Decision Procedures

Jochen Hoenicke

Software Engineering
Albert-Ludwigs-University Freiburg

Summer 2012

DPLL(T)

Suppose we have a $T_{\mathbb{Q}}$-formulae that is not conjunctive:

$$(x \geq 0 \rightarrow y > z) \land (x + y \geq z \rightarrow y \leq z) \land (y \geq 0 \rightarrow x \geq 0) \land x + y \geq z$$

Satisfiability and Conjunctive Theories

Suppose we have a $T_{\mathbb{Q}}$-formulae that is not conjunctive:

$$(x \geq 0 \rightarrow y > z) \wedge (x + y \geq z \rightarrow y \leq z) \wedge (y \geq 0 \rightarrow x \geq 0) \wedge x + y \geq z$$

Our approach so far: Converting to DNF.
Yields in 8 conjuncts that have to be checked separately.

Satisfiability and Conjunctive Theories

Suppose we have a $T_{\mathbb{Q}}$-formulae that is not conjunctive:

$$(x \geq 0 \rightarrow y > z) \wedge (x + y \geq z \rightarrow y \leq z) \wedge (y \geq 0 \rightarrow x \geq 0) \wedge x + y \geq z$$

Our approach so far: Converting to DNF.
Yields in 8 conjuncts that have to be checked separately.

Is there a more efficient way to prove unsatisfiability?

CNF and Propositional Core

Suppose we have the following $T_{\mathbb{Q}}$-formulae:

$$(x \geq 0 \rightarrow y > z) \land (x + y \geq z \rightarrow y \leq z) \land (y \geq 0 \rightarrow x \geq 0) \land x + y \geq z$$

CNF and Propositional Core

Suppose we have the following $T_{\mathbb{Q}}$-formulae:

$$(x \geq 0 \rightarrow y > z) \wedge (x + y \geq z \rightarrow y \leq z) \wedge (y \geq 0 \rightarrow x \geq 0) \wedge x + y \geq z$$

Converting to CNF and restricting to $\leq$:

$$\begin{aligned}(\neg(0 \leq x) \vee \neg(y \leq z)) \wedge (\neg(z \leq x + y) \vee (y \leq z)) \\ \wedge(\neg(0 \leq y) \vee (0 \leq x)) \wedge (z \leq x + y)\end{aligned}$$

CNF and Propositional Core

Suppose we have the following $T_{\mathbb{Q}}$-formulae:

$$(x \geq 0 \rightarrow y > z) \wedge (x + y \geq z \rightarrow y \leq z) \wedge (y \geq 0 \rightarrow x \geq 0) \wedge x + y \geq z$$

Converting to CNF and restricting to $\leq$:

$$(\neg(0 \leq x) \vee \neg(y \leq z)) \wedge (\neg(z \leq x + y) \vee (y \leq z)) \\ \wedge(\neg(0 \leq y) \vee (0 \leq x)) \wedge (z \leq x + y)$$

Now, introduce boolean variables for each atom:

$$P_1 : 0 \leq x \qquad P_2 : y \leq z$$
$$P_3 : z \leq x + y \qquad P_4 : 0 \leq y$$

Gives a propositional formula:

$$(\neg P_1 \vee \neg P_2) \wedge (\neg P_3 \vee P_2) \wedge (\neg P_4 \vee P_1) \wedge P_3$$

DPLL-Algorithm

The core feature of the DPLL-algorithm is Unit Propagation.

$$(\neg P_1 \lor \neg P_2) \land (\neg P_3 \lor P_2) \land (\neg P_4 \lor P_1) \land P_3$$

DPLL-Algorithm

The core feature of the DPLL-algorithm is Unit Propagation.

$$(\neg P_1 \lor \neg P_2) \land (\neg P_3 \lor P_2) \land (\neg P_4 \lor P_1) \land P_3$$

The clause P_3 is a unit clause; set P_3 to $\top$.

DPLL-Algorithm

The core feature of the DPLL-algorithm is Unit Propagation.

$$(\neg P_1 \lor \neg P_2) \land (\neg P_3 \lor P_2) \land (\neg P_4 \lor P_1) \land P_3$$

The clause P_3 is a unit clause; set P_3 to $\top$.
Then $\neg P_3 \lor P_2$ is a unit clause; set P_2 to $\top$.

DPLL-Algorithm

The core feature of the DPLL-algorithm is Unit Propagation.

$$(\neg P_1 \lor \neg P_2) \land (\neg P_3 \lor P_2) \land (\neg P_4 \lor P_1) \land P_3$$

The clause P_3 is a unit clause; set P_3 to $\top$.
Then $\neg P_3 \lor P_2$ is a unit clause; set P_2 to $\top$.
Then $\neg P_1 \lor \neg P_2$ is a unit clause; set P_1 to $\bot$.

DPLL-Algorithm

The core feature of the DPLL-algorithm is Unit Propagation.

$$(\neg P_1 \lor \neg P_2) \land (\neg P_3 \lor P_2) \land (\neg P_4 \lor P_1) \land P_3$$

The clause P_3 is a unit clause; set P_3 to $\top$.
Then $\neg P_3 \lor P_2$ is a unit clause; set P_2 to $\top$.
Then $\neg P_1 \lor \neg P_2$ is a unit clause; set P_1 to $\bot$.
Then $\neg P_4 \lor P_1$ is a unit clause; set P_4 to $\bot$.

