Preprocessing in Pseudo-Boolean Optimization: An Experimental Evaluation

Ruben Martins Inês Lynce Vasco Manquinho

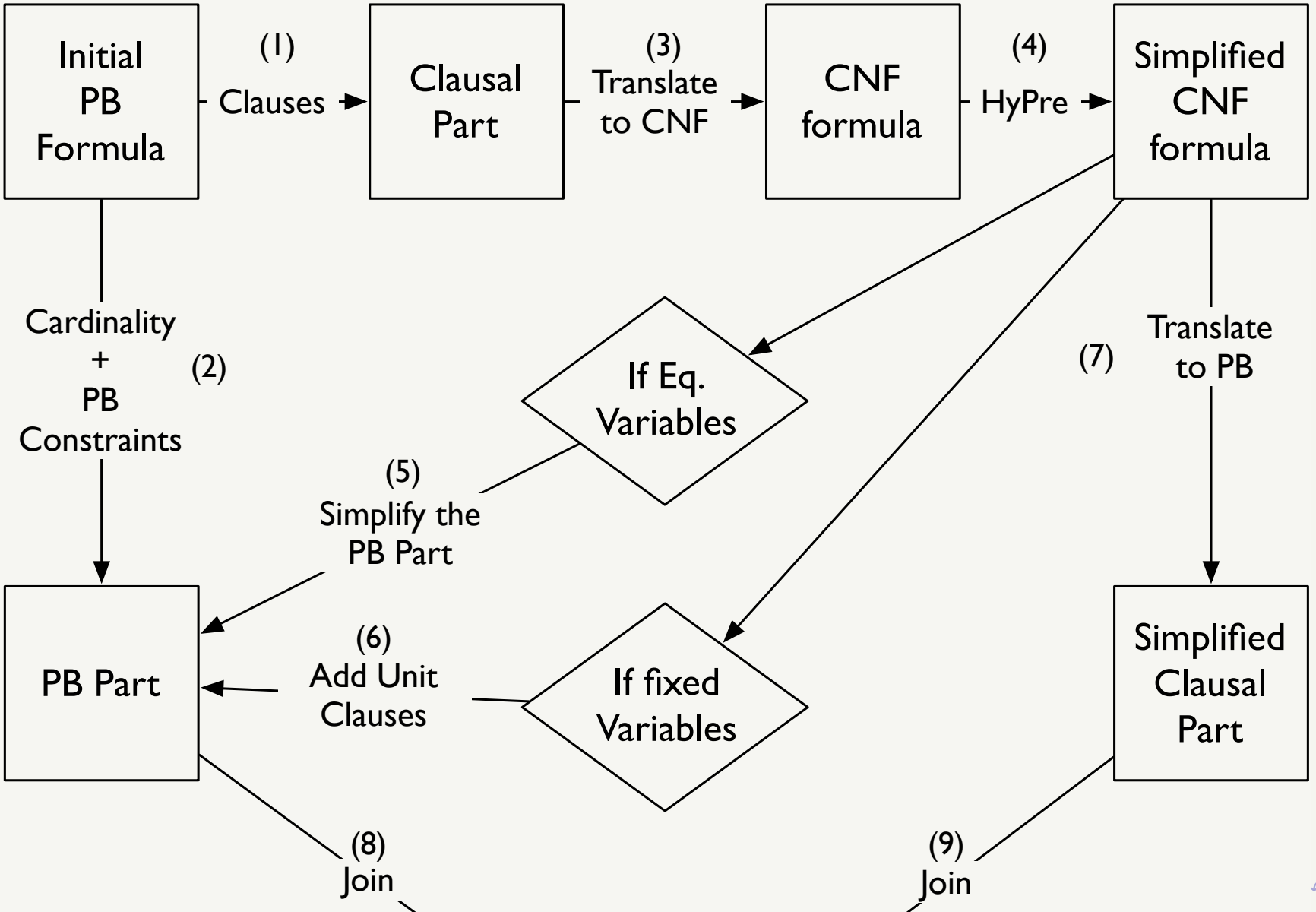
IST/INESC-ID, Technical University of Lisbon, Portugal

20/09/2009 Lisbon, Portugal

Motivation

- Preprocessing is a powerful tool in SAT solving:
  - e.g. `HyPre`, `NiVER`, `SatELite`, `ReVivAl`, `extHyPre`, ...
- PBO is closely related to SAT
- However, preprocessing is almost non existent in PBO
- We propose to benefit from the advances in SAT preprocessing and use them in PBO
- In this talk, we will show how we can use `HyPre` and `extHyPre` to preprocess a PBO instance

HyPre in PBO


Initial PB Formula
(1) Clauses
Clausal Part
(3) Translate to CNF
CNF formula
(4) HyPre
Simplified CNF formula
Cardinality + PB Constraints
(2)
If Eq. Variables
(7) Translate to PB
(5) Simplify the PB Part
PB Part
(6) Add Unit Clauses
If fixed Variables
Simplified Clausal Part
(8) Join
(9) Join

HyPre in PBO

$x_1 \vee x_2 \vee x_3$

$\sim x_1 \vee x_4$

$\sim x_2 \vee x_4$

$\sim x_3 \vee x_4$

HyPre in PBO

$x_1 \vee x_2 \vee x_3$

$x_1 \vee x_2 \vee x_3$

$x_1 \vee x_2 \vee x_3$

$x_1 \vee x_2 \vee x_3$

$\sim x_3 \vee x_4$

Background (SAT and PBO)

- SAT (Boolean Satisfiability)
  - CNF formula $\varphi = \bigvee x_j$
  - *SAT problem* is to dedice if $\bigwedge_i \varphi_i$ is satisfiable
- PBO (Pseudo-Boolean Optimization)
  - PB constraints $\varphi = \sum_j a_j x_j \geq b$
  - Linear objective function $f = \sum_j c_j x_j$
  - *PBO problem* is to find an assignment to x_j such that all φ_i are satisfied and the value of f is optimized

Background (PB Constraints)

- PB constraints can be divided into three categories:
  - *cardinality constraints*:

    e.g. $x_1 + x_2 + x_3 + \overline{x}_4 \geq 2$
  - *clauses*:

    e.g. $x_1 + \overline{x}_2 + x_3 \geq 1$
  - *general PB constraint:*

    e.g. $7x_1 + 4\overline{x}_2 + 5x_3 \geq 7$
- This categories form two distinct parts:
  - The *PB part* is formed by the conjunction of cardinality constraints and general PB constraints
  - The *clausal part* is formed by the conjunction of clauses

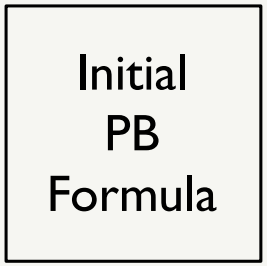
HyPre Preprocessor

- HyPre is based on *Hyper-Binary Resolution* (HypBinRes):

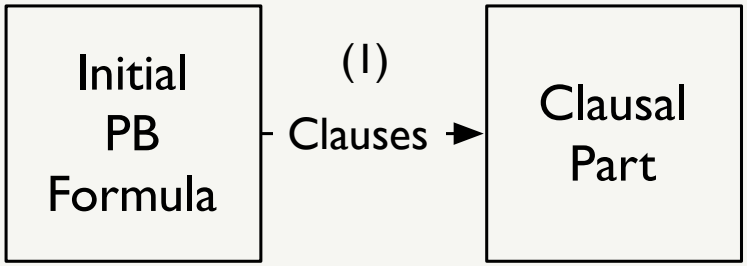
$$\frac{(l_1 \vee \ldots \vee l_n) \quad (\overline{l_1} \vee y) \ldots (\overline{l_{n-1}} \vee y)}{(y \vee l_n)} \quad \text{for } n \geq 2$$

- In addition to HypBinRes, HyPre also simplifies the formula through Equality Reduction and Unit Propagation.

