CS:4350 Logic in Computer Science

Satisfiability Modulo Theories

Cesare Tinelli

Spring 2022

The University of Iowa

Credits

Some of these slides are based on slides originally developed by **Albert Oliveras** at the Technical University of Barcelona and **Dejan Jovanovic** at the New York University. Adapted by permission.

Introduction

Historically:

Automated logical reasoning achieved through uniform theorem-proving procedures for First Order Logic (e.g., resolution, tableaux calculi)

Some success:

However, uniform proof procedures for FOL are not always the best *compromise* between expressiveness and efficiency

Introduction

Historically:

Automated logical reasoning achieved through uniform theorem-proving procedures for First Order Logic (e.g., resolution, tableaux calculi)

Some success:

However, uniform proof procedures for FOL are not always the best *compromise* between expressiveness and efficiency

Introduction

Last 20 years: R&D has focused on

- expressive enough decidable fragments of various logics
- incorporating domain-specific reasoning, e.g., on:
  - temporal reasoning
  - arithmetic reasoning
  - equality reasoning
  - reasoning about certain data structures (arrays, lists, finite sets, . . .)
- combining specialized reasoners modularly

Introduction

Two successful examples of this trend:

SAT: propositional formalization,
boolean reasoning

+ high degree of efficiency
− expressive (all NP-complete problems) but involved encodings

SMT: first-order formalization,
boolean + domain-specific reasoning

+ improves expressivity and scalability
− some (but acceptable) loss of efficiency

Introduction

Two successful examples of this trend:

SAT: propositional formalization,
boolean reasoning

- \+ high degree of efficiency
- − expressive (all NP-complete problems) but involved encodings

SMT: first-order formalization,
boolean + domain-specific reasoning

- \+ improves expressivity and scalability
- − some (but acceptable) loss of efficiency

Introduction

Two successful examples of this trend:

SAT: propositional formalization,
boolean reasoning

- $+$ high degree of efficiency
- $-$ expressive (all NP-complete problems) but involved encodings

SMT: first-order formalization,
boolean + domain-specific reasoning

- $+$ improves expressivity and scalability
- $-$ some (but acceptable) loss of efficiency

Satisfiability Modulo Theories (SMT): Motivation

Some problems are more naturally expressed in logics other than propositional or plain first-order logic

> **Ex:** software verification needs efficient reasoning about equality, arithmetic, memory, data structures, . . .

One needs to check the satisfiability of formulas with respect to, or modulo one or more *background theories*

Satisfiability Modulo Theories (SMT): Motivation

Some problems are more naturally expressed in logics other than propositional or plain first-order logic

> **Ex:** software verification needs efficient reasoning about equality, arithmetic, memory, data structures, ...

One needs to check the satisfiability of formulas with respect to, or modulo one or more *background theories*

The Basic SMT Problem

Determining the satisfiability of a logical formula wrt some combination T of background theories

Example

$$n > 3 * m + 1 \ \wedge \ (\, f\,(n) \leq head\,(l_1) \ \vee \ l_2 = f\,(n) :: l_1 \,)$$

Linear Arithm. (LIA)

Equality (EUF)

Lists (ADT)

The Basic SMT Problem

Determining the satisfiability of a logical formula wrt some combination T of background theories

Example

$$n > 3 * m + 1 \ \wedge \ (f(n) \leq \mathit{head}(l_1) \ \vee \ l_2 = f(n) :: l_1)$$

Linear Arithm. (LIA)	Equality (EUF)	Lists (ADT)

The Basic SMT Problem

Determining the satisfiability of a logical formula wrt some combination T of background theories

Example

$$n > 3 * m + 1 \;\land\; (\, f(n) \le head(l_1) \;\lor\; l_2 = f(n) :: l_1 \,)$$

Linear Arithm.
(LIA)

Equality
(EUF)

Lists
(ADT)

The Basic SMT Problem

Determining the satisfiability of a logical formula wrt some combination T of background theories

Example

$$n > 3 * m + 1 \wedge (f(n) \leq head(l_1) \vee l_2 = f(n) :: l_1)$$

Linear Arithm. (LIA)

Equality (EUF)

Lists (ADT)

The Basic SMT Problem

Determining the satisfiability of a logical formula wrt some combination T of background theories

Example

$$n > 3 * m + 1 \;\wedge\; (\, f(n) \leq head(l_1) \;\vee\; l_2 = f(n) :: l_1 \,)$$

Linear Arithm. (LIA)

Equality (EUF)

Lists (ADT)

Satisfiability Modulo Theories

Given

1. a (many-sorted) logical theory T
2. a first-order formula F

is F satisfiable in a model of T?

SMT Semantics

The theory T can be defined

- axiomatically, as set $\mathbf{A}$ of first-order sentences
- algebraically, as a class $\mathcal{C}$ of interpretations

We call *models of* T the interpretations that satisfy $\mathbf{A}$ / are in $\mathcal{C}$

Some Background Theories of Interest

Uninterpreted Functions	$x = y \rightarrow f(x) = f(y)$
Integer/Real Arithmetic	$2x + y = 0 \wedge 2x - y = 4 \rightarrow x = 1$
Floating Point Arithmetic	$x + 1 \neq \mathrm{NaN} \wedge x < \infty \rightarrow x + 1 > x$
Bit-vectors	$4 \circ (x \gg 2) = (x \mathbin{\&} {\sim}3) + 1$
Strings and RegExs	$x = y \cdot z \wedge z \in ab^* \rightarrow \lvert x\rvert > \lvert y\rvert$
Arrays	$i = j \rightarrow \mathrm{read}(\mathrm{write}(a, i, x), j) = x$
Algebraic Data Types	$x \neq \mathrm{Leaf} \rightarrow \exists\, l, r : \mathrm{Tree}(\alpha).\ \exists\, a : \alpha.\ x = \mathrm{Node}(l, a, r)$
Finite Sets	$e_1 \in x \wedge e_2 \in x \setminus e_1 \rightarrow \exists y, z : \mathrm{Set}(\alpha).\ \lvert y\rvert = \lvert z\rvert \wedge x = y \cup z \wedge y \neq \emptyset$
Finite Relations	$(x, y) \in r \wedge (y, z) \in r \rightarrow (x, z) \in r \bowtie s$

Equality and Uninterpreted Functions (EUF)

Simplest first-order theory with equality, applications of uninterpreted functions, and variables of uninterpreted sorts

For all sorts σ, σ' and function symbols $f : \sigma \to \sigma'$

Reflexivity: $\forall x : \sigma \; x = x$
Symmetry: $\forall x : \sigma \; (x = y \to y = x)$
Transitivity: $\forall x, y : \sigma \; (x = y \land y = z \to x = z)$
Congruence: $\forall \boldsymbol{x}, \boldsymbol{y} : \boldsymbol{\sigma} \; (\boldsymbol{x} = \boldsymbol{y} \to f(\boldsymbol{x}) = f(\boldsymbol{y}))$

Example

$$f(f(f(a))) = b \;\land\; g(f(a), b) = a \;\land\; f(a) \neq a$$

Arrays

Operates over sorts $\mathsf{Array}(\sigma_i, \sigma_e)$, σ_i, σ_e and function symbols

$$\begin{aligned}\mathsf{read} &: \mathsf{Array}(\sigma_i, \sigma_e) \times \sigma_i \to \sigma_e \\ \mathsf{write} &: \mathsf{Array}(\sigma_i, \sigma_e) \times \sigma_i \times \sigma \to \mathsf{Array}(\sigma_i, \sigma_e)\end{aligned}$$

For any index sort σ_i and element sort σ_e

Read-Over-Write-1: $\forall a, i, e.\ \mathsf{read}(\mathsf{write}(a, i, e), i) = e$
Read-Over-Write-2: $\forall a, i, j, e.\ (i \neq j \to \mathsf{read}(\mathsf{write}(a, i, e), j) = \mathsf{read}(a, j))$
Extensionality: $\forall a, b, i.\ (a \neq b \to \exists i.\ \mathsf{read}(a, i) \neq \mathsf{read}(b, i))$

Example

$$\begin{aligned}&\mathsf{write}(\mathsf{write}(a, i, \mathsf{read}(a, j)), j, \mathsf{read}(a, i)) = \\ &\mathsf{write}(\mathsf{write}(a, j, \mathsf{read}(a, i)), i, \mathsf{read}(a, j))\end{aligned}$$

Arithmetics

Restricted fragments, over the reals or the integers, support *efficient* methods:

- Bounds: $x \bowtie k$ with $\bowtie \in \{<, >, \leq, \geq, =\}$
- Difference constraints: $x - y \bowtie k$, with $\bowtie \in \{<, >, \leq, \geq, =\}$
- UTVPI: $\pm x \pm y \bowtie k$, with $\bowtie \in \{<, >, \leq, \geq, =\}$
- Linear arithmetic, e.g: $2x - 3y + 4z \leq 5$
- Non-linear arithmetic, e.g: $2xy + 4xz^2 - 5y \leq 10$

Algebraic Data Types

Family of user-definable theories

Example

$$
\begin{aligned}
\mathrm{Color} &:= \mathrm{red} \mid \mathrm{green} \mid \mathrm{blue} \\
\mathrm{List}(\alpha) &:= \mathrm{nil} \mid (\mathrm{head} : \alpha) :: (\mathrm{tail} : \mathrm{List}(\alpha))
\end{aligned}
$$

Distinctiveness: $\forall h, t \ \mathrm{nil} \neq h :: t$

Exhaustiveness: $\forall l \ (l = \mathrm{nil} \lor \exists h, t.\, h :: t)$

Injectivity: $\forall h_1, h_2, t_1, t_2 \ (h_1 :: t_1 = h_2 :: t_2 \rightarrow h_1 = h_2 \land t_1 = t_2)$

Selectors: $\forall h, t \ (\mathrm{head}(h :: t) = h \land \mathrm{tail}(h :: t) = t)$

Non-circularity: $\forall l, x_1, \ldots, x_n \ l \neq x_1 :: \cdots :: x_n :: l$

Other Interesting Theories

- Floating point arithmetic
- Strings and regular expressions
- Sequences
- Finite sets with cardinality
- Finite multisets
- Finite relations
- Transcendental Functions
- Ordinary differential equations
- …

Reasoning Modulo Theories, Example

$$f(\mathrm{read}(\mathrm{write}(a, i, 3), c - 2)) \neq f(c - i + 1)$$
$$\wedge\ l_1 = c :: d :: e :: \mathrm{nil}$$
$$\wedge\ i + 2 = \mathrm{head}(l_1 \mathbin{@} l_2)$$

Reasoning Modulo Theories, Example

$$
\begin{aligned}
& f(\mathrm{read}(\mathrm{write}(a, i, 3), c - 2)) \neq f(c - i + 1) \\
\wedge\ & l_1 = c :: d :: e :: \mathrm{nil} \\
\wedge\ & i + 2 = \mathrm{head}(l_1 \mathbin{@} l_2)
\end{aligned}
$$

