TECHNISCHE
UNIVERSITÄT
DRESDEN

Unsatisfiability Proofs in SAT Solving with Parity Reasoning

Adrián Rebola-Pardo

EMCL Workshop 2016
February 11th, 2016
Vienna, Austria

Supervisors:
Steffen Hölldobler
Tobias Philipp

CDCL-style SAT solvers

Parity reasoning and unsatisfiability proofs

CDCL-style SAT solvers

branching heuristics

clause removal

symmetry breaking

cardinality resolution

learning schemata

unsatisfiability proofs

parity reasoning

Parity reasoning and unsatisfiability proofs

CDCL-style SAT solvers

branching heuristics

clause removal

symmetry breaking

cardinality resolution

learning schemata

unsatisfiability proofs

parity reasoning

Parity reasoning and unsatisfiability proofs

CDCL-style SAT solvers

branching heuristics

clause removal

symmetry breaking

cardinality resolution

learning schemata

unsatisfiability proofs

parity reasoning

SAT solvers' architecture is complex, and bugs are hard to detect.

- false positives partial interpretations as witnesses
- false negatives unsatisfiability proofs are required

Unless *P* = *coNP*, validating unsatisfiability results is intractable.

Resolution asymmetric tautologies provide proofs for most techniques.

Heule et al. (2013, 2015), Philipp et al. (2014)

Parity reasoning and unsatisfiability proofs

CDCL-style SAT solvers

branching heuristics

clause removal

symmetry breaking

cardinality resolution

learning schemata

unsatisfiability proofs

parity reasoning

CDCL is not polynomially bound in the presence of encoded parity constraints.
Urquhart (1987), Beame et al. (2004)

Parity constraints occur naturally in cryptography.
Massacci et al. (2000)

Polynomial procedures for parity reasoning can be integrated in SAT solvers.
Soos et al. (2009), Laitinen et al. (2014)

Parity reasoning and unsatisfiability proofs

CDCL-style SAT solvers

branching heuristics

clause removal

symmetry breaking

cardinality resolution

learning schemata

unsatisfiability proofs

parity reasoning

Problem generating unsatisfiability proofs for parity reasoning techniques

Biere et al. (2006, 2015)

Parity reasoning is currently disabled when unsatisfiability proofs are required.

Outline

Unsatisfiability proofs in SAT Solving

SAT Solving

SAT problem deciding whether a given CNF formula is satisfiable

Example

$$F = \{ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)$$
$$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6) \}$$

Is F satisfiable?

CDCL-style SAT solving

- try to construct a satisfying interpretation
- learn clauses from conflicts to redirect the search
- satisfiable if a satisfying interpretation is found
- unsatisfiable if the empty clause $or(\ )$ is learned

Problem how to generate an unsatisfiability proof?

Solution record the sequence of learned clauses

Theorem *Beame et al. (2004)*

Learned clauses are linear resolvents

Asymmetric tautologies

How to check if a clause C is a linear resolvent in a CNF formula F?

Definition: unit resolvent in F

$$\text{unit}\ \frac{C \vee D \vee l \qquad C \vee \bar{l} \in F}{C \vee D}$$

Definition: asymmetric tautology in F

$$\text{taut}\ \frac{}{A_0} \qquad C_1 \in F$$

$$\text{unit}\ \frac{A_0 \qquad C_1 \in F}{A_1} \qquad C_2 \in F$$

$$\text{unit}\ \frac{A_1 \qquad C_2 \in F}{\ddots}$$

$$\text{unit}\ \frac{\ddots}{A_{n-1}} \qquad C_n \in F$$

$$\text{unit}\ \frac{A_{n-1} \qquad C_n \in F}{A_n}$$

Proposition

- Asymmetric tautologies can be checked efficiently.
- Linear resolvents (in particular, learned clauses) are asymmetric tautologies.
- Subsumed clauses are asymmetric tautologies.

Resolution asymmetric tautologies

C is a resolution asymmetric tautology in F upon l if, for every resolvent of C with a clause $D \in F$ upon l, their resolvent $C \otimes D$ is an asymmetric tautology in F.

Definition: *Rat* proof system

$F \Rightarrow_{Rat} G$ if:

- $G \subseteq F$
- $G = F \cup \{C\}$ for some asymmetric tautology C in F
- $G = F \cup \{C\}$ for some resolution asymmetric tautology C in F

A *Rat*-derivation of G from F is a chain of *Rat* inferences:

$$F = F_0 \Rightarrow_{Rat} F_1 \Rightarrow_{Rat} F_2 \Rightarrow_{Rat} \cdots \Rightarrow_{Rat} F_{n-1} \Rightarrow_{Rat} F_n = G$$

Theorem

If G is *Rat*-derivable from F and unsatisfiable, then F is unsatisfiable as well.

Parity reasoning

Parity constraints

Parity constraints even/odd number of satisfied variables

X, Y, Z	parity constraints	*expressions of the form* $par(p_1, \ldots, p_n, \top?)$	
A, B	affine formulae	*finite sets of parity constraints*	

$I \models \top$

$I \models X$ iff I satisfies an even number of elements in X

$I \models A$ iff $I \models X$ for all parity constraints $X \in A$

Example

$$X = par(p_1, p_2, p_3)$$
$$Y = par(p_2, p_4, \top)$$

$$I \models p_1 \quad I \models p_2 \quad I \not\models p_3 \quad I \models p_4$$

$$I \models par(p_1, p_2, p_3) \qquad I \not\models par(p_2, p_4, \top)$$

Direct encoding of a parity constraint

smallest CNF formula $\mathcal{D}(X)$ semantically equivalent to X

exponentially-sized on $|X|$

Parity constraints

Parity constraints even/odd number of satisfied variables

may be regarded as congruences modulo 2

X, Y, Z	parity constraints	*expressions of the form* $par(p_1, \ldots, p_n, \top?)$
A, B	affine formulae	*finite sets of parity constraints*

$I \models \top$

$I \models X$ iff I satisfies an even number of elements in X

$I \models A$ iff $I \models X$ for all parity constraints $X \in A$

Example

$$X = par(p_1, p_2, p_3) \qquad p_1 + p_2 + p_3 \approx 0$$

$$Y = par(p_2, p_4, \top) \qquad p_2 + p_4 \approx 1$$

$$I \models p_1 \quad I \models p_2 \quad I \not\models p_3 \quad I \models p_4$$

$$I \models par(p_1, p_2, p_3) \qquad I \not\models par(p_2, p_4, \top)$$

Direct encoding of a parity constraint

smallest CNF formula $\mathcal{D}(X)$ semantically equivalent to X

exponentially-sized on $|X|$

Parity reasoning for SAT solving

Different methods to integrate parity reasoning in SAT solvers:

- Dependent variable elimination *van Maaren et al. (1998)*
- Equivalence reasoning *Li (2000)*
- Gauss-Jordan elimination *Soos et al. (2009)*
- Parity constraint cutting *Soos et al. (2009)*
- Parity reasoning-based clause learning *Laitinen et al. (2014)*

Parity reasoning for SAT solving

Different methods to integrate parity reasoning in SAT solvers:

- Dependent variable elimination *van Maaren et al. (1998)*
- Equivalence reasoning *Li (2000)*
- Gauss-Jordan elimination *Soos et al. (2009)*
- Parity constraint cutting *Soos et al. (2009)*
- Parity reasoning-based clause learning *Laitinen et al. (2014)*

Parity reasoning for SAT solving

Different methods to integrate parity reasoning in SAT solvers:

- Dependent variable elimination *van Maaren et al. (1998)*
- Equivalence reasoning *Li (2000)*
- Gauss-Jordan elimination *Soos et al. (2009)*
- Parity constraint cutting *Soos et al. (2009)*
- Parity reasoning-based clause learning *Laitinen et al. (2014)*

Application to SAT solving simplify detected encodings of parity constraints

original
CNF formula

F

Parity reasoning for SAT solving

Different methods to integrate parity reasoning in SAT solvers:

- Dependent variable elimination *van Maaren et al. (1998)*
- Equivalence reasoning *Li (2000)*
- Gauss-Jordan elimination *Soos et al. (2009)*
- Parity constraint cutting *Soos et al. (2009)*
- Parity reasoning-based clause learning *Laitinen et al. (2014)*

Application to SAT solving simplify detected encodings of parity constraints

detect encoded parity constraints *original CNF formula*

$$A \xmapsto{\mathcal{D}} \mathcal{D}(A) \subseteq F' \cup \mathcal{D}(A) = F$$

Parity reasoning for SAT solving

Different **methods** to integrate parity reasoning in SAT solvers:

- Dependent variable elimination *van Maaren et al. (1998)*
- Equivalence reasoning *Li (2000)*
- **Gauss-Jordan elimination** *Soos et al. (2009)*
- Parity constraint cutting *Soos et al. (2009)*
- Parity reasoning-based clause learning *Laitinen et al. (2014)*

Application to SAT solving simplify detected encodings of parity constraints

detect encoded parity constraints

original CNF formula

$$A \xmapsto{\mathcal{D}} \mathcal{D}(A) \subseteq F' \cup \mathcal{D}(A) = F$$

simplify with Gauss-Jordan elimination

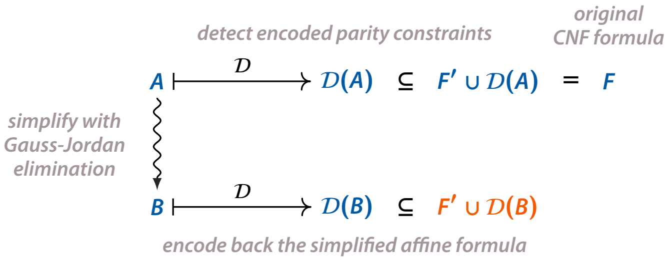
$$A \rightsquigarrow B$$

Parity reasoning for SAT solving

Different **methods** to integrate parity reasoning in SAT solvers:

- Dependent variable elimination *van Maaren et al. (1998)*
- Equivalence reasoning *Li (2000)*
- **Gauss-Jordan elimination** *Soos et al. (2009)*
- Parity constraint cutting *Soos et al. (2009)*
- Parity reasoning-based clause learning *Laitinen et al. (2014)*

Application to SAT solving simplify detected encodings of parity constraints



Proof translations

Unsatisfiability proofs for parity reasoning

detect encoded parity constraints — *original CNF formula*

$$A \overset{\mathcal{D}}{\longmapsto} \mathcal{D}(A) \subseteq F' \cup \mathcal{D}(A) = F$$

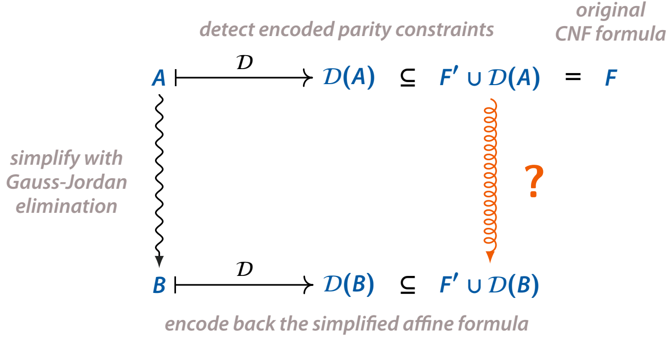
simplify with Gauss-Jordan elimination

$$B \overset{\mathcal{D}}{\longmapsto} \mathcal{D}(B) \subseteq F' \cup \mathcal{D}(B)$$

encode back the simplified affine formula

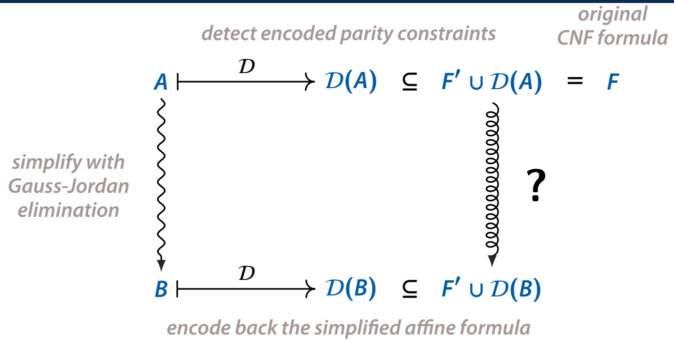
Problem **generate unsatisfiability proofs for Gauss-Jordan elimination**

