Efficient SAT in the Alt-Ergo SMT solver

after the work of the Alt-Ergo team

July 2021

1. Motivations
2. SAT and SMT solvers
3. Albin Coquereau thesis (2018)
4. Efficient SAT in Alt-Ergo SMT
5. Conclusion

1) Motivations

Motivation = software correctness :

- tests are widely used but can't certify that there is no bug
- proofs of program (deductive verification) with proof assistant don't scale

→ automatic provers

Automatic provers :

- what kind of anwser ?
- counterexamples ?
- specialized vs generalist provers ?
- how much time to get the answer ?
- ...

Here we will focus on the algorithmic efficiency.

2) SAT and SMT solvers

SAT : satisfiability of propositional/boolean formula

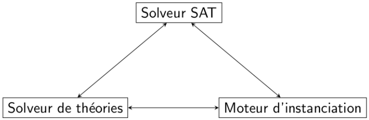
$$A \vee B \Rightarrow C \Leftrightarrow \ (A \wedge B) \vee (B \wedge C)$$

Theory :

- **2-SAT** (SAT with clauses of size 2 in CNF) is solvable in P time
- **3-SAT** (SAT with clauses of size 3 in CNF) NP-complete
- validity of F (tautology) = unsatisfiability of ¬ F (can't find interpretation such that true)

SMT : each literal is a predicate expressed in a specific theory and possibly with quantification, equality, ...

$$(x > 4) \wedge (f(x) < 2.0) \wedge (\forall i : \mathbb{Z}.f(i) = 0)$$



Conjonctive normal form (CNF) :

- boolean formulas are composed of the $\wedge$ of **clauses** and clauses are the $\vee$ of **literals** and literals are a variable or its negation.

$\rightarrow$ $\neg(A \rightarrow B) \vee (C \rightarrow A) \rightsquigarrow (A \vee \neg C) \wedge (\neg B \vee \neg C \vee A)$

Negative normal form (NNF) :

- boolean formulas are composed with $\wedge$, $\vee$ and negation applied to one variable only.

$\rightarrow$ A $\wedge(B \vee \neg C) = (A \wedge B) \vee (A \wedge \neg C)$

$\rightarrow$ not unique !

Unit clauses and Boolean Constrain Propagation (BCP) :

- assignment makes every literal in the clause unsatisfied but leaves a single literal undecided : last one has to be true for the clause to be true

$\rightarrow$ $(\neg A \vee \neg B \vee C) \wedge (\neg C \vee D)$ with A and B assigned leads to a constrain propagated to C and then propagated to D

SAT History :

- 1960 David Putman
- 1962 David Logemann Loveland (DPLL)

$\Rightarrow$ chronological backtracking without learning

- 1967 Tseitin algo to put in CNF (conjonctive normal form) (linear in the number of clauses instead of exponential) $\rightsquigarrow$ $(\ldots \vee \ldots \vee \ldots) \wedge (\ldots \vee \ldots \vee \ldots) \wedge \ldots$

$\rightarrow$ faster boolean constrain propagation (BCP)

- 1996 conflict driven clause learning (CDCL) $\Rightarrow$ non-chronological backjumping + learning
- 2001 optimized BCP

SMT Decision procedure = combination of solvers for differents theories :

- 1980 : **Shostak** algorithm : deciding combination of theories (original paper is wrong, no proof of termination, corrected in 2003) conjonction of equalities with uninterpreted symbols

$\rightarrow$ Alt-Ergo is one of the last to use it

- 1979 : **Nelsen-Oppen** algorithm

Difficulties with SMT solvers :

- combination of decision procedures
- deal with quantification efficiently
- polymorphism
- AC (associative, commutative) symbols

Shostak theories: **canonizer** returns normal form and **solver** take an equality and returns a **substitution** Examples :

- linear integers arithmetic
- pairs, records
- fragment of bitvectors

Algorithm : take each equality $u = v$:

- $u' = \mathrm{canon}(u)$, $v' = \mathrm{canon}(v)$
- $\sigma = \mathrm{solve}(u', v')$
- apply σ to representants of equivalence class
- how to combine canon / solve / sigma such that the algorithm is complete and terminating ?

Some SMT solvers commonly used in Why3 :

- 2002 : CVC
- 2003 : SMT-lib
- 2006 : CVC3, Alt-Ergo (Inria, then 2013 OcamlPro)
- 2007 : Z3
- 2012 : CVC4
- 2017 : Alt-Ergo with floats
- 2021 : CVC5

Alt-Ergo :

- generalist (= not specialized for some specific theory)
- motivations : Why3 (first-order + polymorphism), its input syntax is common with the one used in Why3

History of Alt-Ergo :

- 10/2006 : naive SAT, linear arithmetic
- 02/2007 : polymorphism
- 07/2008 : backjumping
- 07/2009 : AC symbols
- 05/2010 : tableaux method, non-linear arithmetic
- 04/2011 : enum types, graphical interface
- 12/2011 : record types
- 01/2013 : model productions
- 09/2013 : OcamlPro
- 12/2014 : plugin architecture
- 2015-2017 : optimizations
- ...
- 2018 : Albin Coquereau thesis

3) Albin Coquereau thesis (2018)

2018 : Albin Coquereau thesis :

Chapter 3 :

- study of a naive coupling of SAT-CDCL with Alt-Ergo
- Ocaml vs C++ => garbage collector responsible for cache misses
- new SAT is efficient but the new SAT-CDCL/SMT coupling less efficient than the previous SAT-Tableaux/SMT coupling in historical Alt-Ergo

Chapter 4 :

- efficient CDCL(T) in Alt-Ergo

Chapter 5 :

- lib-smt2 syntax support
- syntax extension to polymorphism

Chapter 6 :

- SMT 2018 competition results

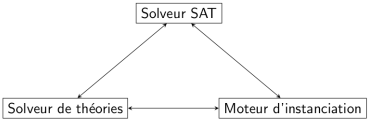
4) Efficient SAT in Alt-Ergo SMT

Four algorithms :

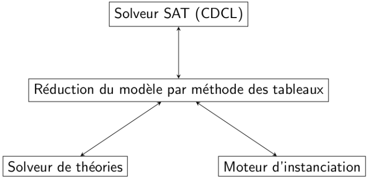
- CDCL
- Tableaux (historique)
- Tableaux assisted by CDCL

