Incremental Encoding of Pseudo-Boolean Goal Functions based on Comparator Networks

Michał Karpiński, Marek Piotrów

Institute of Computer Science
University of Wrocław
Poland

6 July 2020

Outline

1. Definitions and motivation
2. New encoding and analysis
3. Experimental evaluation
4. Conclusions

Definitions and motivation

Pseudo-Boolean Optimization Problem:

$$
\begin{aligned}
\text{minimize} \quad & a_1 x_1 + a_2 x_2 + \cdots + a_n x_n \\
\text{s.t.} \quad & b_{1,1} x_1 + b_{1,2} x_2 + \cdots + b_{1,n} x_n \leq k_1, \\
& b_{2,1} x_1 + b_{2,2} x_2 + \cdots + b_{2,n} x_n \leq k_2, \\
& \vdots \\
& b_{m,1} x_1 + b_{m,2} x_2 + \cdots + b_{m,n} x_n \leq k_m, \\
\text{where} \quad & x_i - \textit{literals},\ a_i, b_{i,j}, k_i \in \mathbb{Z}, i = 1, \ldots, n, j = 1, \ldots, m
\end{aligned}
$$

Definitions and motivation

Applications:

- cumulative scheduling
- logic synthesis
- logic verification

Definitions and motivation

Incremental algorithm:

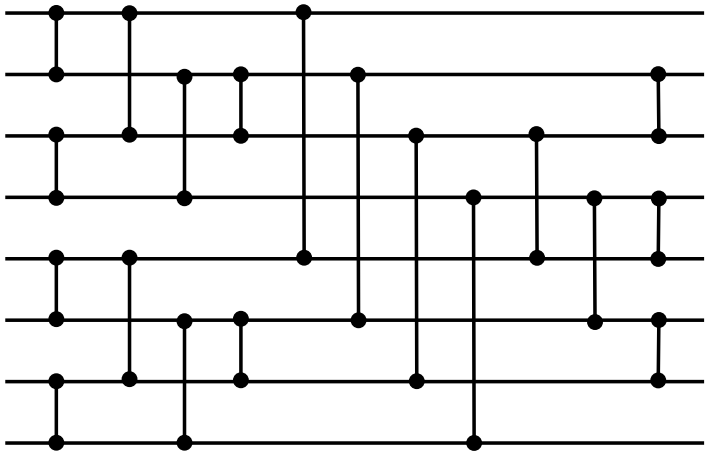
1. Guess the initial bound on the goal function,
2. Translate an input instance into a Boolean formula,
3. Use a SAT-solver to check if the formula is satisfiable,
4. Strengthen (or relax) the constraint depending on SAT-solver output,
5. Translate the goal function into a Boolean formula for the new bound,
6. Repeat from Step 3 until the optimum is found.

Definitions and motivation

Our contribution:

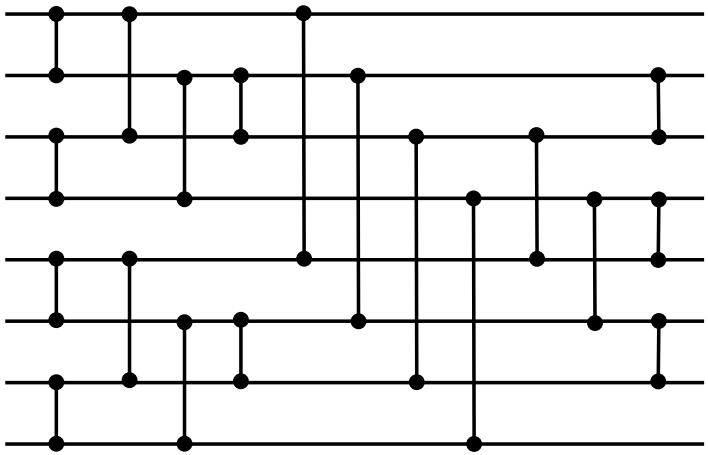
Eliminate Step 5: Translate the goal function into a Boolean formula for the new bound.

Definitions and motivation

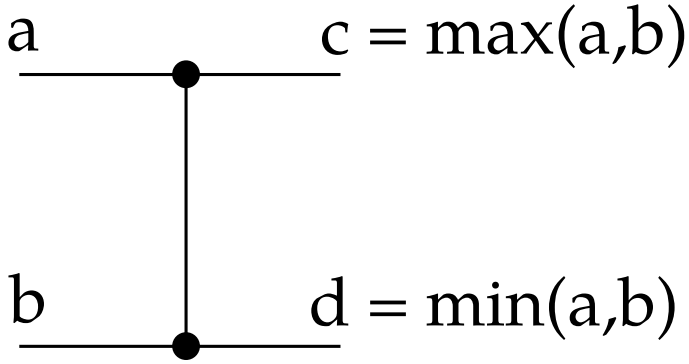


Bather's odd-even sorting network (1968)