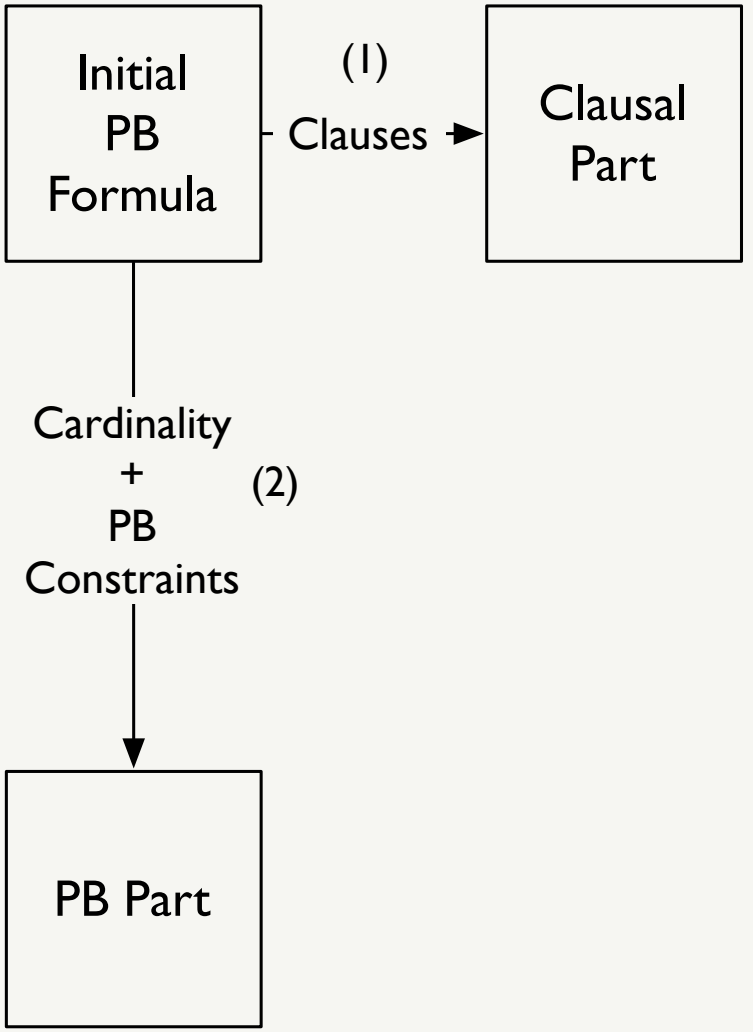
HyPre in PBO



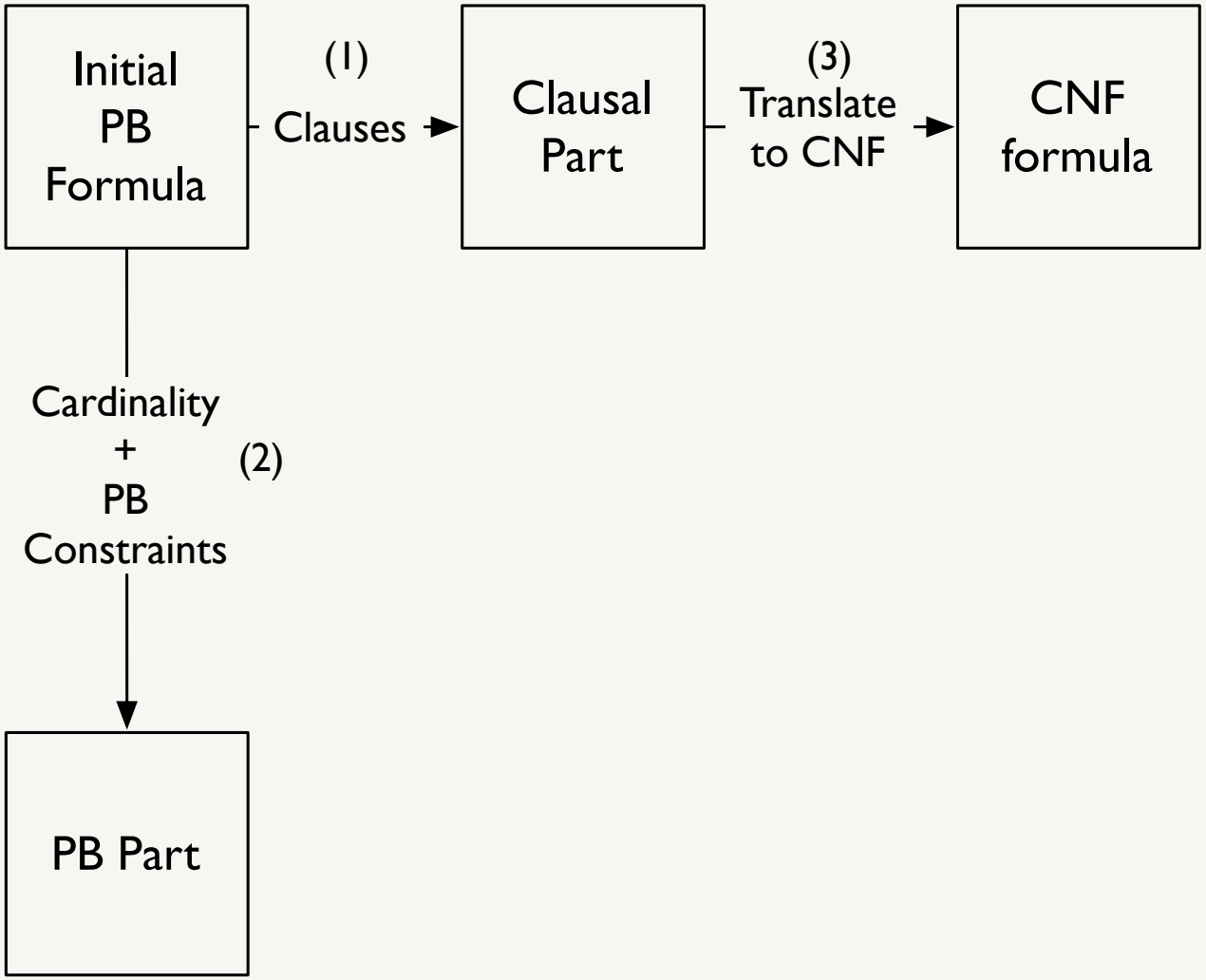
HyPre in PBO



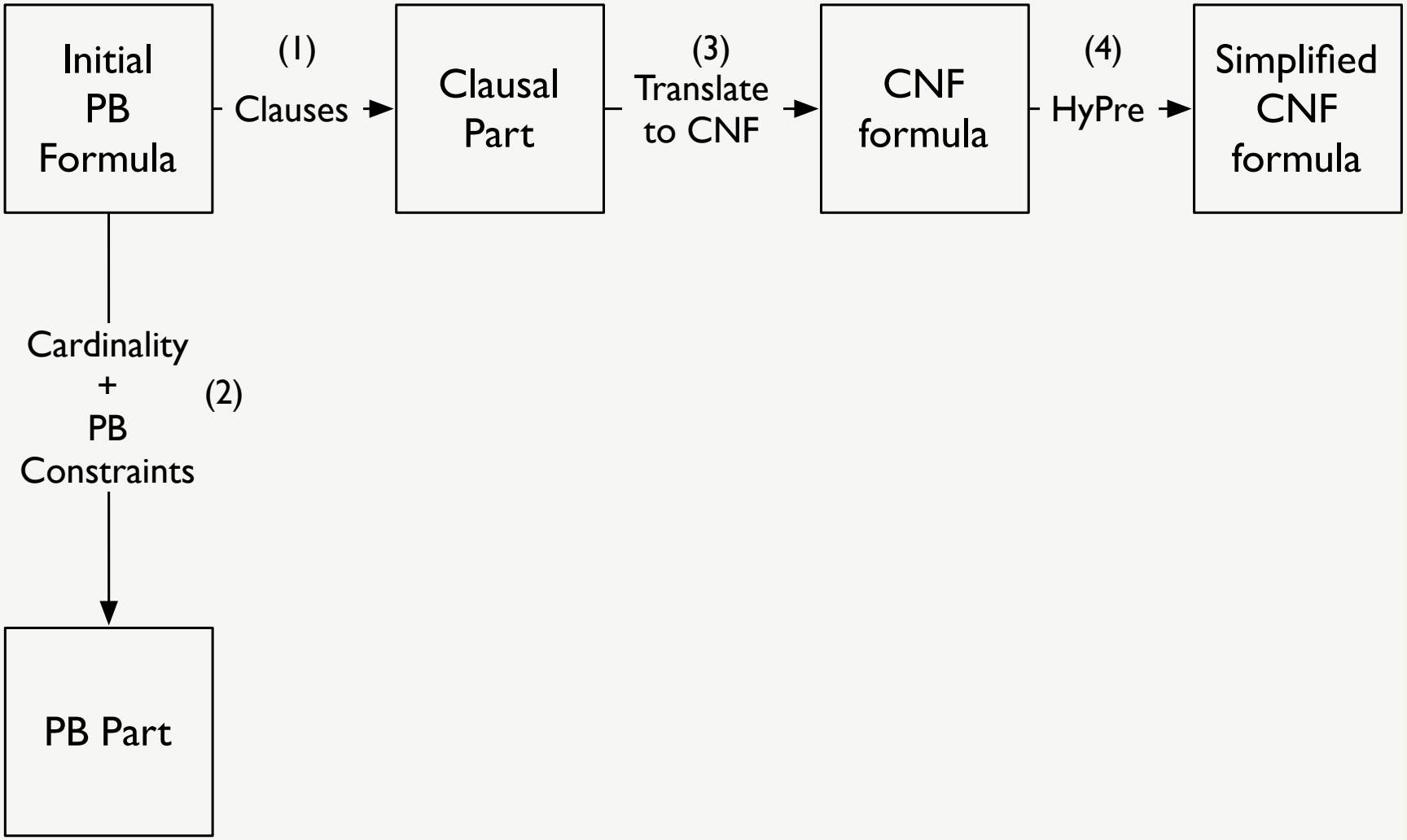
HyPre in PBO


Initial
PB
Formula
(1)
Clauses
Clausal
Part