$\rightarrow$ CDCL + pertinence calculus Tableaux-method

From this :



to this :

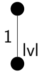


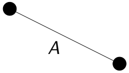
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$

Algorithme DPLL (1962)

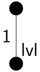
$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$

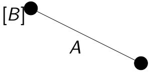




Algorithme DPLL (1962)

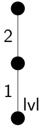
$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$

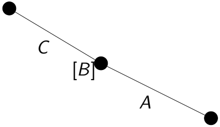




Algorithme DPLL (1962)

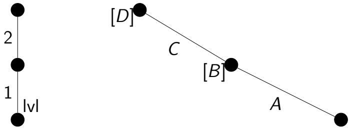
$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$





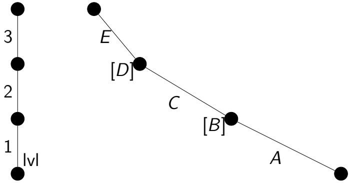
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \lor B) \\
\land\ & (\neg C \lor D) \\
\land\ & (\neg E \lor \neg F) \\
\land\ & (F \lor \neg E \lor \neg B) \\
\land\ & (E \lor G) \\
\land\ & (E \lor \neg G \lor \neg B)
\end{aligned}
$$



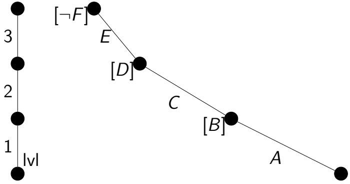
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$



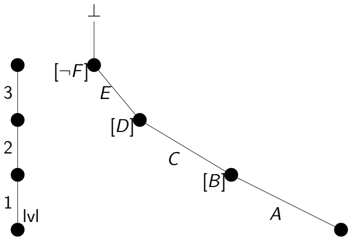
Algorithme DPLL (1962)

$$
\begin{aligned}
&(\neg A \vee B) \\
\wedge\ &(\neg C \vee D) \\
\wedge\ &(\neg E \vee \neg F) \\
\wedge\ &(F \vee \neg E \vee \neg B) \\
\wedge\ &(E \vee G) \\
\wedge\ &(E \vee \neg G \vee \neg B)
\end{aligned}
$$



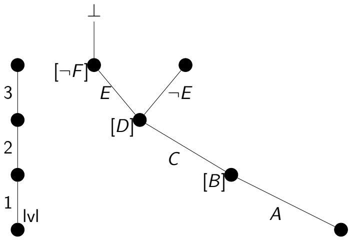
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$



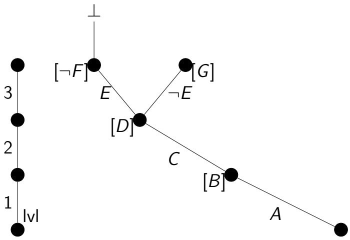
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$



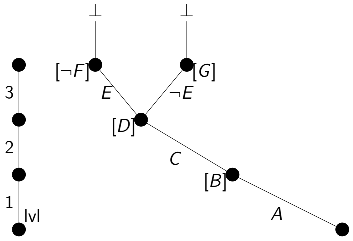
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$



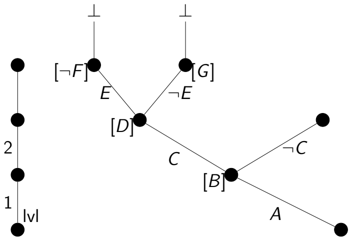
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & {\color{red}(E \vee \neg G \vee \neg B)}
\end{aligned}
$$



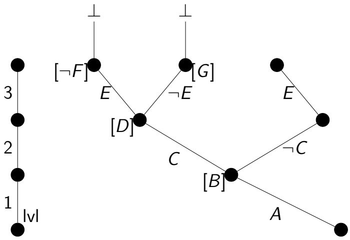
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$



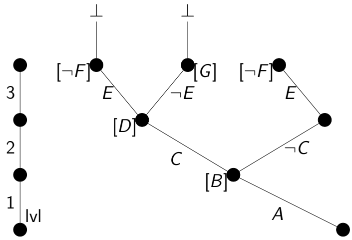
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$



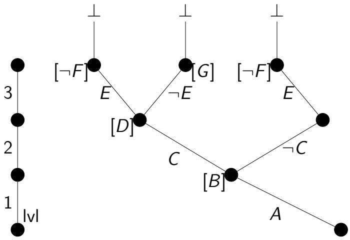
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$



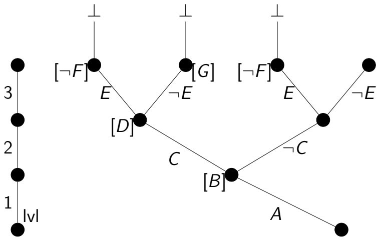
Algorithme DPLL (1962)

$$
\begin{aligned}
&(\neg A \vee B)\\
\wedge\ &(\neg C \vee D)\\
\wedge\ &(\neg E \vee \neg F)\\
\wedge\ &(F \vee \neg E \vee \neg B)\\
\wedge\ &(E \vee G)\\
\wedge\ &(E \vee \neg G \vee \neg B)
\end{aligned}
$$



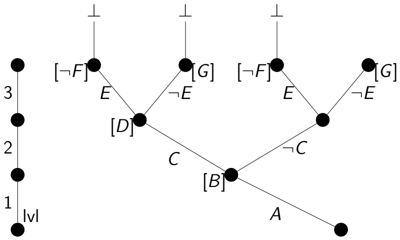
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$



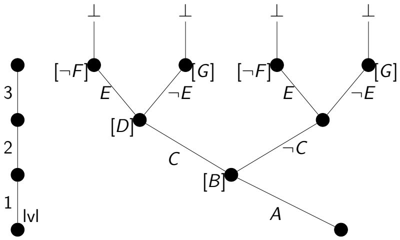
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \lor B) \\
\land\ & (\neg C \lor D) \\
\land\ & (\neg E \lor \neg F) \\
\land\ & (F \lor \neg E \lor \neg B) \\
\land\ & (E \lor G) \\
\land\ & (E \lor \neg G \lor \neg B)
\end{aligned}
$$



