equiv 0 \mod 2 \longrightarrow x = y$$

OVERALL PERFORMANCE

- Test set of 798 instances from customers and public benchmarks
- 96 machines running 2 jobs each on 16-core/2-socket Intel(R) Xeon(R) E5-2630 v3 @ 2.40GHz CPUs
- 62GB of memory per job
- 2-hour time limit, default options, 4 threads
- Results for 14.2
  - 57% faster than 14.1 on instances both versions solve
  - 61 instances more solved within the time limit

COMPUTATIONAL RESULTS

OVERALL PERFORMANCE



COMPUTATIONAL RESULTS

PUBLIC BENCHMARKS

Mittelmann Test	Solved	Time (sec.)[1]
Bench, 1 thr, 2h, 0 gap	81 of 87	240
Bench, 1 thr, 2h, 0 gap (17 seeds avg.)	82 of 87	235
Bench, 4 thr, 2h, 0 gap	85 of 87	121
Bench, 4 thr, 2h, 0 gap (17 seeds avg.)	85 of 87	113
Bench, 12 thr, 2h, 0 gap	82 of 87	95
Bench, 12 thr, 2h, 0 gap (17 seeds avg.)	84 of 87	84
Solvable, 12 thr, 2h, 0 gap	170 of 208	261
Infeasible, 4 thr, 1h	18 of 19	47
Feasible, 4 thr, 1h, 1st feasible	27 of 33	114

[1] Intel(R) Xeon(R) E5-2630 v3 @ 2.40GHz CPU

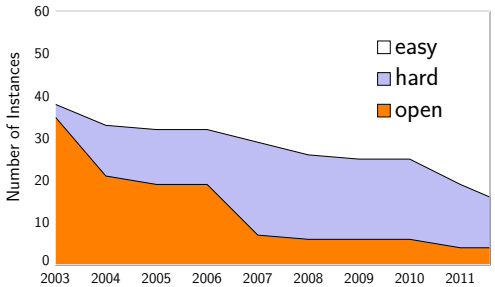
FUTURE DEVELOPMENTS

OUTLOOK

- SAS/OR solvers on SAS Viya
  - SAS Viya: a new, cloud-based architecture
  - Data in Hadoop
  - Call from SAS, Python, Java, Lua
  - Distributed parallel MILP solver
- Solver Improvements
  - New lift-and-project cuts
  - Cut management improvements
  - Probing improvements
  - Even more presolver techniques
  - Simplex: Row basis and advanced crash basis

FUTURE DEVELOPMENTS

CONTRIBUTIONS TO MIPLIB 2017



Since 1992, MIPLIB has been a key driver for computational progress in mixed integer programming.

"Shape the Future of Mixed Integer Programming!"

Go to miplibsubmissions.zib.de to submit your instances to the

6th Mixed Integer Programming Library

Deadline: Februar 28, 2017

An initiative by *Arizona State University, COIN-OR, CPLEX, FICO, Gurobi, MIPCL, MOSEK, NuOPT, SAS, and Zuse Institute Berlin.*

http://support.sas.com/or

The SAS MILP Solver: Current Status And Future Developments