Cardinality
+
PB
Constraints
(2)
PB Part

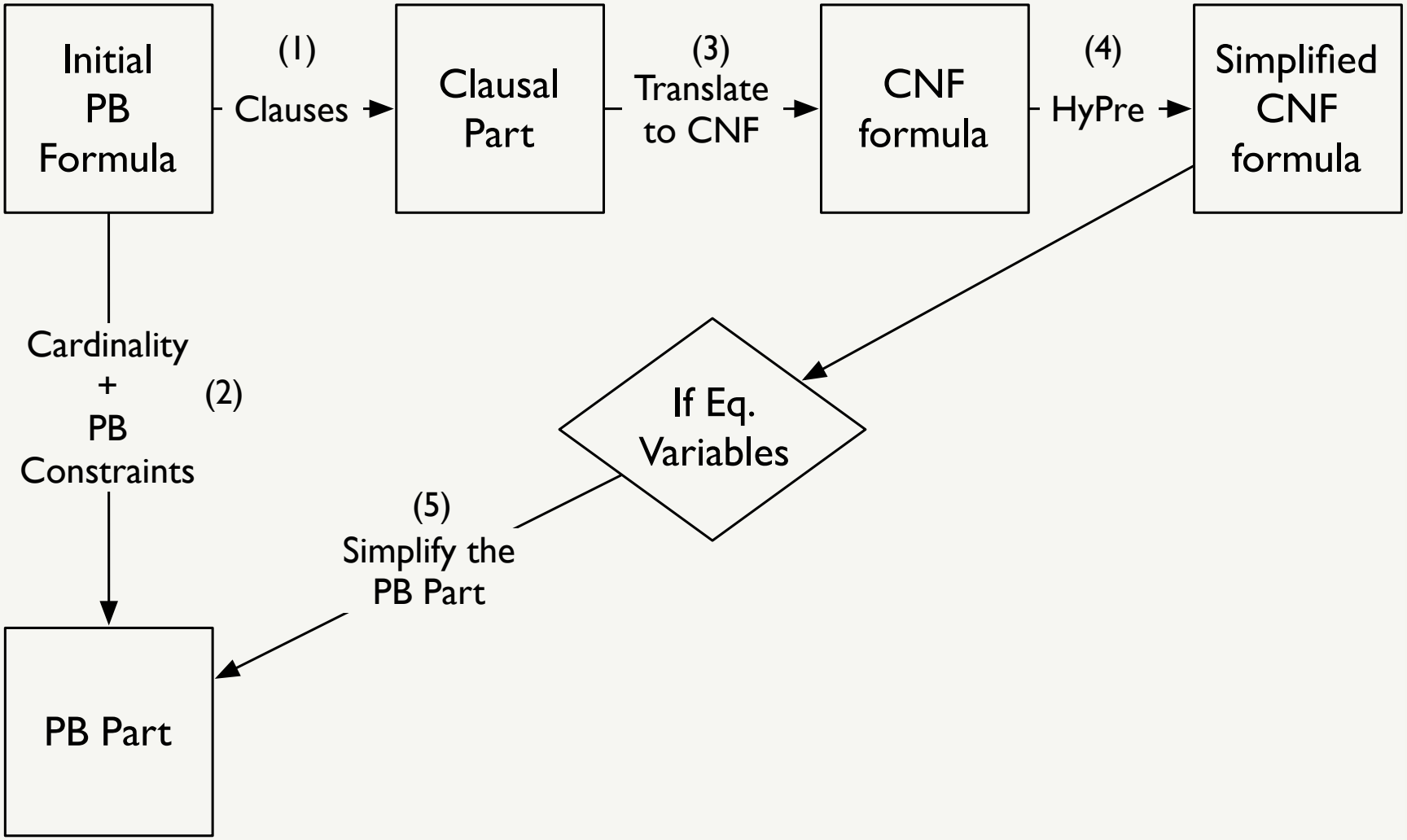
HyPre in PBO


Initial
PB
Formula
(1)
Clauses
Clausal
Part
(3)
Translate
to CNF
CNF
formula
Cardinality
+
PB
Constraints
(2)
PB Part