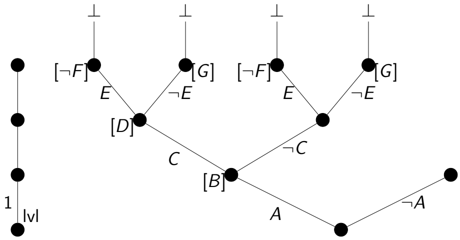
Algorithme DPLL (1962)

$$
\begin{aligned}
&(\neg A \lor B) \\
\land\ &(\neg C \lor D) \\
\land\ &(\neg E \lor \neg F) \\
\land\ &(F \lor \neg E \lor \neg B) \\
\land\ &(E \lor G) \\
\land\ &(E \lor \neg G \lor \neg B)
\end{aligned}
$$



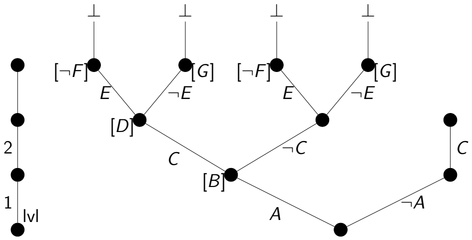
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$



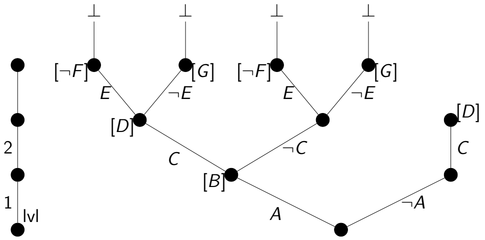
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$



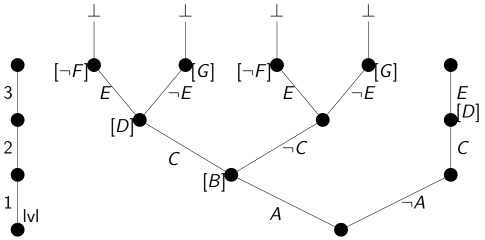
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$



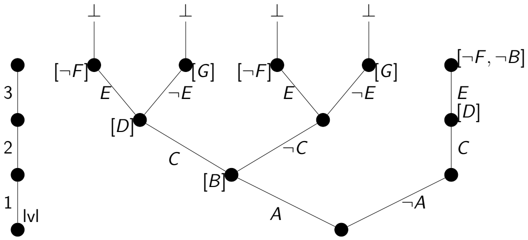
Algorithme DPLL (1962)

$$
\begin{aligned}
&(\neg A \vee B) \\
\wedge\ &(\neg C \vee D) \\
\wedge\ &(\neg E \vee \neg F) \\
\wedge\ &(F \vee \neg E \vee \neg B) \\
\wedge\ &(E \vee G) \\
\wedge\ &(E \vee \neg G \vee \neg B)
\end{aligned}
$$



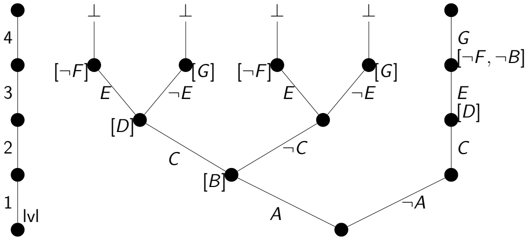
Algorithme DPLL (1962)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B)
\end{aligned}
$$



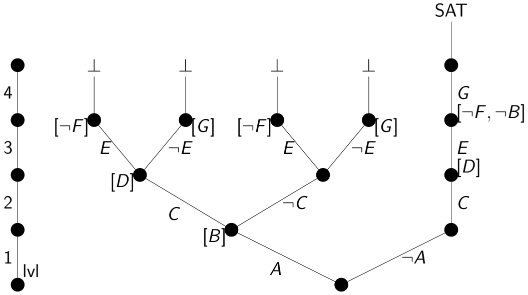
Algorithme DPLL (1962)

$$
\begin{aligned}
&(\neg A \lor B)\\
\land\ &(\neg C \lor D)\\
\land\ &(\neg E \lor \neg F)\\
\land\ &(F \lor \neg E \lor \neg B)\\
\land\ &(E \lor G)\\
\land\ &(E \lor \neg G \lor \neg B)
\end{aligned}
$$



Algorithme DPLL (1962)

$$
\begin{aligned}
&(\neg A \vee B)\\
\wedge\ &(\neg C \vee D)\\
\wedge\ &(\neg E \vee \neg F)\\
\wedge\ &(F \vee \neg E \vee \neg B)\\
\wedge\ &(E \vee G)\\
\wedge\ &(E \vee \neg G \vee \neg B)
\end{aligned}
$$



```
Input: Γ : CNF, Δ: Boolean Model
Output: Satisfiability status
lvl ← 0
while true do
    (Δ, Conflict) ← BCP(Γ, Δ)
    if Conflict ≠ ∅ then
        if lvl = 0 then
            return UNSAT
        else
            lvl ← lvl − 1
            Δ ← backtrack(Γ, Δ)
    else if all variables are assigned in Δ then
        return SAT
    else
        L ← choose(Γ, Δ)
        lvl ← lvl + 1
        Δ ← L :: Δ
```

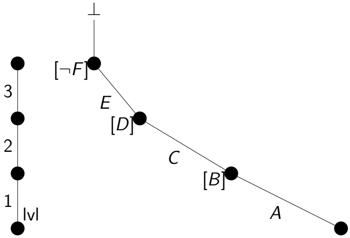
- **when conflict, backtracks to the last non-BCP decision**

→ doesn't learn anything but that there is a conflict

→ backtrack only one level

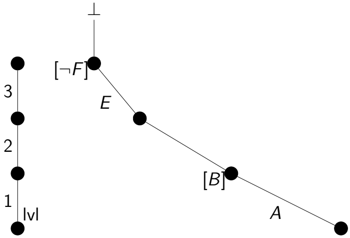
Algorithme CDCL (1996)

$$\begin{aligned}
&(\neg A \vee B)\\
\wedge\, &(\neg C \vee D)\\
\wedge\, &(\neg E \vee \neg F)\\
\wedge &(F \vee \neg E \vee \neg B)\\
\wedge &(E \vee G)\\
\wedge &(E \vee \neg G \vee \neg B)
\end{aligned}$$