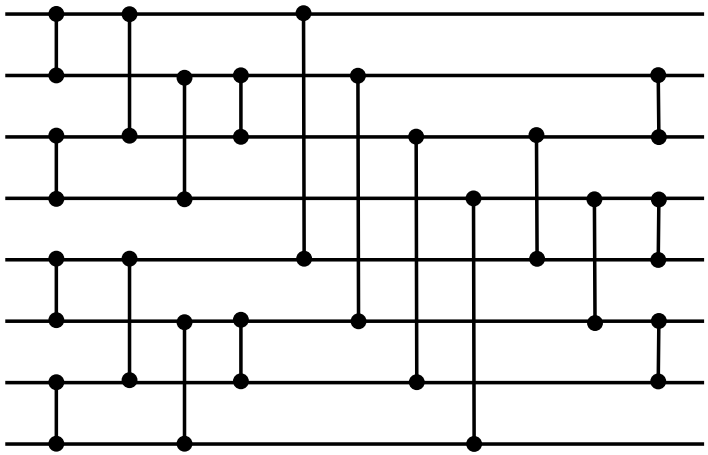
Definitions and motivation



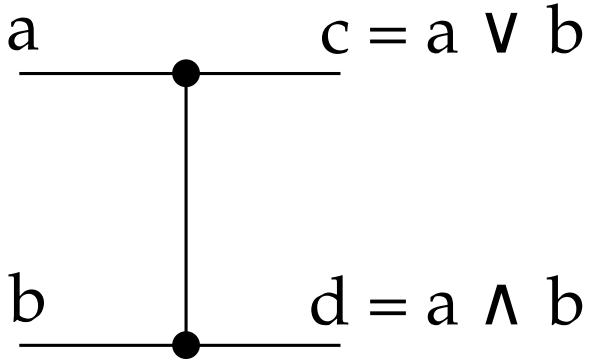
Bather's odd-even sorting network (1968)



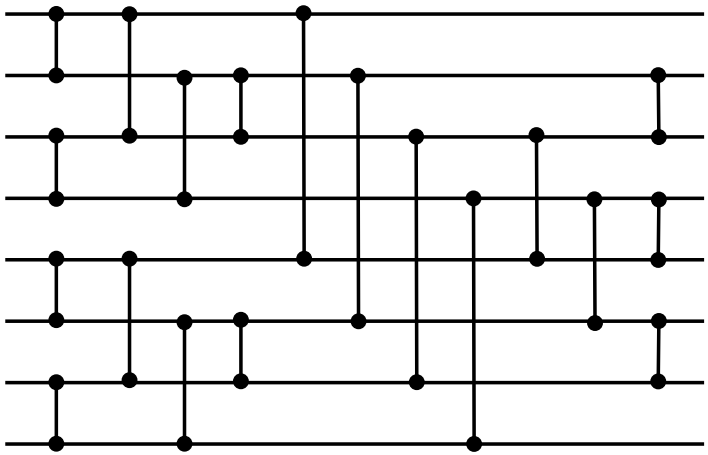
Definitions and motivation



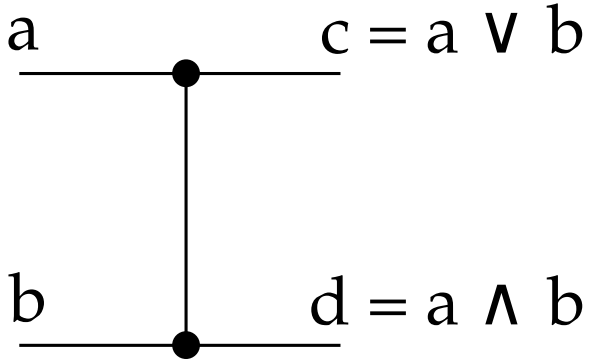
Bather's odd-even sorting network (1968)



Definitions and motivation



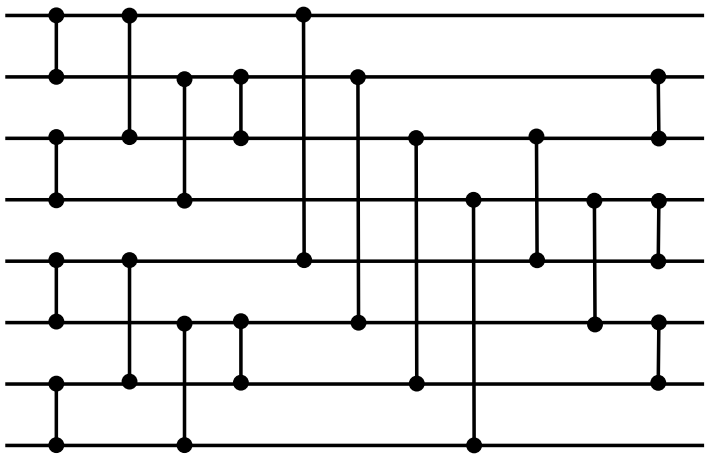
Bather's odd-even sorting network (1968)



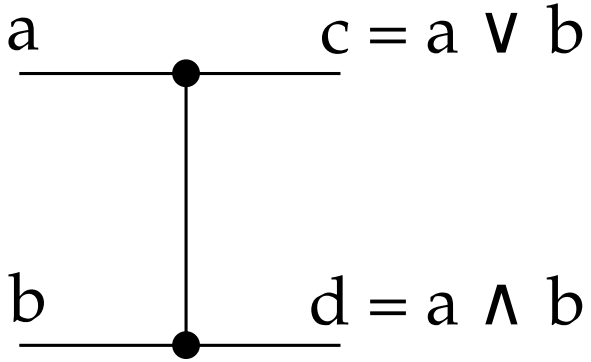
$a \Rightarrow c$, $b \Rightarrow c$,
$a \wedge b \Rightarrow d$,
$c \Rightarrow a \vee b$, $d \Rightarrow a$,
$d \Rightarrow b$.