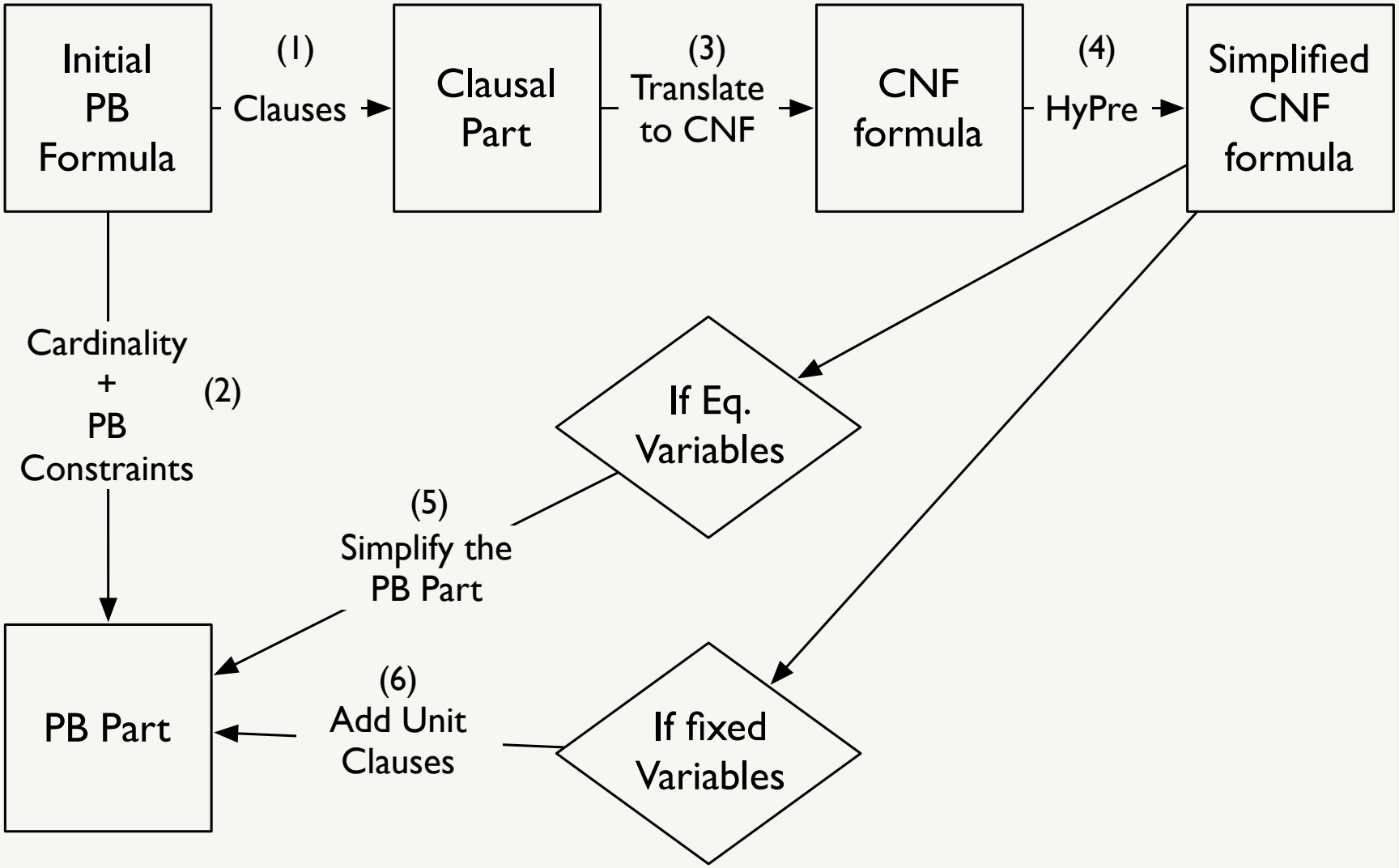
HyPre in PBO


Initial PB Formula
(1)
Clauses
Clausal Part
(3)
Translate to CNF
CNF formula
(4)
HyPre
Simplified CNF formula
Cardinality + PB Constraints
(2)
PB Part

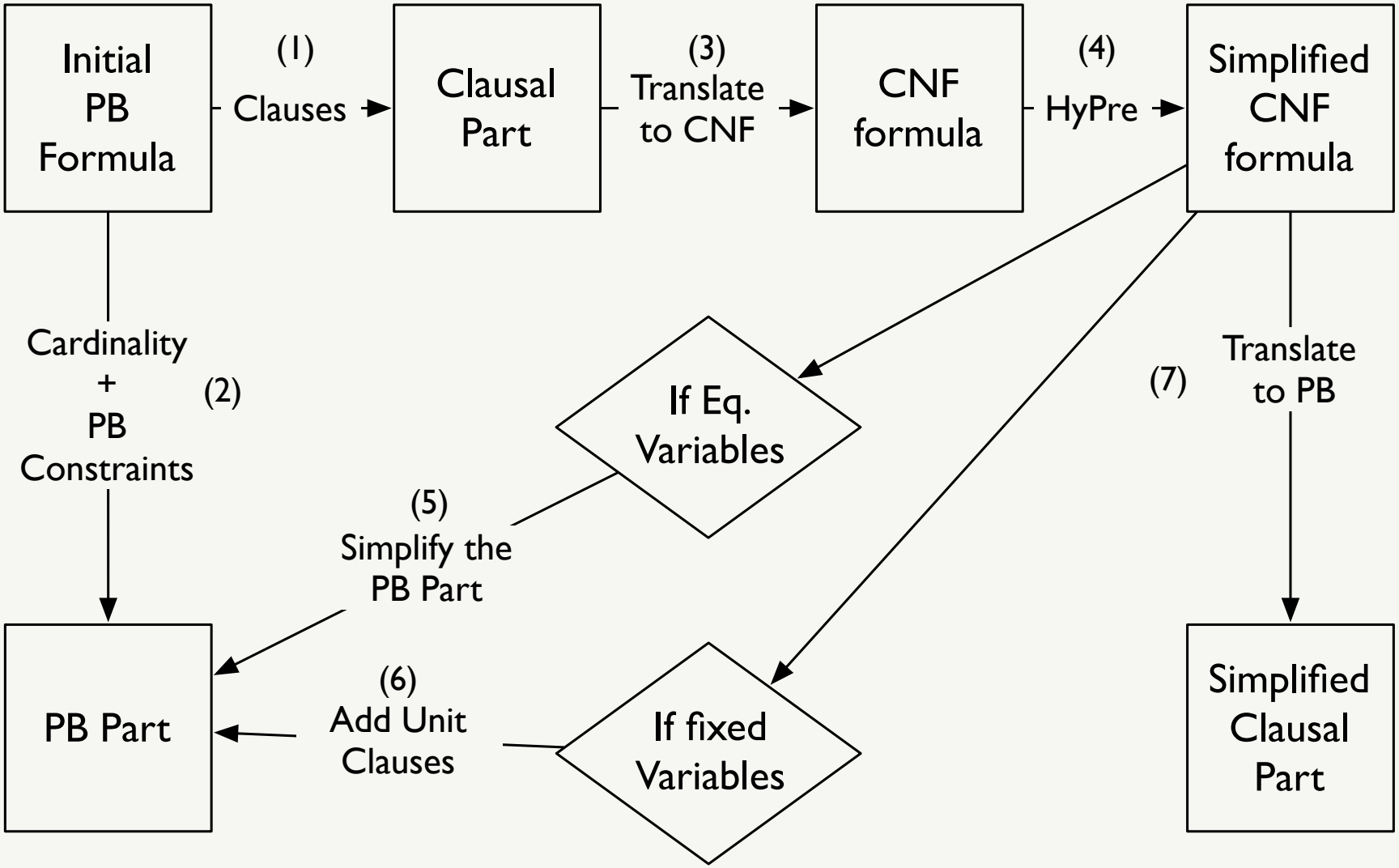
HyPre in PBO


Initial PB Formula
(1) Clauses
Clausal Part
(3) Translate to CNF
CNF formula
(4) HyPre
Simplified CNF formula
Cardinality + PB Constraints
(2)
If Eq. Variables
(5) Simplify the PB Part
PB Part