Algorithme CDCL (1996)

$$
\begin{aligned}
&(\neg A \vee B)\\
\wedge\,&(\neg C \vee D)\\
\wedge\,&(\neg E \vee \neg F)\\
\wedge\,&(F \vee \neg E \vee \neg B)\\
\wedge\,&(E \vee G)\\
\wedge\,&(E \vee \neg G \vee \neg B)
\end{aligned}
$$



Logical resolution :

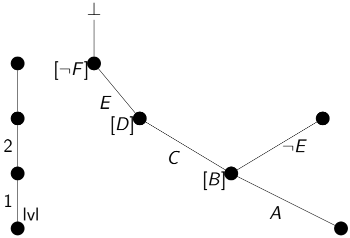
$$(P \vee Q) \wedge (\neg P \vee R) = (Q \vee R)$$

Adding a new learned clause = the logical resolution with the conflict :

$$(\neg A \wedge B) \vee (\neg E \wedge \neg F) \vee (F \wedge \neg E \wedge \neg B) = (\neg A \wedge B) \vee (\neg E \wedge \neg B) = (\neg A \wedge \neg E)$$

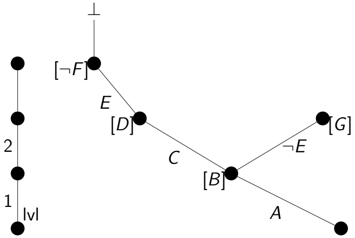
Algorithme CDCL (1996)

$$
\begin{aligned}
& (\neg A \lor B) \\
\land\, & (\neg C \lor D) \\
\land\, & (\neg E \lor \neg F) \\
\land & (F \lor \neg E \lor \neg B) \\
\land & (E \lor G) \\
\land & (E \lor \neg G \lor \neg B) \\
\land & (\neg A \lor \neg E)
\end{aligned}
$$



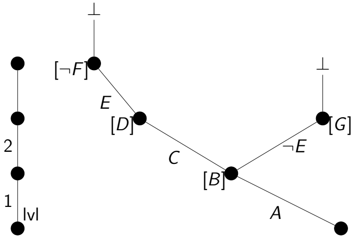
Algorithme CDCL (1996)

$$
\begin{aligned}
&(\neg A \vee B) \\
\wedge\, &(\neg C \vee D) \\
\wedge\, &(\neg E \vee \neg F) \\
\wedge\, &(F \vee \neg E \vee \neg B) \\
\wedge\, &(E \vee G) \\
\wedge\, &(E \vee \neg G \vee \neg B) \\
\wedge\, &(\neg A \vee \neg E)
\end{aligned}
$$



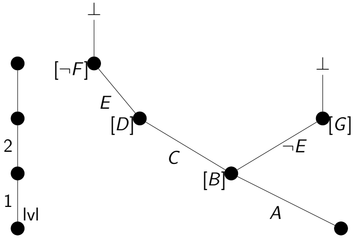
Algorithme CDCL (1996)

$$
\begin{aligned}
& (\neg A \lor B) \\
\land & (\neg C \lor D) \\
\land & (\neg E \lor \neg F) \\
\land & (F \lor \neg E \lor \neg B) \\
\land & (E \lor G) \\
\land & \color{red}{(E \lor \neg G \lor \neg B)} \\
\land & (\neg A \lor \neg E)
\end{aligned}
$$



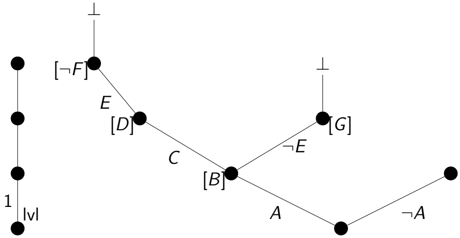
Algorithme CDCL (1996)

$$
\begin{aligned}
& (\neg A \vee B) \\
\wedge\ & (\neg C \vee D) \\
\wedge\ & (\neg E \vee \neg F) \\
\wedge\ & (F \vee \neg E \vee \neg B) \\
\wedge\ & (E \vee G) \\
\wedge\ & (E \vee \neg G \vee \neg B) \\
\wedge\ & (\neg A \vee \neg E) \\
\wedge\ & (\neg A)
\end{aligned}
$$



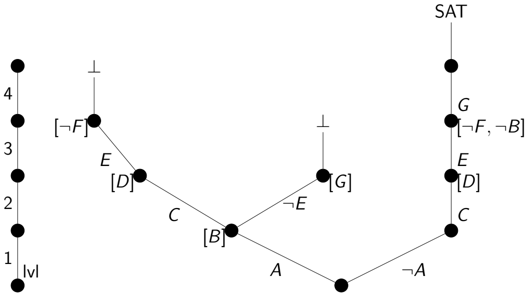
Algorithme CDCL (1996)

$$
\begin{aligned}
&(\neg A \vee B)\\
\wedge\,&(\neg C \vee D)\\
\wedge\,&(\neg E \vee \neg F)\\
\wedge\,&(F \vee \neg E \vee \neg B)\\
\wedge\,&(E \vee G)\\
\wedge\,&(E \vee \neg G \vee \neg B)\\
\wedge\,&(\neg A \vee \neg E)\\
\wedge\,&(\neg A)
\end{aligned}
$$