Definitions and motivation

$$x_1 + x_2 + 2x_3 + 4x_4 < 3$$



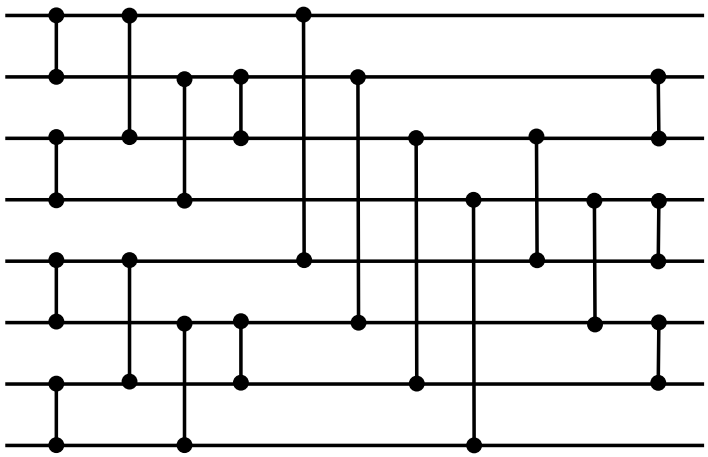
Bather's odd-even sorting network (1968)



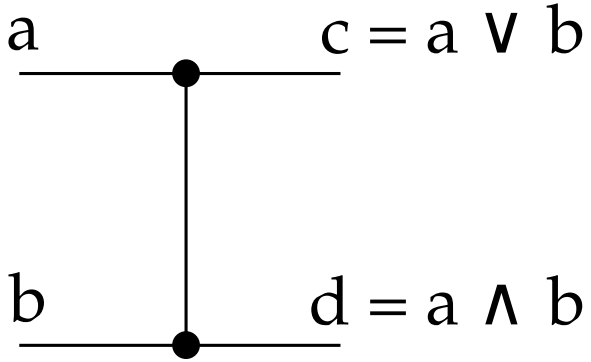
$a \Rightarrow c$, $b \Rightarrow c$,
$a \wedge b \Rightarrow d$,
$c \Rightarrow a \vee b$, $d \Rightarrow a$,
$d \Rightarrow b$.

Definitions and motivation

$$x_1 + x_2 + x_3 + x_3 + x_4 + x_4 + x_4 + x_4 < 3$$



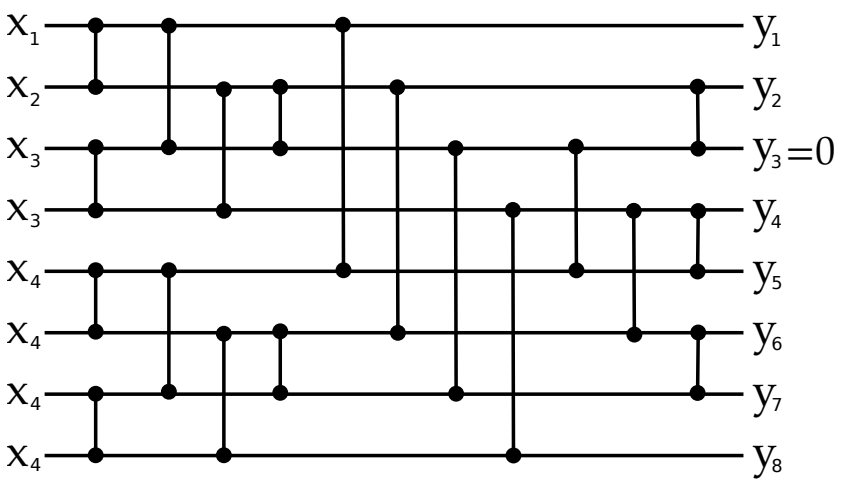
Bather's odd-even sorting network (1968)



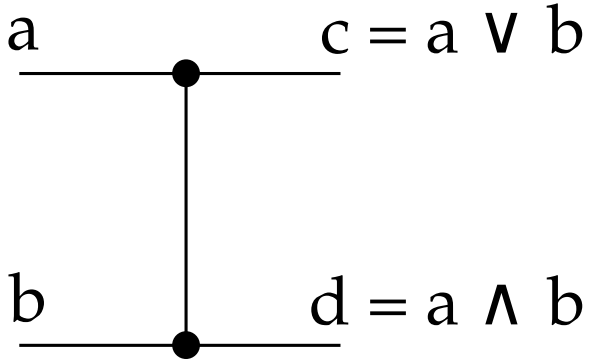
$a \Rightarrow c$, $b \Rightarrow c$,
$a \wedge b \Rightarrow d$,
$c \Rightarrow a \vee b$, $d \Rightarrow a$,
$d \Rightarrow b$.