Unsatisfiability proofs for parity reasoning



Problem **generate unsatisfiability proofs for Gauss-Jordan elimination**
finding a Rat-derivation of $F' \cup \mathcal{D}(B)$ *from* $F' \cup \mathcal{D}(A)$

Unsatisfiability proofs for parity reasoning



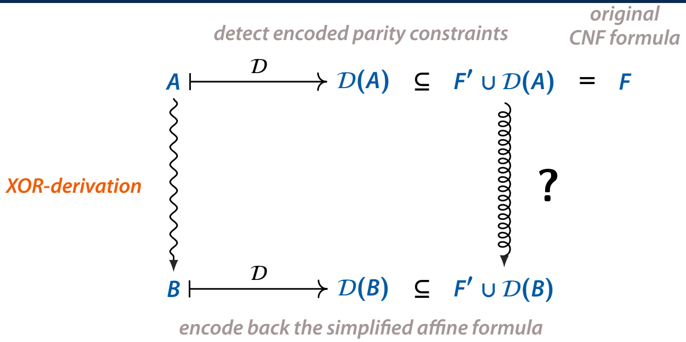
Problem generate unsatisfiability proofs for Gauss-Jordan elimination

finding a Rat-derivation of $F' \cup \mathcal{D}(B)$ *from* $F' \cup \mathcal{D}(A)$

Idea translate Gauss-Jordan elimination steps into *Rat*-derivations

- formalize Gauss-Jordan elimination within a proof system
- translate derivations through the direct encoding
- append the rest of the original CNF formula in every step

Unsatisfiability proofs for parity reasoning



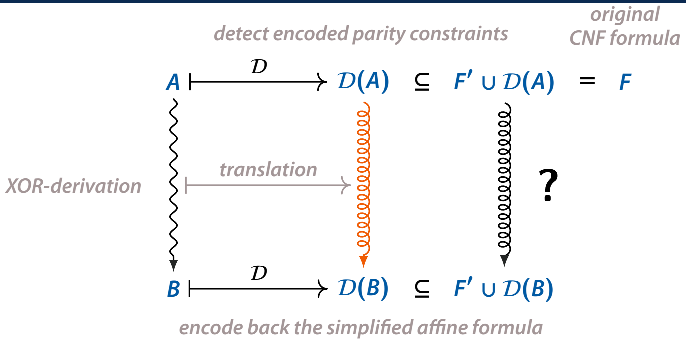
Problem generate unsatisfiability proofs for Gauss-Jordan elimination

finding a Rat-derivation of $F' \cup \mathcal{D}(B)$ *from* $F' \cup \mathcal{D}(A)$

Idea translate Gauss-Jordan elimination steps into *Rat*-derivations

- formalize Gauss-Jordan elimination within a proof system
- translate derivations through the direct encoding
- append the rest of the original CNF formula in every step

Unsatisfiability proofs for parity reasoning



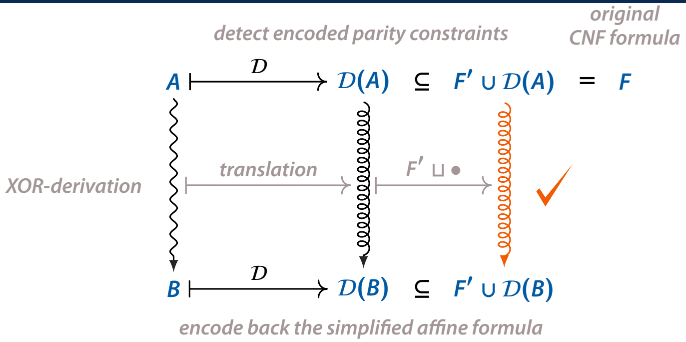
Problem generate unsatisfiability proofs for Gauss-Jordan elimination

finding a Rat-derivation of $F' \cup \mathcal{D}(B)$ *from* $F' \cup \mathcal{D}(A)$

Idea translate Gauss-Jordan elimination steps into *Rat*-derivations

- formalize Gauss-Jordan elimination within a proof system
- translate derivations through the direct encoding
- append the rest of the original CNF formula in every step

Unsatisfiability proofs for parity reasoning



Problem generate unsatisfiability proofs for Gauss-Jordan elimination

finding a Rat-derivation of $F' \cup \mathcal{D}(B)$ *from* $F' \cup \mathcal{D}(A)$

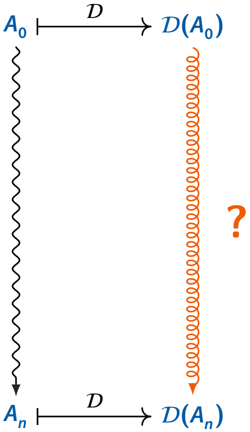
Idea translate Gauss-Jordan elimination steps into *Rat*-derivations

- formalize Gauss-Jordan elimination within a proof system
- translate derivations through the direct encoding
- append the rest of the original CNF formula in every step

Direct translations

Assume an *EXor*-derivation of A_n from A_0.

Direct translations

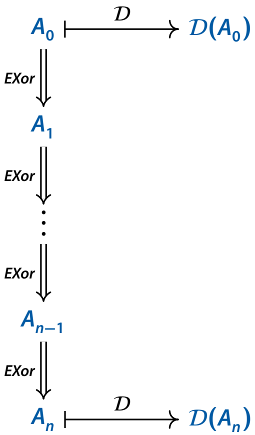


Assume an *EXor*-derivation of A_n from A_0.

Goal

translation through the direct encoding

Direct translations



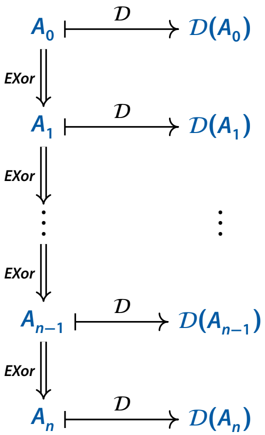
Assume an *EXor*-derivation of A_n from A_0.

Goal

translation through the direct encoding

- translate single *EXor* inferences

Direct translations



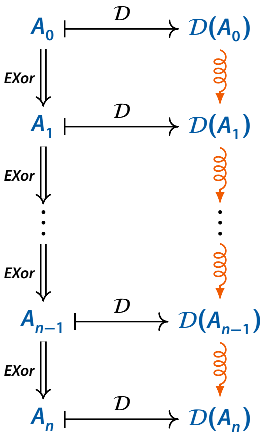
Assume an *EXor*-derivation of A_n from A_0.

Goal

translation through the direct encoding

- translate single *EXor* inferences

Direct translations



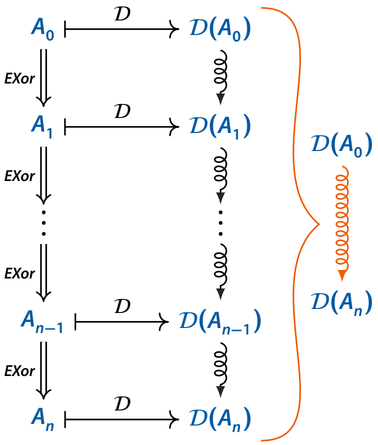
Assume an *EXor*-derivation of A_n from A_0.

Goal

translation through the direct encoding

- translate single *EXor* inferences

Direct translations



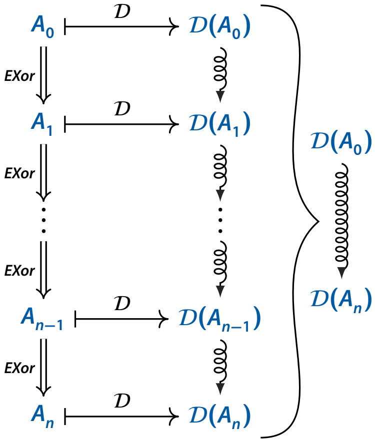
Assume an *EXor*-derivation of A_n from A_0.

Goal

translation through the direct encoding

- translate single *EXor* inferences
- concatenate translations

Direct translations



Assume an *EXor*-derivation of A_n from A_0.

Goal

translation through the direct encoding

- translate single *EXor* inferences
- concatenate translations

Parity constraint deletion

deleting clauses in the direct encoding

XOR definition introduction

clauses in the direct encoding of a XOR definition are resolution asymmetric tautologies

Parity constraint addition

explained next

Translating parity constraint addition inferences

Parity constraint addition inference $A \Rightarrow_{EXor} A \cup \{X \oplus Y\}$ **when** $X, Y \in A$

Goal **derive every clause in** $\mathcal{D}(X \oplus Y)$ **from clauses in** $\mathcal{D}(\{X, Y\})$

Translating parity constraint addition inferences

Parity constraint addition inference $A \Rightarrow_{EXor} A \cup \{X \oplus Y\}$ when $X, Y \in A$

Goal derive every clause in $\mathcal{D}(X \oplus Y)$ from clauses in $\mathcal{D}(\{X, Y\})$

$$\begin{aligned} X &= par(p_1, p_3, p_4, p_5) \\ Y &= par(p_2, p_3, p_4, p_5, \top) \\ X \oplus Y &= par(p_1, p_2, \top) \end{aligned}$$

Translating parity constraint addition inferences

Parity constraint addition inference $A \Rightarrow_{EXor} A \cup \{X \oplus Y\}$ when $X, Y \in A$

Goal derive every clause in $\mathcal{D}(X \oplus Y)$ from clauses in $\mathcal{D}(\{X, Y\})$

$$X = par(p_1, p_3, p_4, p_5)$$
$$Y = par(p_2, p_3, p_4, p_5, \top)$$
$$X \oplus Y = par(p_1, p_2, \top)$$

Consider the clause $or(p_1, p_2) \in \mathcal{D}(X \oplus Y)$.

$$or(p_1, p_2)$$

Translating parity constraint addition inferences

Parity constraint addition inference $A \Rightarrow_{EXor} A \cup \{X \oplus Y\}$ when $X, Y \in A$

Goal derive every clause in $\mathcal{D}(X \oplus Y)$ from clauses in $\mathcal{D}(\{X, Y\})$

$$X = par(p_1, p_3, p_4, p_5)$$
$$Y = par(p_2, p_3, p_4, p_5, \top)$$
$$X \oplus Y = par(p_1, p_2, \top)$$

Consider the clause $or(p_1, p_2) \in \mathcal{D}(X \oplus Y)$.

$$\otimes \frac{or(p_1, p_2, p_3) \qquad or(p_1, p_2, \overline{p_3})}{or(p_1, p_2)}$$

Translating parity constraint addition inferences

Parity constraint addition inference $A \Rightarrow_{EXor} A \cup \{X \oplus Y\}$ when $X, Y \in A$

