Logic and Mechanized Reasoning

DP & DPLL

Marijn J.H. Heule

Carnegie Mellon University

First Midterm Exam

Monday February 19 at 12:30pm

Material covered in the exam:

- All lectures up to (and including) February 7
- All homework through Assignment 4
- Textbook chapters 1-7, excluding Sections 6.3, 6.5, 7.4

Practice exam and solutions on course website

No new homework assigned this week

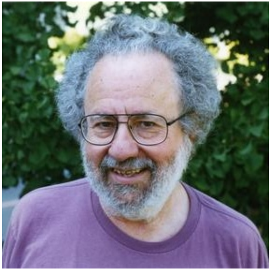
DP Resolution

DPLL

Martin Davis (March 8, 1928 – January 1, 2023)

Martin Davis & Hilary Putnam (1960)
A Computing Procedure for Quantification Theory.
Journal of hte ACM 7(3): 201-215

Martin Davis, George Logemann, & Donald W. Loveland (1962)
A machine program for theorem-proving.
Communications of the ACM 5(7): 394-397



DP Resolution

DPLL

DP Resolution / Variable Elimination [DavisPutnam'60]

Definition (Resolution)

Given two clauses $C = (x \vee p_1 \vee \cdots \vee p_i)$ and $D = (\overline{x} \vee b_1 \vee \cdots \vee b_j)$, the *resolvent* of C and D on variable x (denoted by $C \bowtie_x D$) is $(p_1 \vee \cdots \vee p_i \vee b_1 \vee \cdots \vee b_j)$

Resolution on sets of clauses Γ_x and $\Gamma_{\overline{x}}$ (denoted by $\Gamma_x \bowtie_x \Gamma_{\overline{x}}$) generates all non-tautological resolvents of $C \in \Gamma_x$ and $D \in \Gamma_{\overline{x}}$.

DP Resolution / Variable Elimination [DavisPutnam'60]

Definition (Resolution)

Given two clauses $C = (x \vee p_1 \vee \cdots \vee p_i)$ and $D = (\overline{x} \vee b_1 \vee \cdots \vee b_j)$, the *resolvent* of C and D on variable x (denoted by $C \bowtie_x D$) is $(p_1 \vee \cdots \vee p_i \vee b_1 \vee \cdots \vee b_j)$

Resolution on sets of clauses Γ_x and $\Gamma_{\overline{x}}$ (denoted by $\Gamma_x \bowtie_x \Gamma_{\overline{x}}$) generates all non-tautological resolvents of $C \in \Gamma_x$ and $D \in \Gamma_{\overline{x}}$.

Definition (Variable elimination (VE))

Given a CNF formula Γ, *variable elimination* (or DP resolution) removes a variable x by replacing Γ_x and $\Gamma_{\overline{x}}$ by $\Gamma_x \bowtie_x \Gamma_{\overline{x}}$

DP Resolution / Variable Elimination [DavisPutnam'60]

Definition (Resolution)

Given two clauses $C = (x \vee p_1 \vee \cdots \vee p_i)$ and $D = (\overline{x} \vee b_1 \vee \cdots \vee b_j)$, the *resolvent* of C and D on variable x (denoted by $C \bowtie_x D$) is $(p_1 \vee \cdots \vee p_i \vee b_1 \vee \cdots \vee b_j)$

Resolution on sets of clauses Γ_x and $\Gamma_{\overline{x}}$ (denoted by $\Gamma_x \bowtie_x \Gamma_{\overline{x}}$) generates all non-tautological resolvents of $C \in \Gamma_x$ and $D \in \Gamma_{\overline{x}}$.

Definition (Variable elimination (VE))

Given a CNF formula Γ, *variable elimination* (or DP resolution) removes a variable x by replacing Γ_x and $\Gamma_{\overline{x}}$ by $\Gamma_x \bowtie_x \Gamma_{\overline{x}}$

Proof procedure [DavisPutnam60]

VE is a complete proof procedure. Applying VE until fixpoint results in either the empty formula (satisfiable) or empty clause (unsatisfiable)

Example VE by clause distribution [DavisPutnam'60]

Definition (Variable elimination (VE))

Given a CNF formula Γ, *variable elimination* (or DP resolution) removes a variable x by replacing Γ_x and $\Gamma_{\overline{x}}$ by $\Gamma_x \bowtie_x \Gamma_{\overline{x}}$

