On Performance Variability in Pseudo-Boolean Solving and the Impact of Trivial Model Simplifications

Ambros Gleixner[1,2], Alexander Hoen[1], Jakob Nordström[3,4]

[1]Zuse Institute Berlin [2]HTW Berlin [3]Lund University [4]University of Copenhagen

hoen@zib.de

Pragmatics of SAT

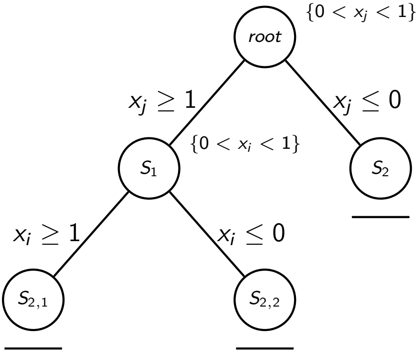
4th July 2023

ZIB ZUSE INSTITUTE BERLIN

htw Hochschule für Technik und Wirtschaft Berlin University of Applied Sciences

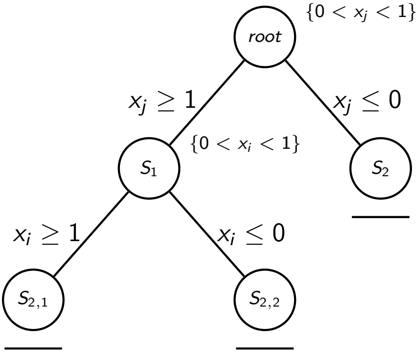
LUNDS UNIVERSITET

Presolving on MIP solvers



[1] Achterberg et al. 2019.

Presolving on MIP solvers

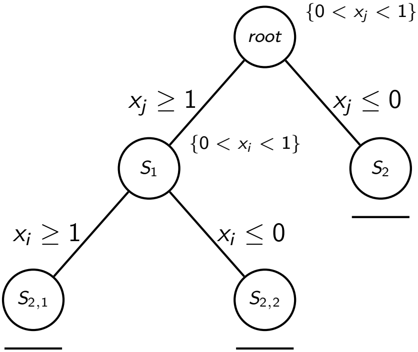


Goals:

- reducing the number of potential nodes in the b&b tree
- reducing the node solving time
- provide numerical robustness

[1] Achterberg et al. 2019.

Presolving on MIP solvers



Goals:

- reducing the number of potential nodes in the b&b tree
- reducing the node solving time
- provide numerical robustness

Impact on MIP: speed-up factor of 9 for instance with solvingtime $\geq$ 10 seconds[1]

What is the impact on a PB solver?

[1] Achterberg et al. 2019.

PaPILO[2]

- Parallel Presolve in Integer in Linear Optimization
- solver independent implementation: provides pre- and postsolving routines
- *exploit parallel hardware*
- *supports multiprecision arithmetic*
- proof logging with VeriPB soon available
- available at `https://github.com/scipopt/papilo`

[2]Gleixner, Gottwald, and Hoen 2023.

Presolvers in PaPILO

PaPILO has 14 presolvers, 11 apply for PB problems.

- Coefficient Strengthening
- (Domain) Propagation
- Dominated Variables
- Dual Fix
- *Dual Infer (continuous only)*
- *Implied integer variables (continuous only)*
- *Parallel Variables (integer only version exists)*
- Parallel Constraints
- (Simple) Probing
- Singleton Variable/Stuffing
- Sparsify
- (Simple) Substitution
- Trivial (Model Clean Up)

Experimental setup

- Timelimit: 1800 seconds
- Testset: PB16 + 5 permutations of PB16
- Solvers: RoundingSAT and PaPILO + RoundingSAT
- $\geq t$ sec: one of the two solvers solved the instances in $\geq t$ seconds

Experimental results on Presolving

PaPILO barely impacts the performance of RoundingSAT.