Algorithme CDCL (1996)

$$\begin{aligned}&(\neg A \vee B)\\ \wedge\,&(\neg C \vee D)\\ \wedge\,&(\neg E \vee \neg F)\\ \wedge\,&(F \vee \neg E \vee \neg B)\\ \wedge\,&(E \vee G)\\ \wedge\,&(E \vee \neg G \vee \neg B)\\ \wedge\,&(\neg A \vee \neg E)\\ \wedge\,&(\neg A)\end{aligned}$$



while *true* **do**
  $(\Delta, \mathit{Conflict}) \leftarrow \mathit{BCP}(\Gamma, \Delta)$
  if $\mathit{Conflict} \neq \emptyset$ **then**
    if $\mathit{lvl} = 0$ **then**
      return UNSAT
    else
      $(L \vee C, \mathit{bj\_lvl}) \leftarrow \mathit{resolve}(\Gamma, \Delta, \mathit{Conflict})$
      $\Gamma \leftarrow \Gamma \cup \{L \vee C\}$
      $(\Delta, \mathit{lvl}) \leftarrow \mathit{backjump}(\Delta, \mathit{bj\_lvl})$
      $\Delta \leftarrow L :: \Delta$
  else if *all variables are assigned in* Δ **then**
    return SAT
  else
    $L \leftarrow \mathit{choose}(\Gamma, \Delta)$
    $\mathit{lvl} \leftarrow \mathit{lvl} + 1$
    $\Delta \leftarrow L :: \Delta$

- **when conflict, backtracks to the latest guess that affects a literal in the learned clause**

$\rightarrow$ mistake $\rightsquigarrow$ clause learning $\rightsquigarrow$ don't do the same mistake

$\rightarrow$ clause learning feeds the BCP mechanism

From SAT CDCL :

while *true* **do**
  $(\Delta, \mathit{Conflict}) \leftarrow \mathit{BCP}(\Gamma, \Delta)$
  if $\mathit{Conflict} \neq \emptyset$ **then**
    if $\mathit{lvl} = 0$ **then**
      return UNSAT
    else
      $(L \vee C, \mathit{bj\_lvl}) \leftarrow \mathit{resolve}(\Gamma, \Delta, \mathit{Conflict})$
      $\Gamma \leftarrow \Gamma \cup \{L \vee C\}$
      $(\Delta, \mathit{lvl}) \leftarrow \mathit{backjump}(\Delta, \mathit{bj\_lvl})$
      $\Delta \leftarrow L :: \Delta$
  else if *all variables are assigned in* Δ **then**
    return SAT
  else
    $L \leftarrow \mathit{choose}(\Gamma, \Delta)$
    $\mathit{lvl} \leftarrow \mathit{lvl} + 1$
    $\Delta \leftarrow L :: \Delta$

To SMT CDCL(Theory) :

while *true* **do**
  $(\Delta, \mathit{Conflict}) \leftarrow \mathit{BCP}(\Gamma, \Delta)$
  if $\mathit{Conflict} \neq \emptyset$ **then**
    ...
  else
    $(T, \mathit{Conflict}) \leftarrow \mathit{theory\_assume}(\Delta, T)$
    if $\mathit{Conflict} \neq \emptyset$ **then**
      if $\mathit{lvl} = 0$ **then**
        return UNSAT
      else
        $(L \vee C, \mathit{bj\_lvl}) \leftarrow \mathit{resolve}(\Gamma, \Delta, T, \mathit{Conflict})$
        $\Gamma \leftarrow \Gamma \cup \{L \vee C\}$
        $(\Delta, T, \mathit{lvl}) \leftarrow \mathit{backjump}(\mathit{bj\_lvl})$
        $\Delta \leftarrow L :: \Delta$
    else if *all variables are assigned in* Δ **then**
      ...

Advantage of the CDCL method :

- fast

Disadvantage of the CDCL method :

- needs CNF

$\rightarrow$ returns the full boolean model

$\rightarrow$ don't take in account the original shape of the formula

$\rightarrow$ too much instanciations in SMT theories

Example :

$$\varphi = (A \vee (B \vee \varphi_1)) \wedge (\neg A \vee (\neg B \vee \neg \varphi_1))$$

assignment $\{A \rightarrow \mathit{true}; B \rightarrow \mathit{false}\}$ is a model for φ
but CDCL won't find it

Solveur SAT par méthode des tableaux d'Alt-Ergo

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{X}$$

Solveur SAT par méthode des tableaux d'Alt-Ergo

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{x_3})}_{x_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{x_5} \wedge G)}^{x_4})}_{x_2}}^{x}$$

[X

Solveur SAT par méthode des tableaux d'Alt-Ergo

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee (\overbrace{\underbrace{(E \vee F)}_{X_5} \wedge G}^{X_4}))}_{X_2}}^{X}$$

$[X; X_1; X_2]$

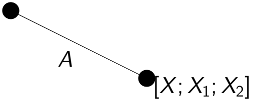
Solveur SAT par méthode des tableaux d'Alt-Ergo

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee (\overbrace{\underbrace{(E \vee F)}_{X_5} \wedge G}^{X_4}))}_{X_2}}^{X}$$

$[X; X_1; X_2]$

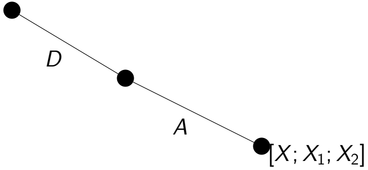
Solveur SAT par méthode des tableaux d'Alt-Ergo

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X_4})}_{X_2}}^{X}$$



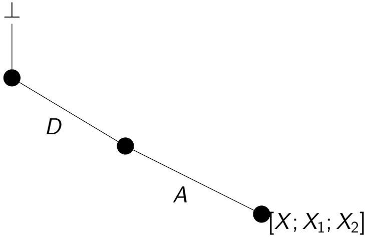
Solveur SAT par méthode des tableaux d'Alt-Ergo

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X_4})}_{X_2}}^{X}$$



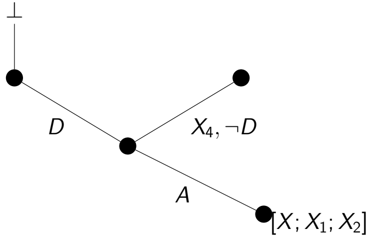
Solveur SAT par méthode des tableaux d'Alt-Ergo

$$\overbrace{\underbrace{(A \lor \overbrace{(B \land C)}^{x_3})}_{x_1} \land \underbrace{(D \lor \overbrace{(\underbrace{(E \lor F)}_{x_5} \land G)}^{x_4})}_{x_2}}^{x}$$