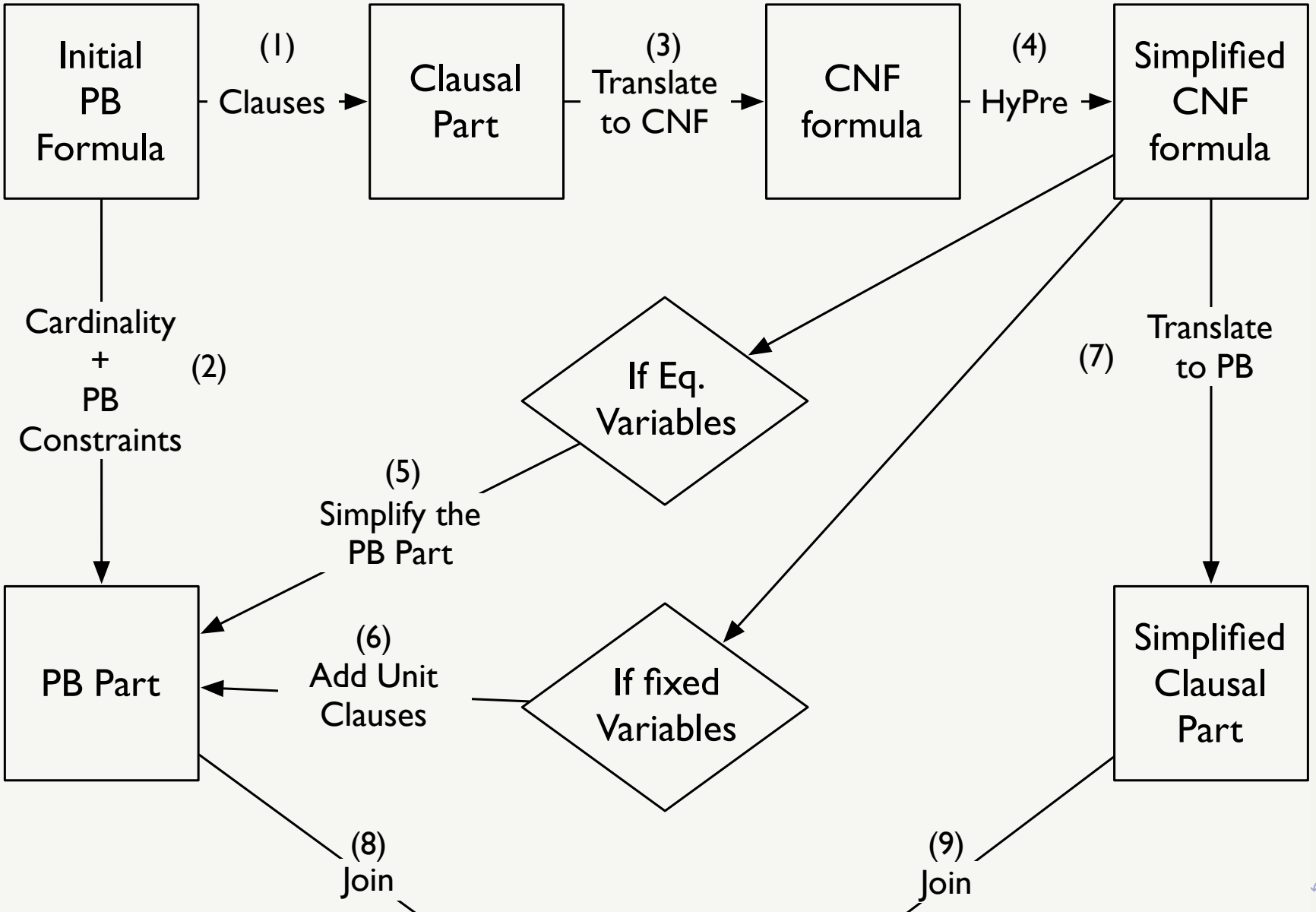
HyPre in PBO


Initial PB Formula
(1) Clauses
Clausal Part
(3) Translate to CNF
CNF formula
(4) HyPre
Simplified CNF formula
Cardinality + PB Constraints
(2)
If Eq. Variables
(5) Simplify the PB Part
PB Part
(6) Add Unit Clauses
If fixed Variables

HyPre in PBO


Initial PB Formula
(1) Clauses
Clausal Part
(3) Translate to CNF
CNF formula
(4) HyPre
Simplified CNF formula
Cardinality + PB Constraints
(2)
If Eq. Variables
(5) Simplify the PB Part
(7) Translate to PB
PB Part
(6) Add Unit Clauses
If fixed Variables
Simplified Clausal Part

HyPre in PBO


Initial PB Formula
(1) Clauses
Clausal Part
(3) Translate to CNF
CNF formula
(4) HyPre
Simplified CNF formula
Cardinality + PB Constraints
(2)
If Eq. Variables
(7) Translate to PB
(5) Simplify the PB Part
PB Part
(6) Add Unit Clauses
If fixed Variables
Simplified Clausal Part
(8) Join
(9) Join

`extHyPre` Preprocessor

- `extHyPre` is based on *Extended Hyper Resolution (ExtHypRes)*, which is an extension of *HypBinRes*:

$$\frac{(l_1 \vee l_2 \vee \ldots \vee l_n) \quad (\overline{l_1} \vee \alpha_1), \ldots, (\overline{l_n} \vee \alpha_n)}{(\bigcup_{1 \leq i \leq n}(\alpha_i))}$$

  where α_i for $1 \leq i \leq n$ are sub-clauses.

- `extHyPre` produces new resolvent clauses that are added to the original formula