Goal derive every clause in $\mathcal{D}(X \oplus Y)$ from clauses in $\mathcal{D}(\{X, Y\})$

$$X = par(p_1, p_3, p_4, p_5)$$
$$Y = par(p_2, p_3, p_4, p_5, \top)$$
$$X \oplus Y = par(p_1, p_2, \top)$$

Consider the clause $or(p_1, p_2) \in \mathcal{D}(X \oplus Y)$.

$$\otimes \frac{\otimes \dfrac{or(p_1, p_2, p_3, p_4) \qquad or(p_1, p_2, p_3, \overline{p_4})}{or(p_1, p_2, p_3)} \qquad or(p_1, p_2, \overline{p_3})}{or(p_1, p_2)}$$

Translating parity constraint addition inferences

Parity constraint addition inference $A \Rightarrow_{EXor} A \cup \{X \oplus Y\}$ when $X, Y \in A$

Goal derive every clause in $\mathcal{D}(X \oplus Y)$ from clauses in $\mathcal{D}(\{X, Y\})$

$$X = par(p_1, p_3, p_4, p_5)$$
$$Y = par(p_2, p_3, p_4, p_5, \top)$$
$$X \oplus Y = par(p_1, p_2, \top)$$

Consider the clause $or(p_1, p_2) \in \mathcal{D}(X \oplus Y)$.

$$\otimes\frac{\otimes\dfrac{or(p_1, p_2, p_3, p_4) \qquad or(p_1, p_2, p_3, \overline{p_4})}{or(p_1, p_2, p_3)} \qquad \dfrac{or(p_1, p_2, \overline{p_3}, p_4) \qquad or(p_1, p_2, \overline{p_3}, \overline{p_4})}{or(p_1, p_2, \overline{p_3})}\otimes}{or(p_1, p_2)}$$

Translating parity constraint addition inferences

Parity constraint addition inference $A \Rightarrow_{EXor} A \cup \{X \oplus Y\}$ when $X, Y \in A$

Goal derive every clause in $\mathcal{D}(X \oplus Y)$ from clauses in $\mathcal{D}(\{X, Y\})$

$$X = par(p_1, p_3, p_4, p_5)$$
$$Y = par(p_2, p_3, p_4, p_5, \top)$$
$$X \oplus Y = par(p_1, p_2, \top)$$

Consider the clause $or(p_1, p_2) \in \mathcal{D}(X \oplus Y)$.

$$\dfrac{\otimes\dfrac{or(p_1, p_2, p_3, p_4) \qquad or(p_1, p_2, p_3, \overline{p_4})}{or(p_1, p_2, p_3)} \qquad \dfrac{or(p_1, p_2, \overline{p_3}, p_4) \qquad or(p_1, p_2, \overline{p_3}, \overline{p_4})}{or(p_1, p_2, \overline{p_3})}\otimes}{or(p_1, p_2)}\otimes$$

Translating parity constraint addition inferences

Parity constraint addition inference $A \Rightarrow_{EXor} A \cup \{X \oplus Y\}$ when $X, Y \in A$

Goal derive every clause in $\mathcal{D}(X \oplus Y)$ from clauses in $\mathcal{D}(\{X, Y\})$

$$X = par(p_1, p_3, p_4, p_5)$$
$$Y = par(p_2, p_3, p_4, p_5, \top)$$
$$X \oplus Y = par(p_1, p_2, \top)$$

Consider the clause $or(p_1, p_2) \in \mathcal{D}(X \oplus Y)$.

$$\otimes\,\dfrac{\otimes\,\dfrac{or(p_1, p_2, p_3, p_4) \qquad or(p_1, p_2, p_3, \overline{p_4})}{or(p_1, p_2, p_3)} \qquad \dfrac{or(p_1, p_2, \overline{p_3}, p_4) \qquad or(p_1, p_2, \overline{p_3}, \overline{p_4})}{or(p_1, p_2, \overline{p_3})}\,\otimes}{or(p_1, p_2)}$$

Proposition Top-level clauses are **asymmetric tautologies** in $\mathcal{D}(\{X, Y\})$

Translating parity constraint addition inferences

Parity constraint addition inference $A \Rightarrow_{EXor} A \cup \{X \oplus Y\}$ when $X, Y \in A$

Goal derive every clause in $\mathcal{D}(X \oplus Y)$ from clauses in $\mathcal{D}(\{X, Y\})$

$$\begin{aligned} X &= par(p_1, p_3, p_4, p_5) \\ Y &= par(p_2, p_3, p_4, p_5, \top) \\ X \oplus Y &= par(p_1, p_2, \top) \end{aligned}$$

Consider the clause $or(p_1, p_2) \in \mathcal{D}(X \oplus Y)$.

$$\dfrac{\dfrac{or(p_1,p_2,p_3,p_4) \qquad or(p_1,p_2,p_3,\overline{p_4})}{or(p_1,p_2,p_3)}\otimes \qquad \dfrac{or(p_1,p_2,\overline{p_3},p_4) \qquad or(p_1,p_2,\overline{p_3},\overline{p_4})}{or(p_1,p_2,\overline{p_3})}\otimes}{or(p_1,p_2)}\otimes$$

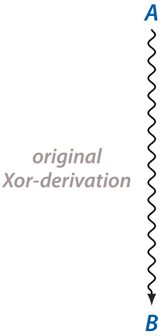
Problem

- Deriving clause E requires exponentially many clauses in $|X| + |Y|$.
- An exponential number of clauses $E \in \mathcal{D}(X \oplus Y)$ must be derived.

Solution bound the size of parity constraints involved in additions

Linear translations

Idea refine the *Xor*-derivation into another one containing bounded-size parity constraints

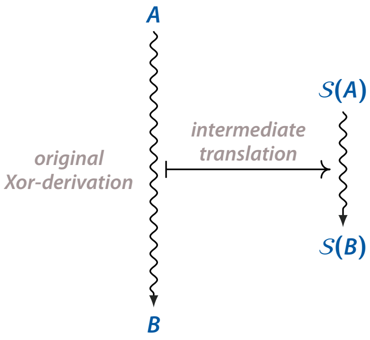


Linear translations

Idea refine the *Xor*-derivation into another one containing bounded-size parity constraints

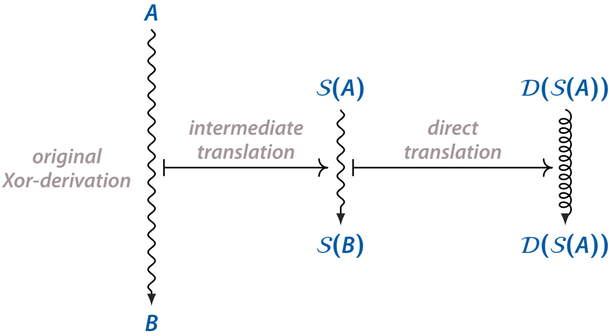
Linear translations

Idea refine the *Xor*-derivation into another one containing bounded-size parity constraints



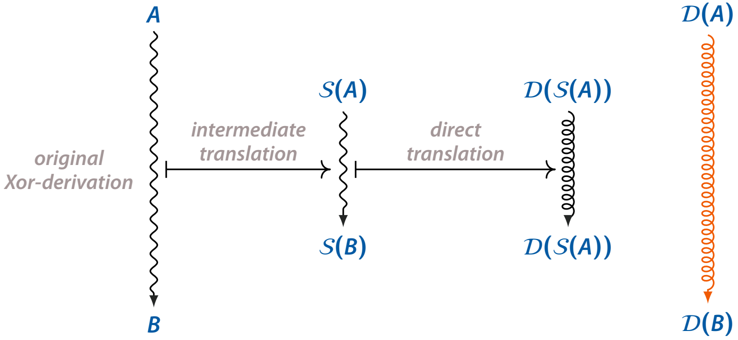
Linear translations

Idea refine the *Xor*-derivation into another one containing bounded-size parity constraints



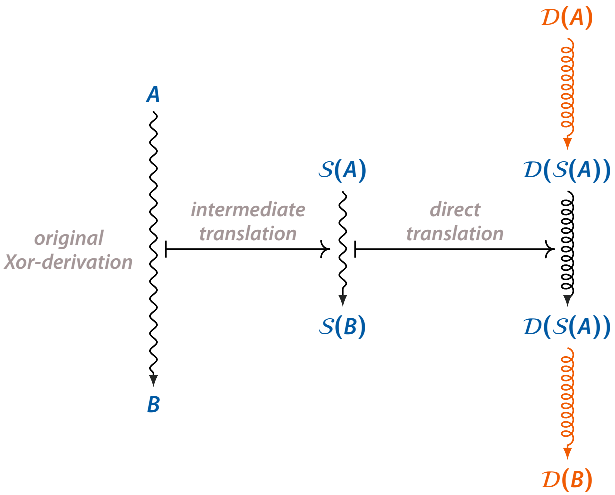
Linear translations

Idea refine the *Xor*-derivation into another one containing bounded-size parity constraints



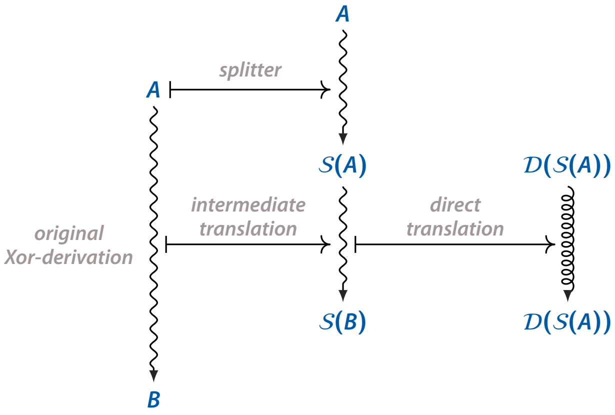
Linear translations

Idea refine the *Xor*-derivation into another one containing bounded-size parity constraints



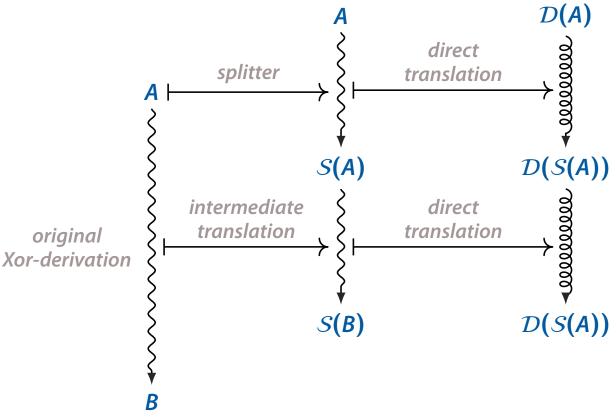
Linear translations

Idea refine the *Xor*-derivation into another one containing bounded-size parity constraints



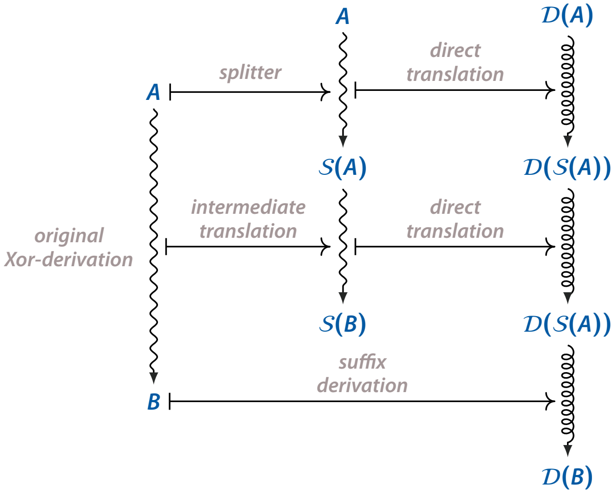
Linear translations

Idea refine the *Xor*-derivation into another one containing bounded-size parity constraints