Definitions and motivation

$$x_1 + x_2 + x_3 + x_3 + x_4 + x_4 + x_4 + x_4 < 3$$



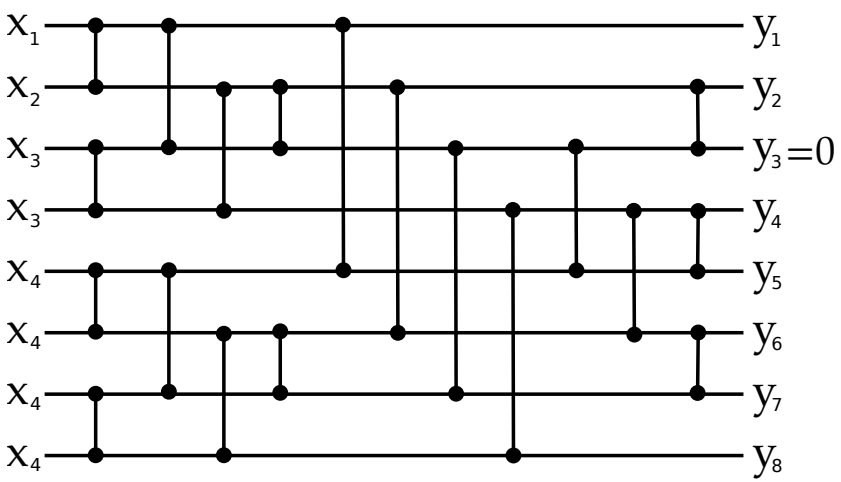
Bather's odd-even sorting network (1968)



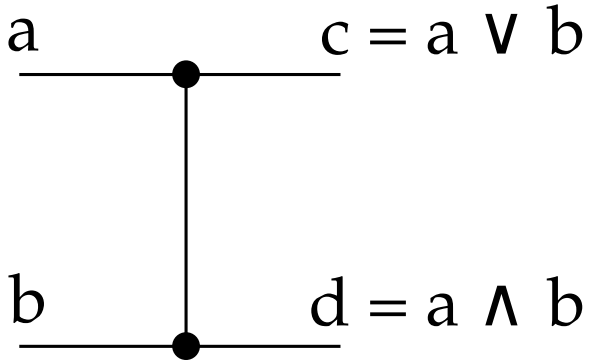
$a \Rightarrow c$, $b \Rightarrow c$,
$a \wedge b \Rightarrow d$,
$c \Rightarrow a \vee b$, $d \Rightarrow a$,
$d \Rightarrow b$.

Definitions and motivation

$$x_1 + x_2 + x_3 + x_3 + x_4 + x_4 + x_4 + x_4 < 3$$



Bather's odd-even sorting network (1968)



$a \Rightarrow c$, $b \Rightarrow c$,
$a \wedge b \Rightarrow d$.

Definitions and motivation

$$3x_1 + 2x_2 + 4x_3 \leq 5, \qquad 30001x_1 + 19999x_2 + 39998x_3 \leq 50007$$

Definitions and motivation

$$3x_1 + 2x_2 + 4x_3 \leq 5, \qquad 30001x_1 + 19999x_2 + 39998x_3 \leq 50007$$

$$(\bar{x}_1 \vee \bar{x}_3) \wedge (\bar{x}_2 \vee \bar{x}_3)$$

Mixed Radix Base Technique

Mixed Radix Base Technique by Niklas Eén and Niklas Sörensson (2006) – authors of MiniSat+.

Mixed Radix Base Technique

Mixed Radix Base Technique by Niklas Eén and Niklas Sörensson (2006) – authors of MiniSat+.

- Positive integer $\mathbf{d} = \langle d_0, \ldots, d_{m-1} \rangle$.
- Base $\mathbf{B}$ is a sequence of positive integers $\langle r_0, \ldots, r_{m-1} \rangle$.
- The value of $\mathbf{d}$ is $v = d_0 w_0 + d_1 w_1 + d_2 w_2 + \cdots + d_{m-1} w_{m-1}$ where $w_0 = 1$ and for $i \geq 0$, $w_{i+1} = w_i r_i$.

Mixed Radix Base Technique

Mixed Radix Base Technique by Niklas Eén and Niklas Sörensson (2006) – authors of MiniSat+.

- Positive integer $\mathbf{d} = \langle d_0, \ldots, d_{m-1} \rangle$.
- Base $\mathbf{B}$ is a sequence of positive integers $\langle r_0, \ldots, r_{m-1} \rangle$.
- The value of $\mathbf{d}$ is $v = d_0 w_0 + d_1 w_1 + d_2 w_2 + \cdots + d_{m-1} w_{m-1}$ where $w_0 = 1$ and for $i \geq 0$, $w_{i+1} = w_i r_i$.