New graph representation of a CNF formula

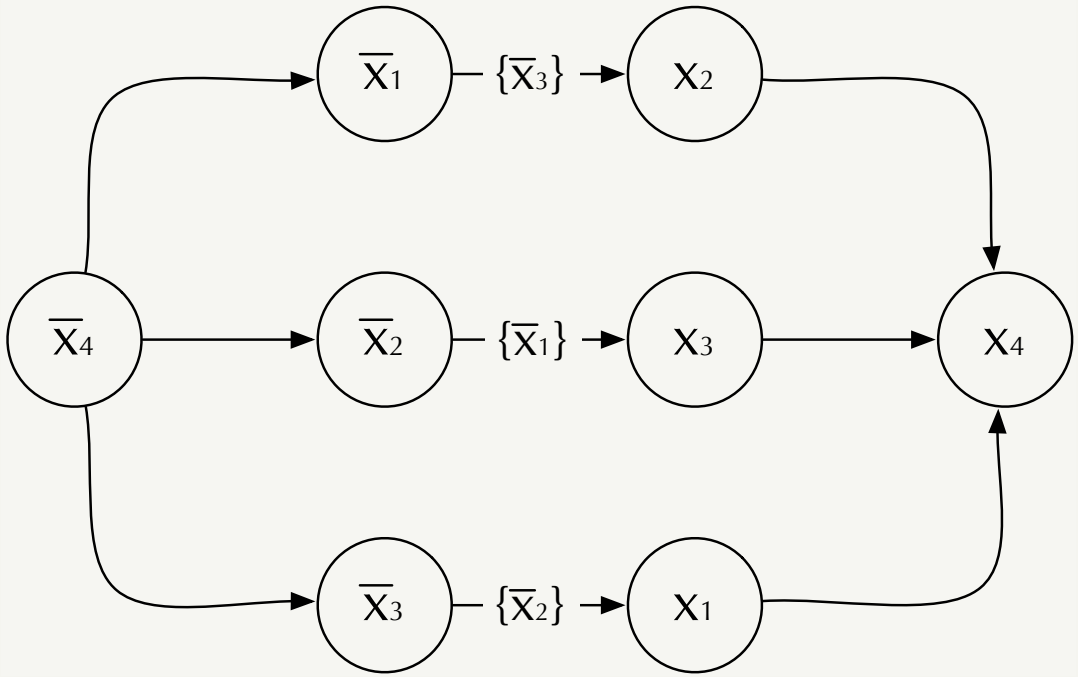
- `extHyPre` is applied through a new graph representation of a CNF formula:
  - The idea behind this representation is to extend the binary implication graph for clauses of any size

$x_1 \vee x_2 \vee x_3$

$\overline{x}_1 \vee x_4$

$\overline{x}_2 \vee x_4$

$\overline{x}_3 \vee x_4$



New graph representation of a CNF formula

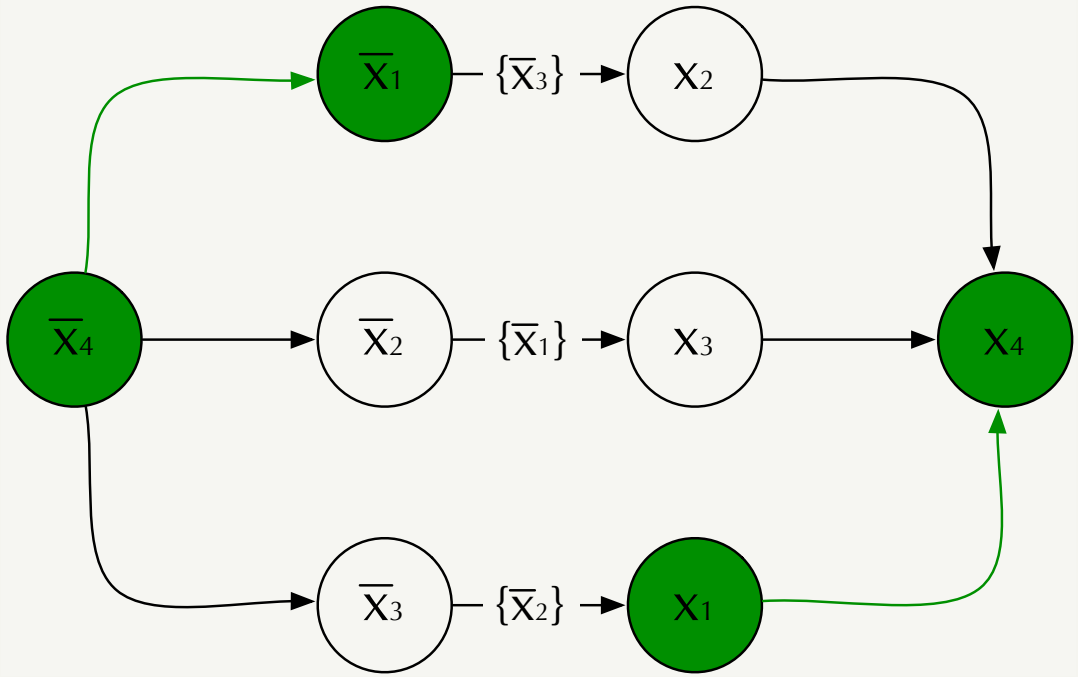
- `extHyPre` is applied through a new graph representation of a CNF formula:
  - The idea behind this representation is to extend the binary implication graph for clauses of any size

$x_1 \vee x_2 \vee x_3$

$\overline{x}_1 \vee x_4$

$\overline{x}_2 \vee x_4$

$\overline{x}_3 \vee x_4$



New graph representation of a CNF formula

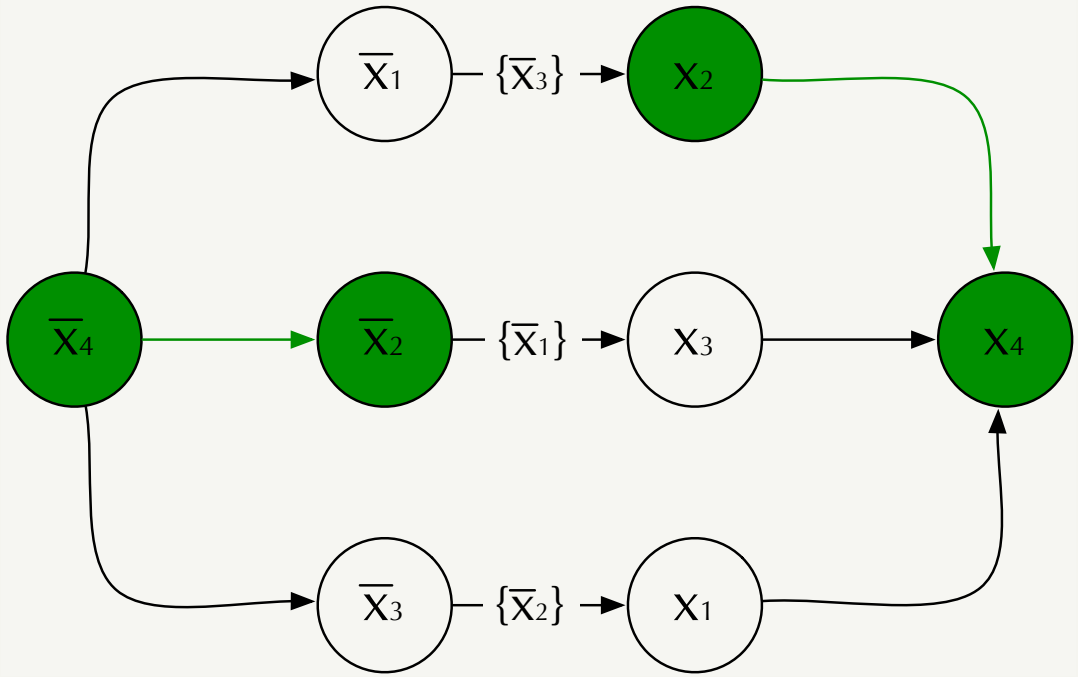
- `extHyPre` is applied through a new graph representation of a CNF formula:
  - The idea behind this representation is to extend the binary implication graph for clauses of any size