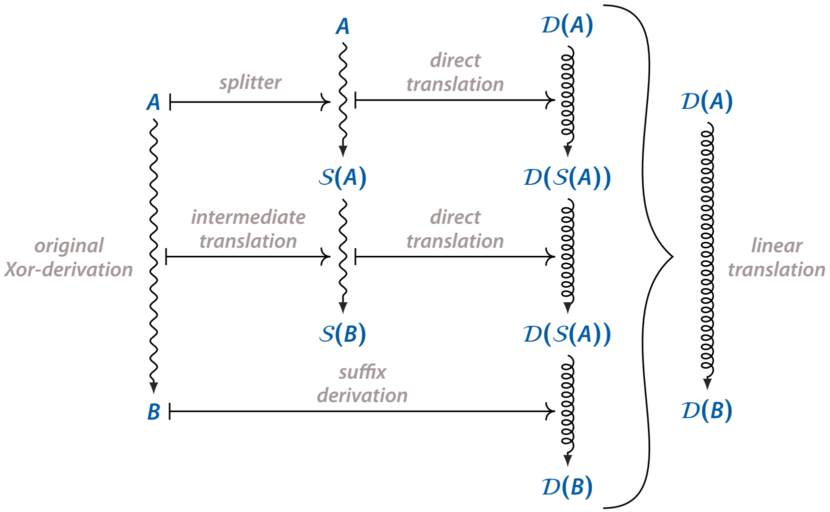
Linear translations

Idea refine the *Xor*-derivation into another one containing bounded-size parity constraints



Linear translations

Idea refine the *Xor*-derivation into another one containing bounded-size parity constraints



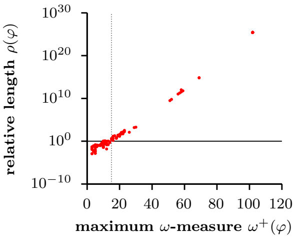
Direct vs. linear translations

Theorem

Direct translations of *Xor*-derivations are exponential in $|F|$.

Theorem

Linear translations of *Xor*-derivations are polynomial in $|F|$.



In practice, direct translations are shorter than linear translations whenever all congruence additions are performed over short parity constraints.

Length approximations are provided so that the most beneficial approach can be chosen beforehand.

Further contributions

A framework for proof systems

A generalized framework for proof systems was introduced.

- Different consequence notions are allowed:

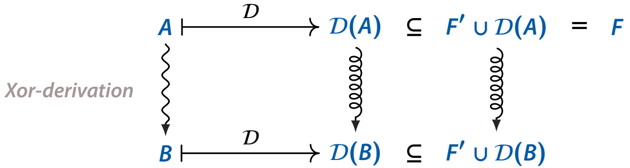
  $F \models G$ iff for every interpretation I, if $I \models F$ then $I \models G$.

  $F \models_{sat} G$ iff, whenever F is satisfiable, G is satisfiable as well.

- This allows to model non-classical proof systems, including *Rat* or *EXor*:
  - $\{or(p_1)\} \not\models \{or(\neg p_2)\}$
  - But $\{or(\neg p_2)\}$ is *Rat*-derivable from $\{or(p_1)\}$!

$$\{or(p_1)\} \Rightarrow_{Rat} \emptyset \Rightarrow_{Rat} \{or(\neg p_2)\}$$

- Criteria to guarantee correctness of derivation composition are provided.



A framework for proof systems

A generalized framework for proof systems was introduced.

- Different consequence notions are allowed:

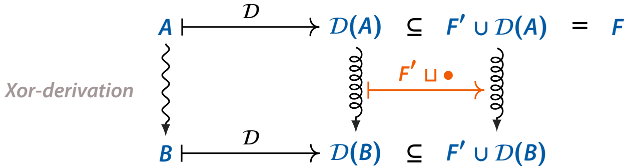
  $F \models G$ iff for every interpretation I, if $I \models F$ then $I \models G$.

  $F \models_{sat} G$ iff, whenever F is satisfiable, G is satisfiable as well.

- This allows to model non-classical proof systems, including *Rat* or *EXor*:
  - $\{or(p_1)\} \not\models \{or(\neg p_2)\}$
  - But $\{or(\neg p_2)\}$ is *Rat*-derivable from $\{or(p_1)\}$!

$$\{or(p_1)\} \Rightarrow_{Rat} \varnothing \Rightarrow_{Rat} \{or(\neg p_2)\}$$

- Criteria to guarantee correctness of derivation composition are provided.



Generalized conflict analysis

An unsatisfiability proof generation schema for generalized conflict analysis and clause learning was developed.

Theorem *Beame et al. (2004)*

If all reason clauses are in *F*, then learned clauses are linear resolvents in *F*.

Generalized conflict analysis

An unsatisfiability proof generation schema for generalized conflict analysis and clause learning was developed.

Theorem

If all reason clauses are consequences of *F*, then learned clauses are linear resolvents in the reason clauses, therefore consequences of *F*.

Generalized conflict analysis

An unsatisfiability proof generation schema for generalized conflict analysis and clause learning was developed.

Theorem

If all reason clauses are consequences of F, then learned clauses are linear resolvents in the reason clauses, therefore consequences of F.

Unsatisfiability proofs for arbitrary conflict analysis methods

generated by providing derivations of reason clauses

Interleaved parity reasoning

reason clauses are obtained Gauss-Jordan elimination

translations can be generated with our approach

Conclusions

Conclusions

- Non-classical proof systems are formalized within an unified framework.
  - unsatisfiability proof generation as derivation translations
  - integration of derivation fragments can be guaranteed
- Unsatisfiability proofs for parity reasoning-based SAT solving is attained.
  - translation of *Xor* derivations through the direct encoding
  - two translation methods: direct and linear translations
  - theoretical and empirical comparisons to choose the shorter
- Future work   generating unsatisfiability proofs for cardinality resolution

Thank you!

CDCL SAT Solving

CDCL SAT Solving

$$F = \{\ or(\neg p_1)\ ,\ or(p_1,p_3)\ ,\ or(\neg p_2,\neg p_3,\neg p_4)\ ,\ or(p_4,p_5)\ ,\ or(\neg p_3,p_4,\neg p_6)\ ,$$
$$or(\neg p_5,p_6)\ ,\ or(p_2,p_5)\ ,\ or(p_2,\neg p_5,\neg p_6)\ \}$$

CDCL-style SAT solving

- try to construct a satisfying interpretation
- learn clauses from conflicts to redirect the search
- satisfiable if a satisfying interpretation is found
- unsatisfiable if the empty clause *or*( ) is learned

CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1,p_3)\ ,\ or(\neg p_2,\neg p_3,\neg p_4)\ ,\ or(p_4,p_5)\ ,\ or(\neg p_3,p_4,\neg p_6)\ ,$
$or(\neg p_5,p_6)\ ,\ or(p_2,p_5)\ ,\ or(p_2,\neg p_5,\neg p_6)\ \}$

[ ]

initialize start with the empty partial interpretation

conflict graph:

reason clauses:

CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1,p_3)\ ,\ or(\neg p_2,\neg p_3,\neg p_4)\ ,\ or(p_4,p_5)\ ,\ or(\neg p_3,p_4,\neg p_6)\ ,$
$or(\neg p_5,p_6)\ ,\ or(p_2,p_5)\ ,\ or(p_2,\neg p_5,\neg p_6)\ \}$

[ ]

unit propagation $\neg p_1$

conflict graph:

reason clauses:

CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$
$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ]$

unit propagation $\neg p_1$

conflict graph:

reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1,p_3)\ ,\ or(\neg p_2,\neg p_3,\neg p_4)\ ,\ or(p_4,p_5)\ ,\ or(\neg p_3,p_4,\neg p_6)\ ,$

$or(\neg p_5,p_6)\ ,\ or(p_2,p_5)\ ,\ or(p_2,\neg p_5,\neg p_6)\ \}$

$[\ \neg p_1\ ]$

unit propagation $\neg p_1$

conflict graph:

reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$

$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ]$

unit propagation p_3

conflict graph:

reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

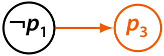
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$
$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ]$

unit propagation p_3

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

CDCL SAT Solving

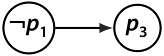
$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$

$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ]$

unit propagation p_3

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

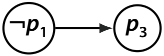
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$
$\quad or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ]$

literal decision p_2

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

CDCL SAT Solving

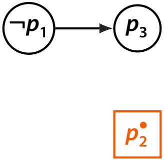
$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$

$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^{\bullet}\ ]$

literal decision p_2

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

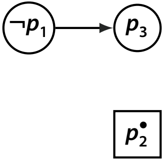
$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$

$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^\bullet\ ]$

literal decision p_2

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

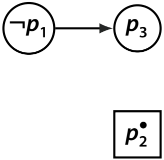
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$
$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^{\bullet}\ ]$

unit propagation $\neg p_4$

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

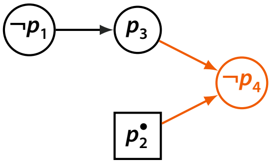
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$
$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^{\bullet}\ ,\ \neg p_4\ ]$

unit propagation $\neg p_4$

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

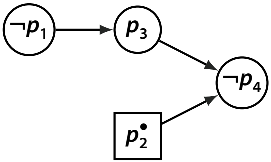
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1,p_3)\ ,\ or(\neg p_2,\neg p_3,\neg p_4)\ ,\ or(p_4,p_5)\ ,\ or(\neg p_3,p_4,\neg p_6)\ ,$
$or(\neg p_5,p_6)\ ,\ or(p_2,p_5)\ ,\ or(p_2,\neg p_5,\neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^{\bullet}\ ,\ \neg p_4\ ]$

unit propagation $\neg p_4$

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$
$\mathcal{R}(p_3) = or(p_1, p_3)$
$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

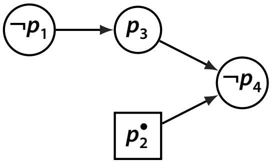
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$
$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^\bullet\ ,\ \neg p_4\ ]$

unit propagation p_5

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

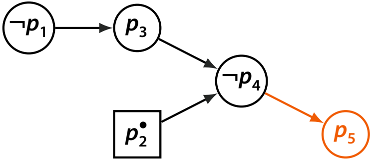
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1,p_3)\ ,\ or(\neg p_2,\neg p_3,\neg p_4)\ ,\ or(p_4,p_5)\ ,\ or(\neg p_3,p_4,\neg p_6)\ ,$
$\quad or(\neg p_5,p_6)\ ,\ or(p_2,p_5)\ ,\ or(p_2,\neg p_5,\neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^{\bullet}\ ,\ \neg p_4\ ,\ p_5\ ]$

unit propagation p_5

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

$\mathcal{R}(p_5) = or(p_4, p_5)$

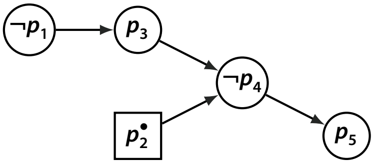
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1,p_3)\ ,\ or(\neg p_2,\neg p_3,\neg p_4)\ ,\ or(p_4,p_5)\ ,\ \mathbf{or(\neg p_3,p_4,\neg p_6)}\ ,$
$\mathbf{or(\neg p_5,p_6)}\ ,\ or(p_2,p_5)\ ,\ or(p_2,\neg p_5,\neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^\bullet\ ,\ \neg p_4\ ,\ p_5\ ]$

unit propagation p_5

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

$\mathcal{R}(p_5) = or(p_4, p_5)$

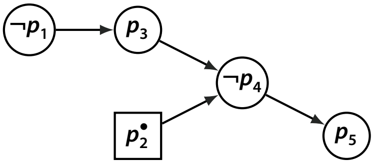
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$
$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^{\bullet}\ ,\ \neg p_4\ ,\ p_5\ ]$

unit propagation $\neg p_6$

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$
$\mathcal{R}(p_3) = or(p_1, p_3)$
$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$
$\mathcal{R}(p_5) = or(p_4, p_5)$