Example VE by clause distribution [DavisPutnam'60]

Definition (Variable elimination (VE))

Given a CNF formula Γ, *variable elimination* (or DP resolution) removes a variable x by replacing Γ_x and $\Gamma_{\overline{x}}$ by $\Gamma_x \bowtie_x \Gamma_{\overline{x}}$

Example of clause distribution

	Γ_x: $(x \vee c)$	Γ_x: $(x \vee \overline{d})$	Γ_x: $(x \vee \overline{a} \vee \overline{b})$
$\Gamma_{\overline{x}}$: $(\overline{x} \vee a)$	$(a \vee c)$	$(a \vee \overline{d})$	$(a \vee \overline{a} \vee \overline{b})$
$\Gamma_{\overline{x}}$: $(\overline{x} \vee b)$	$(b \vee c)$	$(b \vee \overline{d})$	$(b \vee \overline{a} \vee \overline{b})$
$\Gamma_{\overline{x}}$: $(\overline{x} \vee \overline{e} \vee f)$	$(c \vee \overline{e} \vee f)$	$(\overline{d} \vee \overline{e} \vee f)$	$(\overline{a} \vee \overline{b} \vee \overline{e} \vee f)$

Example VE by clause distribution [DavisPutnam'60]

Definition (Variable elimination (VE))

Given a CNF formula Γ, *variable elimination* (or DP resolution) removes a variable x by replacing Γ_x and $\Gamma_{\overline{x}}$ by $\Gamma_x \bowtie_x \Gamma_{\overline{x}}$

Example of clause distribution

	Γ_x: $(x \vee c)$	Γ_x: $(x \vee \overline{d})$	Γ_x: $(x \vee \overline{a} \vee \overline{b})$
$\Gamma_{\overline{x}}$: $(\overline{x} \vee a)$	$(a \vee c)$	$(a \vee \overline{d})$	~~$(a \vee \overline{a} \vee \overline{b})$~~
$\Gamma_{\overline{x}}$: $(\overline{x} \vee b)$	$(b \vee c)$	$(b \vee \overline{d})$	~~$(b \vee \overline{a} \vee \overline{b})$~~
$\Gamma_{\overline{x}}$: $(\overline{x} \vee \overline{e} \vee f)$	$(c \vee \overline{e} \vee f)$	$(\overline{d} \vee \overline{e} \vee f)$	$(\overline{a} \vee \overline{b} \vee \overline{e} \vee f)$

Example VE by clause distribution [DavisPutnam'60]

Definition (Variable elimination (VE))

Given a CNF formula Γ, *variable elimination* (or DP resolution) removes a variable x by replacing Γ_x and $\Gamma_{\overline{x}}$ by $\Gamma_x \bowtie_x \Gamma_{\overline{x}}$

Example of clause distribution

	Γ_x: $(x \vee c)$	$(x \vee \overline{d})$	$(x \vee \overline{a} \vee \overline{b})$
$\Gamma_{\overline{x}}$: $(\overline{x} \vee a)$	$(a \vee c)$	$(a \vee \overline{d})$	~~$(a \vee \overline{a} \vee \overline{b})$~~
$(\overline{x} \vee b)$	$(b \vee c)$	$(b \vee \overline{d})$	~~$(b \vee \overline{a} \vee \overline{b})$~~
$(\overline{x} \vee \overline{e} \vee f)$	$(c \vee \overline{e} \vee f)$	$(\overline{d} \vee \overline{e} \vee f)$	$(\overline{a} \vee \overline{b} \vee \overline{e} \vee f)$

In the example: $|\Gamma_x \bowtie \Gamma_{\overline{x}}| > |\Gamma_x| + |\Gamma_{\overline{x}}|$

Exponential growth of clauses in general

DP Resolution and Pure Literals

Proposition

Given a CNF formula Γ with pure literal p, the effect of applying the pure literal rule on p is the same as the effect of applying DP resolution on p.

True or false?

DP Resolution and Pure Literals

Proposition

Given a CNF formula Γ with pure literal p, the effect of applying the pure literal rule on p is the same as the effect of applying DP resolution on p.

True or false?

Proof.