Theory of Linear Integer Arithmetic

Reasoning Modulo Theories, Example

$$f(\mathrm{read}(\mathrm{write}(a, i, 3), c - 2)) \neq f(c - i + 1)$$
$$\wedge\ l_1 = c :: d :: e :: \mathrm{nil}$$
$$\wedge\ i + 2 = \mathrm{head}(l_1 \mathbin{@} l_2)$$

Theory of Algebraic Data Types

Reasoning Modulo Theories, Example

$$f(\mathrm{read}(\mathrm{write}(a, i, 3), c - 2)) \neq f(c - i + 1)$$
$$\wedge\ l_1 = c :: d :: e :: \mathrm{nil}$$
$$\wedge\ i + 2 = \mathrm{head}(l_1 \mathbin{@} l_2)$$

Theory of Arrays

Reasoning Modulo Theories, Example

$$f(\mathrm{read}(\mathrm{write}(a, i, 3), c - 2)) \neq f(c - i + 1)$$
$$\wedge \;\; l_1 = c :: d :: e :: \mathrm{nil}$$
$$\wedge \;\; i + 2 = \mathrm{head}(l_1 \mathbin{@} l_2)$$

Theory of Equality and Uninterpreted Functions

Reasoning Modulo Theories, Example

$$f(\mathrm{read}(\mathrm{write}(a, i, 3), c - 2)) \neq f(c - i + 1)$$
$$\wedge \;\; l_1 = c :: d :: e :: \mathrm{nil}$$
$$\wedge \;\; i + 2 = \mathrm{head}(l_1 \mathbin{@} l_2)$$

Reasoning Modulo Theories, Example

$$f(\text{read}(\text{write}(a, i, 3), c-2)) \neq f(c-i+1)$$

$$\wedge \;\; l_1 = c :: d :: e :: \text{nil}$$

$$\wedge \;\; i+2 = \text{head}(l_1 \mathbin{@} l_2)$$

$$l_1 = c :: d :: e :: \text{nil} \;\models_{\text{ADT}}\; \text{head}(l_1 \mathbin{@} l_2) = c$$

Reasoning Modulo Theories, Example

$$f(\text{read}(\text{write}(a, i, 3), c - 2)) \neq f(c - i + 1)$$

$$\wedge \;\; l_1 = c :: d :: e :: \text{nil}$$

$$\wedge \;\; i + 2 = c$$

Reasoning Modulo Theories, Example

$$f(\mathrm{read}(\mathrm{write}(a, i, 3), c - 2)) \neq f(c - i + 1)$$
$$\wedge\ l_1 = c :: d :: e :: \mathrm{nil}$$
$$\wedge\ i + 2 = c$$

$$i + 2 = c \models_{\mathrm{EUF}} \begin{array}{rcl} c - 2 & = & i + 2 - 2 \\ c - i + 1 & = & i + 2 - i + 1 \end{array}$$

Reasoning Modulo Theories, Example

$$f(\mathrm{read}(\mathrm{write}(a, i, 3), i + 2 - 2)) \neq f(i + 2 - i + 1)$$

$$\wedge \ \ l_1 = c :: d :: e :: \mathrm{nil}$$

$$\wedge \ \ c = i + 2$$

Reasoning Modulo Theories, Example

$$f(\mathrm{read}(\mathrm{write}(a, i, 3), i + 2 - 2)) \neq f(i + 2 - i + 1)$$

$$\wedge\ l_1 = c :: d :: e :: \mathrm{nil}$$

$$\wedge\ c = i + 2$$

$$\models_{\mathrm{LIA}} \quad \begin{array}{rcl} i + 2 - 2 & = & i \\ i + 2 - i + 1 & = & 3 \end{array}$$

Reasoning Modulo Theories, Example

$f(\text{read}(\text{write}(a, i, 3), i)) \neq f(3)$

$\wedge\ l_1 = c :: d :: e :: \text{nil}$

$\wedge\ c = i + 2$

Reasoning Modulo Theories, Example

$$f(\mathrm{read}(\mathrm{write}(a, i, 3), i)) \neq f(3)$$
$$\wedge \; l_1 = c :: d :: e :: \mathrm{nil}$$
$$\wedge \; c = i + 2$$

$$\models_{\mathsf{A}} \; \mathrm{read}(\mathrm{write}(a, i, 3), i) = 3$$

Reasoning Modulo Theories, Example

$$f(3) \neq f(3)$$
$$\wedge \ \ l_1 = c :: d :: e :: \mathrm{nil}$$
$$\wedge \ \ c = i + 2$$

Reasoning Modulo Theories, Example

$f(3) \neq f(3)$

$\wedge \; l_1 = c :: d :: e :: \mathrm{nil}$

$\wedge \; c = i + 2$

$$f(3) \neq f(3) \models_{\mathrm{EUF}} \bot$$

Reasoning Modulo Theories, Example

$$f(3) \neq f(3)$$
$$\wedge \;\; l_1 = c :: d :: e :: \mathrm{nil}$$
$$\wedge \;\; c = i + 2$$

Unsatisfiable!

Solving SMT Problems

Fact: Many theories have efficient decision procedures
for the satisfiability of conjunctions of literals

Problem: In practice, we need to deal with

1. arbitrary Boolean combinations of literals
2. literals over more than one theory
3. formulas with quantifiers

Solving SMT Problems

Fact: Many theories have efficient decision procedures
for the satisfiability of conjunctions of literals

Problem: In practice, we need to deal with

1. arbitrary Boolean combinations of literals
2. literals over more than one theory
3. formulas with quantifiers

Solving SMT Problems

Fact: Many theories have efficient decision procedures
for the satisfiability of conjunctions of literals

Problem: In practice, we need to deal with

1. arbitrary Boolean combinations of literals
2. literals over more than one theory
3. formulas with quantifiers

Satisfiability Modulo a Theory T

$F, F_1, \ldots, F_n$ formulas, T a theory

F is *satisfiable in* T, or T*-satisfiable*, if it is satisfiable in a model of T

F is *unsatisfiable in* T, or T*-unsatisfiable*, if it is not T satisfiable

$F_1, \ldots, F_n$ *entail* F *in* T, or T*-entail* F, written $F_1, \ldots, F_n \models_T F$
if $F_1 \wedge \cdots \wedge F_n \wedge F$ is T-unsatisfiable

Satisfiability Modulo a Theory T

Note:

The T-satisfiability of quantifier-free formulas is decidable iff the T-satisfiability of conjunctions/sets of literals is decidable

Satisfiability Modulo a Theory T

Note:

The T-satisfiability of quantifier-free formulas is decidable iff the T-satisfiability of conjunctions/sets of literals is decidable

(Convert the formula in DNF and check if any of its disjuncts is T-sat)

Satisfiability Modulo a Theory T

Note:

The T-satisfiability of quantifier-free formulas is decidable iff the T-satisfiability of conjunctions/sets of literals is decidable

Problem: In practice, dealing with Boolean combinations of literals is as hard as in propositional logic

Satisfiability Modulo a Theory T

Note:

The T-satisfiability of quantifier-free formulas is decidable iff
the T-satisfiability of conjunctions/sets of literals is decidable

Problem: In practice, dealing with Boolean combinations of literals is as hard as in propositional logic

Solution: Exploit propositional satisfiability technology

Lifting SAT Technology to SMT

Two main approaches:

Lifting SAT Technology to SMT

Two main approaches:

1. *Eager*
   - translate the input formula F to an equisatisfiable propositional formula P
   - feed P to any SAT solver

Lifting SAT Technology to SMT

Two main approaches:

2. *Lazy*

- abstract the input formula F to a propositional formula A in CNF
- feed A to a DPLL-based SAT solver
- use a theory-specific solver to refine the abstraction and guide the SAT solver

Lifting SAT Technology to SMT

Two main approaches:

2. *Lazy*

- abstract the input formula F to a propositional formula A in CNF
- feed A to a DPLL-based SAT solver
- use a theory-specific solver to refine the abstraction and guide the SAT solver

We will focus on the lazy approach here

(Very) Lazy Approach for SMT, Example

$$g(a) = c \quad \wedge \quad f(g(a)) \neq f(c) \ \vee \ g(a) = d \quad \wedge \quad c \neq d$$

Theory T**:** Equality with Uninterpreted Functions

(Very) Lazy Approach for SMT, Example

$$g(a) = c \quad \wedge \quad f(g(a)) \neq f(c) \;\vee\; g(a) = d \quad \wedge \quad c \neq d$$

Simplest setting:

- Off-line SAT solver
- Non-incremental *theory solver*
  for conjunctions of equalities and disequalities
- Theory atoms abstracted to propositional atoms
  (e.g., $g(a) = c$ abstracted to p_1)

(Very) Lazy Approach for SMT – Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

Notation:

- $\overline{a} \stackrel{\text{def}}{=} \neg a$
- $\overline{\overline{a}} \stackrel{\text{def}}{=} a$
- $\{\, p_1, \, \overline{p}_2, \, \overline{p}_3, \, p_4 \,\} \stackrel{\text{def}}{=} \{\, p_1 \mapsto 1, \, p_2 \mapsto 0, \, p_3 \mapsto 0, \, \overline{p}_4 \mapsto 1 \,\}$

(Very) Lazy Approach for SMT – Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

1. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4\}$ to SAT solver
2. SAT solver returns satisfying assignment $\{p_1,\ \overline{p}_2,\ \overline{p}_4\}$
3. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4\}$ to SAT solver
4. SAT solver returns new satisfying assignment $\{p_1,\ p_3,\ \overline{p}_4\}$
5. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ to SAT solver
6. SAT solver finds $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ unsat

(Very) Lazy Approach for SMT – Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

1. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4\}$ to SAT solver
2. SAT solver returns satisfying assignment $\{p_1, \overline{p}_2, \overline{p}_4\}$
3. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4\}$ to SAT solver
4. SAT solver returns new satisfying assignment $\{p_1, p_3, \overline{p}_4\}$
5. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ to SAT solver
6. SAT solver finds $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ unsat

(Very) Lazy Approach for SMT – Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

1. Send $\{p_1, \ \overline{p}_2 \vee p_3, \ \overline{p}_4\}$ to SAT solver
2. SAT solver returns satisfying assignment $\{p_1, \overline{p}_2, \overline{p}_4\}$
   Theory solver finds concretization of $\{p_1, \overline{p}_2, \overline{p}_4\}$
   ($\{\, g(a) = c, \ f(g(a)) \neq f(c), \ c \neq d \,\}$) unsat
3. Send $\{p_1, \ \overline{p}_2 \vee p_3, \ \overline{p}_4, \ \overline{p}_1 \vee p_2 \vee p_4\}$ to SAT solver
4. SAT solver returns new satisfying assignment $\{p_1, p_3, \overline{p}_4\}$
5. Send $\{p_1, \ \overline{p}_2 \vee p_3, \ \overline{p}_4, \ \overline{p}_1 \vee p_2, \ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ to SAT solver
6. SAT solver finds $\{p_1, \ \overline{p}_2 \vee p_3, \ \overline{p}_4, \ \overline{p}_1 \vee p_2 \vee p_4, \ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ unsat