CDCL SAT Solving

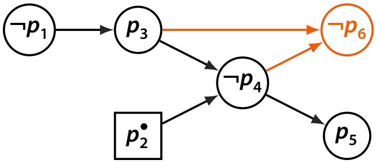
$F = \{ or(\neg p_1) , or(p_1,p_3) , or(\neg p_2,\neg p_3,\neg p_4) , or(p_4,p_5) , or(\neg p_3,p_4,\neg p_6) ,$

$or(\neg p_5,p_6) , or(p_2,p_5) , or(p_2,\neg p_5,\neg p_6) \}$

$[\neg p_1 , p_3 , p_2^{\bullet} , \neg p_4 , p_5 , \neg p_6]$

unit propagation $\neg p_6$

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

$\mathcal{R}(p_5) = or(p_4, p_5)$

$\mathcal{R}(\neg p_6) = or(\neg p_3, p_4, \neg p_6)$

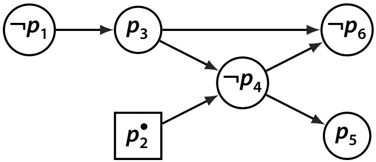
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1,p_3)\ ,\ or(\neg p_2,\neg p_3,\neg p_4)\ ,\ or(p_4,p_5)\ ,\ or(\neg p_3,p_4,\neg p_6)\ ,$
$\quad or(\neg p_5,p_6)\ ,\ or(p_2,p_5)\ ,\ or(p_2,\neg p_5,\neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^{\bullet}\ ,\ \neg p_4\ ,\ p_5\ ,\ \neg p_6\ ]$

unit propagation $\neg p_6$

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

$\mathcal{R}(p_5) = or(p_4, p_5)$

$\mathcal{R}(\neg p_6) = or(\neg p_3, p_4, \neg p_6)$

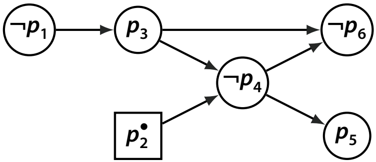
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$
$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^\bullet\ ,\ \neg p_4\ ,\ p_5\ ,\ \neg p_6\ ]$

conflict $or(\neg p_5, p_6)$

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

$\mathcal{R}(p_5) = or(p_4, p_5)$

$\mathcal{R}(\neg p_6) = or(\neg p_3, p_4, \neg p_6)$

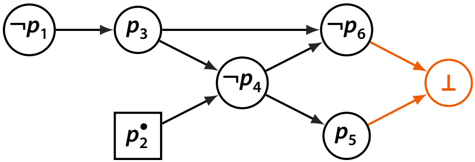
CDCL SAT Solving

$F = \{ or(\neg p_1) , or(p_1,p_3) , or(\neg p_2,\neg p_3,\neg p_4) , or(p_4,p_5) , or(\neg p_3,p_4,\neg p_6) ,$
$or(\neg p_5,p_6) , or(p_2,p_5) , or(p_2,\neg p_5,\neg p_6) \}$

$[\, \neg p_1 , p_3 , p_2^{\bullet} , \neg p_4 , p_5 , \neg p_6 \,]$

conflict $or(\neg p_5, p_6)$

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$
$\mathcal{R}(p_3) = or(p_1, p_3)$
$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$
$\mathcal{R}(p_5) = or(p_4, p_5)$
$\mathcal{R}(\neg p_6) = or(\neg p_3, p_4, \neg p_6)$
$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

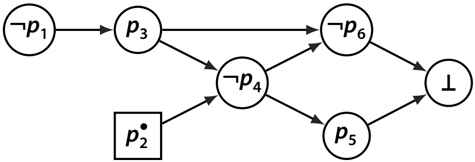
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$
$\quad or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^{\bullet}\ ,\ \neg p_4\ ,\ p_5\ ,\ \neg p_6\ ]$

conflict $or(\neg p_5, p_6)$ learn a new clause and backtrack

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

$\mathcal{R}(p_5) = or(p_4, p_5)$

$\mathcal{R}(\neg p_6) = or(\neg p_3, p_4, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

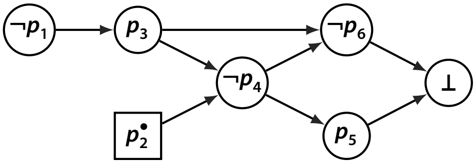
CDCL SAT Solving

$F = \{ or(\neg p_1) , or(p_1,p_3) , or(\neg p_2,\neg p_3,\neg p_4) , or(p_4,p_5) , or(\neg p_3,p_4,\neg p_6) ,$
$or(\neg p_5,p_6) , or(p_2,p_5) , or(p_2,\neg p_5,\neg p_6) \}$

$[\neg p_1 , p_3 , p_2^{\bullet} , \neg p_4 , p_5 , \neg p_6]$

conflict $or(\neg p_5, p_6)$ learn a new clause and backtrack

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

$\mathcal{R}(p_5) = or(p_4, p_5)$

$\mathcal{R}(\neg p_6) = or(\neg p_3, p_4, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

CDCL SAT Solving

$F = \{ or(\neg p_1) , or(p_1, p_3) , or(\neg p_2, \neg p_3, \neg p_4) , or(p_4, p_5) , or(\neg p_3, p_4, \neg p_6) ,$
$or(\neg p_5, p_6) , or(p_2, p_5) , or(p_2, \neg p_5, \neg p_6) \}$

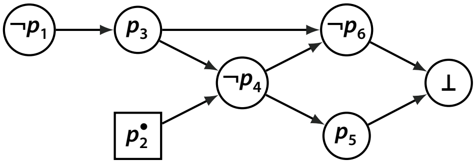
$[\neg p_1 , p_3 , p_2^{\bullet} , \neg p_4 , p_5 , \neg p_6]$

conflict $or(\neg p_5, p_6)$ **learn a new clause** and backtrack

learned clauses entailed clauses that prune the search space

obtained by linear resolution from reason clauses

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

$\mathcal{R}(p_5) = or(p_4, p_5)$

$\mathcal{R}(\neg p_6) = or(\neg p_3, p_4, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$

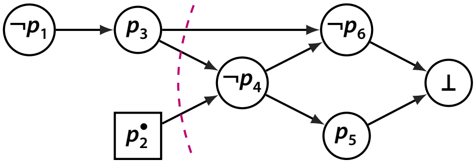
$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^\bullet\ ,\ \neg p_4\ ,\ p_5\ ,\ \neg p_6\ ]$

conflict $or(\neg p_5, p_6)$ learn a new clause and backtrack

$$or(\neg p_2, \neg p_3) = \mathcal{R}(\bot) \otimes \mathcal{R}(\neg p_6) \otimes \mathcal{R}(p_5) \otimes \mathcal{R}(\neg p_4)$$

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

$\mathcal{R}(p_5) = or(p_4, p_5)$

$\mathcal{R}(\neg p_6) = or(\neg p_3, p_4, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

CDCL SAT Solving

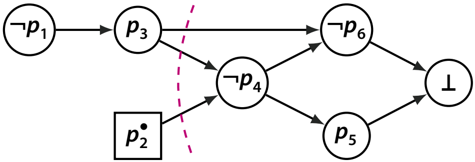
$F = \{\ or(\neg p_1)\ ,\ or(p_1,p_3)\ ,\ or(\neg p_2,\neg p_3,\neg p_4)\ ,\ or(p_4,p_5)\ ,\ or(\neg p_3,p_4,\neg p_6)\ ,$
$or(\neg p_5,p_6)\ ,\ or(p_2,p_5)\ ,\ or(p_2,\neg p_5,\neg p_6)\ ,\ or(\neg p_2,\neg p_3)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^{\bullet}\ ,\ \neg p_4\ ,\ p_5\ ,\ \neg p_6\ ]$

conflict $or(\neg p_5, p_6)$ learn a new clause and backtrack

$$or(\neg p_2, \neg p_3) = \mathcal{R}(\bot) \otimes \mathcal{R}(\neg p_6) \otimes \mathcal{R}(p_5) \otimes \mathcal{R}(\neg p_4)$$

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

$\mathcal{R}(p_5) = or(p_4, p_5)$

$\mathcal{R}(\neg p_6) = or(\neg p_3, p_4, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

CDCL SAT Solving

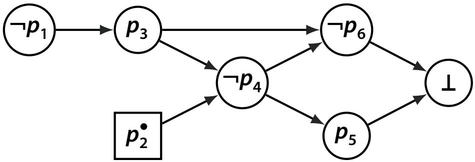
$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$

$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ ,\ or(\neg p_2, \neg p_3)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^\bullet\ ,\ \neg p_4\ ,\ p_5\ ,\ \neg p_6\ ]$

conflict $or(\neg p_5, p_6)$ learn a new clause and backtrack

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

$\mathcal{R}(p_5) = or(p_4, p_5)$

$\mathcal{R}(\neg p_6) = or(\neg p_3, p_4, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

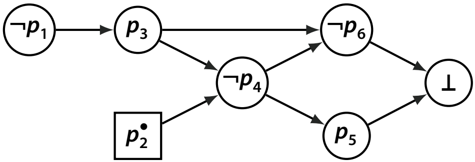
CDCL SAT Solving

$F = \{ or(\neg p_1) , or(p_1,p_3) , or(\neg p_2,\neg p_3,\neg p_4) , or(p_4,p_5) , or(\neg p_3,p_4,\neg p_6) ,$
$or(\neg p_5,p_6) , or(p_2,p_5) , or(p_2,\neg p_5,\neg p_6) , or(\neg p_2, \neg p_3) \}$

$[\neg p_1 , p_3 , p_2^\bullet , \neg p_4 , p_5 , \neg p_6]$

conflict $or(\neg p_5, p_6)$ learn a new clause and backtrack

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$
$\mathcal{R}(p_3) = or(p_1, p_3)$
$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$
$\mathcal{R}(p_5) = or(p_4, p_5)$
$\mathcal{R}(\neg p_6) = or(\neg p_3, p_4, \neg p_6)$
$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

CDCL SAT Solving

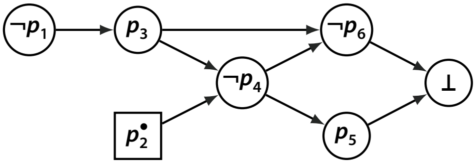
$F = \{\, or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$
$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ ,\ or(\neg p_2, \neg p_3)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ p_2^\bullet\ ,\ \neg p_4\ ,\ p_5\ ,\ \neg p_6\ ]$

conflict $or(\neg p_5, p_6)$ learn a new clause and **backtrack**

backtracking undo decisions by dropping latter literals in the interpretation

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

$\mathcal{R}(p_5) = or(p_4, p_5)$

$\mathcal{R}(\neg p_6) = or(\neg p_3, p_4, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