The core feature of the DPLL-algorithm is Unit Propagation.

$$(\neg P_1 \vee \neg P_2) \wedge (\neg P_3 \vee P_2) \wedge (\neg P_4 \vee P_1) \wedge P_3$$

The clause P_3 is a unit clause; set P_3 to $\top$.
Then $\neg P_3 \vee P_2$ is a unit clause; set P_2 to $\top$.
Then $\neg P_1 \vee \neg P_2$ is a unit clause; set P_1 to $\bot$.
Then $\neg P_4 \vee P_1$ is a unit clause; set P_4 to $\bot$.

Only solution is $P_3 \wedge P_2 \wedge \neg P_1 \wedge \neg P_4$.

Only solution is $P_3 \wedge P_2 \wedge \neg P_1 \wedge \neg P_4$.

$$P_1 : 0 \leq x \qquad\qquad P_2 : y \leq z$$
$$P_3 : z \leq x + y \qquad\qquad P_4 : 0 \leq y$$

This gives the conjunctive $T_{\mathbb{Q}}$-formula

$$z \leq x + y \wedge y \leq z \wedge x < 0 \wedge y < 0.$$

DPLL(T) with Learning (CDCL)

We describe DPLL(T) by a set of rules modifying a configuration.
A configuration is a triple

$$\langle M, F, C \rangle\,,$$

where

- M (model) is a sequence of literals (that are currently set to true) interspersed with backtracking points denoted by $\square$.
- F (formula) is a formula in CNF, i. e., a set of clauses where each clause is a set of literals.
- C (conflict) is either $\top$ or a conflict clause (a set of literals). A conflict clause C is a clause with $F \Rightarrow C$ and $M \not\models C$. Thus, a conflict clause shows $M \not\models F$.

We describe the algorithm by a set of rules, which each describe a set of transitions between configurations, e. g.,

Explain $\dfrac{\langle M, F, C \cup \{\ell\}\rangle}{\langle M, F, C \cup \{\ell_1, \dots, \ell_k\}\rangle}$ where $\ell \notin C$, $\{\ell_1, \dots, \ell_k, \bar{\ell}\} \in F$, and $\bar{\ell_1}, \dots, \bar{\ell_k} \prec \bar{\ell}$ in M.

Here, $\bar{\ell_1}, \dots, \bar{\ell_k} \prec \ell$ in M means the literals $\bar{\ell_1}, \dots, \bar{\ell_k}$ occur in the sequence M before the literal ℓ (and all literals appear in M).

We describe the algorithm by a set of rules, which each describe a set of transitions between configurations, e. g.,

Explain $\dfrac{\langle M, F, C \cup \{\ell\}\rangle}{\langle M, F, C \cup \{\ell_1, \dots, \ell_k\}\rangle}$ where $\ell \notin C$, $\{\ell_1, \dots, \ell_k, \bar{\ell}\} \in F$, and $\bar{\ell_1}, \dots, \bar{\ell_k} \prec \bar{\ell}$ in M.

Here, $\bar{\ell_1}, \dots, \bar{\ell_k} \prec \ell$ in M means the literals $\bar{\ell_1}, \dots, \bar{\ell_k}$ occur in the sequence M before the literal ℓ (and all literals appear in M).

Example: for $M = P_1\bar{P}_3\bar{P}_2\bar{P}_4$, $F = \{\{P_1\}, \{P_3, \bar{P}_4\}\}$, and $C = \{P_2\}$ the transition

$$\langle M, F, \{P_2, P_4\}\rangle \longrightarrow \langle M, F, \{P_2, P_3\}\rangle$$

is possible.

Rules for CDCL (Conflict Driven Clause Learning)

Decide $\dfrac{\langle M, F, \top\rangle}{\langle M \cdot \Box \cdot \ell, F, \top\rangle}$ where $\ell \in \mathit{lit}(F)$, $\ell, \bar{\ell} \notin M$

Decide $\dfrac{\langle M, F, \top\rangle}{\langle M \cdot \Box \cdot \ell, F, \top\rangle}$ where $\ell \in \mathit{lit}(F)$, $\ell, \bar{\ell}$ $\not\in$ M

Propagate $\dfrac{\langle M, F, \top\rangle}{\langle M \cdot \ell, F, \top\rangle}$ where $\{\ell_1, \dots, \ell_k, \ell\} \in F$ and $\bar{\ell_1}, \dots, \bar{\ell_k}$ in M, $\ell, \bar{\ell}$ $\not\in$ M.

Rules for CDCL (Conflict Driven Clause Learning)

Decide $\dfrac{\langle M, F, \top\rangle}{\langle M \cdot \square \cdot \ell, F, \top\rangle}$ where $\ell \in \mathit{lit}(F)$, $\ell, \bar{\ell} \not\in M$

Propagate $\dfrac{\langle M, F, \top\rangle}{\langle M \cdot \ell, F, \top\rangle}$ where $\{\ell_1, \ldots, \ell_k, \ell\} \in F$ and $\bar{\ell}_1, \ldots, \bar{\ell}_k$ in M, $\ell, \bar{\ell} \not\in M$.

Conflict $\dfrac{\langle M, F, \top\rangle}{\langle M, F, \{\ell_1, \ldots, \ell_k\}\rangle}$ where $\{\ell_1, \ldots, \ell_k\} \in F$ and $\bar{\ell}_1, \ldots, \bar{\ell}_k$ in M.