(Very) Lazy Approach for SMT – Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

1. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4\}$ to SAT solver

2. SAT solver returns satisfying assignment $\{p_1,\ \overline{p}_2,\ \overline{p}_4\}$
   Theory solver finds concretization of $\{p_1,\ \overline{p}_2,\ \overline{p}_4\}$
   ($\{\, g(a) = c,\ f(g(a)) \neq f(c),\ c \neq d \,\}$) unsat

3. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4\}$ to SAT solver

4. SAT solver returns new satisfying assignment $\{p_1,\ p_3,\ \overline{p}_4\}$

5. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ to SAT solver

6. SAT solver finds $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ unsat

(Very) Lazy Approach for SMT – Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

1. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4\}$ to SAT solver
2. SAT solver returns satisfying assignment $\{p_1,\ \overline{p}_2,\ \overline{p}_4\}$
   Theory solver finds concretization of $\{p_1,\ \overline{p}_2,\ \overline{p}_4\}$
   ($\{\, g(a) = c,\ f(g(a)) \neq f(c),\ c \neq d \,\}$) unsat
3. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4\}$ to SAT solver
4. SAT solver ret

New clause *blocks* previous assignment

5. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ to SAT solver
6. SAT solver finds $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ unsat

(Very) Lazy Approach for SMT – Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

1. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4\}$ to SAT solver
2. SAT solver returns satisfying assignment $\{p_1,\ \overline{p}_2,\ \overline{p}_4\}$
   Theory solver finds concretization of $\{p_1,\ \overline{p}_2,\ \overline{p}_4\}$
   $(\{\, g(a) = c,\ f(g(a)) \neq f(c),\ c \neq d \,\})$ unsat
3. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4\}$ to SAT solver
4. SAT solver returns new satisfying assignment $\{p_1,\ p_3,\ \overline{p}_4\}$
5. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ to SAT solver
6. SAT solver finds $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ unsat

(Very) Lazy Approach for SMT – Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

1. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4\}$ to SAT solver
2. SAT solver returns satisfying assignment $\{p_1,\ \overline{p}_2,\ \overline{p}_4\}$
   Theory solver finds concretization of $\{p_1,\ \overline{p}_2,\ \overline{p}_4\}$
   ($\{\ g(a) = c,\ f(g(a)) \neq f(c),\ c \neq d\ \}$) unsat
3. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4\}$ to SAT solver
4. SAT solver returns new satisfying assignment $\{p_1,\ p_3,\ \overline{p}_4\}$
   Theory solver finds $\{p_1,\ p_3,\ \overline{p}_4\}$ unsat
5. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ to SAT solver
6. SAT solver finds $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ unsat

(Very) Lazy Approach for SMT – Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

1. Send $\{p_1, \overline{p}_2 \vee p_3, \overline{p}_4\}$ to SAT solver
2. SAT solver returns satisfying assignment $\{p_1, \overline{p}_2, \overline{p}_4\}$
   Theory solver finds concretization of $\{p_1, \overline{p}_2, \overline{p}_4\}$
   $(\{\, g(a) = c,\ f(g(a)) \neq f(c),\ c \neq d \,\})$ unsat
3. Send $\{p_1, \overline{p}_2 \vee p_3, \overline{p}_4, \overline{p}_1 \vee p_2 \vee p_4\}$ to SAT solver
4. SAT solver returns new satisfying assignment $\{p_1, p_3, \overline{p}_4\}$
   Theory solver finds $\{p_1, p_3, \overline{p}_4\}$ unsat
5. Send $\{p_1, \overline{p}_2 \vee p_3, \overline{p}_4, \overline{p}_1 \vee p_2, \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ to SAT solver
6. SAT solver finds $\{p_1, \overline{p}_2 \vee p_3, \overline{p}_4, \overline{p}_1 \vee p_2 \vee p_4, \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ unsat

(Very) Lazy Approach for SMT – Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

1. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4\}$ to SAT solver
2. SAT solver returns satisfying assignment $\{p_1,\ \overline{p}_2,\ \overline{p}_4\}$
   Theory solver finds concretization of $\{p_1,\ \overline{p}_2,\ \overline{p}_4\}$ $(\{\, g(a) = c,\ f(g(a)) \neq f(c),\ c \neq d \,\})$ unsat
3. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4\}$ to SAT solver
4. SAT solver returns new satisfying assignment $\{p_1,\ p_3,\ \overline{p}_4\}$
   Theory solver finds $\{p_1,\ p_3,\ \overline{p}_4\}$ unsat
5. Send $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ to SAT solver
6. SAT solver finds $\{p_1,\ \overline{p}_2 \vee p_3,\ \overline{p}_4,\ \overline{p}_1 \vee p_2 \vee p_4,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ unsat

(Very) Lazy Approach for SMT – Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

1. Send $\{p_1, \ \overline{p}_2 \vee p_3, \ \overline{p}_4\}$ to SAT solver
2. SAT solver returns satisfying assignment $\{p_1, \overline{p}_2, \overline{p}_4\}$

   Theory solver finds concretization of $\{p_1, \overline{p}_2, \overline{p}_4\}$

   $(\{\, g(a) = c, \ f(g(a)) \neq f(c), \ c \neq d \,\})$ unsat
3. Send $\{p_1, \ \overline{p}$

   Done! The original formula is unsatisfiable in EUF!
4. SAT solver returns new satisfying assignment $\{p_1, p_3, \overline{p}_4\}$

   Theory solver finds $\{p_1, p_3, \overline{p}_4\}$ unsat
5. Send $\{p_1, \ \overline{p}_2 \vee p_3, \ \overline{p}_4, \ \overline{p}_1 \vee p_2, \ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ to SAT solver
6. SAT solver finds $\{p_1, \ \overline{p}_2 \vee p_3, \ \overline{p}_4, \ \overline{p}_1 \vee p_2 \vee p_4, \ \overline{p}_1 \vee \overline{p}_3 \vee p_4\}$ unsat

Lazy Approach – Enhancements

Several enhancements are possible to increase efficiency:

- Check T-satisfiability only of full propositional model
- Check T-satisfiability of partial assignment M as it grows

- If M is T-unsatisfiable, add $\neg M$ as a clause
- If M is T-unsatisfiable, identify a T-unsatisfiable subset M_0 of M and add $\neg M_0$ as a clause

- If M is T-unsatisfiable, add clause and restart
- If M is T-unsatisfiable, bactrack to some point where the assignment was still T-satisfiable

Lazy Approach – Enhancements

Several enhancements are possible to increase efficiency:

- Check T-satisfiability only of full propositional model
- Check T-satisfiability of partial assignment M as it grows

- If M is T-unsatisfiable, add $\neg M$ as a clause
- If M is T-unsatisfiable, identify a T-unsatisfiable subset M_0 of M and add $\neg M_0$ as a clause

- If M is T-unsatisfiable, add clause and restart
- If M is T-unsatisfiable, bactrack to some point where the assignment was still T-satisfiable

Lazy Approach – Enhancements

Several enhancements are possible to increase efficiency:

- Check T-satisfiability only of full propositional model
- Check T-satisfiability of partial assignment M as it grows
- If M is T-unsatisfiable, add $\neg M$ as a clause
- If M is T-unsatisfiable, identify a T-unsatisfiable subset M_0 of M and add $\neg M_0$ as a clause
- If M is T-unsatisfiable, add clause and restart
- If M is T-unsatisfiable, bactrack to some point where the assignment was still T-satisfiable

Lazy Approach – Enhancements

Several enhancements are possible to increase efficiency:

- Check T-satisfiability only of full propositional model
- Check T-satisfiability of partial assignment M as it grows
- If M is T-unsatisfiable, add $\neg M$ as a clause
- If M is T-unsatisfiable, identify a T-unsatisfiable subset M_0 of M and add $\neg M_0$ as a clause
- If M is T-unsatisfiable, add clause and restart
- If M is T-unsatisfiable, bactrack to some point where the assignment was still T-satisfiable

Lazy Approach – Enhancements

Several enhancements are possible to increase efficiency:

- Check T-satisfiability only of full propositional model
- Check T-satisfiability of partial assignment M as it grows
- If M is T-unsatisfiable, add $\neg M$ as a clause
- If M is T-unsatisfiable, identify a T-unsatisfiable subset M_0 of M and add $\neg M_0$ as a clause
- If M is T-unsatisfiable, add clause and restart
- If M is T-unsatisfiable, bactrack to some point where the assignment was still T-satisfiable

Lazy Approach – Enhancements

Several enhancements are possible to increase efficiency:

- Check T-satisfiability only of full propositional model
- Check T-satisfiability of partial assignment M as it grows
- If M is T-unsatisfiable, add $\neg M$ as a clause
- If M is T-unsatisfiable, identify a T-unsatisfiable subset M_0 of M and add $\neg M_0$ as a clause
- If M is T-unsatisfiable, add clause and restart
- If M is T-unsatisfiable, bactrack to some point where the assignment was still T-satisfiable

Lazy Approach – Enhancements

Several enhancements are possible to increase efficiency:

- Check T-satisfiability only of full propositional model
- Check T-satisfiability of partial assignment M as it grows
- If M is T-unsatisfiable, add $\neg M$ as a clause
- If M is T-unsatisfiable, identify a T-unsatisfiable subset M_0 of M and add $\neg M_0$ as a clause
- If M is T-unsatisfiable, add clause and restart
- If M is T-unsatisfiable, bactrack to some point where the assignment was still T-satisfiable

Lazy Approach, Main Benefits

Every tool does what it is good at:

- SAT solver takes care of Boolean information
- Theory solver takes care of theory information

The SAT solver works only with propositional clauses

The theory solver works only with conjunctions of (FOL) literals

Lazy Approach, Main Benefits

Every tool does what it is good at:

- SAT solver takes care of Boolean information
- Theory solver takes care of theory information

The SAT solver works only with propositional clauses

The theory solver works only with conjunctions of (FOL) literals

Lazy Approach, Main Benefits

Every tool does what it is good at:

- SAT solver takes care of Boolean information
- Theory solver takes care of theory information

The SAT solver works only with propositional clauses

The theory solver works only with conjunctions of (FOL) literals

The Original DPLL Procedure

Recall

Modern SAT solvers are based on the DPLL procedure

DPLL tries to build incrementally a satisfying truth assignment M for a formula F in CNF

M is grown by

- deducing by unit propagation the truth value of a literal from M and F, or
- guessing a truth value

The procedure backtracks on each wrong guess and tries the opposite value

An Abstract Transition System for DPLL

States:

$$\text{fail} \qquad \text{or} \qquad \langle M, F \rangle$$

where

- M is a sequence of literals and *decision points* $\bullet$ denoting a partial truth *assignment*
- F is a set of clauses denoting a CNF *formula*

Definition If $M = M_0 \bullet M_1 \bullet \cdots \bullet M_n$ where each M_i contains no decision points