CDCL SAT Solving

$F = \{ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$

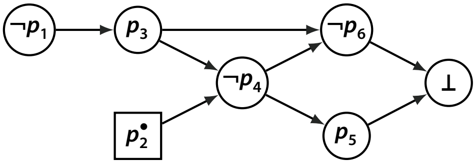
$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ ,\ or(\neg p_2, \neg p_3) \}$

$[\ \neg p_1\ ,\ p_3\ ]$

conflict $or(\neg p_5, p_6)$ **learn a new clause and backtrack**

backtracking **undo decisions by dropping latter literals in the interpretation**

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_4) = or(\neg p_2, \neg p_3, \neg p_4)$

$\mathcal{R}(p_5) = or(p_4, p_5)$

$\mathcal{R}(\neg p_6) = or(\neg p_3, p_4, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

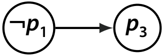
$$F = \{\, or(\neg p_1)\,,\, or(p_1, p_3)\,,\, or(\neg p_2, \neg p_3, \neg p_4)\,,\, or(p_4, p_5)\,,\, or(\neg p_3, p_4, \neg p_6)\,,$$
$$or(\neg p_5, p_6)\,,\, or(p_2, p_5)\,,\, or(p_2, \neg p_5, \neg p_6)\,,\, or(\neg p_2, \neg p_3) \,\}$$

$$[\, \neg p_1 \,,\, p_3 \,]$$

conflict $or(\neg p_5, p_6)$ **learn a new clause and backtrack**

backtracking **undo decisions by dropping latter literals in the interpretation**

conflict graph:



reason clauses:

$$\mathcal{R}(\neg p_1) = or(\neg p_1)$$
$$\mathcal{R}(p_3) = or(p_1, p_3)$$

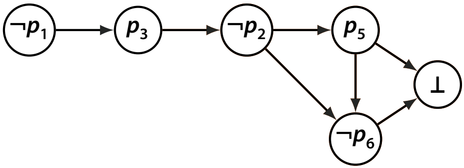
CDCL SAT Solving

$F = \{ or(\neg p_1) , or(p_1,p_3) , or(\neg p_2,\neg p_3,\neg p_4) , or(p_4,p_5) , or(\neg p_3,p_4,\neg p_6) ,$
$or(\neg p_5,p_6) , or(p_2,p_5) , or(p_2,\neg p_5,\neg p_6) , or(\neg p_2, \neg p_3) \}$

$[\neg p_1 , p_3 , \neg p_2 , p_5 , \neg p_6]$

conflict $or(\neg p_5, p_6)$ learn a new clause and backtrack

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_2) = or(\neg p_2, \neg p_3)$

$\mathcal{R}(p_5) = or(p_2, p_5)$

$\mathcal{R}(\neg p_6) = or(p_2, \neg p_5, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

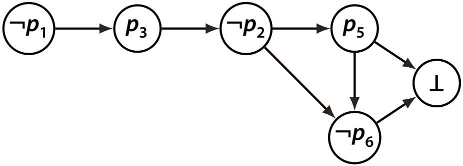
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$
$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ ,\ or(\neg p_2, \neg p_3)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ \neg p_2\ ,\ p_5\ ,\ \neg p_6\ ]$

conflict $or(\neg p_5, p_6)$ learn a new clause and backtrack

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_2) = or(\neg p_2, \neg p_3)$

$\mathcal{R}(p_5) = or(p_2, p_5)$

$\mathcal{R}(\neg p_6) = or(p_2, \neg p_5, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

CDCL SAT Solving

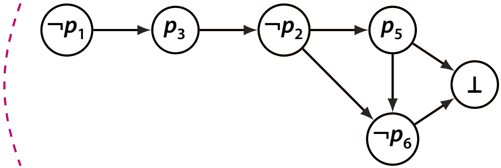
$F = \{ or(\neg p_1) , or(p_1,p_3) , or(\neg p_2,\neg p_3,\neg p_4) , or(p_4,p_5) , or(\neg p_3,p_4,\neg p_6) ,$
$or(\neg p_5,p_6) , or(p_2,p_5) , or(p_2,\neg p_5,\neg p_6) , or(\neg p_2, \neg p_3) \}$

$[\neg p_1 , p_3 , \neg p_2 , p_5 , \neg p_6]$

conflict $or(\neg p_5, p_6)$ learn a new clause and backtrack

$$or(\,) = \mathcal{R}(\bot) \otimes \mathcal{R}(\neg p_6) \otimes \mathcal{R}(p_5) \otimes \mathcal{R}(\neg p_2) \otimes \mathcal{R}(p_3) \otimes \mathcal{R}(\neg p_1)$$

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_2) = or(\neg p_2, \neg p_3)$

$\mathcal{R}(p_5) = or(p_2, p_5)$

$\mathcal{R}(\neg p_6) = or(p_2, \neg p_5, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

CDCL SAT Solving

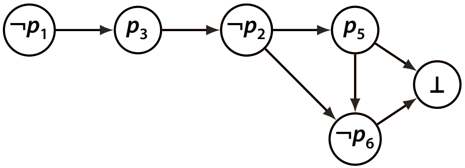
$$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$$
$$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ ,\ or(\neg p_2, \neg p_3)\ ,\ or(\ )\ \}$$

$$[\ \neg p_1\ ,\ p_3\ ,\ \neg p_2\ ,\ p_5\ ,\ \neg p_6\ ]$$

conflict $or(\neg p_5, p_6)$ **learn a new clause and backtrack**

$$or(\ ) = \mathcal{R}(\bot) \otimes \mathcal{R}(\neg p_6) \otimes \mathcal{R}(p_5) \otimes \mathcal{R}(\neg p_2) \otimes \mathcal{R}(p_3) \otimes \mathcal{R}(\neg p_1)$$

conflict graph:



reason clauses:

$$\mathcal{R}(\neg p_1) = or(\neg p_1)$$
$$\mathcal{R}(p_3) = or(p_1, p_3)$$
$$\mathcal{R}(\neg p_2) = or(\neg p_2, \neg p_3)$$
$$\mathcal{R}(p_5) = or(p_2, p_5)$$
$$\mathcal{R}(\neg p_6) = or(p_2, \neg p_5, \neg p_6)$$
$$\mathcal{R}(\bot) = or(\neg p_5, p_6)$$

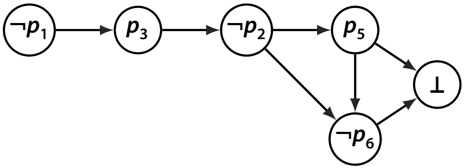
CDCL SAT Solving

$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$
$\quad or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ ,\ or(\neg p_2, \neg p_3)\ ,\ or(\,)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ \neg p_2\ ,\ p_5\ ,\ \neg p_6\ ]$

conflict $\quad or(\neg p_5, p_6)$ $\qquad$ **learn a new clause and backtrack**

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_2) = or(\neg p_2, \neg p_3)$

$\mathcal{R}(p_5) = or(p_2, p_5)$

$\mathcal{R}(\neg p_6) = or(p_2, \neg p_5, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

CDCL SAT Solving

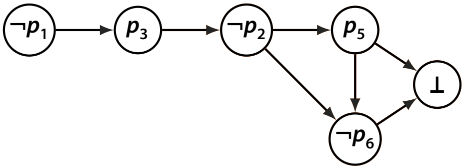
$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$

$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ ,\ or(\neg p_2, \neg p_3)\ ,\ or(\,)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ \neg p_2\ ,\ p_5\ ,\ \neg p_6\ ]$

conflict $or(\neg p_5, p_6)$ learn a new clause and backtrack

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_2) = or(\neg p_2, \neg p_3)$

$\mathcal{R}(p_5) = or(p_2, p_5)$

$\mathcal{R}(\neg p_6) = or(p_2, \neg p_5, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

CDCL SAT Solving

$F = \{ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$

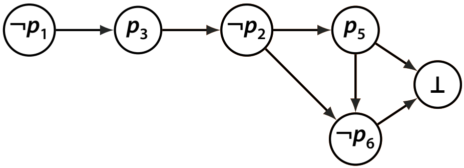
$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ ,\ or(\neg p_2, \neg p_3)\ ,\ or(\,)\ \}$

$[\ \neg p_1\ ,\ p_3\ ,\ \neg p_2\ ,\ p_5\ ,\ \neg p_6\ ]$

conflict $or(\neg p_5, p_6)$ learn a new clause and backtrack

backtracking not necessary because $or(\,)$ was learned

conflict graph:



reason clauses:

$\mathcal{R}(\neg p_1) = or(\neg p_1)$

$\mathcal{R}(p_3) = or(p_1, p_3)$

$\mathcal{R}(\neg p_2) = or(\neg p_2, \neg p_3)$

$\mathcal{R}(p_5) = or(p_2, p_5)$

$\mathcal{R}(\neg p_6) = or(p_2, \neg p_5, \neg p_6)$

$\mathcal{R}(\bot) = or(\neg p_5, p_6)$

CDCL SAT Solving

$$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,\ or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$$

F is unsatisfiable

CDCL SAT Solving

$$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,\ or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$$

F is unsatisfiable

Problem how to generate an unsatisfiability proof?

CDCL SAT Solving

$$F = \{\ or(\neg p_1)\ ,\ or(p_1, p_3)\ ,\ or(\neg p_2, \neg p_3, \neg p_4)\ ,\ or(p_4, p_5)\ ,\ or(\neg p_3, p_4, \neg p_6)\ ,$$
$$or(\neg p_5, p_6)\ ,\ or(p_2, p_5)\ ,\ or(p_2, \neg p_5, \neg p_6)\ \}$$

F is unsatisfiable

Problem how to generate an unsatisfiability proof?

Solution record the sequence of learned clauses
check whether they are linear resolvents

Linear translations

Splitting of a parity constraint

Assume a total ordering on variables.

Splitting of a parity constraint using Tseitin variables

$X = par(\quad p_1, \quad p_2, \quad \ldots, \quad p_n, \quad \top ?)$ $\qquad (p_1 < p_2 < \cdots < p_n)$

Splitting of a parity constraint

Assume a **total ordering** on variables.

Splitting of a parity constraint using Tseitin variables

$$
\begin{array}{lllllllll}
 & par(x_0, & & & & & & &) \\
 & par(x_0, p_1, x_1, & & & & & & &) \\
 & par(& x_1, p_2, x_2 & & & & & &) \\
 & & & \ddots & & & & & \\
 & par(& & & x_{n-1}, p_n, x_n & & & &) \\
\oplus & par(& & & & x_n, \mathrm{T?}) & & & \\
\hline
X = & par(& p_1, & p_2, & \ldots, & p_n, & \mathrm{T?}) & &
\end{array}
$$

$(p_1 < p_2 < \cdots < p_n)$

Splitting of a parity constraint

Assume a total ordering on variables.

Splitting of a parity constraint using Tseitin variables

$$
\begin{array}{ll}
 & par(x_0, \quad) \\
 & par(x_0, p_1, x_1, \quad) \\
 & par(\quad x_1, p_2, x_2 \quad) \\
 & \quad \ddots \\
 & par(\quad x_{n-1}, p_n, x_n \quad) \\
\oplus & par(\quad x_n, \top ?) \\
\hline
X = & par(\ p_1,\ p_2,\ \ldots,\ p_n,\ \top ?)
\end{array}
\quad \Bigg\} = \mathcal{S}(X)
$$

splitting of X

$(p_1 < p_2 < \cdots < p_n)$

Splitting of a parity constraint

Assume a **total ordering** on variables.