Rules for CDCL (Conflict Driven Clause Learning)

Decide	$\dfrac{\langle M, F, \top\rangle}{\langle M \cdot \square \cdot \ell, F, \top\rangle}$	where $\ell \in lit(F)$, $\ell, \bar{\ell} \not\in M$
Propagate	$\dfrac{\langle M, F, \top\rangle}{\langle M \cdot \ell, F, \top\rangle}$	where $\{\ell_1, \ldots, \ell_k, \ell\} \in F$ and $\bar{\ell}_1, \ldots, \bar{\ell}_k$ in M, $\ell, \bar{\ell} \not\in M$.
Conflict	$\dfrac{\langle M, F, \top\rangle}{\langle M, F, \{\ell_1, \ldots, \ell_k\}\rangle}$	where $\{\ell_1, \ldots, \ell_k\} \in F$ and $\bar{\ell}_1, \ldots, \bar{\ell}_k$ in M.
Explain	$\dfrac{\langle M, F, C \cup \{\ell\}\rangle}{\langle M, F, C \cup \{\ell_1, \ldots, \ell_k\}\rangle}$	where $\ell \notin C$, $\{\ell_1, \ldots, \ell_k, \bar{\ell}\} \in F$, and $\bar{\ell}_1, \ldots, \bar{\ell}_k \prec \bar{\ell}$ in M.

Rules for CDCL (Conflict Driven Clause Learning)

Decide $\dfrac{\langle M, F, \top\rangle}{\langle M \cdot \square \cdot \ell, F, \top\rangle}$ where $\ell \in \mathit{lit}(F)$, $\ell, \bar{\ell}$ $\not\in$ M

Propagate $\dfrac{\langle M, F, \top\rangle}{\langle M \cdot \ell, F, \top\rangle}$ where $\{\ell_1, \dots, \ell_k, \ell\} \in F$ and $\bar{\ell_1}, \dots, \bar{\ell_k}$ in M, $\ell, \bar{\ell}$ $\not\in$ M.

Conflict $\dfrac{\langle M, F, \top\rangle}{\langle M, F, \{\ell_1, \dots, \ell_k\}\rangle}$ where $\{\ell_1, \dots, \ell_k\} \in F$ and $\bar{\ell_1}, \dots, \bar{\ell_k}$ in M.

Explain $\dfrac{\langle M, F, C \cup \{\ell\}\rangle}{\langle M, F, C \cup \{\ell_1, \dots, \ell_k\}\rangle}$ where $\ell \notin C$, $\{\ell_1, \dots, \ell_k, \bar{\ell}\} \in F$, and $\bar{\ell_1}, \dots, \bar{\ell_k} \prec \bar{\ell}$ in M.

Learn $\dfrac{\langle M, F, C\rangle}{\langle M, F \cup \{C\}, C\rangle}$ where $C \neq \top$, $C \notin F$.

Rules for CDCL (Conflict Driven Clause Learning)

Decide	$\dfrac{\langle M, F, \top\rangle}{\langle M \cdot \Box \cdot \ell, F, \top\rangle}$	where $\ell \in \mathit{lit}(F)$, $\ell, \bar{\ell} \not\text{in}\ M$
Propagate	$\dfrac{\langle M, F, \top\rangle}{\langle M \cdot \ell, F, \top\rangle}$	where $\{\ell_1, \dots, \ell_k, \ell\} \in F$ and $\bar{\ell}_1, \dots, \bar{\ell}_k$ in M, $\ell, \bar{\ell} \not\text{in}\ M$.
Conflict	$\dfrac{\langle M, F, \top\rangle}{\langle M, F, \{\ell_1, \dots, \ell_k\}\rangle}$	where $\{\ell_1, \dots, \ell_k\} \in F$ and $\bar{\ell}_1, \dots, \bar{\ell}_k$ in M.
Explain	$\dfrac{\langle M, F, C \cup \{\ell\}\rangle}{\langle M, F, C \cup \{\ell_1, \dots, \ell_k\}\rangle}$	where $\ell \notin C$, $\{\ell_1, \dots, \ell_k, \bar{\ell}\} \in F$, and $\bar{\ell}_1, \dots, \bar{\ell}_k \prec \bar{\ell}$ in M.
Learn	$\dfrac{\langle M, F, C\rangle}{\langle M, F \cup \{C\}, C\rangle}$	where $C \neq \top$, $C \notin F$.
Back	$\dfrac{\langle M, F, \{\ell_1, \dots, \ell_k, \ell\}\rangle}{\langle M' \cdot \ell, F, \top\rangle}$	where $\{\ell_1, \dots, \ell_k, \ell\} \in F$, $M = M' \cdot \Box \cdots \bar{\ell} \cdots$, and $\bar{\ell}_1, \dots, \bar{\ell}_k$ in M'.

Example: DPLL with Learning

$$P_1 \land (\neg P_2 \lor P_3) \land (\neg P_4 \lor P_3) \land (P_2 \lor P_4) \land (\neg P_1 \lor \neg P_4 \lor \neg P_3) \land (P_4 \lor \neg P_3)$$

The algorithm starts with $M = \epsilon$, $C = \top$ and
$F = \{\{P_1\}, \{\bar{P}_2, P_3\}, \{\bar{P}_4, P_3\}, \{P_2, P_4\}, \{\bar{P}_1, \bar{P}_4, \bar{P}_3\}, \{P_4, \bar{P}_3\}\}$.