1. M_i is *decision level* i of M
2. $M^{[i]} \stackrel{\text{def}}{=} M_0 \bullet \cdots \bullet M_i$

An Abstract Transition System for DPLL

States:

$$\text{fail} \qquad \text{or} \qquad \langle M, F \rangle$$

Initial state:

$\langle \varepsilon, F_0 \rangle$ where ε is the empty sequence and F_0 is the input CNF

Expected final states:

fail if F_0 is unsatisfiable

$\langle M, G \rangle$ otherwise, where

- G is equivalent to F_0 and
- M satisfies G

Transition Rule Notation

Transition rules in *guarded assignment form*

$$\frac{P_1 \quad \cdots \quad P_n}{\mathsf{M}' = e_1 \quad \mathsf{F}' = e_2}$$

updating M, F or both when premises $P_1, \ldots, P_n$ all hold

Note: When convenient, will treat M as the set of its literals

Transition Rules for Original DPLL

Extending M

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M}\ l} \text{ Propagate}$$

Note: The order of literal in clauses is not meaningful

$$\frac{l \in \mathrm{Lit}(\mathsf{F}) \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M} \bullet l} \text{ Decide}$$

Notation: $\mathrm{Lit}(F) \stackrel{\mathrm{def}}{=} \{l \mid l \text{ literal of } F\} \cup \{\bar{l} \mid l \text{ literal of } F\}$

Transition Rules for Original DPLL

Extending M

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M}\ l} \ \text{Propagate}$$

Note: The order of literal in clauses is not meaningful

$$\frac{l \in \mathrm{Lit}(\mathsf{F}) \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M} \bullet l} \ \text{Decide}$$

Notation: $\mathrm{Lit}(F) \stackrel{\mathrm{def}}{=} \{l \mid l \text{ literal of } F\} \cup \{\bar{l} \mid l \text{ literal of } F\}$

Transition Rules for Original DPLL

Repairing M

$$\frac{l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \bullet \notin \mathsf{M}}{\mathsf{fail}} \;\textsf{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \mathsf{M} = M \bullet l\, N \quad \bullet \notin N}{\mathsf{M}' = M\, \bar{l}} \;\textsf{Backtrack}$$

Note: Last premise of Backtrack enforces chronological backtracking

Transition Rules for Original DPLL

Repairing M

$$\frac{l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \bullet \notin \mathsf{M}}{\mathsf{fail}} \ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \mathsf{M} = M \bullet l \; N \quad \bullet \notin N}{\mathsf{M}' = M \; \bar{l}} \ \text{Backtrack}$$

Note: Last premise of Backtrack enforces chronological backtracking

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M}\; l}\ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathsf{F}) \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M} \bullet l}\ \text{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \bullet \notin \mathsf{M}}{\mathsf{fail}}\ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \mathsf{M} = M \bullet l\, N \quad \bullet \notin N}{\mathsf{M}' = M\, \bar{l}}\ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\ l}\ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l}\ \text{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\text{fail}}\ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}}\ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

One execution:

	M	F	Rule
1	ε	F_0	

M	F	Rule

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\ l} \ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \ \text{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}} \ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

One execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	

M	F	Rule

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M}\ l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathsf{F}) \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \bullet \notin \mathsf{M}}{\mathsf{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \mathsf{M} = M \bullet l\ N \quad \bullet \notin N}{\mathsf{M}' = M\ \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \overline{a} \vee e, \quad a \vee \overline{c} \vee \overline{r}, \quad \overline{r} \vee a \vee \overline{e}, \quad \overline{e} \vee c, \quad \overline{a} \vee \overline{e} \vee \overline{c}, \quad r$$

One execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide $\overline{a}$
3	$r \bullet \overline{a}$	F_0	

M	F	Rule

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\; l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

One execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide $\bar{a}$
3	$r \bullet \bar{a}$	F_0	Propagate on $\bar{r} \vee a \vee \bar{e}$
4	$r \bullet \bar{a}\ \bar{e}$	F_0	

M	F	Rule

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\; l}\ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l}\ \text{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\text{fail}}\ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}}\ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

One execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide $\bar{a}$
3	$r \bullet \bar{a}$	F_0	Propagate on $\bar{r} \vee a \vee \bar{e}$
4	$r \bullet \bar{a}\ \bar{e}$	F_0	Propagate on $a \vee e \vee c$
5	$r \bullet \bar{a}\ \bar{e}\ c$	F_0	

M	F	Rule

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\ l} \ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \ \text{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\text{fail}} \ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

One execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide $\bar{a}$
3	$r \bullet \bar{a}$	F_0	Propagate on $\bar{r} \vee a \vee \bar{e}$
4	$r \bullet \bar{a}\ \bar{e}$	F_0	Propagate on $a \vee e \vee c$
5	$r \bullet \bar{a}\ \bar{e}\ c$	F_0	Backtrack on $a \vee \bar{c} \vee \bar{r}$

	M	F	Rule
6	$r\ a$	F_0	

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\; l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

One execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide $\bar{a}$
3	$r \bullet \bar{a}$	F_0	Propagate on $\bar{r} \vee a \vee \bar{e}$
4	$r \bullet \bar{a}\ \bar{e}$	F_0	Propagate on $a \vee e \vee c$
5	$r \bullet \bar{a}\ \bar{e}\ c$	F_0	Backtrack on $a \vee \bar{c} \vee \bar{r}$

	M	F	Rule
6	$r\ a$	F_0	Propagate on $\bar{a} \vee e$
7	$r\ a\ e$	F_0	

DPLL Execution, Example 1

$$\frac{l_1 \vee \dots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \dots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\ l}\ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l}\ \text{Decide} \qquad \frac{l_1 \vee \dots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \dots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}}\ \text{Fail}$$

$$\frac{l_1 \vee \dots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \dots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}}\ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

One execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide $\bar{a}$
3	$r \bullet \bar{a}$	F_0	Propagate on $\bar{r} \vee a \vee \bar{e}$
4	$r \bullet \bar{a}\ \bar{e}$	F_0	Propagate on $a \vee e \vee c$
5	$r \bullet \bar{a}\ \bar{e}\ c$	F_0	Backtrack on $a \vee \bar{c} \vee \bar{r}$

	M	F	Rule
6	$r\ a$	F_0	Propagate on $\bar{a} \vee e$
7	$r\ a\ e$	F_0	Propagate on $\bar{e} \vee c$
8	$r\ a\ e\ c$	F_0	

DPLL Execution, Example 1

$$\frac{l_1 \lor \cdots \lor l_n \lor l \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M}\ l} \ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathsf{F}) \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M} \bullet l} \ \text{Decide} \qquad \frac{l_1 \lor \cdots \lor l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \bullet \notin \mathsf{M}}{\mathsf{fail}} \ \text{Fail}$$

$$\frac{l_1 \lor \cdots \lor l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \mathsf{M} = M \bullet l\ N \quad \bullet \notin N}{\mathsf{M}' = M\ \bar{l}} \ \text{Backtrack}$$

$$F_0 \quad = \quad a \lor e \lor c, \quad \bar{a} \lor e, \quad a \lor \bar{c} \lor \bar{r}, \quad \bar{r} \lor a \lor \bar{e}, \quad \bar{e} \lor c, \quad \bar{a} \lor \bar{e} \lor \bar{c}, \quad r$$

One execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide $\bar{a}$
3	$r \bullet \bar{a}$	F_0	Propagate on $\bar{r} \lor a \lor \bar{e}$
4	$r \bullet \bar{a}\ \bar{e}$	F_0	Propagate on $a \lor e \lor c$
5	$r \bullet \bar{a}\ \bar{e}\ c$	F_0	Backtrack on $a \lor \bar{c} \lor \bar{r}$

	M	F	Rule
6	$r\ a$	F_0	Propagate on $\bar{a} \lor e$
7	$r\ a\ e$	F_0	Propagate on $\bar{e} \lor c$
8	$r\ a\ e\ c$	F_0	Fail on $\bar{a} \lor \bar{e} \lor \bar{c}$
9	fail		

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M}\; l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathsf{F}) \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \bullet \notin \mathsf{M}}{\mathsf{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \mathsf{M} = M \bullet l\, N \quad \bullet \notin N}{\mathsf{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

Another execution:

	M	F	Rule
1	ε	F_0	

M	F	Rule

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\ l} \ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \ \text{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\text{fail}} \ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

Another execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	

M	F	Rule

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\; l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

Another execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide e
3	$r \bullet e$	F_0	

M	F	Rule

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\ l}\ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l}\ \text{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\text{fail}}\ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\ \bar{l}}\ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \overline{a} \vee e, \quad a \vee \overline{c} \vee \overline{r}, \quad \overline{r} \vee a \vee \overline{e}, \quad \overline{e} \vee c, \quad \overline{a} \vee \overline{e} \vee \overline{c}, \quad r$$

Another execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide e
3	$r \bullet e$	F_0	Propagate on $\overline{e} \vee c$
4	$r \bullet e\ c$	F_0	

M	F	Rule

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\ l}\ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l}\ \text{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\text{fail}}\ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\ \bar{l}}\ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

Another execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide e
3	$r \bullet e$	F_0	Propagate on $\bar{e} \vee c$
4	$r \bullet e\ c$	F_0	Propagate on $\bar{a} \vee \bar{e} \vee \bar{c}$
5	$r \bullet e\ c\ \bar{a}$	F_0	

M	F	Rule

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\ l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\ \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

Another execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide e
3	$r \bullet e$	F_0	Propagate on $\bar{e} \vee c$
4	$r \bullet e\ c$	F_0	Propagate on $\bar{a} \vee \bar{e} \vee \bar{c}$
5	$r \bullet e\ c\ \bar{a}$	F_0	Backtrack on $a \vee \bar{c} \vee \bar{r}$

	M	F	Rule
6	$r\ \bar{e}$	F_0	

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\, l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

Another execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide e
3	$r \bullet e$	F_0	Propagate on $\bar{e} \vee c$
4	$r \bullet e\ c$	F_0	Propagate on $\bar{a} \vee \bar{e} \vee \bar{c}$
5	$r \bullet e\ c\ \bar{a}$	F_0	Backtrack on $a \vee \bar{c} \vee \bar{r}$

	M	F	Rule
6	$r\ \bar{e}$	F_0	Propagate on $\bar{a} \vee e$
7	$r\ \bar{e}\ \bar{a}$	F_0	

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\ l}\ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l}\ \text{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\text{fail}}\ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}}\ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

Another execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide e
3	$r \bullet e$	F_0	Propagate on $\bar{e} \vee c$
4	$r \bullet e\ c$	F_0	Propagate on $\bar{a} \vee \bar{e} \vee \bar{c}$
5	$r \bullet e\ c\ \bar{a}$	F_0	Backtrack on $a \vee \bar{c} \vee \bar{r}$

	M	F	Rule
6	$r\ \bar{e}$	F_0	Propagate on $\bar{a} \vee e$
7	$r\ \bar{e}\ \bar{a}$	F_0	Propagate on $a \vee e \vee c$
8	$r\ \bar{e}\ \bar{a}\ c$	F_0	