Example: The number $\langle 2, 4, 10 \rangle_{\mathbf{B}}$ in base $\mathbf{B} = \langle 3, 5 \rangle$ is interpreted as:
$2 \times \mathbf{1} + 4 \times \mathbf{3} + 10 \times \mathbf{15} = 164$ (values of w_i's in boldface).

Mixed Radix Base Technique

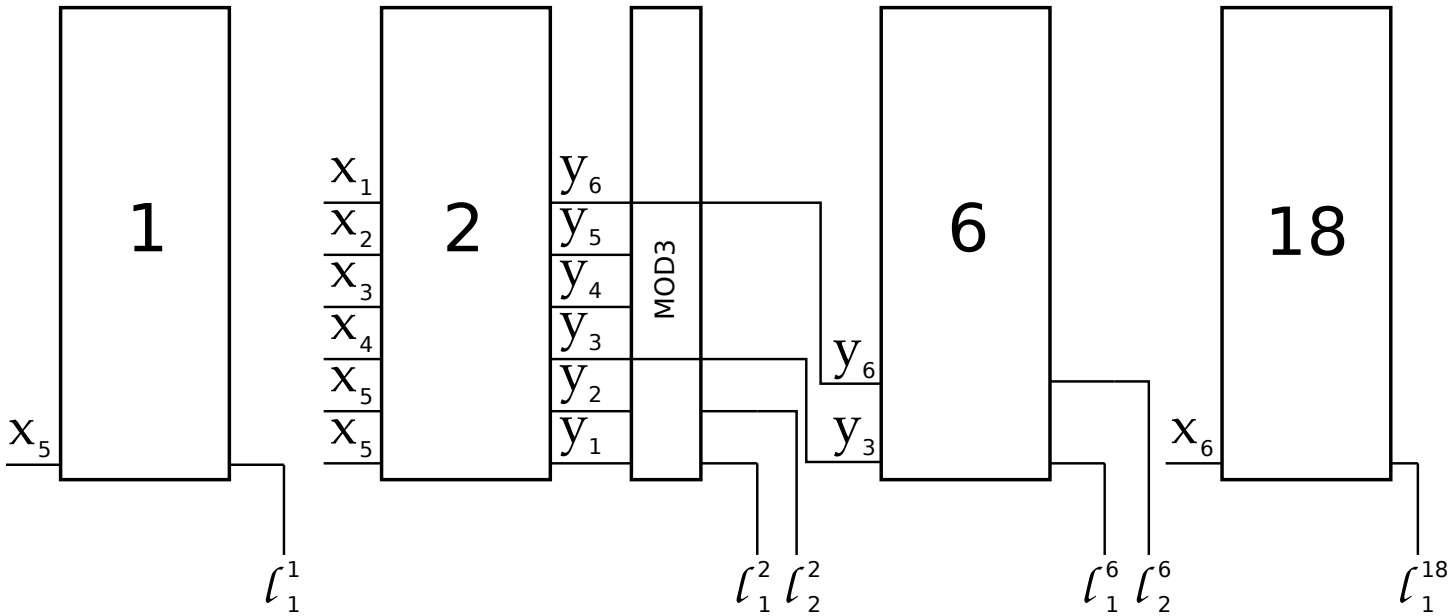
For PB-constraint with coefficients $\{a_1, \ldots, a_n\}$ do:

1. Find a "suitable" finite base $\mathbf{B}$ for the set of coefficients.
2. For each element w_i (computed from $\mathbf{B}$) construct a sorting network where the inputs to i-th sorter will be those digits $\mathbf{d}$ (from the coefficients) where d_i is non-zero, plus the potential carry bits from the $(i-1)$-th sorter.

Mixed Radix Base Technique

$$2x_1 + 2x_2 + 2x_3 + 2x_4 + 5x_5 + 18x_6 \leq 22,\ \mathbf{B} = \langle 2, 3, 3 \rangle$$

$$\mathbf{1} \cdot (x_5) + \mathbf{2} \cdot (x_1 + x_2 + x_3 + x_4 + 2x_5) + \mathbf{6} \cdot (0) + \mathbf{18} \cdot (x_6)$$



$22 = \langle 0, 2, 0, 1 \rangle_{\mathbf{B}}$

Enforce a constraint ≤ 22: $(l_1^{18} \Rightarrow \neg l_1^6) \wedge (l_1^{18} \Rightarrow \neg(l_2^2 \wedge l_1^1))$.

Mixed Radix Base Technique

Simplification of the inequality (NaPS):

Add $13 = \langle 1, 0, 2, 0 \rangle_{\mathbf{B}}$ to both sides:

$$2x_1 + 2x_2 + 2x_3 + 2x_4 + 5x_5 + 18x_6 + 13 < 36,\ \mathbf{B} = \langle 2, 3, 3 \rangle$$

$36 = \langle 0, 0, 0, 2 \rangle_{\mathbf{B}}$

$$\mathbf{1} \cdot (1 + x_5) + \mathbf{2} \cdot (x_1 + x_2 + x_3 + x_4 + 2x_5) + \mathbf{6} \cdot (1 + 1) + \mathbf{18} \cdot (x_6)$$

Enforce a constraint < 36: $\neg l_2^{18}$.