$x_1 \vee x_2 \vee x_3$

$\overline{x}_1 \vee x_4$

$\overline{x}_2 \vee x_4$

$\overline{x}_3 \vee x_4$



New graph representation of a CNF formula

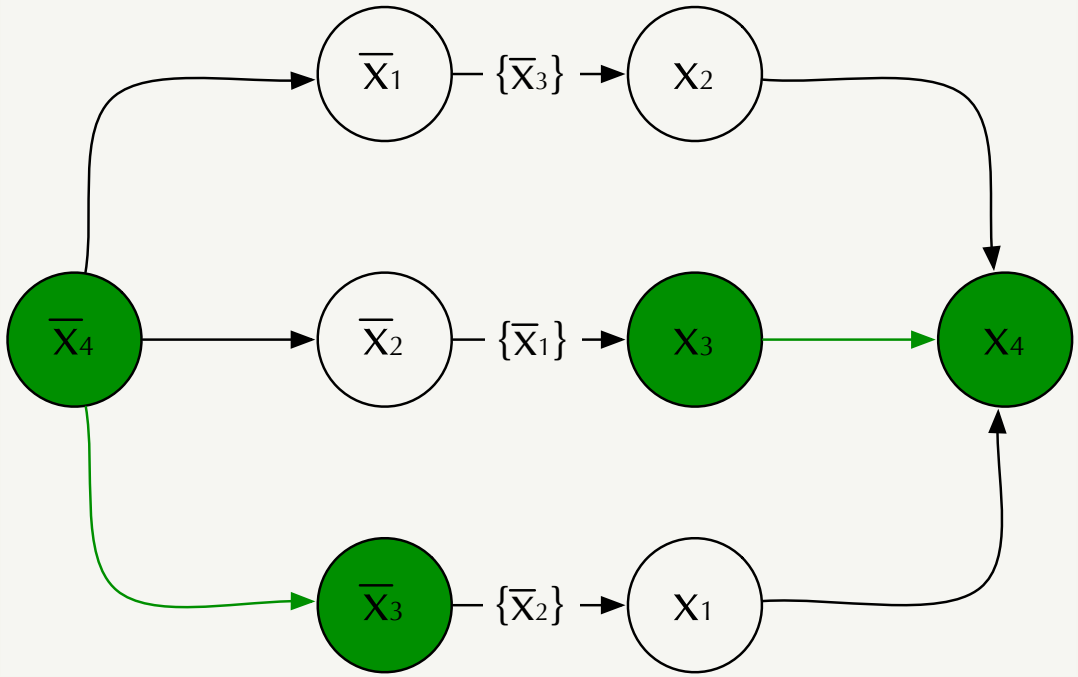
- `extHyPre` is applied through a new graph representation of a CNF formula:
  - The idea behind this representation is to extend the binary implication graph for clauses of any size

$x_1 \vee x_2 \vee x_3$

$\overline{x}_1 \vee x_4$

$\overline{x}_2 \vee x_4$

$\overline{x}_3 \vee x_4$



New graph representation of a CNF formula

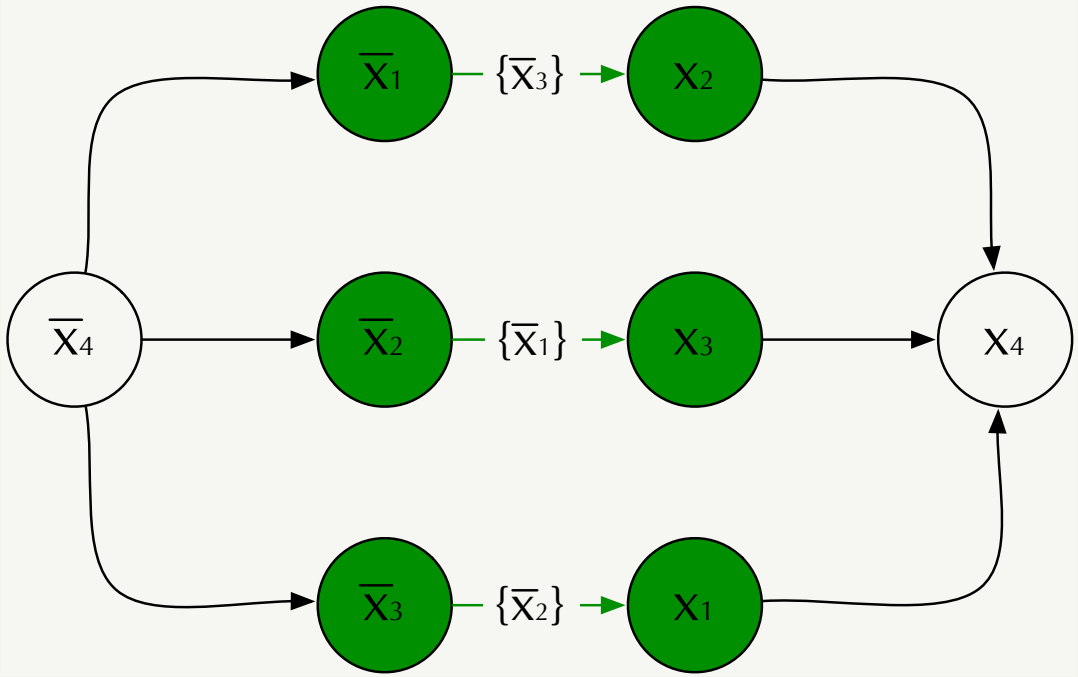
- `extHyPre` is applied through a new graph representation of a CNF formula:
  - The idea behind this representation is to extend the binary implication graph for clauses of any size

$x_1 \vee x_2 \vee x_3$

$\overline{x}_1 \vee x_4$

$\overline{x}_2 \vee x_4$

$\overline{x}_3 \vee x_4$



extHyPre in PBO

Enrich the graph representation:

- Information from *cardinality constraints*:
  - Given $x_1, \ldots, x_n \geq b$
  - Suppose that $n - b + 1$ literals are falsified
  - Then if one of the $b + 1$ literals is falsified it implies that all the remaining b literals are satisfied

- $x_1 + x_2 + x_3 + x_4 \geq 2$

$$\overline{x_1} \Rightarrow_{\{\overline{x_2}\}} x_3 \wedge \overline{x_1} \Rightarrow_{\{\overline{x_2}\}} x_4$$
$$\overline{x_2} \Rightarrow_{\{\overline{x_3}\}} x_4 \wedge \overline{x_2} \Rightarrow_{\{\overline{x_3}\}} x_1$$
$$\overline{x_3} \Rightarrow_{\{\overline{x_4}\}} x_1 \wedge \overline{x_3} \Rightarrow_{\{\overline{x_4}\}} x_2$$
$$\overline{x_4} \Rightarrow_{\{\overline{x_1}\}} x_2 \wedge \overline{x_4} \Rightarrow_{\{\overline{x_1}\}} x_3$$

extHyPre in PBO

Enrich the graph representation:

- Information from *general PB constraints*:
  - Probing on each literal of each *general PB constraint*
  - $2x_1 + x_2 + x_3 \geq 2$

    $\overline{x_1} \Rightarrow x_2$
    $\overline{x_1} \Rightarrow x_3$

Clause Selection Heuristics

- `extHyPre` uses two fixed parameters:
  - maximum size of the resolvent : 75
  - percentage of clauses to be considered: 10%
- Heuristics for clause selection:
  - random - `extHyPre(75,10,r)`
  - larger clauses - `extHyPre(75,10,c)`
  - clauses with higher weight - `extHyPre(75,10,d)`

Experimental Results (HyPre I)

- Average size of the 309 instances to be preprocessed by HyPre and average time of preprocessing