DPLL Execution, Example 1

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\; l}\ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l}\ \text{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}}\ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}}\ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}, \quad r$$

Another execution:

	M	F	Rule
1	ε	F_0	Propagate on r
2	r	F_0	Decide e
3	$r \bullet e$	F_0	Propagate on $\bar{e} \vee c$
4	$r \bullet e\ c$	F_0	Propagate on $\bar{a} \vee \bar{e} \vee \bar{c}$
5	$r \bullet e\ c\ \bar{a}$	F_0	Backtrack on $a \vee \bar{c} \vee \bar{r}$

	M	F	Rule
6	$r\ \bar{e}$	F_0	Propagate on $\bar{a} \vee e$
7	$r\ \bar{e}\ \bar{a}$	F_0	Propagate on $a \vee e \vee c$
8	$r\ \bar{e}\ \bar{a}\ c$	F_0	Fail on $a \vee \bar{c} \vee \bar{r}$
9		fail	

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M}\, l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathsf{F}) \quad l \notin \mathsf{M} \quad \bar{l} \notin \mathsf{M}}{\mathsf{M}' = \mathsf{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \bullet \notin \mathsf{M}}{\mathsf{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M} \quad \mathsf{M} = M \bullet l\, N \quad \bullet \notin N}{\mathsf{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\, l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

One execution:

	M	F	Rule
1	ε	F_0	

M	F	Rule

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\, l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit(F)} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\text{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

One execution:

	M	F	Rule
1	ε	F_0	Decide $\bar{c}$
2	$\bullet\, \bar{c}$	F_0	

M	F	Rule

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\; l} \;\textsf{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \;\textsf{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathsf{fail}} \;\textsf{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \;\textsf{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

One execution:

	M	F	Rule
1	ε	F_0	Decide $\bar{c}$
2	$\bullet\, \bar{c}$	F_0	Propagate on $\bar{e} \vee c$
3	$\bullet\, \bar{c}\, \bar{e}$	F_0	

M	F	Rule

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\; l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\; \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

One execution:

	M	F	Rule
1	ε	F_0	Decide $\bar{c}$
2	$\bullet\, \bar{c}$	F_0	Propagate on $\bar{e} \vee c$
3	$\bullet\, \bar{c}\; \bar{e}$	F_0	Propagate on $a \vee e \vee c$
4	$\bullet\, \bar{c}\; \bar{e}\; a$	F_0	

M	F	Rule

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\, l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

One execution:

	M	F	Rule
1	ε	F_0	Decide $\bar{c}$
2	$\bullet\, \bar{c}$	F_0	Propagate on $\bar{e} \vee c$
3	$\bullet\, \bar{c}\, \bar{e}$	F_0	Propagate on $a \vee e \vee c$
4	$\bullet\, \bar{c}\, \bar{e}\, a$	F_0	Backtrack on $\bar{a} \vee e$

	M	F	Rule
5	c	F_0	

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\, l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\text{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

One execution:

	M	F	Rule
1	ε	F_0	Decide $\bar{c}$
2	$\bullet\, \bar{c}$	F_0	Propagate on $\bar{e} \vee c$
3	$\bullet\, \bar{c}\, \bar{e}$	F_0	Propagate on $a \vee e \vee c$
4	$\bullet\, \bar{c}\, \bar{e}\, a$	F_0	Backtrack on $\bar{a} \vee e$

	M	F	Rule
5	c	F_0	Decide on $\bar{e}$
6	$c \bullet \bar{e}$	F_0	

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\, l} \ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \ \text{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\text{fail}} \ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

One execution:

	M	F	Rule
1	ε	F_0	Decide $\bar{c}$
2	$\bullet\, \bar{c}$	F_0	Propagate on $\bar{e} \vee c$
3	$\bullet\, \bar{c}\, \bar{e}$	F_0	Propagate on $a \vee e \vee c$
4	$\bullet\, \bar{c}\, \bar{e}\, a$	F_0	Backtrack on $\bar{a} \vee e$

	M	F	Rule
5	c	F_0	Decide on $\bar{e}$
6	$c \bullet \bar{e}$	F_0	Propagate on $\bar{a} \vee e$
7	$c \bullet \bar{e}\, \bar{a}$	F_0	

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\; l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\text{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

One execution:

	M	F	Rule
1	ε	F_0	Decide $\bar{c}$
2	$\bullet\, \bar{c}$	F_0	Propagate on $\bar{e} \vee c$
3	$\bullet\, \bar{c}\; \bar{e}$	F_0	Propagate on $a \vee e \vee c$
4	$\bullet\, \bar{c}\; \bar{e}\; a$	F_0	Backtrack on $\bar{a} \vee e$

	M	F	Rule
5	c	F_0	Decide on $\bar{e}$
6	$c \bullet \bar{e}$	F_0	Propagate on $\bar{a} \vee e$
7	$c \bullet \bar{e}\; \bar{a}$	F_0	Decide on r
8	$c \bullet \bar{e}\; \bar{a} \bullet r$	F_0	

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\, l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\text{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

One execution:

	M	F	Rule
1	ε	F_0	Decide $\bar{c}$
2	$\bullet\, \bar{c}$	F_0	Propagate on $\bar{e} \vee c$
3	$\bullet\, \bar{c}\, \bar{e}$	F_0	Propagate on $a \vee e \vee c$
4	$\bullet\, \bar{c}\, \bar{e}\, a$	F_0	Backtrack on $\bar{a} \vee e$

	M	F	Rule
5	c	F_0	Decide on $\bar{e}$
6	$c \bullet \bar{e}$	F_0	Propagate on $\bar{a} \vee e$
7	$c \bullet \bar{e}\, \bar{a}$	F_0	Decide on r
8	$c \bullet \bar{e}\, \bar{a} \bullet r$	F_0	

F_0 satisfied by $\{\, a \mapsto 0,\ c \mapsto 1,\ e \mapsto 0,\ r \mapsto 1 \,\}$

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\; l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit(F)} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

Another execution:

	M	F	Rule
1	ε	F_0	

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\, l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \overline{a} \vee e, \quad a \vee \overline{c} \vee \overline{r}, \quad \overline{r} \vee a \vee \overline{e}, \quad \overline{e} \vee c, \quad \overline{a} \vee \overline{e} \vee \overline{c}$$

Another execution:

	M	F	Rule
1	ε	F_0	Decide a
2	$\bullet\, a$	F_0	

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\ l}\ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l}\ \text{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\text{fail}}\ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}}\ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \overline{a} \vee e, \quad a \vee \overline{c} \vee \overline{r}, \quad \overline{r} \vee a \vee \overline{e}, \quad \overline{e} \vee c, \quad \overline{a} \vee \overline{e} \vee \overline{c}$$

Another execution:

	M	F	Rule
1	ε	F_0	Decide a
2	$\bullet\, a$	F_0	Propagate on $\overline{a} \vee e$
3	$\bullet\, a\ e$	F_0	

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\, l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

Another execution:

	M	F	Rule
1	ε	F_0	Decide a
2	$\bullet\, a$	F_0	Propagate on $\bar{a} \vee e$
3	$\bullet\, a\; e$	F_0	Propagate on $\bar{e} \vee c$
4	$\bullet\, a\; e\; c$	F_0	

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M}\, l} \text{ Propagate}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad l \notin \mathrm{M} \quad \bar{l} \notin \mathrm{M}}{\mathrm{M}' = \mathrm{M} \bullet l} \text{ Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \bullet \notin \mathrm{M}}{\mathrm{fail}} \text{ Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M} \quad \mathrm{M} = M \bullet l\, N \quad \bullet \notin N}{\mathrm{M}' = M\, \bar{l}} \text{ Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

Another execution:

	M	F	Rule
1	ε	F_0	Decide a
2	$\bullet\, a$	F_0	Propagate on $\bar{a} \vee e$
3	$\bullet\, a\ e$	F_0	Propagate on $\bar{e} \vee c$
4	$\bullet\, a\ e\ c$	F_0	Decide r
5	$\bullet\, a\ e\ c \bullet r$	F_0	

DPLL Execution, Example 2

$$\frac{l_1 \vee \cdots \vee l_n \vee l \in F \quad \bar{l}_1, \ldots, \bar{l}_n \in M \quad l \notin M \quad \bar{l} \notin M}{M' = M\, l}\ \text{Propagate}$$

$$\frac{l \in \mathrm{Lit}(F) \quad l \notin M \quad \bar{l} \notin M}{M' = M \bullet l}\ \text{Decide} \qquad \frac{l_1 \vee \cdots \vee l_n \in F \quad \bar{l}_1, \ldots, \bar{l}_n \in M \quad \bullet \notin M}{\text{fail}}\ \text{Fail}$$

$$\frac{l_1 \vee \cdots \vee l_n \in F \quad \bar{l}_1, \ldots, \bar{l}_n \in M \quad M = M \bullet l\, N \quad \bullet \notin N}{M' = M\, \bar{l}}\ \text{Backtrack}$$

$$F_0 \quad = \quad a \vee e \vee c, \quad \bar{a} \vee e, \quad a \vee \bar{c} \vee \bar{r}, \quad \bar{r} \vee a \vee \bar{e}, \quad \bar{e} \vee c, \quad \bar{a} \vee \bar{e} \vee \bar{c}$$

Another execution:

	M	F	Rule
1	ε	F_0	Decide a
2	$\bullet\, a$	F_0	Propagate on $\bar{a} \vee e$
3	$\bullet\, a\ e$	F_0	Propagate on $\bar{e} \vee c$
4	$\bullet\, a\ e\ c$	F_0	Decide r
5	$\bullet\, a\ e\ c \bullet r$	F_0	

F_0 satisfied by

$\{\, a \mapsto 1,\ c \mapsto 1,\ e \mapsto 1,\ r \mapsto 1 \,\}$

From DPLL to CDCL Solvers

Modern SAT solvers have more sophisticated ways to recover from wrong decisions

They implement

- conflict-driven (CD) backjumping
  instead of (chronological) backtracking
- selective clause learning (CL)
  to help focus later search
- restart strategies
  to get out of unproductive search paths

An Abstract Transition System for CDCL

States:

$$\text{fail} \qquad \text{or} \qquad \langle M, C, F \rangle$$

Extend DPLL state with a component C
whose value is either none or a *conflict clause*

An Abstract Transition System for CDCL

States:

$$\text{fail} \qquad \text{or} \qquad \langle M, C, F \rangle$$

Initial state:

$\langle \varepsilon, \text{none}, F_0 \rangle$ where F_0 is the input CNF

Expected final states:

fail if F_0 is unsatisfiable

$\langle M, \text{none}, G \rangle$ otherwise, where

- G is equivalent to F_0 and
- M satisfies G

From DPLL to CDCL rules

Replace Backtrack with

$$\frac{C = \text{none} \quad l_1 \vee \cdots \vee l_n \in F \quad \bar{l}_1, \ldots, \bar{l}_n \in M}{C' = l_1 \vee \cdots \vee l_n} \ \text{Conflict}$$

$$\frac{C = l \vee D \quad l_1 \vee \cdots \vee l_n \vee \bar{l} \in F \quad \bar{l}_1, \ldots, \bar{l}_n \prec_M \bar{l}}{C' = l_1 \vee \cdots \vee l_n \vee D} \ \text{Explain}$$

$$\frac{C = l_1 \vee \cdots \vee l_n \vee l \quad \text{lev}\,\bar{l}_1, \ldots, \text{lev}\,\bar{l}_n \leq i < \text{lev}\,\bar{l}}{C' = \text{none} \quad M' = M^{[i]}\, l} \ \text{Backjump}$$

From DPLL to CDCL rules

Replace Backtrack with

$$\frac{\mathsf{C} = \mathsf{none} \quad l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M}}{\mathsf{C}' = l_1 \vee \cdots \vee l_n} \ \text{Conflict}$$

$$\frac{\mathsf{C} = l \vee D \quad l_1 \vee \cdots \vee l_n \vee \bar{l} \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \prec_{\mathsf{M}} \bar{l}}{\mathsf{C}' = l_1 \vee \cdots \vee l_n \vee D} \ \text{Explain}$$

$$\frac{\mathsf{C} = l_1 \vee \cdots \vee l_n \vee l \quad \mathrm{lev}\,\bar{l}_1, \ldots, \mathrm{lev}\,\bar{l}_n \leq i < \mathrm{lev}\,\bar{l}}{\mathsf{C}' = \mathsf{none} \quad \mathsf{M}' = \mathsf{M}^{[i]}\, l} \ \text{Backjump}$$

Notation: $l \prec_{\mathsf{M}} l'$ if l occurs before l' in M
$\mathrm{lev}\, l = i$ iff l occurs in decision level i of M

From DPLL to CDCL rules

Replace Backtrack with

$$\frac{\mathsf{C} = \text{none} \quad l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M}}{\mathsf{C}' = l_1 \vee \cdots \vee l_n} \text{ Conflict}$$

$$\frac{\mathsf{C} = l \vee D \quad l_1 \vee \cdots \vee l_n \vee \bar{l} \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \prec_{\mathsf{M}} \bar{l}}{\mathsf{C}' = l_1 \vee \cdots \vee l_n \vee D} \text{ Explain}$$

$$\frac{\mathsf{C} = l_1 \vee \cdots \vee l_n \vee l \quad \text{lev}\,\bar{l}_1, \ldots, \text{lev}\,\bar{l}_n \leq i < \text{lev}\,\bar{l}}{\mathsf{C}' = \text{none} \quad \mathsf{M}' = \mathsf{M}^{[i]}\, l} \text{ Backjump}$$

Maintain invariant: $\mathsf{F} \models_{\mathrm{p}} \mathsf{C}$ and $\mathsf{M} \not\models_{\mathrm{p}} \mathsf{C}$ when $\mathsf{C} \neq \text{none}$
where $\models_{\mathrm{p}}$ denotes propositional entailment

From DPLL to CDCL rules

Modify Fail to

$$\frac{C \neq \text{none} \qquad \bullet \notin M}{\text{fail}}\ \text{Fail}$$

CDCL Execution Example

$$\frac{\mathsf{C} = \mathsf{none} \quad l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M}}{\mathsf{C}' = l_1 \vee \cdots \vee l_n}\ \text{Conflict}$$

$$\frac{\mathsf{C} = l \vee D \quad l_1 \vee \cdots \vee l_n \vee \bar{l} \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \prec_{\mathsf{M}} \bar{l}}{\mathsf{C}' = l_1 \vee \cdots \vee l_n \vee D}\ \text{Explain}$$

$$\frac{\mathsf{C} = l_1 \vee \cdots \vee l_n \vee l \quad \mathsf{lev}\,\bar{l}_1, \ldots, \mathsf{lev}\,\bar{l}_n \leq i < \mathsf{lev}\,\bar{l}}{\mathsf{C}' = \mathsf{none} \quad \mathsf{M}' = \mathsf{M}^{[i]}\, l}\ \text{Backjump} \qquad \frac{\mathsf{C} \neq \mathsf{none} \quad \bullet \notin \mathsf{M}}{\mathsf{fail}}\ \text{Fail}$$

$$F_0 \quad = \quad p_1, \quad \overline{p}_1 \vee p_2, \quad \overline{p}_3 \vee p_4, \quad \overline{p}_5 \vee \overline{p}_6, \quad \overline{p}_1 \vee \overline{p}_5 \vee p_7, \quad \overline{p}_2 \vee \overline{p}_5 \vee p_6 \vee \overline{p}_7$$

CDCL Execution Example

$$\frac{\mathrm{C} = \text{none} \quad l_1 \vee \cdots \vee l_n \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathrm{M}}{\mathrm{C}' = l_1 \vee \cdots \vee l_n}\ \text{Conflict}$$

$$\frac{\mathrm{C} = l \vee D \quad l_1 \vee \cdots \vee l_n \vee \bar{l} \in \mathrm{F} \quad \bar{l}_1, \ldots, \bar{l}_n \prec_{\mathrm{M}} \bar{l}}{\mathrm{C}' = l_1 \vee \cdots \vee l_n \vee D}\ \text{Explain}$$

$$\frac{\mathrm{C} = l_1 \vee \cdots \vee l_n \vee l \quad \text{lev}\,\bar{l}_1, \ldots, \text{lev}\,\bar{l}_n \leq i < \text{lev}\,\bar{l}}{\mathrm{C}' = \text{none} \quad \mathrm{M}' = \mathrm{M}^{[i]}\, l}\ \text{Backjump} \qquad \frac{\mathrm{C} \neq \text{none} \quad \bullet \notin \mathrm{M}}{\text{fail}}\ \text{Fail}$$

$$F_0 = p_1, \quad \bar{p}_1 \vee p_2, \quad \bar{p}_3 \vee p_4, \quad \bar{p}_5 \vee \bar{p}_6, \quad \bar{p}_1 \vee \bar{p}_5 \vee p_7, \quad \bar{p}_2 \vee \bar{p}_5 \vee p_6 \vee \bar{p}_7$$

	M	F	C	Rule
1	ε	F_0	none	Propagate on p_1
2	p_1	F_0	none	Propagate on $\bar{p}_1 \vee p_2$
3	$p_1\ p_2$	F_0	none	Decide p_3
4	$p_1\ p_2 \bullet p_3$	F_0	none	Propagate on $\bar{p}_3 \vee p_4$
5	$p_1\ p_2 \bullet p_3\ p_4$	F_0	none	Decide p_5
6	$p_1\ p_2 \bullet p_3\ p_4 \bullet p_5$	F_0	none	Propagate on $\bar{p}_5 \vee \bar{p}_6$
7	$p_1\ p_2 \bullet p_3\ p_4 \bullet p_5\ \bar{p}_6$	F_0	none	Propagate on $\bar{p}_1 \vee \bar{p}_4 \vee p_7$

CDCL Execution Example

$$\frac{\mathsf{C} = \text{none} \quad l_1 \vee \cdots \vee l_n \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \in \mathsf{M}}{\mathsf{C}' = l_1 \vee \cdots \vee l_n}\ \text{Conflict}$$

$$\frac{\mathsf{C} = l \vee D \quad l_1 \vee \cdots \vee l_n \vee \bar{l} \in \mathsf{F} \quad \bar{l}_1, \ldots, \bar{l}_n \prec_{\mathsf{M}} \bar{l}}{\mathsf{C}' = l_1 \vee \cdots \vee l_n \vee D}\ \text{Explain}$$

$$\frac{\mathsf{C} = l_1 \vee \cdots \vee l_n \vee l \quad \text{lev}\,\bar{l}_1, \ldots, \text{lev}\,\bar{l}_n \leq i < \text{lev}\,\bar{l}}{\mathsf{C}' = \text{none} \quad \mathsf{M}' = \mathsf{M}^{[i]}\, l}\ \text{Backjump} \qquad \frac{\mathsf{C} \neq \text{none} \quad \bullet \notin \mathsf{M}}{\text{fail}}\ \text{Fail}$$

$$F_0 \quad = \quad p_1, \quad \bar{p}_1 \vee p_2, \quad \bar{p}_3 \vee p_4, \quad \bar{p}_5 \vee \bar{p}_6, \quad \bar{p}_1 \vee \bar{p}_5 \vee p_7, \quad \bar{p}_2 \vee \bar{p}_5 \vee p_6 \vee \bar{p}_7$$

	M	F	C	Rule
7	$p_1\ p_2 \bullet p_3\ p_4 \bullet p_5\ \bar{p}_6$	F_0	none	Propagate on $\bar{p}_1 \vee \bar{p}_4 \vee p_7$
8	$p_1\ p_2 \bullet p_3\ p_4 \bullet p_5\ \bar{p}_6\ p_7$	F_0	none	Conflict on $\bar{p}_2 \vee \bar{p}_5 \vee p_6 \vee \bar{p}_7$
9	$p_1\ p_2 \bullet p_3\ p_4 \bullet p_5\ \bar{p}_6\ p_7$	F_0	$\bar{p}_2 \vee \bar{p}_5 \vee p_6 \vee \bar{p}_7$	Explain with $\bar{p}_1 \vee \bar{p}_5 \vee p_7$
10	$p_1\ p_2 \bullet p_3\ p_4 \bullet p_5\ \bar{p}_6\ p_7$	F_0	$\bar{p}_1 \vee \bar{p}_2 \vee \bar{p}_5 \vee p_6$	Explain with $\bar{p}_5 \vee \bar{p}_6$
11	$p_1\ p_2 \bullet p_3\ p_4 \bullet p_5\ \bar{p}_6\ p_7$	F_0	$\bar{p}_1 \vee \bar{p}_2 \vee \bar{p}_5$	Backjump
12	$p_1\ p_2\ \bar{p}_5$	F_0	none	Decide p_3
13	$p_1\ p_2\ \bar{p}_5 \bullet p_3$	F_0	none	...