True. The pure literal rule assign p to true, which has the effect that all clauses containing p are removed. Applying DP resolution on p also removes all clauses containing literal p, because $\Gamma_p \bowtie \Gamma_{\neg p}$ is empty. □

VE by substitution [EenBiere07]

General idea

Detect definitions $x = \mathrm{DEF}(p_1, \ldots, p_n)$ in the formula and use them to reduce the number of added clauses

VE by substitution [EenBiere07]

General idea

Detect definitions $x = \mathrm{DEF}(p_1, \ldots, p_n)$ in the formula and use them to reduce the number of added clauses

Possible gates

definition	D_x	$D_{\overline{x}}$
$\mathrm{AND}(p_1, \ldots, p_n)$	$(x \vee \overline{p}_1 \vee \cdots \vee \overline{p}_n)$	$(\overline{x} \vee p_1), \ldots, (\overline{x} \vee p_n)$
$\mathrm{OR}(p_1, \ldots, p_n)$	$(x \vee \overline{p}_1), \ldots, (x \vee \overline{p}_n)$	$(\overline{x} \vee p_1 \vee \cdots \vee p_n)$
$\mathrm{ITE}(c, t, f)$	$(x \vee \overline{c} \vee \overline{t}), (x \vee c \vee \overline{f})$	$(\overline{x} \vee \overline{c} \vee t), (\overline{x} \vee c \vee f)$

VE by substitution [EenBiere07]

General idea

Detect definitions $x = \mathrm{DEF}(p_1, \ldots, p_n)$ in the formula and use them to reduce the number of added clauses

Possible gates

definition	D_x	$D_{\overline{x}}$
$\mathrm{AND}(p_1, \ldots, p_n)$	$(x \vee \overline{p}_1 \vee \cdots \vee \overline{p}_n)$	$(\overline{x} \vee p_1), \ldots, (\overline{x} \vee p_n)$
$\mathrm{OR}(p_1, \ldots, p_n)$	$(x \vee \overline{p}_1), \ldots, (x \vee \overline{p}_n)$	$(\overline{x} \vee p_1 \vee \cdots \vee p_n)$
$\mathrm{ITE}(c, t, f)$	$(x \vee \overline{c} \vee \overline{t}), (x \vee c \vee \overline{f})$	$(\overline{x} \vee \overline{c} \vee t), (\overline{x} \vee c \vee f)$

Variable elimination by substitution [EenBiere07]

Let $R_x = \Gamma_x \setminus D_x$; $R_{\overline{x}} = \Gamma_{\overline{x}} \setminus D_{\overline{x}}$.

Replace $\Gamma_x \wedge \Gamma_{\overline{x}}$ by $D_x \bowtie_x R_{\overline{x}} \wedge D_{\overline{x}} \bowtie_x R_x$.

Always less than $\Gamma_x \bowtie_x \Gamma_{\overline{x}}$! (if x is a definition)

VE by substitution [EenBiere'07]

Example of gate extraction: $x = \mathrm{AND}(a, b)$

$$\Gamma_x = (x \vee c) \wedge (x \vee \bar{d}) \wedge (x \vee \bar{a} \vee \bar{b})$$
$$\Gamma_{\bar{x}} = (\bar{x} \vee a) \wedge (\bar{x} \vee b) \wedge (\bar{x} \vee \bar{e} \vee f)$$

VE by substitution [EenBiere'07]

Example of gate extraction: $x = \mathrm{AND}(a, b)$

$$\Gamma_x = (x \vee c) \wedge (x \vee \overline{d}) \wedge (x \vee \overline{a} \vee \overline{b})$$
$$\Gamma_{\overline{x}} = (\overline{x} \vee a) \wedge (\overline{x} \vee b) \wedge (\overline{x} \vee \overline{e} \vee f)$$

Example of substitution

		R_x		D_x
		$(x \vee c)$	$(x \vee \overline{d})$	$(x \vee \overline{a} \vee \overline{b})$
$D_{\overline{x}}$	$(\overline{x} \vee a)$	$(a \vee c)$	$(a \vee \overline{d})$	
	$(\overline{x} \vee b)$	$(b \vee c)$	$(b \vee \overline{d})$	
$R_{\overline{x}}$	$(\overline{x} \vee \overline{e} \vee f)$			$(\overline{a} \vee \overline{b} \vee \overline{e} \vee f)$

VE by substitution [EenBiere'07]

Example of gate extraction: $x = \text{AND}(a, b)$

$$\Gamma_x = (x \vee c) \wedge (x \vee \bar{d}) \wedge (x \vee \bar{a} \vee \bar{b})$$
$$\Gamma_{\bar{x}} = (\bar{x} \vee a) \wedge (\bar{x} \vee b) \wedge (\bar{x} \vee \bar{e} \vee f)$$