			RoundingSAT		PaPILO + RoundingSAT			
		instances	solved	time	solved	time	vars%	cons%
dec	all	5361	3713	25.86	3700	24.59	39	32
	$\geq$ 10 sec	858	792	157.34	779	168.28	16	13
	$\geq$ 100 sec	595	529	388.49	516	395.03	17	15
	$\geq$ 1000 sec	232	166	888.20	153	973.09	20	17
opt	all	2628	1648	37.29	1642	44.30	38	31
	$\geq$ 10 sec	338	294	163.46	288	198.00	30	25
	$\geq$ 100 sec	244	200	471.92	194	475.19	27	23
	$\geq$ 1000 sec	144	100	728.14	94	848.73	25	19

Table 1: Impact of PaPILO on the performance of RoundingSAT with 1800 seconds on PB16

Experimental results on Presolving - solved by both

RoundingSAT gains a speed-up by PaPILO on instances solved by both solvers.

			RoundingSAT		PaPILO + RoundingSAT			
		instances	solved	time	solved	time	vars%	cons%
dec	$\geq$ 10 sec	713	713	111.63	713	116.83	14	11
	$\geq$ 100 sec	450	450	302.46	450	292.34	15	12
	$\geq$ 1000 sec	87	87	967.01	87	923.22	13	11
opt	$\geq$ 10 sec	244	244	92.84	244	108.77	33	28
	$\geq$ 100 sec	150	150	367.01	150	311.59	29	26
	$\geq$ 1000 sec	50	50	774.39	50	712.99	28	22

Table 2: Impact of PaPILO on the performance of RoundingSAT with 1800 seconds on PB16

Experimental results on Presolving- unpermuted

			RoundingSAT		PaPILO + RoundingSAT			
		instances	solved	time	solved	time	vars%	cons%
dec	all	5361	3713	25.86	3700	24.59	39	32
	$\geq$ 10 sec	858	792	157.34	779	168.28	16	13
	$\geq$ 100 sec	595	529	388.49	516	395.03	17	15
	$\geq$ 1000 sec	232	166	888.20	153	973.09	20	17
unpermuted	$\geq$ 1000 sec	42	26	1162.29	32	861.31	17	19
opt	all	2628	1648	37.29	1642	44.30	38	31
	$\geq$ 10 sec	338	294	163.46	288	198.00	30	25
	$\geq$ 100 sec	244	200	471.92	194	475.19	27	23
	$\geq$ 1000 sec	144	100	728.14	94	848.73	25	19
unpermuted	$\geq$ 1000 sec	18	12	1069.28	13	785.53	21	29

Table 3: Impact of PaPILO on the performance of RoundingSAT with 1800 seconds on PB16

Measuring Performance Variability

Definition:

Seemingly inconsequential changes can cause significant differences in performance.

Measuring Performance Variability

Definition:

Seemingly inconsequential changes can cause significant differences in performance.

For one model, let $X_1, \ldots, X_n$ be the solving times for the n permutations, where X_1 is reserved for the unpermuted "default" version of the model:

- $VS = \sqrt{\frac{1}{n\mu^2} \cdot \left(\sum_{p=1}^{n}(X_p - \mu)^2\right)}$ (Lodi and Tramontani 2013)
- $\text{spread} = \max(X_1, \ldots, X_n) - \min(X_1, \ldots, X_n)$
- $\text{freq} = \left|\{(p, q) : 1 \leq p < q \leq n \wedge \text{status}_p \neq \text{status}_q\}\right| / \binom{n}{2}$

Experimental results on Variability on decision

		models	solved	time	VS	spread	freq
RS	all	1398	1089.8	15.24	0.13	2.04	0.02
	≥ 10 sec	301	268.8	101.43	0.43	107.64	0.09
	≥ 100 sec	181	148.8	315.27	0.46	369.71	0.15
	≥ 1000 sec	82	49.8	846.66	0.48	1073.54	0.33
	≥ 1800 sec	60	27.8	1003.95	0.44	1125.54	0.45
SCIP	all	1397	736.3	76.63	0.08	1.19	0.01
	≥ 10 sec	206	174.3	180.72	0.38	137.31	0.10
	≥ 100 sec	146	115.3	463.77	0.37	331.21	0.14
	≥ 1000 sec	76	45.3	1045.86	0.33	610.75	0.27
	≥ 1800 sec	50	19.3	1055.56	0.38	658.41	0.41

Table 4: Variability statistics on decision models for SCIP and ROUNDINGSAT.