Example: DPLL with Learning

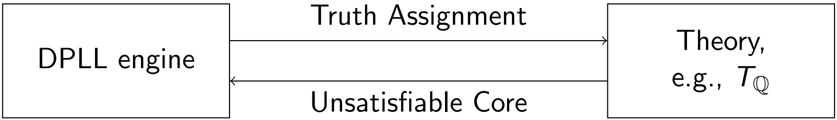
$P_1 \wedge (\neg P_2 \vee P_3) \wedge (\neg P_4 \vee P_3) \wedge (P_2 \vee P_4) \wedge (\neg P_1 \vee \neg P_4 \vee \neg P_3) \wedge (P_4 \vee \neg P_3)$

The algorithm starts with $M = \epsilon$, $C = \top$ and
$F = \{\{P_1\}, \{\bar{P}_2, P_3\}, \{\bar{P}_4, P_3\}, \{P_2, P_4\}, \{\bar{P}_1, \bar{P}_4, \bar{P}_3\}, \{P_4, \bar{P}_3\}\}$.

$\langle \epsilon, F, \top \rangle \xrightarrow{\text{Propagate}} \langle P_1, F, \top \rangle \xrightarrow{\text{Decide}} \langle P_1 \Box \bar{P}_2, F, \top \rangle \xrightarrow{\text{Propagate}}$
$\langle P_1 \Box \bar{P}_2 P_4, F, \top \rangle \xrightarrow{\text{Propagate}} \langle P_1 \Box \bar{P}_2 P_4 P_3, F, \top \rangle \xrightarrow{\text{Conflict}}$
$\langle P_1 \Box \bar{P}_2 P_4 P_3, F, \{\bar{P}_1, \bar{P}_4, \bar{P}_3\} \rangle \xrightarrow{\text{Explain}} \langle P_1 \Box \bar{P}_2 P_4 P_3, F, \{\bar{P}_1, \bar{P}_4\} \rangle \xrightarrow{\text{Learn}}$
$\langle P_1 \Box \bar{P}_2 P_4 P_3, F', \{\bar{P}_1, \bar{P}_4\} \rangle \xrightarrow{\text{Back}} \langle P_1 \bar{P}_4, F', \top \rangle \xrightarrow{\text{Propagate}}$
$\langle P_1 \bar{P}_4 P_2 P_3, F', \top \rangle \xrightarrow{\text{Conflict}} \langle P_1 \bar{P}_4 P_2 P_3, F', \{P_4, \bar{P}_3\} \rangle \xrightarrow{\text{Explain}}$
$\langle P_1 \bar{P}_4 P_2 P_3, F', \{P_4, \bar{P}_2\} \rangle \xrightarrow{\text{Explain}} \langle P_1 \bar{P}_4 P_2 P_3, F', \{P_4\} \rangle \xrightarrow{\text{Explain}}$
$\langle P_1 \bar{P}_4 P_2 P_3, F', \{\bar{P}_1\} \rangle \xrightarrow{\text{Explain}} \langle P_1 \bar{P}_4 P_2 P_3, F', \emptyset \rangle \xrightarrow{\text{Learn}}$
$\langle P_1 \bar{P}_4 P_2 P_3, F' \cup \{\emptyset\}, \emptyset \rangle$

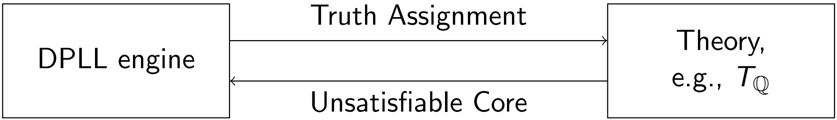
where $F' = F \cup \{\{\bar{P}_1, \bar{P}_4\}\}$.

The DPLL/CDCL algorithm is combined with a Decision Procedures for a Theory



DPLL(T): DPLL Modulo Theory

The DPLL/CDCL algorithm is combined with a Decision Procedures for a Theory



DPLL takes the propositional core of a formula,
assigns truth-values to atoms.

Theory takes a conjunctive formula (conjunction of literals),
returns a minimal unsatisfiable core.

Suppose we have a decision procedure for a conjunctive theory, e.g., Simplex Algorithm for $T_{\mathbb{Q}}$.

Given an unsatisfiable conjunction of literals $\ell_1 \land \cdots \land \ell_n$.

Minimal Unsatisfiable Core

Suppose we have a decision procedure for a conjunctive theory, e.g., Simplex Algorithm for $T_{\mathbb{Q}}$.

Given an unsatisfiable conjunction of literals $\ell_1 \wedge \cdots \wedge \ell_n$.
Find a subset $\text{UnsatCore} = \{\ell_{i_1}, \ldots, \ell_{i_m}\}$, such that

Minimal Unsatisfiable Core

Suppose we have a decision procedure for a conjunctive theory, e.g., Simplex Algorithm for $T_{\mathbb{Q}}$.

Given an unsatisfiable conjunction of literals $\ell_1 \wedge \cdots \wedge \ell_n$.
Find a subset UnsatCore $= \{\ell_{i_1}, \ldots, \ell_{i_m}\}$, such that

- $\ell_{i_1} \wedge \ldots \wedge \ell_{i_m}$ is unsatisfiable.

Minimal Unsatisfiable Core

Suppose we have a decision procedure for a conjunctive theory, e.g., Simplex Algorithm for $T_{\mathbb{Q}}$.

Given an unsatisfiable conjunction of literals $\ell_1 \wedge \dots \wedge \ell_n$.
Find a subset UnsatCore $= \{\ell_{i_1}, \dots, \ell_{i_m}\}$, such that

- $\ell_{i_1} \wedge \ldots \wedge \ell_{i_m}$ is unsatisfiable.
- For each subset of UnsatCore the conjunction is satisfiable.

Minimal Unsatisfiable Core

Suppose we have a decision procedure for a conjunctive theory, e.g., Simplex Algorithm for $T_{\mathbb{Q}}$.