Benchmark	#	Vars	Constraints			Time
			Cls	Card	PB	
aksoy	79	15,259.90	53,156.57	0.13	0.00	0.21
logic-synthesis	74	1,767.91	1,605.35	0.00	0.00	0.01
primes-dimacs-cnf	130	1,688.45	11,537.59	0.00	0.00	0.04
routing	10	679.20	2,023.00	24.00	0.00	0.01
synthesis-ptl-cmos	8	375.75	879.26	0.00	0.00	0.01
haplotype	8	19,926.75	1,186,541.00	3,260.75	0.00	1.30

Experimental Results (HyPre II)

- Impact of HyPre preprocessing on bsolo, Pueblo and minisat+

Benchmark	#	bsolo		Pueblo		minisat+	
		w/o pre	HyPre	w/o pre	HyPre	w/o pre	HyPre
aksoy	79	26	*24*	15	15	24	24
logic-synthesis	74	51	*50*	31	**32**	31	31
primes-dimacs-cnf	130	69	69	75	**79**	79	**83**
routing	10	9	*8*	10	*9*	10	10
synthesis-ptl-cmos	8	6	6	1	1	1	1
haplotype	8	0	0	0	0	8	8
Total	309	161	*157*	132	**136**	153	**157**

Experimental Results (HyPre III)

- Details of some instances of HyPre preprocessing on bsolo

Benchmark	Instance	bsolo	
		w/o pre	HyPre
aksoy	f20c10b_016_area_delay	**25** ✔	25
	fir07_area_delay	**16** ✔	17
	fir09_area_opers	**34** ✔	43
	fir08_trarea_ac	**1769** ✔	2235
	f20c10b_029_area_delay	27	**24** ✔
	fir07_area_opers	34	**32** ✔
logic-synthesis	prom2.pi	**287** ✔	302
routing	s4-4-3-2pb	**64** ✔	64

Experimental Results (HyPre III)

- Details of some instances of HyPre preprocessing on Pueblo

Benchmark	Instance	Pueblo	
		w/o pre	HyPre
logic-synthesis	sao2.b	25	**25** ✔
primes-dimacs-cnf	ssa7552-038	1448	**1448** ✔
	ssa7552-158	1327	**1327** ✔
	ssa7552-159	1327	**1327** ✔
	ssa7552-160	1359	**1359** ✔
routing	s4-4-3-7pb	**64** ✔	64

Experimental Results (HyPre III)

- Details of some instances of HyPre preprocessing on minisat+

Benchmark	Instance	minisat+	
		w/o pre	HyPre
primes-dimacs-cnf	ssa7552-038	1448	**1448** ✔
	ssa7552-158	1327	**1327** ✔
	ssa7552-159	1327	**1327** ✔
	ssa7552-160	1359	**1359** ✔

Experimental Results (extHyPre I)

- Average time in seconds of extHyPre preprocessing using the different clause selection heuristics

Benchmark	#	extHyPre (75,10,c)	extHyPre (75,10,d)	extHyPre (75,10,r)
aksoy	79	78.51	95.83	80.61
logic-synthesis	74	3.48	3.51	3.43
primes-dimacs-cnf	130	0.34	0.39	0.62
radar	12	0.12	0.18	0.12
routing	10	0.13	0.13	0.11
synthesis-ptl-cmos	8	0.04	0.05	0.04
testset	6	0.01	0.01	0.01
ttp	8	4.54	3.08	3.99

Experimental Results (extHyPre II)

- Impact of extHyPre preprocessing on bsolo

Benchmark	#	bsolo			
		w/o pre	extHypre (75,10,c)	extHypre (75,10,d)	extHypre (75,10,r)
aksoy	79	26	26	*25*	*23*
logic-synthesis	74	51	51	51	51
primes-dimacs-cnf	130	69	**71**	**70**	**70**
radar	12	6	6	6	6
routing	10	9	*8*	**10**	9
synthesis-ptl-cmos	8	6	6	6	6
testset	6	6	6	6	6
ttp	8	2	2	2	2
Total	327	175	**176**	**176**	*173*

Experimental Results (extHyPre II)

- Details on some instances of extHyPre preprocessing on bsolo:

Benchmark	Instance	bsolo			
		w/o pre	extHypre (75,10,c)	extHypre (75,10,d)	extHypre (75,10,r)
aksoy	fir07_area_delay	**16** ✔	**16** ✔	18	17
	fir09_area_opers	**34** ✔	**34** ✔	**34** ✔	36
	fir09_area_partials	**35** ✔	**35** ✔	**35** ✔	35
primes-dimacs-cnf	ii32b1	191	**191** ✔	191	**191** ✔
	ii32e4.opb	364	**364** ✔	**364** ✔	364
routing	s4-4-3-2pb	**64** ✔	66	**64** ✔	**64** ✔
	s4-4-3-3pb	68	72	**62** ✔	70

Experimental Results (extHyPre III)

- Impact of extHyPre preprocessing and Pueblo

Benchmark	#	Pueblo			
		w/o pre	extHypre (75,10,c)	extHypre (75,10,d)	extHypre (75,10,r)
aksoy	79	15	15	15	15
logic-synthesis	74	31	**32**	**32**	31
primes-dimacs-cnf	130	75	**78**	**77**	**78**
radar	12	0	0	0	0
routing	10	9	10	10	10
synthesis-ptl-cmos	8	6	6	6	6
testset	6	6	6	6	6
ttp	8	2	2	2	2
Total	327	140	**144**	**143**	**143**

Experimental Results (extHyPre III)

- Details on some instances of extHyPre preprocessing on Pueblo:

Benchmark	Instance	Pueblo			
		w/o pre	extHypre (75,10,c)	extHypre (75,10,d)	extHypre (75,10,r)
logic-synthesis	C880.a	**87** ✔	**87** ✔	**87** ✔	87
	sao2.b	25	**25** ✔	**25** ✔	**25** ✔
primes-dimacs-cnf	hanoi5	1931	**1931** ✔	**1931** ✔	**1931** ✔
	ii32c2	207	**207** ✔	**207** ✔	**207** ✔
	ii32e2	235	**235** ✔	235	**235** ✔

Experimental Results (extHyPre IV)

- Impact of extHyPre preprocessing on minisat+

Benchmark	#	minisat+			
		w/o pre	extHypre (75,10,c)	extHypre (75,10,d)	extHypre (75,10,r)
aksoy	79	24	24	**27**	**27**
logic-synthesis	74	31	*30*	*30*	*30*
primes-dimacs-cnf	130	79	**80**	**80**	79
radar	12	0	0	0	0
routing	10	10	10	10	10
synthesis-ptl-cmos	8	1	1	1	1
testset	6	4	4	4	4
ttp	8	2	2	2	2
Total	327	151	151	**154**	**153**

Experimental Results (extHyPre IV)

- Details on some instances of extHyPre preprocessing on minisat+:

Benchmark	Instance	minisat+			
		w/o pre	extHypre (75,10,c)	extHypre (75,10,d)	extHypre (75,10,r)
aksoy	fir05_area_delay	14	14	**14** ✔	**14** ✔
	matrix_5x3_5	21	21	**20** ✔	**20** ✔
	matrix_5x3_6	20	20	**20** ✔	**20** ✔
logic-synthesis	5xp1.b	**12** ✔	14	14	14
primes-dimacs-cnf	ii32d2	372	**372** ✔	**372** ✔	372

Conclusions & Future Work

- Applying SAT techniques in PBO preprocessing lead to a few improvements
- This take us to believe that further research can lead to better results
- In the future we plan to improve the efficiency of our PBO version of `extHyPre` and study different heuristics
- We also plan to incorporate other SAT techniques into PBO preprocessing