Example of substitution

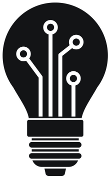
		R_x		D_x
		$(x \vee c)$	$(x \vee \bar{d})$	$(x \vee \bar{a} \vee \bar{b})$
$D_{\bar{x}}$	$(\bar{x} \vee a)$	$(a \vee c)$	$(a \vee \bar{d})$	
	$(\bar{x} \vee b)$	$(b \vee c)$	$(b \vee \bar{d})$	
$R_{\bar{x}}$	$(\bar{x} \vee \bar{e} \vee f)$			$(\bar{a} \vee \bar{b} \vee \bar{e} \vee f)$

using substitution: $|\Gamma_x \bowtie \Gamma_{\bar{x}}| < |\Gamma_x| + |\Gamma_{\bar{x}}|$

DP Resolution

DPLL

SAT Solver Paradigms Overview



DPLL: Aims at finding a small search-tree by selecting effective splitting variables (e.g. via looking ahead).

Strength: Effective on small, hard formulas.

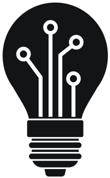
Weakness: Expensive.

SAT Solver Paradigms Overview

DPLL: Aims at finding a small search-tree by selecting effective splitting variables (e.g. via looking ahead).

Strength: Effective on small, hard formulas.

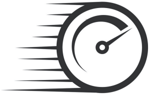
Weakness: Expensive.



Local search: Given a full assignment for a formula Γ, flip the truth values of variables until satisfying Γ.

Strength: Can quickly find solutions for hard formulas.

Weakness: Cannot prove unsatisfiability.

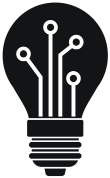


SAT Solver Paradigms Overview

DPLL: Aims at finding a small search-tree by selecting effective splitting variables (e.g. via looking ahead).

Strength: Effective on small, hard formulas.

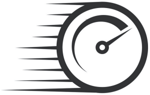
Weakness: Expensive.



Local search: Given a full assignment for a formula Γ, flip the truth values of variables until satisfying Γ.

Strength: Can quickly find solutions for hard formulas.

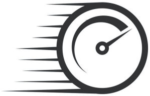
Weakness: Cannot prove unsatisfiability.



Conflict-driven clause learning (CDCL): Makes fast decisions and converts conflicts into learned clauses.

Strength: Effective on large, "easy" formulas.

Weakness: Hard to parallelize.



DPLL: Introduction

Davis Putnam Logemann Loveland [DP60,DLL62]

Recursive procedure that in each recursive call:

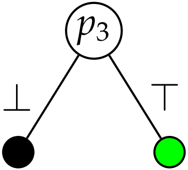
- Simplifies the formula (using unit propagation)
- Splits the formula into two subformulas
    - Variable selection heuristics (which variable to split on)
    - Direction heuristics (which subformula to explore first)

DPLL: Example

$$\Gamma_{\text{DPLL}} := (p_1 \vee p_2 \vee \neg p_3) \wedge (\neg p_1 \vee p_2 \vee p_3) \wedge (\neg p_1 \vee \neg p_2 \vee p_3) \wedge (p_1 \vee p_3) \wedge (\neg p_1 \vee \neg p_3)$$

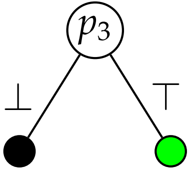
DPLL: Example

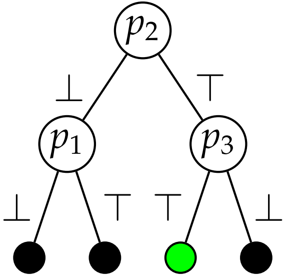
$$\Gamma_{\mathrm{DPLL}} := (p_1 \vee p_2 \vee \neg p_3) \wedge (\neg p_1 \vee p_2 \vee p_3) \wedge (\neg p_1 \vee \neg p_2 \vee p_3) \wedge (p_1 \vee p_3) \wedge (\neg p_1 \vee \neg p_3)$$



DPLL: Example

$$\Gamma_{\text{DPLL}} := (p_1 \vee p_2 \vee \neg p_3) \wedge (\neg p_1 \vee p_2 \vee p_3) \wedge (\neg p_1 \vee \neg p_2 \vee p_3) \wedge (p_1 \vee p_3) \wedge (\neg p_1 \vee \neg p_3)$$





DPLL: Slightly Harder Example

Construct a DPLL tree for:

$$(p \vee q \vee \neg r) \wedge (\neg p \vee \neg q \vee r) \wedge$$
$$(q \vee r \vee \neg s) \wedge (\neg q \vee \neg r \vee s) \wedge$$
$$(p \vee r \vee s) \wedge (\neg p \vee \neg r \vee \neg s) \wedge$$
$$(\neg p \vee q \vee s)$$

What is a good heuristic?