CDCL rules with learning

Also add

$$\frac{\mathsf{F} \models_{\mathrm{p}} C \quad C \notin \mathsf{F}}{\mathsf{F}' = \mathsf{F} \cup \{C\}}\ \text{Learn}$$

$$\frac{\mathsf{C} = \mathsf{none} \quad \mathsf{F} = G \cup \{C\} \quad G \models_{\mathrm{p}} C}{\mathsf{F}' = G}\ \text{Forget}$$

$$\frac{}{\mathsf{M}' = \mathsf{M}^{[0]} \quad \mathsf{C}' = \mathsf{none}}\ \text{Restart}$$

Note: Learn can be applied to *any* clause stored in C when $\mathsf{C} \neq \mathsf{none}$

Modeling Modern SAT Solvers

At their core, modern SAT solvers are implementations of the CDCL transition system with rules

Propagate, Decide, Conflict, Explain, Backjump,

Learn, Forget, Restart

$$\textit{Basic CDCL} \stackrel{\text{def}}{=} \{\ \text{Propagate, Decide, Conflict, Explain, Backjump}\ \}$$

$$\textit{CDCL} \stackrel{\text{def}}{=} \text{Basic CDCL} + \{\ \text{Learn, Forget, Restart}\ \}$$

Modeling Modern SAT Solvers

At their core, modern SAT solvers are implementations of the CDCL transition system with rules

Propagate, Decide, Conflict, Explain, Backjump,

Learn, Forget, Restart

$$\textit{Basic CDCL} \stackrel{\text{def}}{=} \{\text{Propagate}, \text{Decide}, \text{Conflict}, \text{Explain}, \text{Backjump}\}$$

$$\textit{CDCL} \stackrel{\text{def}}{=} \text{Basic CDCL} + \{\text{Learn}, \text{Forget}, \text{Restart}\}$$

The Basic CDCL System – Correctness

Irreducible state: state to which no Basic CDCL rules apply

Execution: sequence of transitions allowed by the rules and starting with $\mathsf{M} = \varepsilon$ and $\mathsf{C} = \mathsf{none}$

Exhausted execution: execution ending in an irreducible state

The Basic CDCL System – Correctness

Irreducible state: state to which no Basic CDCL rules apply

Execution: sequence of transitions allowed by the rules and starting with $M = \varepsilon$ and $C = \text{none}$

Exhausted execution: execution ending in an irreducible state

Theorem 1 (Strong Termination)
Every execution in Basic CDCL is finite.

Note: This is not so immediate, because of Backjump

The Basic CDCL System – Correctness

Irreducible state: state to which no Basic CDCL rules apply

Execution: sequence of transitions allowed by the rules and starting with $\mathsf{M} = \varepsilon$ and $\mathsf{C} = \mathsf{none}$

Exhausted execution: execution ending in an irreducible state

Theorem 1 (Strong Termination)

Every execution in Basic CDCL is finite.

Lemma 2

Every exhausted execution ends with either $\mathsf{C} = \mathsf{none}$ *or* fail.

The Basic CDCL System – Correctness

Irreducible state: state to which no Basic CDCL rules apply

Execution: sequence of transitions allowed by the rules and starting with $\mathsf{M} = \varepsilon$ and $\mathsf{C} = \mathsf{none}$

Exhausted execution: execution ending in an irreducible state

Theorem 1 (Soundness)

For every exhausted execution starting with $\mathsf{F} = F_0$ *and ending with* fail, *the clause set* F_0 *is unsatisfiable.*

The Basic CDCL System – Correctness

Irreducible state: state to which no Basic CDCL rules apply

Execution: sequence of transitions allowed by the rules and starting with $\mathsf{M} = \varepsilon$ and $\mathsf{C} = \mathsf{none}$

Exhausted execution: execution ending in an irreducible state

Theorem 1 (Soundness)

For every exhausted execution starting with $\mathsf{F} = F_0$ and ending with fail, the clause set F_0 is unsatisfiable.

Theorem 2 (Completeness)

For every exhausted execution starting with $\mathsf{F} = F_0$ and ending with $\mathsf{C} = \mathsf{none}$, the clause set F_0 is satisfied by M.

The CDCL System – Strategies

Applying

- one Basic CDCL rule between each two Learn applications and
- Restart less and less often

ensures termination

The CDCL System – Strategies

A common basic strategy applies the rules with the following priorities:

1. If $n > 0$ conflicts have been found so far,
   increase n and apply Restart
2. If a clause is falsified by M, apply Conflict
3. Keep applying Explain until Backjump is applicable
4. Apply Learn
5. Apply Backjump
6. Apply Propagate to completion
7. Apply Decide

The CDCL System – Correctness

Theorem 3 (Termination)
Every execution in which
(a) Learn/Forget are applied only finitely many times and
(b) Restart is applied with increased periodicity
is finite.

Theorem 4 (Soundness)
As before.

Theorem 5 (Completeness)
As before.

The CDCL System – Correctness

Theorem 3 (Termination)

Every execution in which
(a) Learn/Forget are applied only finitely many times and
(b) Restart is applied with increased periodicity
is finite.

Theorem 4 (Soundness)

As before.

Theorem 5 (Completeness)

As before.

From SAT to SMT

Same sort of states $\langle \mathsf{M}, \mathsf{C}, \mathsf{F} \rangle$ and transitions as in CDCL system

Differences:

- F contains quantifier-free clauses from some theory T
- M is a sequence of theory literals and decision points
- the CDCL system augmented with rules

  T-Conflict, T-Propagate, T-Explain
- maintains invariant: $\mathsf{F} \models_T \mathsf{C}$ and $\mathsf{M} \models_{\mathrm{p}} \neg\mathsf{C}$ when $\mathsf{C} \neq \mathsf{none}$

Recall: $F \models_T G$ iff every model of T that satisfies F satisfies G as well

SMT-level Rules

A theory T

SMT-level Rules

A theory T

$$\frac{C = \text{none} \quad l_1, \ldots, l_n \in M \quad l_1, \ldots, l_n \models_T \bot}{C := \bar{l}_1 \vee \cdots \vee \bar{l}_n}\ T\text{-Conflict}$$

Note: $\models_T$ is decided by theory solver

SMT-level Rules

A theory T

$$\frac{\mathrm{C} = \mathrm{none} \quad l_1, \ldots, l_n \in \mathrm{M} \quad l_1, \ldots, l_n \models_T \bot}{\mathrm{C} := \bar{l}_1 \vee \cdots \vee \bar{l}_n}\ T\text{-Conflict}$$

$$\frac{l \in \mathrm{Lit}(\mathrm{F}) \quad \mathrm{M} \models_T l \quad l, \bar{l} \notin \mathrm{M}}{\mathrm{M} := \mathrm{M}\ l}\ T\text{-Propagate}$$

Note: $\models_T$ is decided by theory solver

SMT-level Rules

A theory T

$$\frac{\mathsf{C} = \text{none} \quad l_1, \ldots, l_n \in \mathsf{M} \quad l_1, \ldots, l_n \models_T \bot}{\mathsf{C} := \bar{l}_1 \vee \cdots \vee \bar{l}_n} \; T\text{-Conflict}$$

$$\frac{l \in \mathrm{Lit}(\mathsf{F}) \quad \mathsf{M} \models_T l \quad l, \bar{l} \notin \mathsf{M}}{\mathsf{M} := \mathsf{M}\, l} \; T\text{-Propagate}$$

$$\frac{\mathsf{C} = l \vee D \quad \bar{l}_1, \ldots, \bar{l}_n \models_T \bar{l} \quad \bar{l}_1, \ldots, \bar{l}_n \prec_{\mathsf{M}} \bar{l}}{\mathsf{C} := l_1 \vee \cdots \vee l_n \vee D} \; T\text{-Explain}$$

Note: $\models_T$ is decided by theory solver

Modeling the Very Lazy Theory Approach

T-Conflict is enough to model the naive integration of SAT solvers and theory solvers seen in the earlier EUF example

Modeling the Very Lazy Theory Approach

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

$$F_0 \quad = \quad p_1, \quad \overline{p}_2 \vee p_3, \quad \overline{p}_4$$

Modeling the Very Lazy Theory Approach

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

$$F_0 \quad = \quad p_1, \quad \overline{p}_2 \vee p_3, \quad \overline{p}_4$$

	M	F	C	rule
0	ε	F_0	none	Propagate$^+$
1	$p_1\ \overline{p}_4$	F_0	none	Decide
2	$p_1\ \overline{p}_4 \bullet \overline{p}_2$	F_0	none	T-Conflict
3	$p_1\ \overline{p}_4 \bullet \overline{p}_2$	F_0	$\overline{p}_1 \vee p_2 \vee p_4$	Learn
4	$p_1\ \overline{p}_4 \bullet \overline{p}_2$	$F_0,\ \overline{p}_1 \vee p_2 \vee p_4$	$\overline{p}_1 \vee p_2 \vee p_4$	Restart
5	$p_1\ \overline{p}_4$	$F_0,\ \overline{p}_1 \vee p_2 \vee p_4$	none	Propagate$^+$
6	$p_1\ \overline{p}_4\ p_2\ p_3$	$F_0,\ \overline{p}_1 \vee p_2 \vee p_4$	none	T-Conflict
7	$p_1\ \overline{p}_4\ p_2\ p_3$	$F_0,\ \overline{p}_1 \vee p_2 \vee p_4$	$\overline{p}_1 \vee \overline{p}_3 \vee p_4$	Learn
8	$p_1\ \overline{p}_4\ p_2\ p_3$	$F_0,\ \overline{p}_1 \vee p_2 \vee p_4,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4$	$\overline{p}_1 \vee \overline{p}_3 \vee p_4$	Restart
9	$p_1\ \overline{p}_4\ p_2\ p_3$	$F_0,\ \overline{p}_1 \vee p_2 \vee p_4,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4$	none	Conflict
10	$p_1\ \overline{p}_4\ p_2\ p_3$	$F_0,\ \overline{p}_1 \vee p_2 \vee p_4,\ \overline{p}_1 \vee \overline{p}_3 \vee p_4$	$\overline{p}_1 \vee \overline{p}_3 \vee p_4$	Fail
11		fail		

A Better Lazy Approach

The very lazy approach can be improved considerably with

- An *on-line* SAT engine,
  which can accept new input clauses on the fly
- an *incremental and explicating* T-solver,
  which can

A Better Lazy Approach

The very lazy approach can be improved considerably with

- An *on-line* SAT engine,
  which can accept new input clauses on the fly

- an *incremental and explicating* $\mathcal{T}$-solver,
  which can

A Better Lazy Approach

The very lazy approach can be improved considerably with

- An *on-line* SAT engine,
  which can accept new input clauses on the fly
- an *incremental and explicating* T-solver,
  which can
  1. check the T-satisfiability of M as it is extended and
  2. identify a small T-unsatisfiable subset of M
     once M becomes T-unsatisfiable

A Better Lazy Approach

The very lazy approach can be improved considerably with

- An *on-line* SAT engine,
  which can accept new input clauses on the fly
- an *incremental and explicating* T-solver,
  which can
  1. check the T-satisfiability of M as it is extended and
  2. identify a small T-unsatisfiable subset of M
     once M becomes T-unsatisfiable

A Better Lazy Approach

The very lazy approach can be improved considerably with

- An *on-line* SAT engine,
  which can accept new input clauses on the fly
- an *incremental and explicating* T-solver,
  which can
  1. check the T-satisfiability of M as it is extended and
  2. identify a small T-unsatisfiable subset of M
     once M becomes T-unsatisfiable