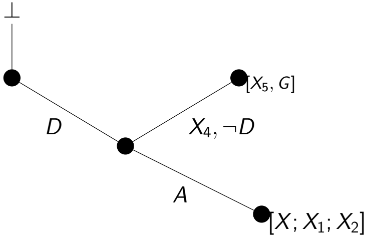
Solveur SAT par méthode des tableaux d'Alt-Ergo

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X_4})}_{X_2}}^{X}$$



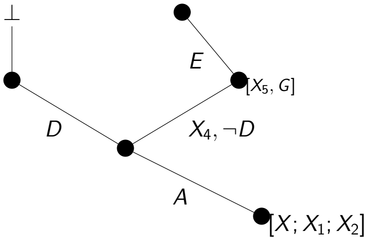
Solveur SAT par méthode des tableaux d'Alt-Ergo

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{x_3})}_{x_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{\color{blue}{x_5}} \wedge \color{blue}{G})}^{x_4})}_{x_2}}^{x}$$



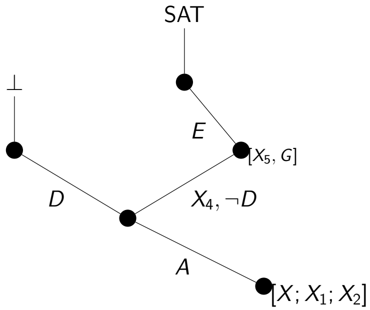
Solveur SAT par méthode des tableaux d'Alt-Ergo

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{\color{blue}{X_5}} \wedge G)}^{X_4})}_{X_2}}^{X}$$



Solveur SAT par méthode des tableaux d'Alt-Ergo

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{x_3})}_{x_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{x_5} \wedge G)}^{x_4})}_{x_2}}^{x}$$



Modèle booléen renvoyé par le solveur SAT : $\{A; \neg D; E; G\}$

X_i n'ont de l'influence que sur la partie SAT

Input: Φ : Set of formulas in Negatif Normal Form
Output: Satisfiability status, explanation

```
Function solve(Φ)
    (Φ, Error) ← propagate(Φ)        // Boolean constraint propagation
    if Error then
        reason ← explain_conflict()
        return (UNSAT,reason)
    else
        if ∃A ∨ B ∈ Φ then
            (Φ, Error) ← (assume(Φ, [{A}]))              // decide on A
            if Error then
                reason ← explain_conflict()
                return (UNSAT,reason)
            else
                (status, reason) ← solve(Φ)
                if status ≠ UNSAT then
                    return (status,reason)
                else
                    if A ∈ reason then
                        (Φ, Error) ← (assume(Φ, [{¬A}; {B}]))
                        // backtrack and propagate ¬A and B
                        if Error then
                            reason ← explain_conflict()
                            return (UNSAT,reason)
                        else
                            return solve(Φ)
                    else
                        return (UNSAT,reason)      // backjump further
        else
            return (SAT,∅)                         // Φ is empty
```

- "proxy" variables
- **when conflict, backtracks is changing literal (we were taking negation with the DPLL/CDCL)**
- truth value of a variable $\rightsquigarrow$ truth value of a disjonction
- negative normal form

Reduction of the literals sent to the SMT solver components (decision procedure + instanciation) with the Tableau-method :

- keep the original formula
- graph search : "pertinent" literals to keep are the ones assigned to true, during the graph search

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \lor \overbrace{(B \land C)}^{X_3})}_{X_1} \land \underbrace{(D \lor \overbrace{(\underbrace{(E \lor F)}_{X_5} \land G)}^{X4})}_{X_2}}^{X}$$

Modèle booléen partiel du solveur SAT :

$$A, G = \top$$
$$B, D = \bot$$
$$E, F, C = -$$

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{\mathrm{X}}$$

$X_1 \wedge X_2$

Modèle booléen partiel du solveur SAT :

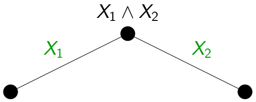
$$A, G = \top$$

$$B, D = \bot$$

$$E, F, C = -$$

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{\mathrm{X}}$$

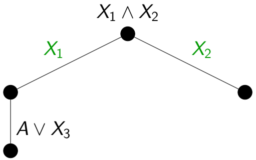


Modèle booléen partiel du solveur SAT :

$$A, G = \top$$
$$B, D = \bot$$
$$E, F, C = -$$

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{\mathrm{X}}$$

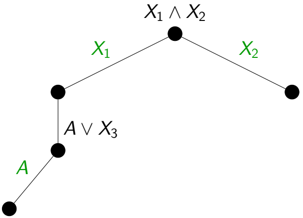


Modèle booléen partiel du solveur SAT :

$$A, G = \top$$
$$B, D = \bot$$
$$E, F, C = -$$

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{X}$$

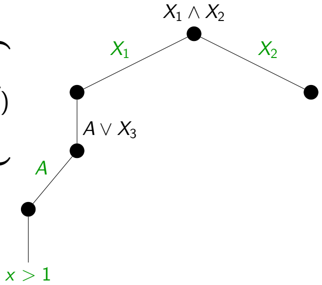


Modèle booléen partiel du solveur SAT :

$$A, G = \top$$
$$B, D = \bot$$
$$E, F, C = -$$

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{\mathrm{X}}$$

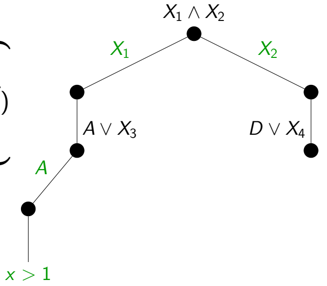


Modèle booléen partiel du solveur SAT :

$$A, G = \top$$
$$B, D = \bot$$
$$E, F, C = -$$

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{\mathrm{X}}$$

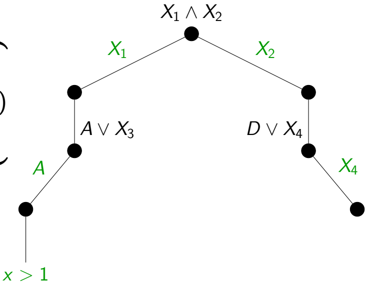


Modèle booléen partiel du solveur SAT :