New encoding and analysis

Our improvement: use **assumptions** rather than constant 1's.

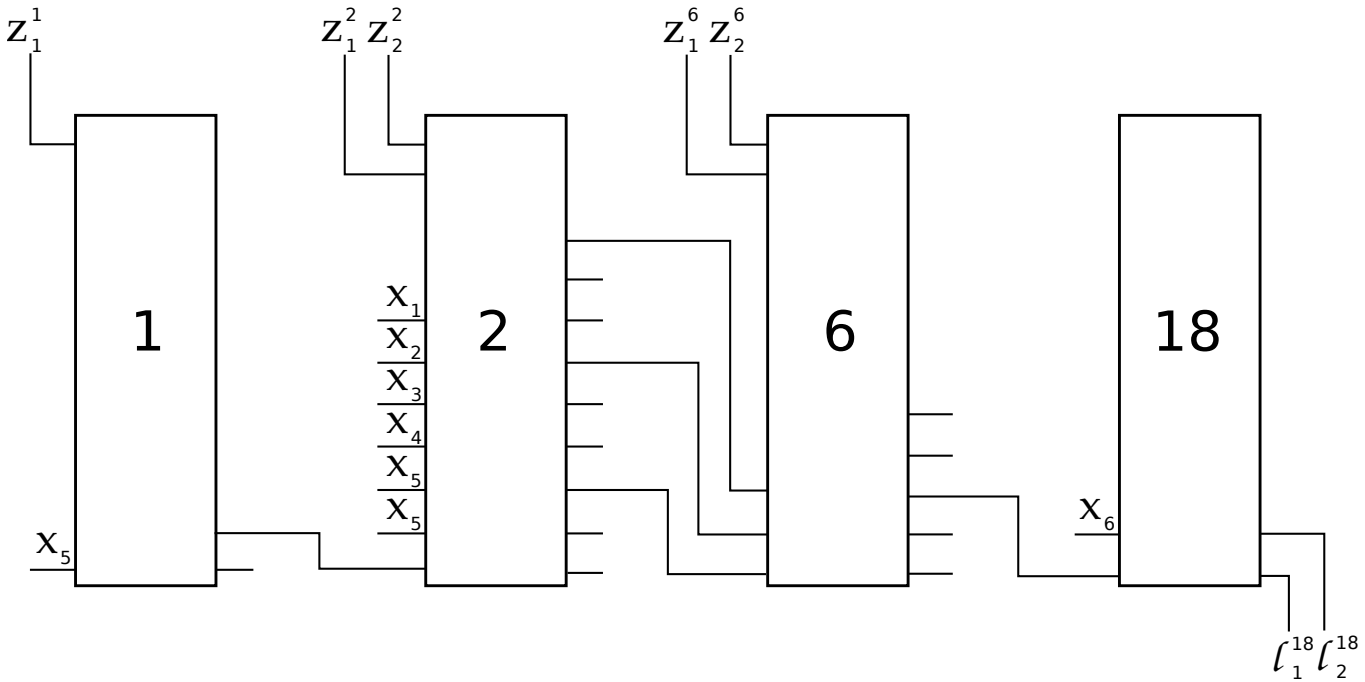
Assumptions is a mechanism, which allows to accumulate and reuse knowledge from one iteration to the next and, in consequence, the provided input formula need not to be rebuilt during computation.

New encoding and analysis

$$2x_1 + 2x_2 + 2x_3 + 2x_4 + 5x_5 + 18x_6 \leq 22,\ \mathbf{B} = \langle 2, 3, 3 \rangle$$

Idea: create assumption variables $z_1^1, z_1^2, z_2^2, z_1^6, z_2^6$.

$$\mathbf{1}\cdot(z_1^1 + x_5) + \mathbf{2}\cdot(z_1^2 + z_2^2 + x_1 + x_2 + x_3 + x_4 + 2x_5) + \mathbf{6}\cdot(z_1^6 + z_2^6) + \mathbf{18}\cdot(x_6)$$



Enforce < 36: set $z_1^1 = 1$, $z_1^2 = z_2^2 = 0$, $z_1^6 = z_2^6 = 1$, and add $\neg l_2^{18}$.

Experimental evaluation

- KP-MiniSat+ (https://github.com/karpiu/kp-minisatp).
- SAT-solver used is COMiniSatPS by Chanseok Oh.
- Three variants:
  - **KP-MSP++** – new algorithm with assumptions processing improvement due to Hickey and Bacchus (2019),
  - **KP-MSP+-** – new algorithm but without the improvement,
  - **KP-MSP–** – without new algorithm and improvement.

Experimental evaluation

Solvers:

- **PBLib** - ver. 1.2.1, by Tobias Philipp and Peter Steinke,
- **NaPS** - ver. 1.02b by Masahiko Sakai and Hidetomo Nabeshima,
- **MS+** - original MiniSat+ by Eén and Sörensson,
- **MS+COM** - MiniSat+ with COMiniSatPS as an underlying SAT-solver.