DPLL: Slightly Harder Example

Construct a DPLL tree for:

$$\begin{aligned}&(p \vee q \vee \neg r) \wedge (\neg p \vee \neg q \vee r) \wedge \\ &(q \vee r \vee \neg s) \wedge (\neg q \vee \neg r \vee s) \wedge \\ &(p \vee r \vee s) \wedge (\neg p \vee \neg r \vee \neg s) \wedge \\ &(\neg p \vee q \vee s)\end{aligned}$$

What is a good heuristic?

A cheap and reasonably effective heuristic is MOMS:
Maximum Occurrence in clauses of Minimum Size

DPLL: Pseudocode

DPLL (τ, Γ)

$\tau' :=$ Simplify (τ, Γ)

if $[\![\Gamma]\!]_{\tau'} = \top$ **then return** `satisfiable`

if $[\![\Gamma]\!]_{\tau'} = \bot$ **then return** `unsatisfiable`

$l_{\text{decision}} :=$ Decide (τ', Γ)

if (DPLL($\tau' \cup l_{\text{decision}} := \top, \Gamma) =$ `satisfiable`) **then**

  return `satisfiable`

return DPLL $(\tau' \cup l_{\text{decision}} := \bot, \Gamma)$

DPLL: Demo in Lean

```
-- textbook: dpllSat
partial def dpllSatAux (τ : PropAssignment) (φ : CnfForm) : Option (PropAssignment × CnfForm) :=
  if φ.hasEmpty then none
  else match pickSplit? φ with
  -- No variables left to split on, we found a solution.
  | none => some (τ, φ)
  -- Split on `x`.
  -- `<|>` is the "or else" operator which tries one action, and if that failed tries the other.
  | some x => goWithNew x τ φ <|> goWithNew (-x) τ φ

where
  /-- Assigns `x` to true and continues out DPLL. -/
  goWithNew (x : Lit) (τ : PropAssignment) (φ : CnfForm) : Option (PropAssignment × CnfForm) :=
    let (τ', φ') := propagateWithNew x τ φ
    dpllSatAux τ' φ'

/-- Solve `φ` using DPLL. Return a satisfying assignment if found, otherwise `none`. -/
def dpllSat (φ : CnfForm) : Option PropAssignment :=
  let ⟨τ, φ⟩ := propagateUnits [] φ
  (dpllSatAux τ φ).map fun ⟨τ, _⟩ => τ
-- end textbook: dpllSat
```

DPLL: Look-aheads

DPLL with selection of (effective) decision variables by look-aheads on variables

DPLL: Look-aheads

DPLL with selection of (effective) decision variables by look-aheads on variables

Look-ahead:

- Assign a variable to a truth value

DPLL: Look-aheads

DPLL with selection of (effective) decision variables by look-aheads on variables

Look-ahead:

- Assign a variable to a truth value
- Simplify the formula

DPLL: Look-aheads

DPLL with selection of (effective) decision variables by look-aheads on variables

Look-ahead:

- Assign a variable to a truth value
- Simplify the formula
- Measure the reduction

DPLL: Look-aheads

DPLL with selection of (effective) decision variables by look-aheads on variables

Look-ahead:

- Assign a variable to a truth value
- Simplify the formula
- Measure the reduction
- Learn if possible

DPLL: Look-aheads

DPLL with selection of (effective) decision variables by look-aheads on variables

Look-ahead:

- Assign a variable to a truth value
- Simplify the formula
- Measure the reduction
- Learn if possible
- Backtrack

DPLL: Look-ahead Reduction Heuristics

- Number of satisfied clauses

DPLL: Look-ahead Reduction Heuristics

- Number of satisfied clauses
- Number of implied variables

DPLL: Look-ahead Reduction Heuristics

- Number of satisfied clauses
- Number of implied variables
- New (reduced, not satisfied) clauses
  - Smaller clauses more important
  - Weights based on occurrences