A Better Lazy Approach

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

$$F_0 \quad = \quad p_1, \quad \overline{p}_2 \vee p_3, \quad \overline{p}_4$$

	M	F	C	rule
1	ε	F_0	none	Propagate$^+$
2	$p_1\ \overline{p}_4$	F_0	none	Decide
3	$p_1\ \overline{p}_4 \bullet \overline{p}_2$	F_0	none	T-Conflict
4	$p_1\ \overline{p}_4 \bullet \overline{p}_2$	F_0	$\overline{p}_1 \vee p_2$	Backjump
5	$p_1\ \overline{p}_4\ p_2$	F_0	none	Propagate
6	$p_1\ \overline{p}_4\ p_2\ p_3$	F_0	none	T-Conflict
7	$p_1\ \overline{p}_4\ p_2\ p_3$	F_0	$\overline{p}_1 \vee \overline{p}_3 \vee p_4$	Fail
8		fail		

Lazy Approach – Strategies

Ignoring Restart, for simplicity,

a common strategy is to apply the rules using the following priorities:

1. If a clause is falsified by the current assignment M, apply Conflict
2. If M is T-unsatisfiable, apply T-Conflict
3. Apply Fail or Explain+Learn+Backjump as appropriate
4. Apply Propagate
5. Apply Decide

Lazy Approach – Strategies

Ignoring Restart, for simplicity,

a common strategy is to apply the rules using the following priorities:

1. If a clause is falsified by the current assignment M, apply Conflict
2. If M is T-unsatisfiable, apply T-Conflict
3. Apply Fail or Explain+Learn+Backjump as appropriate
4. Apply Propagate
5. Apply Decide

Note: Depending on the cost of checking the T-satisfiability of M, Step (2) can be applied with lower frequency or priority

Theory Propagation

With T-Conflict as the only theory rule, the theory solver is used just to validate the choices of the SAT solver

Theory Propagation

With T-Conflict as the only theory rule, the theory solver is used just to validate the choices of the SAT solver

With T-Propagate and T-Explain, it can also be used to guide the solver search

$$\frac{l \in \mathrm{Lit}(\mathsf{F}) \quad \mathsf{M} \models_T l \quad l, \bar{l} \notin \mathsf{M}}{\mathsf{M} := \mathsf{M}\ l}\ T\text{-Propagate}$$

$$\frac{\mathsf{C} = l \vee D \quad \bar{l}_1, \ldots, \bar{l}_n \models_T \bar{l} \quad \bar{l}_1, \ldots, \bar{l}_n \prec_{\mathsf{M}} \bar{l}}{\mathsf{C} := l_1 \vee \cdots \vee l_n \vee D}\ T\text{-Explain}$$

Theory Propagation Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

$$F_0 \quad = \quad p_1, \quad \overline{p}_2 \vee p_3, \quad \overline{p}_4$$

Theory Propagation Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

$$F_0 \quad = \quad p_1, \quad \overline{p}_2 \vee p_3, \quad \overline{p}_4$$

	M	F	C	rule
1	ε	F_0	none	Propagate
1	p_1	F_0	none	Propagate
1	$p_1\ \overline{p}_4$	F_0	none	T-Propagate $(p_1 \models_T p_2)$
1	$p_1\ \overline{p}_4\ p_2$	F_0	none	T-Propagate $(p_1,\ \overline{p}_4 \models_T \overline{p}_3)$
1	$p_1\ \overline{p}_4\ p_2\ \overline{p}_3$	F_0	none	Conflict
1	$p_1\ \overline{p}_4\ p_2\ \overline{p}_3$	F_0	$\overline{p}_2 \vee p_3$	Fail
		fail		

Theory Propagation Example

$$\underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

$$F_0 \quad = \quad p_1, \quad \overline{p}_2 \vee p_3, \quad \overline{p}_4$$

	M	F	C	rule
1	ε	F_0	none	Propagate
1	p_1	F_0	none	Propagate
1	$p_1\ \overline{p}_4$	F_0	none	T-Propagate $(p_1 \models_T p_2)$
1	$p_1\ \overline{p}_4\ p_2$	F_0	none	T-Propagate $(p_1,\ \overline{p}_4 \models_T \overline{p}_3)$
1	$p_1\ \overline{p}_4\ p_2\ \overline{p}_3$	F_0	none	Conflict
1	$p_1\ \overline{p}_4\ p_2\ \overline{p}_3$	F_0	$\overline{p}_2 \vee p_3$	Fail
		fail		

Note: T-propagation eliminates search altogether in this case, no applications of Decide are needed!

Theory Propagation Example 2

$$\underbrace{g(a) = e}_{p_0} \vee \underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

$$F_0 \quad = \quad p_0 \vee p_1, \quad \overline{p}_2 \vee p_3, \quad \overline{p}_4$$

Theory Propagation Example 2

$$\underbrace{g(a) = e}_{p_0} \vee \underbrace{g(a) = c}_{p_1} \quad \wedge \quad \underbrace{f(g(a)) \neq f(c)}_{\overline{p}_2} \vee \underbrace{g(a) = d}_{p_3} \quad \wedge \quad \underbrace{c \neq d}_{\overline{p}_4}$$

$$F_0 \quad = \quad p_0 \vee p_1, \quad \overline{p}_2 \vee p_3, \quad \overline{p}_4$$

	M	F	C	rule
1	ε	F_0	none	Propagate
2	$\overline{p}_4$	F_0	none	Decide
3	$\overline{p}_4 \bullet p_1$	F_0	none	T-Propagate $(p_1 \models_T p_2)$
4	$\overline{p}_4 \bullet p_1\ p_2$	F_0	none	T-Propagate $(p_1,\ \overline{p}_4 \models_T \overline{p}_3)$
5	$\overline{p}_4 \bullet p_1\ p_2\ \overline{p}_3$	F_0	none	Conflict
6	$\overline{p}_4 \bullet p_1\ p_2\ \overline{p}_3$	F_0	$\overline{p}_2 \vee p_3$	T-Explain
7	$\overline{p}_4 \bullet p_1\ p_2\ \overline{p}_3$	F_0	$\overline{p}_1 \vee p_3$	T-Explain
8	$\overline{p}_4 \bullet p_1\ p_2\ \overline{p}_3$	F_0	$\overline{p}_1 \vee p_4$	Backjump
9	$\overline{p}_4\ \overline{p}_1$	F_0	none	$\cdots$
$\cdots$				

Theory Propagation Features

- With exhaustive theory propagation every assignment M is T-satisfiable (since $M\ l$ is T-unsatisfiable iff $M \models_T \bar{l}$)
- For theory propagation to be effective in practice, it needs specialized theory solvers
- For some theories, e.g., difference logic, detecting T-entailed literals is cheap and so theory propagation is extremely effective
- For others, e.g., the theory of equality, detecting all T-entailed literals is too expensive
- If T-Propagate is not applied exhaustively, T-Conflict is needed to repair T-unsatisfiable assignments

Theory Propagation Features

- With exhaustive theory propagation every assignment M is T-satisfiable (since $M\ l$ is T-unsatisfiable iff $M \models_T \bar{l}$)
- For theory propagation to be effective in practice, it needs specialized theory solvers
- For some theories, e.g., difference logic, detecting T-entailed literals is cheap and so theory propagation is extremely effective
- For others, e.g., the theory of equality, detecting all T-entailed literals is too expensive
- If T-Propagate is not applied exhaustively, T-Conflict is needed to repair T-unsatisfiable assignments

Theory Propagation Features

- With exhaustive theory propagation every assignment M is T-satisfiable (since $M\,l$ is T-unsatisfiable iff $M \models_T \bar{l}$)
- For theory propagation to be effective in practice, it needs specialized theory solvers
- For some theories, e.g., difference logic, detecting T-entailed literals is cheap and so theory propagation is extremely effective
- For others, e.g., the theory of equality, detecting all T-entailed literals is too expensive
- If T-Propagate is not applied exhaustively, T-Conflict is needed to repair T-unsatisfiable assignments

Theory Propagation Features

- With exhaustive theory propagation every assignment M is T-satisfiable (since $M\ l$ is T-unsatisfiable iff $M \models_T \bar{l}$)
- For theory propagation to be effective in practice, it needs specialized theory solvers
- For some theories, e.g., difference logic, detecting T-entailed literals is cheap and so theory propagation is extremely effective
- For others, e.g., the theory of equality, detecting all T-entailed literals is too expensive
- If T-Propagate is not applied exhaustively, T-Conflict is needed to repair T-unsatisfiable assignments

Theory Propagation Features

- With exhaustive theory propagation every assignment M is T-satisfiable (since $M\ l$ is T-unsatisfiable iff $M \models_T \bar{l}$)
- For theory propagation to be effective in practice, it needs specialized theory solvers
- For some theories, e.g., difference logic, detecting T-entailed literals is cheap and so theory propagation is extremely effective
- For others, e.g., the theory of equality, detecting all T-entailed literals is too expensive
- If T-Propagate is not applied exhaustively, T-Conflict is needed to repair T-unsatisfiable assignments

Modeling Modern Lazy SMT Solvers

At their core,

modern lazy SMT solvers are implementations of the transition system with rules

(1) Propagate, Decide, Conflict, Explain, Backjump, Fail

(2) T-Conflict, T-Propagate, T-Explain

(3) Learn, Forget, Restart

Modeling Modern Lazy SMT Solvers

At their core,

modern lazy SMT solvers are implementations of the transition system with rules

(1) Propagate, Decide, Conflict, Explain, Backjump, Fail

(2) T-Conflict, T-Propagate, T-Explain

(3) Learn, Forget, Restart

Basic CDCL Modulo Theories $\stackrel{\text{def}}{=}$ (1) + (2)

CDCL Modulo Theories $\stackrel{\text{def}}{=}$ (1) + (2) + (3)

Correctness of CDCL Modulo Theories

Irreducible state: state to which no Basic CDCL MT rules apply

Execution: sequence of transitions allowed by the rules and starting with $M = \varepsilon$ and $C = \text{none}$

Exhausted execution: execution ending in an irreducible state

Correctness of CDCL Modulo Theories

Irreducible state: state to which no Basic CDCL MT rules apply

Execution: sequence of transitions allowed by the rules and starting with $\mathsf{M} = \varepsilon$ and $\mathsf{C} = \mathsf{none}$

Exhausted execution: execution ending in an irreducible state

Theorem 6 (Termination)

Every execution in which
(a) Learn/Forget are applied only finitely many times and
(b) Restart is applied with increased periodicity
is finite.

Lemma 7

Every exhausted execution ends with either $\mathsf{C} = \mathsf{none}$ *or* fail.

Correctness of CDCL Modulo Theories

Irreducible state: state to which no Basic CDCL MT rules apply

Execution: sequence of transitions allowed by the rules and starting with $\mathsf{M} = \varepsilon$ and $\mathsf{C} = \mathsf{none}$

Exhausted execution: execution ending in an irreducible state

Theorem 6 (Soundness)

For every exhausted execution starting with $\mathsf{F} = F_0$ *and ending with* fail, *the clause set* F_0 *is* T*-unsatisfiable.*

Theorem 7 (Completeness)

For every exhausted execution starting with $\mathsf{F} = F_0$ *and ending with* $\mathsf{C} = \mathsf{none}$, F_0 *is* T*-satisfiable; specifically,* M *is* T*-satisfiable and* $\mathsf{M} \models_{\mathrm{p}} F_0$.