$$A, G = \top$$
$$B, D = \bot$$
$$E, F, C = -$$

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{\mathrm{X}}$$

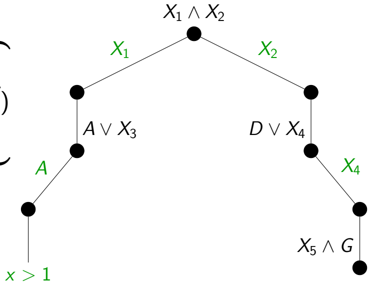


Modèle booléen partiel du solveur SAT :

$$A, G = \top$$
$$B, D = \bot$$
$$E, F, C = -$$

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{\mathrm{X}}$$

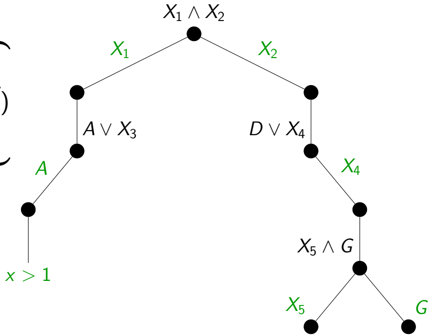


Modèle booléen partiel du solveur SAT :

$$A, G = \top$$
$$B, D = \bot$$
$$E, F, C = -$$

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \lor \overbrace{(B \land C)}^{X_3})}_{X_1} \land \underbrace{(D \lor \overbrace{(\underbrace{(E \lor F)}_{X_5} \land G)}^{X4})}_{X_2}}^{X}$$

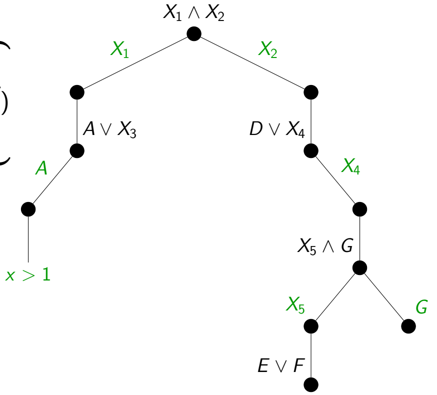


Modèle booléen partiel du solveur SAT :

$$A, G = \top$$
$$B, D = \bot$$
$$E, F, C = -$$

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{X}$$



Modèle booléen partiel du solveur SAT :

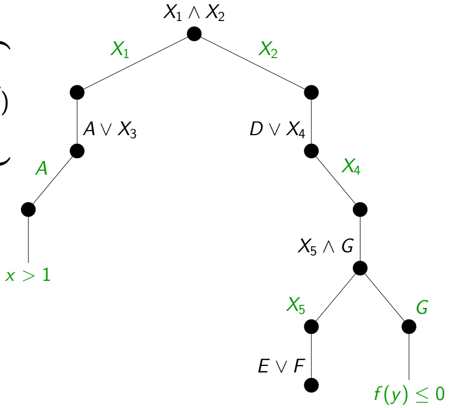
$$A, G = \top$$
$$B, D = \bot$$
$$E, F, C = -$$

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{\mathrm{x}}$$

Modèle booléen partiel du solveur SAT :

$$A, G = \top$$
$$B, D = \bot$$
$$E, F, C = -$$

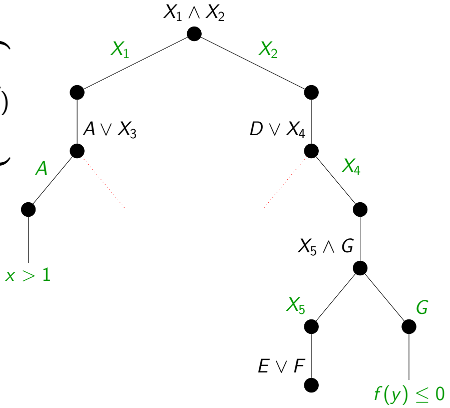


Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \lor \overbrace{(B \land C)}^{X_3})}_{X_1} \land \underbrace{(D \lor \overbrace{(\underbrace{(E \lor F)}_{X_5} \land G)}^{X4})}_{X_2}}^{\mathrm{X}}$$

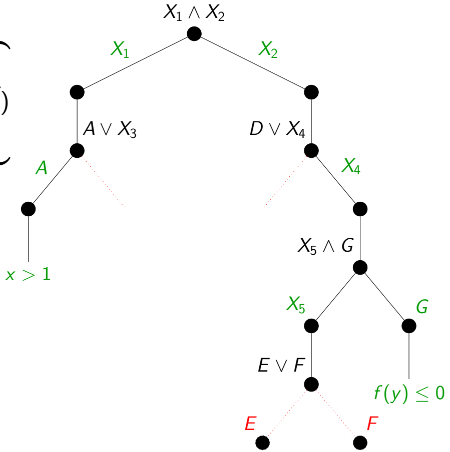
Modèle booléen partiel du solveur SAT :

$$A, G = \top$$
$$B, D = \bot$$
$$E, F, C = -$$



Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{X}$$



Modèle booléen partiel du solveur SAT :

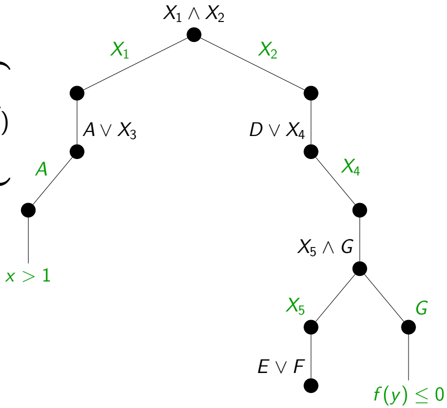
$$A, G = \top$$

$$B, D = \bot$$

$$E, F, C = -$$

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{x}$$



Modèle booléen complet du solveur SAT :