Experimental evaluation

- Machines: Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz.
- Timeout limit: 1800 seconds.
- Memory limit: 15 GB.

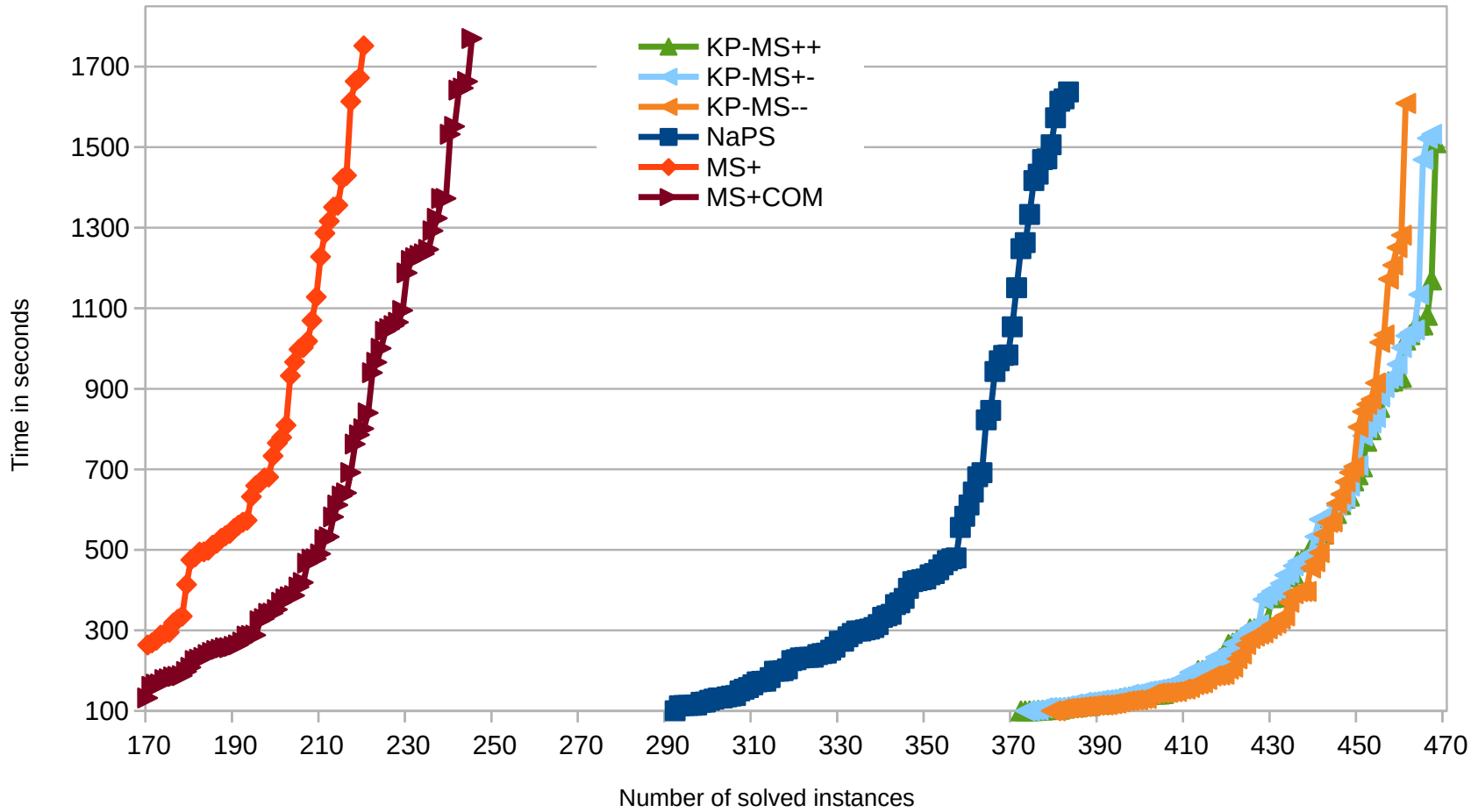
Experimental evaluation

Pseudo-Boolean Competition 2016:

- **OPT-BIGINT-LIN** - 1109 instances of optimization problems with big coefficients in the constraints (at least one constraint with a sum of coefficients greater than 2^{20}). An objective function is present. The solver must find a solution with the best possible value of the objective function.
- **OPT-SMALLINT-LIN** - 1600 instances of optimization problems. Like OPT-BIGINT-LIN but with small coefficients in the constraints (no constraint with sum of coefficients greater than 2^{20}).