DPLL: Learning Necessary Assignments

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

DPLL: Learning Necessary Assignments

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top\}$$

DPLL: Learning Necessary Assignments

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top, p_2 = \top\}$$

DPLL: Learning Necessary Assignments

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top, p_2 = \top, p_3 = \top\}$$

DPLL: Learning Necessary Assignments

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top, p_2 = \top, p_3 = \top, p_4 = \top\}$$

DPLL: Learning Necessary Assignments

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top, p_2 = \top, p_3 = \top, p_4 = \top\}$$

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

DPLL: Learning Necessary Assignments

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top, p_2 = \top, p_3 = \top, p_4 = \top\}$$

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \bot\}$$

DPLL: Learning Necessary Assignments

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top, p_2 = \top, p_3 = \top, p_4 = \top\}$$

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \bot, p_6 = \bot\}$$

DPLL: Learning Necessary Assignments

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top, p_2 = \top, p_3 = \top, p_4 = \top\}$$

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \bot, p_6 = \bot, p_3 = \top\}$$

DPLL: Look-ahead Autarky Detection

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

DPLL: Look-ahead Autarky Detection

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top\}$$

DPLL: Look-ahead Autarky Detection

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top, p_2 = \top\}$$

DPLL: Look-ahead Autarky Detection

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top, p_2 = \top, p_3 = \top\}$$

DPLL: Look-ahead Autarky Detection

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top, p_2 = \top, p_3 = \top, p_4 = \top\}$$

DPLL: Look-ahead Autarky Detection

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top, p_2 = \top, p_3 = \top, p_4 = \top\}$$

Γ_{LEARN} satisfiability equivalent to $(p_5 \vee \overline{p}_6)$

DPLL: Look-ahead Autarky Detection

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_1 = \top, p_2 = \top, p_3 = \top, p_4 = \top\}$$

Γ_{LEARN} satisfiability equivalent to $(p_5 \vee \overline{p}_6)$

Could reduce computational cost on UNSAT

DPLL: Look-ahead 1-Autarky Learning

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

DPLL: Look-ahead 1-Autarky Learning

$$\Gamma_{\text{LEARN}} := (\bar{p}_1 \vee \bar{p}_3 \vee p_4) \wedge (\bar{p}_1 \vee \bar{p}_2 \vee p_3) \wedge (\bar{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\bar{p}_1 \vee p_4 \vee \bar{p}_5) \wedge (p_1 \vee \bar{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \bar{p}_6)$$

$$\tau = \{p_2 = \bot\}$$

DPLL: Look-ahead 1-Autarky Learning

$$\Gamma_{\textsc{learn}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$\tau = \{p_2 = \bot, p_1 = \bot\}$

DPLL: Look-ahead 1-Autarky Learning

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_2 = \bot, p_1 = \bot, p_6 = \bot\}$$

DPLL: Look-ahead 1-Autarky Learning

$$\Gamma_{\text{LEARN}} := (\bar{p}_1 \vee \bar{p}_3 \vee p_4) \wedge (\bar{p}_1 \vee \bar{p}_2 \vee p_3) \wedge$$
$$(\bar{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\bar{p}_1 \vee p_4 \vee \bar{p}_5) \wedge$$
$$(p_1 \vee \bar{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \bar{p}_6)$$

$$\tau = \{p_2 = \bot, p_1 = \bot, p_6 = \bot, p_3 = \top\}$$

DPLL: Look-ahead 1-Autarky Learning

$$\Gamma_{\text{LEARN}} := (\overline{p}_1 \vee \overline{p}_3 \vee p_4) \wedge (\overline{p}_1 \vee \overline{p}_2 \vee p_3) \wedge (\overline{p}_1 \vee p_2) \wedge (p_1 \vee p_3 \vee p_6) \wedge (\overline{p}_1 \vee p_4 \vee \overline{p}_5) \wedge (p_1 \vee \overline{p}_6) \wedge (p_4 \vee p_5 \vee p_6) \wedge (p_5 \vee \overline{p}_6)$$

$$\tau = \{p_2 = \bot, p_1 = \bot, p_6 = \bot, p_3 = \top\}$$

(local) 1-autarky resolvents:
$(\overline{p}_2 \vee \overline{p}_4)$ and $(\overline{p}_2 \vee \overline{p}_5)$