Splitting of a parity constraint using Tseitin variables

$$
\left.
\begin{array}{ll}
 & par(x_0, \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad) \\
 & par(x_0, p_1, x_1, \qquad\qquad\qquad\qquad\qquad\quad) \\
 & par(\qquad\quad x_1, p_2, x_2 \qquad\qquad\qquad\qquad) \\
 & \qquad\qquad\qquad\qquad\qquad \ddots \\
 & par(\qquad\qquad\qquad\qquad\qquad x_{n-1}, p_n, x_n \quad) \\
\oplus & par(\qquad\qquad\qquad\qquad\qquad\qquad\qquad x_n, \top?) \\
\hline
X = & par(\quad p_1, \quad p_2, \quad \dots, \quad p_n, \quad \top?)
\end{array}
\right\} = \mathcal{S}(X) \quad \text{splitting of } X
$$

$(p_1 < p_2 < \cdots < p_n)$

Definition: linear encoding of a parity constraint

$$\mathcal{L}(X) = \mathcal{D}(\mathcal{S}(X))$$

Proposition

$I \models X$ iff $J \models \mathcal{L}(X)$ for an interpretation J agreeing with I on the variables of X

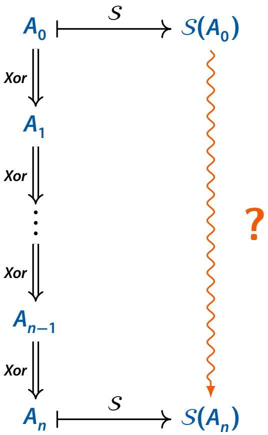
.

Intermediate translation of an Xor-derivation

Assume an *Xor*-derivation of A_n from A_0.

$$
\begin{array}{c}
A_0 \\
\Big\Downarrow{\scriptstyle Xor} \\
A_1 \\
\Big\Downarrow{\scriptstyle Xor} \\
\vdots \\
\Big\Downarrow{\scriptstyle Xor} \\
A_{n-1} \\
\Big\Downarrow{\scriptstyle Xor} \\
A_n
\end{array}
$$

Intermediate translation of an Xor-derivation

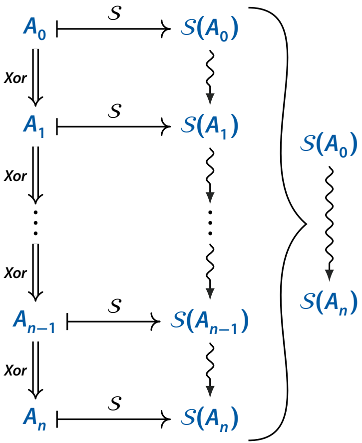


Assume an *Xor*-derivation of A_n from A_0.

Goal

translation through the splitting

Intermediate translation of an Xor-derivation



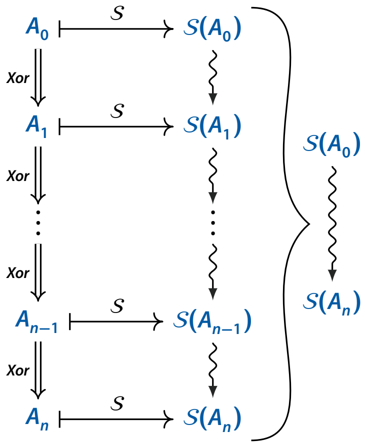
Assume an *Xor*-derivation of A_n from A_0.

Goal

translation through the splitting

- translate single *Xor* inferences
- concatenate translations

Intermediate translation of an Xor-derivation



Assume an *Xor*-derivation of A_n from A_0.

Goal

translation through the splitting

- translate single *Xor* inferences
- concatenate translations

Parity constraint deletion

deleting parity constraints in the splitting

Parity constraint addition

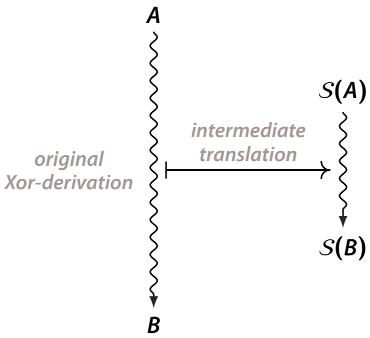
stepwise addding parity constraints in the splitting

Linear translations

Linear translations

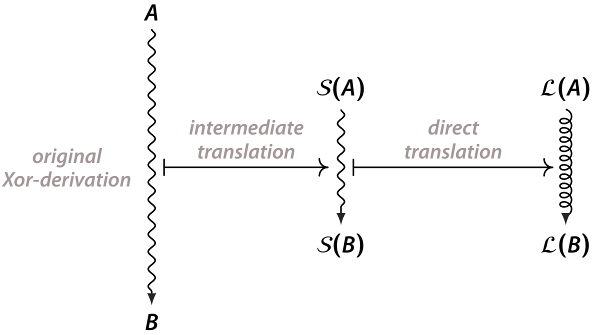
Goal generate a translation through the direct encoding of an *Xor*-derivation

Linear translations



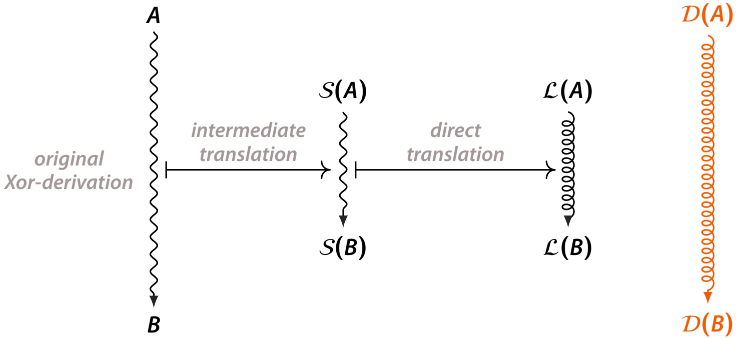
Intermediate translation an *EXor* translation through the splitting with bounded-sized parity constraints

Linear translations



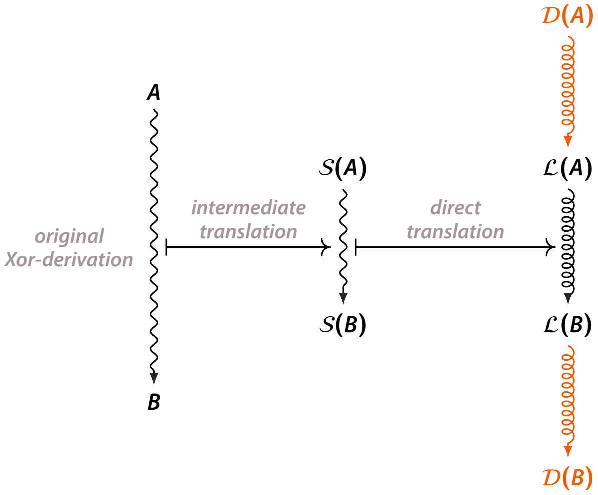
Lift obtained by direct translation from the splitting

Linear translations



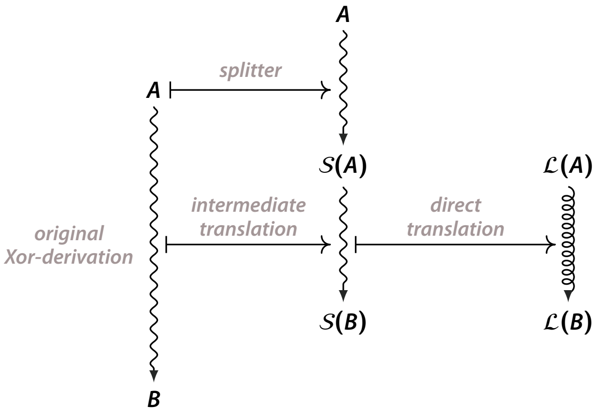
Lift obtained by direct translation from the splitting... but we need a translation through the direct encoding!

Linear translations



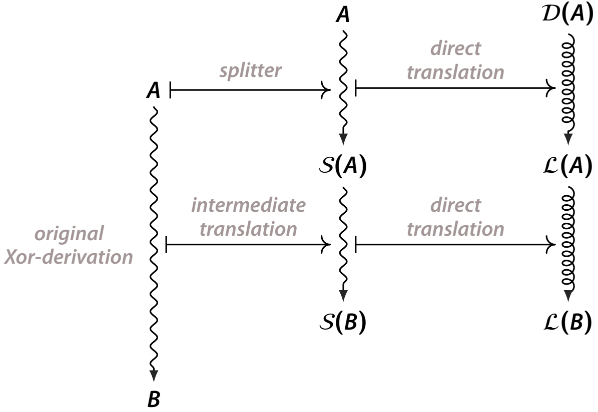
Idea generate *Rat*-derivations between direct and linear translations

Linear translations



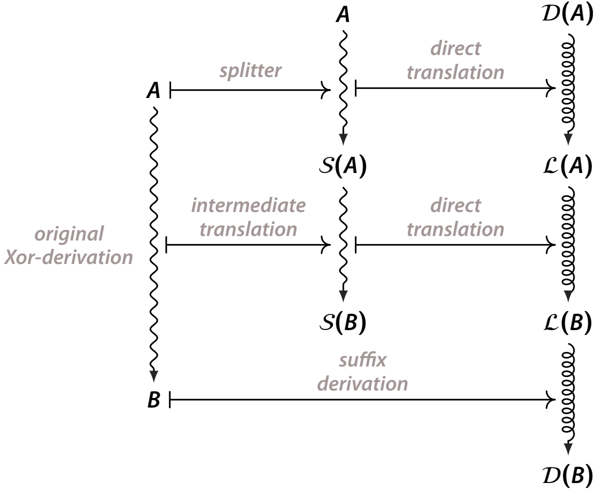
Splitter an *EXor*-derivation iteratively splits premise parity constraints

Linear translations



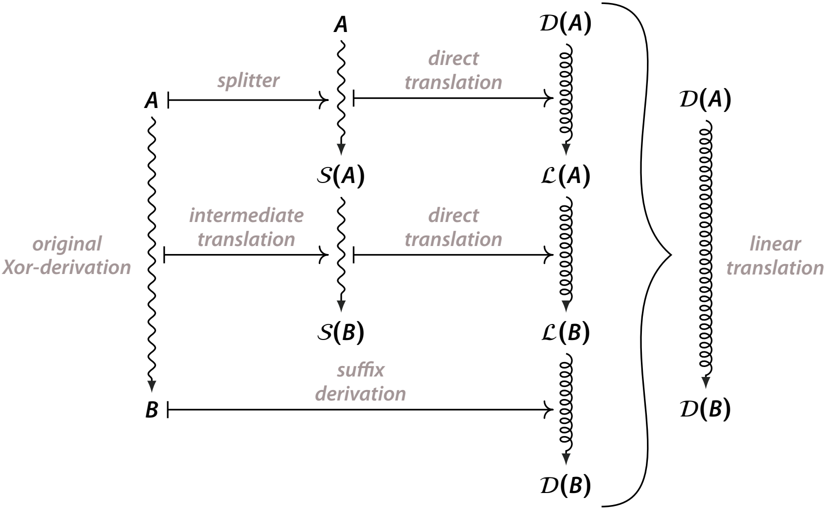
Prefix derivation **obtained by direct translation from the splitting**

Linear translations



Suffix derivation clauses in the direct encoding of a parity constraint are resolution asymmetric tautologies in their linear encoding

Linear translations



Linear translation concatenation of the prefix derivation, the lift and the suffix derivation

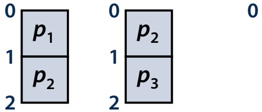
Translating parity constraint addition inferences through the splitting