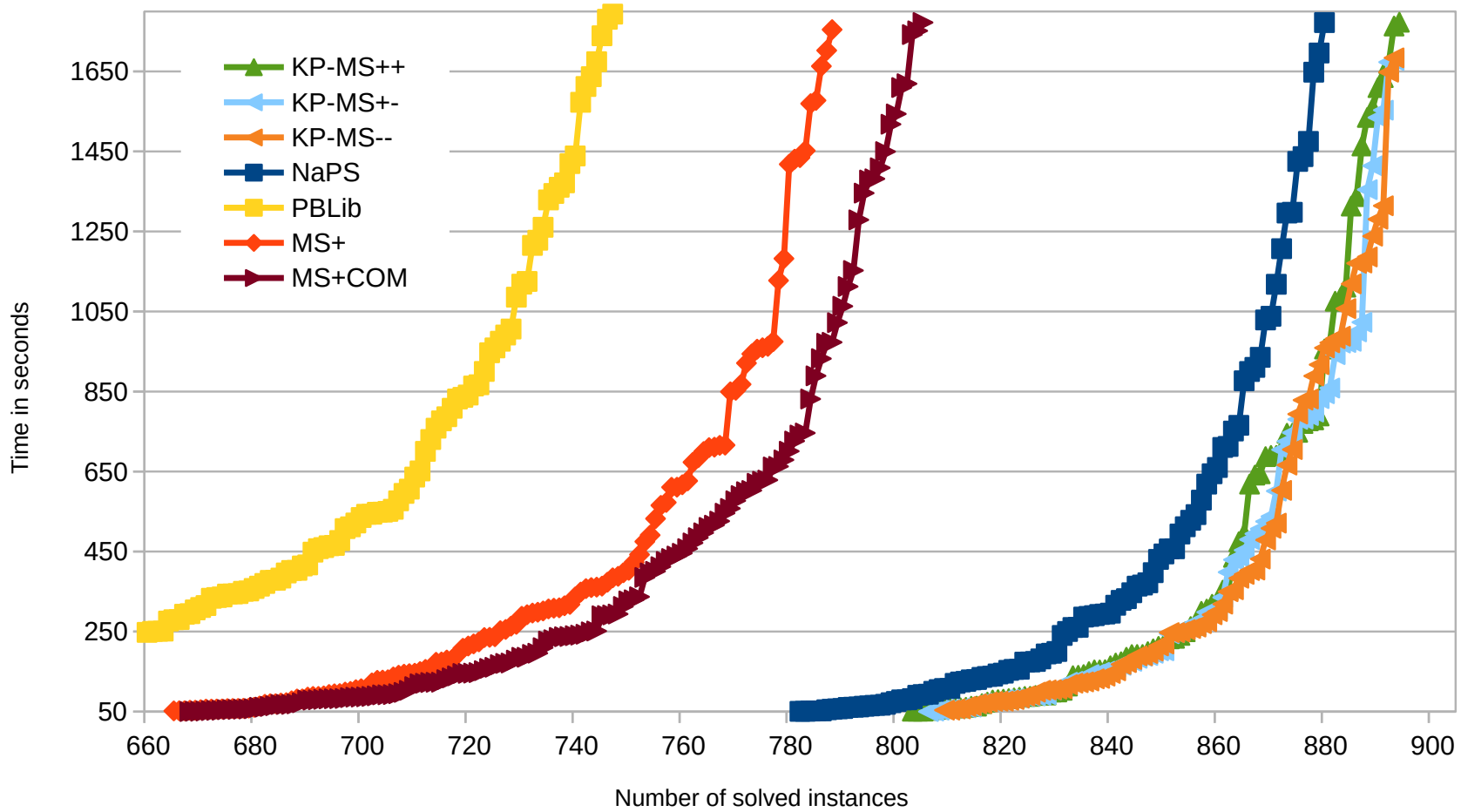
Experimental evaluation

OPT-BIGINT-LIN:



Experimental evaluation

OPT-SMALLINT-LIN:



Experimental evaluation

OPT-BIGINT-LIN:

solver	solved	Opt	UnSat	cpu (s)	scpu (s)	avg(scpu)	smem(MB)	avg(smem)
KP-MSP++	**468**	**395**	**73**	1046518	44424	94.9	208035	444.5
KP-MSP+-	467	**395**	72	**1037085**	44886	96.1	213973	458.2
KP-MSP--	461	389	72	1039499	**37672**	**81.7**	283681	615.4
NaPS	383	314	69	1314536	51557	134.6	245533	641.1
MS+	220	149	71	1647958	47759	217.1	**42181**	191.7
MS+COM	245	174	71	1609433	54234	221.4	46336	**189.1**

Experimental evaluation

OPT-SMALLINT-LIN:

solver	solved	Opt	UnSat	cpu (s)	scpu (s)	avg(scpu)	smem(MB)	avg(smem)
KP-MSP++	**894**	808	86	1282788	43556	48.7	164223	183.7
KP-MSP+-	893	806	**87**	1278926	38474	43.1	162405	181.9
KP-MSP--	893	**809**	84	**1278722**	**37747**	**42.3**	153619	172.0
NaPS	887	803	84	1310006	40376	45.5	186760	210.6
PBLib	747	691	56	1611247	74993	100.4	112993	151.3
MS+	788	715	73	1515166	53566	68.0	113606	144.2
MS+COM	805	734	71	1491269	60270	74.9	**106886**	**132.8**

The end

Thank You!