Given an unsatisfiable conjunction of literals $\ell_1 \land \cdots \land \ell_n$.
Find a subset $\text{UnsatCore} = \{\ell_{i_1}, \ldots, \ell_{i_m}\}$, such that

- $\ell_{i_1} \land \ldots \land \ell_{i_m}$ is unsatisfiable.
- For each subset of UnsatCore the conjunction is satisfiable.

Possible approach: check for each literal whether it can be omitted.
$\longrightarrow$ n calls to decision procedure.

Most decision procedures can give small unsatisfiable cores for free.

Unsatisfiable Core and Conflict Clause

Theory returns an unsatisfiable core:

- a conjunction of literals from current truth assignment
- that is unsatisfible.

Unsatisfiable Core and Conflict Clause

Theory returns an unsatisfiable core:

- a conjunction of literals from current truth assignment
- that is unsatisfible.

DPLL learns conflict clauses, a disjunction of literals

- that are implied by the formula
- and in conflict to current truth assignment.

Unsatisfiable Core and Conflict Clause

Theory returns an unsatisfiable core:

- a conjunction of literals from current truth assignment
- that is unsatisfible.

DPLL learns conflict clauses, a disjunction of literals

- that are implied by the formula
- and in conflict to current truth assignment.

Thus the negation of an unsatisfiable core is a conflict clause.

DPLL(T)

The DPLL part only needs one new rule:

$$\text{TConflict} \quad \frac{\langle M, F, \top \rangle}{\langle M, F, C \rangle} \quad \begin{array}{l}\text{where } M \text{ is unsatisfiable in the theory} \\ \text{and } \neg C \text{ an unsatisfiable core of } M.\end{array}$$

Example: DPLL(T)

$$F : \; y \geq 1 \land (x \geq 0 \to y \leq 0) \land (x \leq 1 \to y \leq 0)$$

$$F: \; y \geq 1 \land (x \geq 0 \rightarrow y \leq 0) \land (x \leq 1 \rightarrow y \leq 0)$$

Atomic propositions:

$$P_1 : y \geq 1 \qquad\qquad P_2 : x \geq 0$$
$$P_3 : y \leq 0 \qquad\qquad P_4 : x \leq 1$$

Example: DPLL(T)

$$F : \; y \geq 1 \land (x \geq 0 \rightarrow y \leq 0) \land (x \leq 1 \rightarrow y \leq 0)$$

Atomic propositions:

$$P_1 : y \geq 1 \qquad\qquad P_2 : x \geq 0$$
$$P_3 : y \leq 0 \qquad\qquad P_4 : x \leq 1$$

Propositional core of F in CNF:

$$F_0 : (P_1) \land (\neg P_2 \lor P_3) \land (\neg P_4 \lor P_3)$$

Running DPLL(T)

$F_0 : \{\{P_1\}, \{\bar{P}_2, P_3\}, \{\bar{P}_4, P_3\}\}$
$P_1 : y \geq 1 \quad P_2 : x \geq 0 \quad P_3 : y \leq 0 \quad P_4 : x \leq 1$

Running DPLL(T)

$F_0 : \{\{P_1\}, \{\bar{P}_2, P_3\}, \{\bar{P}_4, P_3\}\}$
$P_1 : y \geq 1 \quad P_2 : x \geq 0 \quad P_3 : y \leq 0 \quad P_4 : x \leq 1$

$\langle \epsilon, F_0, \top \rangle \xrightarrow{\text{Propagate}} \langle P_1, F_0, \top \rangle \xrightarrow{\text{Decide}} \langle P_1 \Box P_3, F_0, \top \rangle \xrightarrow{\text{TConflict}}$
$\langle P_1 \Box P_3, F_0, \{\bar{P}_1, \bar{P}_3\} \rangle \xrightarrow{\text{Learn}} \langle P_1 \Box P_3, F_1, \{\bar{P}_1, \bar{P}_3\} \rangle \xrightarrow{\text{Back}}$
$\langle P_1 \bar{P}_3, F_1, \top \rangle \xrightarrow{\text{Propagate}} \langle P_1 \bar{P}_3 \bar{P}_2, F_1, \top \rangle \xrightarrow{\text{Propagate}}$
$\langle P_1 \bar{P}_3 \bar{P}_2 \bar{P}_4, F_1, \top \rangle \xrightarrow{\text{TConflict}} \langle P_1 \bar{P}_3 \bar{P}_2 \bar{P}_4, F_1, \{P_2, P_4\} \rangle \xrightarrow{\text{Explain}}$
$\langle P_1 \bar{P}_3 \bar{P}_2 \bar{P}_4, F_1, \{P_2, P_3\} \rangle \xrightarrow{\text{Explain}} \langle P_1 \bar{P}_3 \bar{P}_2 \bar{P}_4, F_1, \{P_3\} \rangle \xrightarrow{\text{Explain}}$
$\langle P_1 \bar{P}_3 \bar{P}_2 \bar{P}_4, F_1, \{\bar{P}_1\} \rangle \xrightarrow{\text{Explain}} \langle P_1 \bar{P}_3 \bar{P}_2 \bar{P}_4, F_1, \emptyset \rangle \xrightarrow{\text{Learn}}$
$\langle P_1 \bar{P}_3 \bar{P}_2 \bar{P}_4, F_1 \cup \{\emptyset\}, \emptyset \rangle$

where $F_1 := F_0 \cup \{\{\bar{P}_1, \bar{P}_3\}\}$

No further step is possible; the formula F is unsatisfiable.