$$A, G = \top$$

$$B, D = \bot$$

$$C = \bot$$

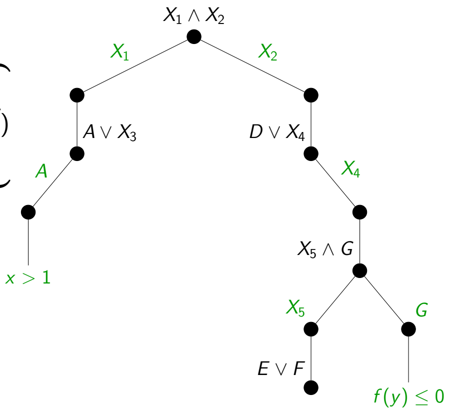
$$E, F = \top$$

Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{\mathrm{x}}$$

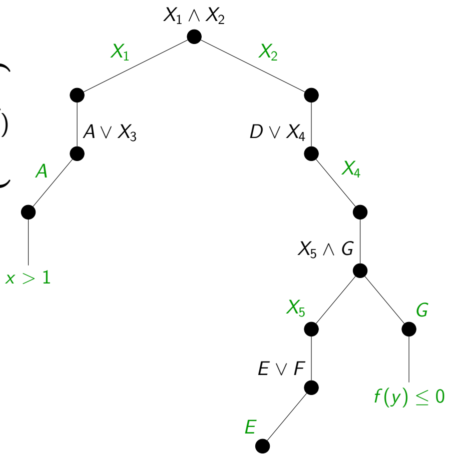
Modèle booléen complet du solveur SAT :

$$A, G = \top$$
$$B, D = \bot$$
$$C = \bot$$
$$E, F = \top$$



Réduction du modèle par méthode des tableaux

$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{X}$$



Modèle booléen complet du solveur SAT :

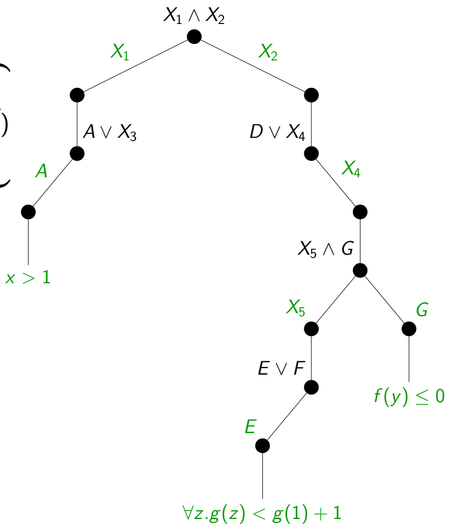
$$A, G = \top$$
$$B, D = \bot$$
$$C = \bot$$
$$E, F = \top$$

Réduction du modèle par méthode des tableaux

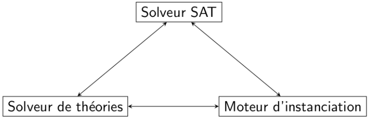
$$\overbrace{\underbrace{(A \vee \overbrace{(B \wedge C)}^{X_3})}_{X_1} \wedge \underbrace{(D \vee \overbrace{(\underbrace{(E \vee F)}_{X_5} \wedge G)}^{X4})}_{X_2}}^{\mathrm{X}}$$

Modèle booléen complet du solveur SAT :

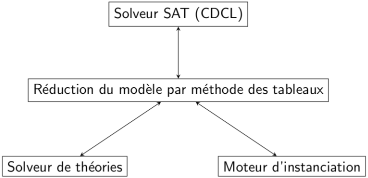
$$A, G = \top$$
$$B, D = \bot$$
$$C = \bot$$
$$E, F = \top$$



From this :



to this :



$$(A \vee (B \vee C)) \wedge (D \vee ((E \vee F) \wedge G))$$

- CDCL $\rightsquigarrow$ complete boolean model $\{A, \neg B, \neg C, \neg D, E, F, G\}$
- CDCL-tableaux $\rightsquigarrow$ reduced boolean model $\{A, E, G\}$ sent to the combinator of theories or instanciation engine

CDCL(*Tableaux*(*T*))

Résultats

	# buts	cdcl	tableaux	cdcl + tableaux
BWARE–DAB	860	98.7% (258s)	100% (417s)	**100%(47s)**
BWARE–RCS3	2256	98.7% (742s)	98.9% (685s)	**99.0%(725s)**
BWARE–p4	9341	98.4% (2097s)	99.3% (2279s)	**99.4(790s)**
BWARE–p9	371	64.7% (1104s)	67.9% (342s)	**72.2%(492s)**
EACSL	959	75.6% (64s)	**93.3% (258s)**	92.3%(293s)
SPARK	16773	80.5% (1769s)	83.6% (2757s)	**84.0%(2298s)**
WHY3	2003	38.9% (616s)	**72.0% (1876s)**	69.8% (1471s)
Total	32563	84.5% (6652s)	89.0% (8617s)	**89.1%(6119s)**

- Solution efficace (-29% en temps)
- Permet à Alt-Ergo d'être performant sur des problèmes fortement booléens
- Tout en restant performant sur des problèmes quantifiés

Results :

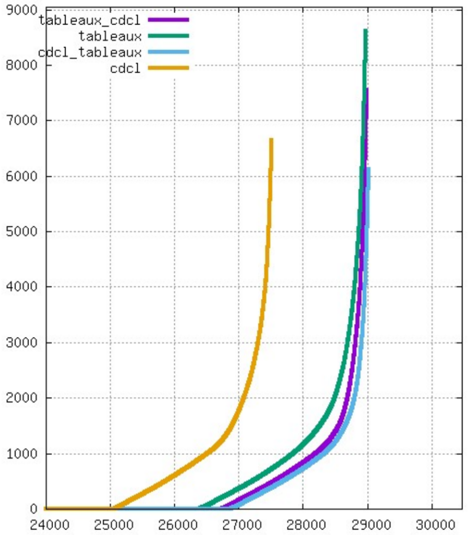


FIGURE 4.19 – Graphique du temps de résolution des solveurs SAT d'Alt-Ergo en fonction du nombre de buts résolus sur des fichiers issus de la preuve de programme.

5) Conclusion

SMT solvers :

- hard to be efficient and generalist
- heuristics working on some problems but not on the others
- result of $\sim$ 40 years of research and experiments

Alt-Ergo has now four core solvers :

- `cdcl` performance on boolean problems
- `tableaux` to reduce boolean problems
- `tableaux-cdcl` not good enough
- `cdcl-tableaux` $\Rightarrow$ best performance

$\rightarrow$ default since Alt-Ergo 2.3.0 (February 2019)