Translating parity constraint addition inferences

Example $par(p_1, p_2, \top) \oplus par(p_2, p_3) = par(p_1, p_3, \top)$

Parity constraints	$par(p_1, p_2, \top)$	$par(p_2, p_3)$	$par(p_1, p_3, \top)$
Splitting matrix	$par(x_0)$	$par(y_0)$	$par(z_0)$
	$par(x_0, p_1, x_1)$	$par(y_0, p_2, y_1)$	$par(z_0, p_1, z_1)$
	$par(x_1, p_2, x_2)$	$par(y_1, p_3, y_2)$	$par(z_1, p_3, z_2)$
Independent parity constraint	$par(x_2, \top)$	$par(y_2)$	$par(z_2, \top)$

symmetric difference of sorted lists

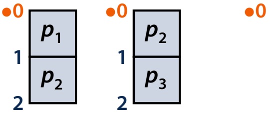


Translating parity constraint addition inferences

Example $par(p_1, p_2, \top) \oplus par(p_2, p_3) = par(p_1, p_3, \top)$

Parity constraints	$par(p_1, p_2, \top)$	$par(p_2, p_3)$	$par(p_1, p_3, \top)$
Splitting matrix	$par(x_0)$	$par(y_0)$	$par(z_0)$
	$par(x_0, p_1, x_1)$	$par(y_0, p_2, y_1)$	$par(z_0, p_1, z_1)$
	$par(x_1, p_2, x_2)$	$par(y_1, p_3, y_2)$	$par(z_1, p_3, z_2)$
Independent parity constraint	$par(x_2, \top)$	$par(y_2)$	$par(z_2, \top)$

symmetric difference of sorted lists

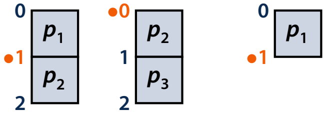


Translating parity constraint addition inferences

Example $par(p_1, p_2, \top) \oplus par(p_2, p_3) = par(p_1, p_3, \top)$

Parity constraints	$par(p_1, p_2, \top)$	$par(p_2, p_3)$	$par(p_1, p_3, \top)$
Splitting matrix	$par(x_0)$	$par(y_0)$	$par(z_0)$
	$par(x_0, p_1, x_1)$	$par(y_0, p_2, y_1)$	$par(z_0, p_1, z_1)$
	$par(x_1, p_2, x_2)$	$par(y_1, p_3, y_2)$	$par(z_1, p_3, z_2)$
Independent parity constraint	$par(x_2, \top)$	$par(y_2)$	$par(z_2, \top)$

symmetric difference of sorted lists

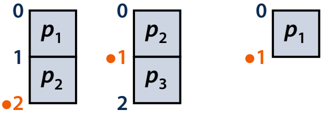


Translating parity constraint addition inferences

Example $par(p_1, p_2, \top) \oplus par(p_2, p_3) = par(p_1, p_3, \top)$

Parity constraints	$par(p_1, p_2, \top)$	$par(p_2, p_3)$	$par(p_1, p_3, \top)$
Splitting matrix	$par(x_0)$	$par(y_0)$	$par(z_0)$
	$par(x_0, p_1, x_1)$	$par(y_0, p_2, y_1)$	$par(z_0, p_1, z_1)$
	$par(x_1, p_2, x_2)$	$par(y_1, p_3, y_2)$	$par(z_1, p_3, z_2)$
Independent parity constraint	$par(x_2, \top)$	$par(y_2)$	$par(z_2, \top)$

symmetric difference of sorted lists

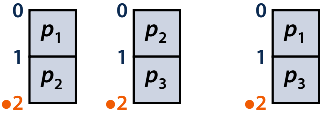


Translating parity constraint addition inferences

Example $par(p_1, p_2, \top) \oplus par(p_2, p_3) = par(p_1, p_3, \top)$

Parity constraints	$par(p_1, p_2, \top)$	$par(p_2, p_3)$	$par(p_1, p_3, \top)$
Splitting matrix	$par(x_0)$	$par(y_0)$	$par(z_0)$
	$par(x_0, p_1, x_1)$	$par(y_0, p_2, y_1)$	$par(z_0, p_1, z_1)$
	$par(x_1, p_2, x_2)$	$par(y_1, p_3, y_2)$	$par(z_1, p_3, z_2)$
Independent parity constraint	$par(x_2, \top)$	$par(y_2)$	$par(z_2, \top)$

symmetric difference of sorted lists

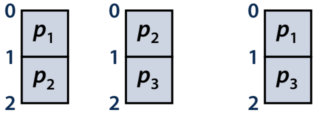


Translating parity constraint addition inferences

Example $par(p_1, p_2, \top) \oplus par(p_2, p_3) = par(p_1, p_3, \top)$

Parity constraints	$par(p_1, p_2, \top)$	$par(p_2, p_3)$	$par(p_1, p_3, \top)$
Splitting matrix	$par(x_0)$	$par(y_0)$	$par(z_0)$
	$par(x_0, p_1, x_1)$	$par(y_0, p_2, y_1)$	$par(z_0, p_1, z_1)$
	$par(x_1, p_2, x_2)$	$par(y_1, p_3, y_2)$	$par(z_1, p_3, z_2)$
Independent parity constraint	$par(x_2, \top)$	$par(y_2)$	$par(z_2, \top)$

symmetric difference of sorted lists

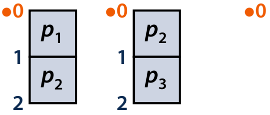


Translating parity constraint addition inferences

Example $par(p_1, p_2, \top) \oplus par(p_2, p_3) = par(p_1, p_3, \top)$

Parity constraints	$par(p_1, p_2, \top)$	$par(p_2, p_3)$	$par(p_1, p_3, \top)$
Splitting matrix	$par(x_0)$	$par(y_0)$	$par(z_0)$
	$par(x_0, p_1, x_1)$	$par(y_0, p_2, y_1)$	$par(z_0, p_1, z_1)$
	$par(x_1, p_2, x_2)$	$par(y_1, p_3, y_2)$	$par(z_1, p_3, z_2)$
Independent parity constraint	$par(x_2, \top)$	$par(y_2)$	$par(z_2, \top)$

symmetric difference of sorted lists



counter parity constraint $par(x_0, y_0, z_0)$

$$\oplus\dfrac{par(x_0) \qquad par(y_0)}{par(x_0, y_0)} \qquad \dfrac{}{par(z_0)}\text{def}$$

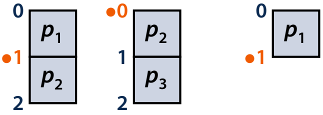
$$\oplus\dfrac{par(x_0, y_0) \qquad par(z_0)}{par(x_0, y_0, z_0)}$$

Translating parity constraint addition inferences

Example $par(p_1, p_2, \top) \oplus par(p_2, p_3) = par(p_1, p_3, \top)$

Parity constraints	$par(p_1, p_2, \top)$	$par(p_2, p_3)$	$par(p_1, p_3, \top)$
Splitting matrix	$par(x_0)$	$par(y_0)$	$par(z_0)$
	$par(x_0, p_1, x_1)$	$par(y_0, p_2, y_1)$	$par(z_0, p_1, z_1)$
	$par(x_1, p_2, x_2)$	$par(y_1, p_3, y_2)$	$par(z_1, p_3, z_2)$
Independent parity constraint	$par(x_2, \top)$	$par(y_2)$	$par(z_2, \top)$

symmetric difference of sorted lists



counter parity constraint $par(x_1, y_0, z_1)$

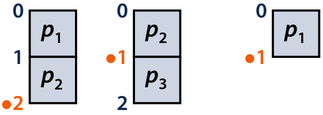
$$\oplus\dfrac{\oplus\dfrac{par(x_0, y_0, z_0) \qquad par(x_0, p_1, x_1)}{par(x_1, y_0, z_0, p_1)} \qquad \dfrac{}{par(z_0, p_1, z_1)}\,\text{def}}{par(x_1, y_0, z_1)}$$

Translating parity constraint addition inferences

Example $par(p_1, p_2, \top) \oplus par(p_2, p_3) = par(p_1, p_3, \top)$

Parity constraints	$par(p_1, p_2, \top)$	$par(p_2, p_3)$	$par(p_1, p_3, \top)$
Splitting matrix	$par(x_0)$	$par(y_0)$	$par(z_0)$
	$par(x_0, p_1, x_1)$	$par(y_0, p_2, y_1)$	$par(z_0, p_1, z_1)$
	$par(x_1, p_2, x_2)$	$par(y_1, p_3, y_2)$	$par(z_1, p_3, z_2)$
Independent parity constraint	$par(x_2, \top)$	$par(y_2)$	$par(z_2, \top)$

symmetric difference of sorted lists



counter parity constraint $par(x_2, y_1, z_1)$

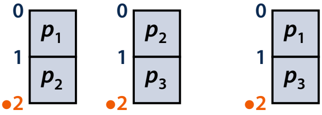
$$\dfrac{\oplus\dfrac{par(x_1, y_0, z_1) \qquad par(x_1, p_2, x_2)}{par(x_2, y_0, z_1, p_2)} \qquad par(y_0, p_2, y_1)}{par(x_2, y_1, z_1)}\ \oplus$$

Translating parity constraint addition inferences

Example $par(p_1, p_2, \top) \oplus par(p_2, p_3) = par(p_1, p_3, \top)$

Parity constraints	$par(p_1, p_2, \top)$	$par(p_2, p_3)$	$par(p_1, p_3, \top)$
Splitting matrix	$par(x_0)$	$par(y_0)$	$par(z_0)$
	$par(x_0, p_1, x_1)$	$par(y_0, p_2, y_1)$	$par(z_0, p_1, z_1)$
	$par(x_1, p_2, x_2)$	$par(y_1, p_3, y_2)$	$par(z_1, p_3, z_2)$
Independent parity constraint	$par(x_2, \top)$	$par(y_2)$	$par(z_2, \top)$

symmetric difference of sorted lists



counter parity constraint $par(x_2, y_2, z_2)$

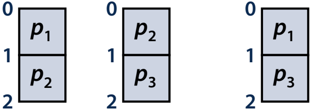
$$\oplus\dfrac{\oplus\dfrac{par(x_2, y_1, z_1) \qquad par(y_1, p_3, y_2)}{par(x_2, y_2, z_1, p_3)} \qquad \dfrac{}{par(z_1, p_3, z_2)}\,\text{def}}{par(x_2, y_2, z_2)}$$

Translating parity constraint addition inferences

Example $par(p_1, p_2, \top) \oplus par(p_2, p_3) = par(p_1, p_3, \top)$

Parity constraints	$par(p_1, p_2, \top)$	$par(p_2, p_3)$	$par(p_1, p_3, \top)$
Splitting matrix	$par(x_0)$	$par(y_0)$	$par(z_0)$
	$par(x_0, p_1, x_1)$	$par(y_0, p_2, y_1)$	$par(z_0, p_1, z_1)$
	$par(x_1, p_2, x_2)$	$par(y_1, p_3, y_2)$	$par(z_1, p_3, z_2)$
Independent parity constraint	$par(x_2, \top)$	$par(y_2)$	$par(z_2, \top)$

symmetric difference of sorted lists



independent parity constraint $par(z_2, \top)$

$$\oplus\dfrac{\oplus\dfrac{par(x_2, y_2, z_2) \qquad par(x_2, \top)}{par(y_2, z_2, \top)} \qquad par(y_2)}{par(z_2, \top)}$$