Correctness of DPLL(T)

Theorem (Correctness of DPLL(T))

Let F be a Σ-formula and F' its propositional core. Let

$$\langle \epsilon, F', \top \rangle = \langle M_0, F_0, C_0 \rangle \longrightarrow \ldots \longrightarrow \langle M_n, F_n, C_n \rangle$$

be a maximal sequence of rule application of DPLL(T).

Then F is T-satisfiable iff C_n is $\top$.

Correctness of DPLL(T)

Theorem (Correctness of DPLL(T))

Let F be a Σ-formula and F' its propositional core. Let

$$\langle \epsilon, F', \top \rangle = \langle M_0, F_0, C_0 \rangle \longrightarrow \ldots \longrightarrow \langle M_n, F_n, C_n \rangle$$

be a maximal sequence of rule application of DPLL(T).

Then F is T-satisfiable iff C_n is $\top$.

Before proving the theorem, we note some important invariants:

Correctness of DPLL(T)

Theorem (Correctness of DPLL(T))

Let F be a Σ-formula and F' its propositional core. Let

$$\langle \epsilon, F', \top \rangle = \langle M_0, F_0, C_0 \rangle \longrightarrow \ldots \longrightarrow \langle M_n, F_n, C_n \rangle$$

be a maximal sequence of rule application of DPLL(T).

Then F is T-satisfiable iff C_n is $\top$.

Before proving the theorem, we note some important invariants:

- M_i never contains a literal more than once.
- M_i never contains ℓ and $\overline{\ell}$.

Correctness of DPLL(T)

Theorem (Correctness of DPLL(T))

Let F be a Σ-formula and F' its propositional core. Let

$$\langle \epsilon, F', \top \rangle = \langle M_0, F_0, C_0 \rangle \longrightarrow \ldots \longrightarrow \langle M_n, F_n, C_n \rangle$$

be a maximal sequence of rule application of DPLL(T).

Then F is T-satisfiable iff C_n is $\top$.

Before proving the theorem, we note some important invariants:

- M_i never contains a literal more than once.
- M_i never contains ℓ and $\bar{\ell}$.
- Every $\square$ in M_i is followed immediately by a literal.

Correctness of DPLL(T)

Theorem (Correctness of DPLL(T))

Let F be a Σ-formula and F' its propositional core. Let

$$\langle \epsilon, F', \top \rangle = \langle M_0, F_0, C_0 \rangle \longrightarrow \ldots \longrightarrow \langle M_n, F_n, C_n \rangle$$

be a maximal sequence of rule application of DPLL(T).

Then F is T-satisfiable iff C_n is $\top$.

Before proving the theorem, we note some important invariants:

- M_i never contains a literal more than once.
- M_i never contains ℓ and $\overline{\ell}$.
- Every $\square$ in M_i is followed immediately by a literal.
- If $C_i = \{\ell_1, \ldots, \ell_k\}$ then $\overline{\ell_1}, \ldots, \overline{\ell_k}$ in M.

Theorem (Correctness of DPLL(T))

Let F be a Σ-formula and F' its propositional core. Let

$$\langle \epsilon, F', \top \rangle = \langle M_0, F_0, C_0 \rangle \longrightarrow \ldots \longrightarrow \langle M_n, F_n, C_n \rangle$$

be a maximal sequence of rule application of DPLL(T).

Then F is T-satisfiable iff C_n is $\top$.

Before proving the theorem, we note some important invariants:

- M_i never contains a literal more than once.
- M_i never contains ℓ and $\bar{\ell}$.
- Every $\Box$ in M_i is followed immediately by a literal.
- If $C_i = \{\ell_1, \ldots, \ell_k\}$ then $\bar{\ell_1}, \ldots, \bar{\ell_k}$ in M.
- C_i is always implied by F_i (or the theory).

Theorem (Correctness of DPLL(T))

Let F be a Σ-formula and F' its propositional core. Let

$$\langle \epsilon, F', \top \rangle = \langle M_0, F_0, C_0 \rangle \longrightarrow \ldots \longrightarrow \langle M_n, F_n, C_n \rangle$$

be a maximal sequence of rule application of DPLL(T).

Then F is T-satisfiable iff C_n is $\top$.

Before proving the theorem, we note some important invariants:

- M_i never contains a literal more than once.
- M_i never contains ℓ and $\bar{\ell}$.
- Every $\square$ in M_i is followed immediately by a literal.
- If $C_i = \{\ell_1, \ldots, \ell_k\}$ then $\bar{\ell}_1, \ldots, \bar{\ell}_k$ in M.
- C_i is always implied by F_i (or the theory).
- F is equivalent to F_i for all steps i of the computation.

Correctness of DPLL(T)

Theorem (Correctness of DPLL(T))

Let F be a Σ-formula and F' its propositional core. Let

$$\langle \epsilon, F', \top \rangle = \langle M_0, F_0, C_0 \rangle \longrightarrow \ldots \longrightarrow \langle M_n, F_n, C_n \rangle$$

be a maximal sequence of rule application of DPLL(T).

Then F is T-satisfiable iff C_n is $\top$.

Before proving the theorem, we note some important invariants:

- M_i never contains a literal more than once.
- M_i never contains ℓ and $\overline{\ell}$.
- Every $\Box$ in M_i is followed immediately by a literal.
- If $C_i = \{\ell_1, \ldots, \ell_k\}$ then $\overline{\ell_1}, \ldots, \overline{\ell_k}$ in M.
- C_i is always implied by F_i (or the theory).
- F is equivalent to F_i for all steps i of the computation.
- If a literal ℓ in M is not immediately preceded by $\Box$, then F contains a clause $\{\ell, \ell_1, \ldots, \ell_k\}$ and $\overline{\ell_1}, \ldots, \overline{\ell_k} \prec \ell$ in M.