Visualization of the variability results on decision

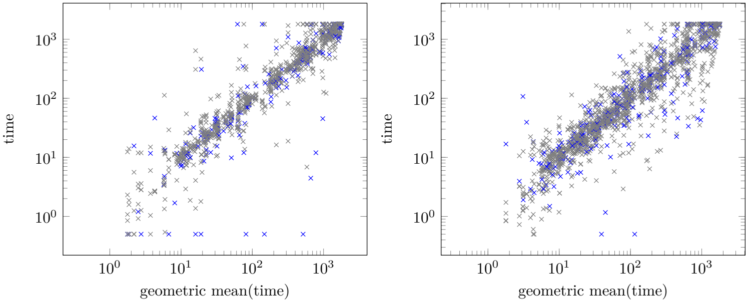


Figure 1: Distribution of running times for SCIP (left) and ROUNDINGSAT (right) on PB16 dec with filter "≥ 10 sec". Each $\times$ marks a permutation of a model. The y-axis gives the time for each of these instances and the x-axis gives the shifted geometric mean of runtimes of all permutations of a model. Unpermuted models (Permutation 1) are marked in blue

Experimental results on Variability on optimization

		models	solved	time	VS	spread	freq
RS	all	532	295.2	60.35	0.13	2.01	0.04
	$\geq$ 10 sec	86	59.2	220.82	0.62	561.12	0.28
	$\geq$ 100 sec	73	46.2	366.29	0.65	1043.05	0.33
	$\geq$ 1000 sec	59	32.2	432.03	0.68	1430.93	0.40
	$\geq$ 1800 sec	52	25.2	485.71	0.63	1472.21	0.46
SCIP	all	532	325.2	116.27	0.11	4.74	0.02
	$\geq$ 10 sec	187	173.2	190.91	0.27	125.38	0.07
	$\geq$ 100 sec	137	123.2	364.13	0.30	260.32	0.09
	$\geq$ 1000 sec	41	27.2	1047.99	0.31	621.46	0.30
	$\geq$ 1800 sec	26	12.2	1075.84	0.35	601.38	0.47

Table 5: Variability statistics on optimization models for SCIP and RoundingSAT.

Visualization of the variability results on optimization

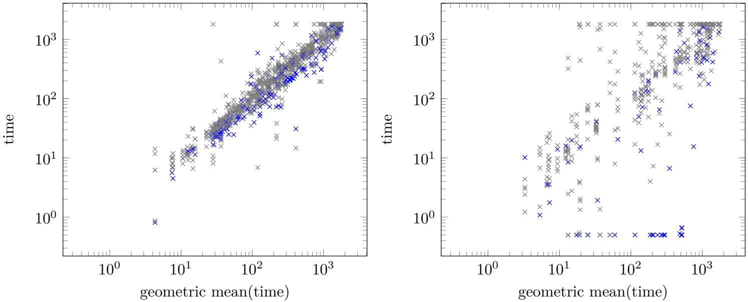


Figure 2: Distribution of running times for SCIP (left) and ROUNDINGSAT (right) on PB16 opt with filter "$\geq 10\,\text{sec}$". Each $\times$ marks a permutation of a model. The y-axis gives the time for each of these instances and the x-axis gives the shifted geometric mean of runtimes of all permutations of a model. Unpermuted models (Permutation 1) are marked in blue

Conclusion

- Presolving is one of the most important heuristics in mixed-integer programming.
- The majority of these techniques are also suited for pseudo-boolean problems.
- Time-limit is too small to show if Presolving helps on harder instances.
- Overall PaPILO barely affects the performance of RoundingSAT.
- Presolving helps RoundingSAT on specific subsets instances (all-optimal).
- Performance Variability may impacts the effect of presolving.

Thank you for your attention!

Questions?