Correctness proof

Proof: If the sequence ends with $\langle M_n, F_n, \top \rangle$ and there is no rule applicable, then:

Correctness proof

Proof: If the sequence ends with $\langle M_n, F_n, \top \rangle$ and there is no rule applicable, then:

- Since Decide is not applicable, all literals of F_n appear in M_n either positively or negatively.

Correctness proof

Proof: If the sequence ends with $\langle M_n, F_n, \top\rangle$ and there is no rule applicable, then:

- Since Decide is not applicable, all literals of F_n appear in M_n either positively or negatively.
- Since Conflict is not applicable, for each clause at least one literal appears in M_n positively.

Correctness proof

Proof: If the sequence ends with $\langle M_n, F_n, \top \rangle$ and there is no rule applicable, then:

- Since Decide is not applicable, all literals of F_n appear in M_n either positively or negatively.
- Since Conflict is not applicable, for each clause at least one literal appears in M_n positively.
- Since TConflict is not applicable, the conjunction of truth assignments of M_n is satisfiable by a model I.

Correctness proof

Proof: If the sequence ends with $\langle M_n, F_n, \top\rangle$ and there is no rule applicable, then:

- Since Decide is not applicable, all literals of F_n appear in M_n either positively or negatively.
- Since Conflict is not applicable, for each clause at least one literal appears in M_n positively.
- Since TConflict is not applicable, the conjunction of truth assignments of M_n is satisfiable by a model I.

Thus, I is a model for F_n, which is equivalent to F.

Correctness proof

Proof: If the sequence ends with $\langle M_n, F_n, \top\rangle$ and there is no rule applicable, then:

- Since Decide is not applicable, all literals of F_n appear in M_n either positively or negatively.
- Since Conflict is not applicable, for each clause at least one literal appears in M_n positively.
- Since TConflict is not applicable, the conjunction of truth assignments of M_n is satisfiable by a model I.

Thus, I is a model for F_n, which is equivalent to F.

If the sequence ends with $\langle M_n, F_n, C_n\rangle$ with $C_n \neq \top$.

Proof: If the sequence ends with $\langle M_n, F_n, \top\rangle$ and there is no rule applicable, then:

- Since Decide is not applicable, all literals of F_n appear in M_n either positively or negatively.
- Since Conflict is not applicable, for each clause at least one literal appears in M_n positively.
- Since TConflict is not applicable, the conjunction of truth assignments of M_n is satisfiable by a model I.

Thus, I is a model for F_n, which is equivalent to F.

If the sequence ends with $\langle M_n, F_n, C_n\rangle$ with $C_n \neq \top$.
Assume $C_n = \{\ell_1, \dots, \ell_k, \ell\} \neq \emptyset$. W.l.o.g., $\bar{\ell}_1, \dots, \bar{\ell}_k \prec \ell$. Then:

Correctness proof

Proof: If the sequence ends with $\langle M_n, F_n, \top \rangle$ and there is no rule applicable, then:

- Since Decide is not applicable, all literals of F_n appear in M_n either positively or negatively.
- Since Conflict is not applicable, for each clause at least one literal appears in M_n positively.
- Since TConflict is not applicable, the conjunction of truth assignments of M_n is satisfiable by a model I.

Thus, I is a model for F_n, which is equivalent to F.

If the sequence ends with $\langle M_n, F_n, C_n \rangle$ with $C_n \neq \top$.
Assume $C_n = \{\ell_1, \dots, \ell_k, \ell\} \neq \emptyset$. W.l.o.g., $\bar{\ell}_1, \dots, \bar{\ell}_k \prec \ell$. Then:

- Since Learn is not applicable, $C_n \in F_n$.

Correctness proof

Proof: If the sequence ends with $\langle M_n, F_n, \top \rangle$ and there is no rule applicable, then:

- Since Decide is not applicable, all literals of F_n appear in M_n either positively or negatively.
- Since Conflict is not applicable, for each clause at least one literal appears in M_n positively.
- Since TConflict is not applicable, the conjunction of truth assignments of M_n is satisfiable by a model I.

Thus, I is a model for F_n, which is equivalent to F.

If the sequence ends with $\langle M_n, F_n, C_n \rangle$ with $C_n \neq \top$.
Assume $C_n = \{\ell_1, \dots, \ell_k, \ell\} \neq \emptyset$. W.l.o.g., $\bar{\ell}_1, \dots, \bar{\ell}_k \prec \ell$. Then:

- Since Learn is not applicable, $C_n \in F_n$.
- Since Explain is not applicable $\bar{\ell}$ must be immediately preceded by $\square$.

Correctness proof

Proof: If the sequence ends with $\langle M_n, F_n, \top\rangle$ and there is no rule applicable, then:

- Since Decide is not applicable, all literals of F_n appear in M_n either positively or negatively.
- Since Conflict is not applicable, for each clause at least one literal appears in M_n positively.
- Since TConflict is not applicable, the conjunction of truth assignments of M_n is satisfiable by a model I.

Thus, I is a model for F_n, which is equivalent to F.

If the sequence ends with $\langle M_n, F_n, C_n\rangle$ with $C_n \neq \top$.
Assume $C_n = \{\ell_1, \dots, \ell_k, \ell\} \neq \emptyset$. W.l.o.g., $\overline{\ell_1}, \dots, \overline{\ell_k} \prec \ell$. Then:

- Since Learn is not applicable, $C_n \in F_n$.
- Since Explain is not applicable $\overline{\ell}$ must be immediately preceded by $\Box$.
- However, then Back is applicable, contradiction!

Correctness proof

Proof: If the sequence ends with $\langle M_n, F_n, \top \rangle$ and there is no rule applicable, then:

- Since Decide is not applicable, all literals of F_n appear in M_n either positively or negatively.
- Since Conflict is not applicable, for each clause at least one literal appears in M_n positively.
- Since TConflict is not applicable, the conjunction of truth assignments of M_n is satisfiable by a model I.

Thus, I is a model for F_n, which is equivalent to F.

If the sequence ends with $\langle M_n, F_n, C_n \rangle$ with $C_n \neq \top$.
Assume $C_n = \{\ell_1, \dots, \ell_k, \ell\} \neq \emptyset$. W.l.o.g., $\bar{\ell}_1, \dots, \bar{\ell}_k \prec \ell$. Then:

- Since Learn is not applicable, $C_n \in F_n$.
- Since Explain is not applicable $\bar{\ell}$ must be immediately preceded by $\square$.
- However, then Back is applicable, contradiction!

Therefore, the assumption was wrong and $C_n = \emptyset (= \bot)$.
Since F implies C_n, F is not satisfiable.

Total Correctness of DPLL with Learning

Theorem (Termination of DPLL)

Let F be a propositional formula. Then every sequence

$$\langle \epsilon, F, \top \rangle = \langle M_0, F_0, C_0 \rangle \longrightarrow \langle M_1, F_1, C_1 \rangle \longrightarrow \ldots$$

terminates.

Proof of Total Correctness

We define some well-ordering on the domains:

Proof of Total Correctness

We define some well-ordering on the domains:

- We define $M \prec M'$ if $M\Box\Box$ comes lexicographically before $M'\Box\Box$, where every literal is considered to be smaller than $\Box$.
  Example: $\ell_1\ell_2(\Box\Box) \prec \ell_1\Box\bar{\ell}_2\ell_3(\Box\Box) \prec \ell_1\Box\bar{\ell}_2(\Box\Box) \prec \ell_1(\Box\Box)$

Proof of Total Correctness

We define some well-ordering on the domains:

- We define $M \prec M'$ if $M\square\square$ comes lexicographically before $M'\square\square$, where every literal is considered to be smaller than $\square$.
  Example: $\ell_1\ell_2(\square\square) \prec \ell_1\square\bar{\ell_2}\ell_3(\square\square) \prec \ell_1\square\bar{\ell_2}(\square\square) \prec \ell_1(\square\square)$
- For a sequence $M = \bar{\ell_1}\ldots\bar{\ell_n}$, the conflict clauses are ordered by:
  $C \prec_M C'$, iff $C \neq \top, C' = \top$ or for some $k \leq n$:
  $C \cap \{\ell_{k+1},\ldots,\ell_n\} = C' \cap \{\ell_{k+1},\ldots\ell_n\}$ and $\ell_k \notin C, \ell_k \in C'$.
  Example: $\emptyset \prec_{\bar{\ell_1}\bar{\ell_2}\bar{\ell_3}} \{\ell_2\} \prec_{\bar{\ell_1}\bar{\ell_2}\bar{\ell_3}} \{\ell_1,\ell_3\} \prec_{\bar{\ell_1}\bar{\ell_2}\bar{\ell_3}} \{\ell_2,\ell_3\} \prec_{\bar{\ell_1}\bar{\ell_2}\bar{\ell_3}} \top$

These are well-orderings, because the domains are finite.

We define some well-ordering on the domains:

- We define $M \prec M'$ if $M\Box\Box$ comes lexicographically before $M'\Box\Box$, where every literal is considered to be smaller than $\Box$.
  Example: $\ell_1\ell_2(\Box\Box) \prec \ell_1\Box\bar{\ell}_2\ell_3(\Box\Box) \prec \ell_1\Box\bar{\ell}_2(\Box\Box) \prec \ell_1(\Box\Box)$
- For a sequence $M = \bar{\ell}_1 \dots \bar{\ell}_n$, the conflict clauses are ordered by:
  $C \prec_M C'$, iff $C \neq \top, C' = \top$ or for some $k \leq n$:
  $C \cap \{\ell_{k+1}, \dots, \ell_n\} = C' \cap \{\ell_{k+1}, \dots \ell_n\}$ and $\ell_k \notin C, \ell_k \in C'$.
  Example: $\emptyset \prec_{\bar{\ell}_1\bar{\ell}_2\bar{\ell}_3} \{\ell_2\} \prec_{\bar{\ell}_1\bar{\ell}_2\bar{\ell}_3} \{\ell_1, \ell_3\} \prec_{\bar{\ell}_1\bar{\ell}_2\bar{\ell}_3} \{\ell_2, \ell_3\} \prec_{\bar{\ell}_1\bar{\ell}_2\bar{\ell}_3} \top$

These are well-orderings, because the domains are finite.

Termination Proof: Every rule application decreases the value of $\langle M_i, F_i, C_i\rangle$ according to the well-ordering:

$$\langle M, F, C\rangle \prec \langle M', F', C'\rangle, \text{ iff} \begin{cases} M \prec M', \\ \text{or } M = M', C \prec_M C', \\ \text{or } M = M', C = C', C \in F, C \notin F'. \end{cases}$$