

Decision Procedures for Verification

Part 1. Propositional Logic (2)

27.10.2022

Viorica Sofronie-Stokkermans

sofronie@uni-koblenz.de

Last time

Propositional Logic

1.1 Syntax

- Language
 - propositional variables
 - logical symbols \Rightarrow Boolean combinations
- Propositional Formulae

1.2 Semantics

- Valuations
- Truth value of a formula in a valuation
- Models, Validity, and Satisfiability
- Some important equivalences
- CNF/DNF

1.4 Normal Forms

- CNF and DNF
- Computing CNF/DNF by rewriting the formulae
- Structure-Preserving Translation for CNF
- Optimized translation using polarity

1.4 Normal Forms

We define **conjunctions** of formulas as follows:

$$\bigwedge_{i=1}^0 F_i = \top.$$

$$\bigwedge_{i=1}^1 F_i = F_1.$$

$$\bigwedge_{i=1}^{n+1} F_i = \bigwedge_{i=1}^n F_i \wedge F_{n+1}.$$

and analogously **disjunctions**:

$$\bigvee_{i=1}^0 F_i = \perp.$$

$$\bigvee_{i=1}^1 F_i = F_1.$$

$$\bigvee_{i=1}^{n+1} F_i = \bigvee_{i=1}^n F_i \vee F_{n+1}.$$

Literals and Clauses

A **literal** is either a propositional variable P or a negated propositional variable $\neg P$.

A **clause** is a (possibly empty) disjunction of literals.

CNF and DNF

A formula is in **conjunctive normal form (CNF, clause normal form)**, if it is a conjunction of disjunctions of literals (or in other words, a conjunction of clauses).

A formula is in **disjunctive normal form (DNF)**, if it is a disjunction of conjunctions of literals.

CNF and DNF

Checking the validity of CNF formulas or the unsatisfiability of DNF formulas is easy:

A formula in CNF is valid, if and only if each of its disjunctions contains a pair of complementary literals P and $\neg P$.

Conversely, a formula in DNF is unsatisfiable, if and only if each of its conjunctions contains a pair of complementary literals P and $\neg P$.

On the other hand, checking the unsatisfiability of CNF formulas or the validity of DNF formulas is known to be coNP-complete.

Conversion to CNF/DNF

Proposition 1.8:

For every formula there is an equivalent formula in CNF (and also an equivalent formula in DNF).

Proof:

We consider the case of CNF.

Apply the following rules as long as possible (modulo associativity and commutativity of \wedge and \vee):

Step 1: Eliminate equivalences:

$$(F \leftrightarrow G) \Rightarrow_K (F \rightarrow G) \wedge (G \rightarrow F)$$

Conversion to CNF/DNF

Step 2: Eliminate implications:

$$(F \rightarrow G) \Rightarrow_K (\neg F \vee G)$$

Step 3: Push negations downward:

$$\neg(F \vee G) \Rightarrow_K (\neg F \wedge \neg G)$$

$$\neg(F \wedge G) \Rightarrow_K (\neg F \vee \neg G)$$

Step 4: Eliminate multiple negations:

$$\neg\neg F \Rightarrow_K F$$

The formula obtained from a formula F after applying steps 1-4 is called the **negation normal form (NNF)** of F

Conversion to CNF/DNF

Step 5: Push disjunctions downward:

$$(F \wedge G) \vee H \Rightarrow_K (F \vee H) \wedge (G \vee H)$$

Step 6: Eliminate \top and \perp :

$$(F \wedge \top) \Rightarrow_K F$$

$$(F \wedge \perp) \Rightarrow_K \perp$$

$$(F \vee \top) \Rightarrow_K \top$$

$$(F \vee \perp) \Rightarrow_K F$$

$$\neg \perp \Rightarrow_K \top$$

$$\neg \top \Rightarrow_K \perp$$

Conversion to CNF/DNF

Proving termination is easy for most of the steps; only step 3 and step 5 are a bit more complicated.

The resulting formula is equivalent to the original one and in CNF.

The conversion of a formula to DNF works in the same way, except that disjunctions have to be pushed downward in step 5.

Complexity

Conversion to CNF (or DNF) may produce a formula whose size is **exponential** in the size of the original one.

Satisfiability-preserving Transformations

The goal

“find a formula G in CNF such that $\models F \leftrightarrow G$ ”

is unpractical.

But if we relax the requirement to

“find a formula G in CNF such that $F \models \perp$ iff $G \models \perp$ ”

we can get an efficient transformation.

Satisfiability-preserving Transformations

Idea:

A formula $F[F']$ is satisfiable iff $F[P] \wedge (P \leftrightarrow F')$ is satisfiable (where P new propositional variable that works as abbreviation for F').

We can use this rule recursively for all subformulas in the original formula (this introduces a linear number of new propositional variables).

Conversion of the resulting formula to CNF increases the size only by an additional factor (each formula $P \leftrightarrow F'$ gives rise to at most one application of the distributivity law).

Optimized Transformations

A further improvement is possible by taking the **polarity** of the subformula F into account.

Assume that F contains neither \rightarrow nor \leftrightarrow . A subformula F' of F has **positive polarity** in F , if it occurs below an even number of negation signs; it has **negative polarity** in F , if it occurs below an odd number of negation signs.

Optimized Transformations

Proposition 1.9:

Let $F[F']$ be a formula containing neither \rightarrow nor \leftrightarrow ; let P be a propositional variable not occurring in $F[F']$.

If F' has positive polarity in F , then $F[F']$ is satisfiable if and only if $F[P] \wedge (P \rightarrow F')$ is satisfiable.

If F' has negative polarity in F , then $F[F']$ is satisfiable if and only if $F[P] \wedge (F' \rightarrow P)$ is satisfiable.

Proof:

Exercise.

This satisfiability-preserving transformation to clause form is also called **structure-preserving transformation to clause form**.

Optimized Transformations

Example: Let $F = (Q_1 \wedge Q_2) \vee (R_1 \wedge R_2)$.

The following are equivalent:

- $F \models \perp$
- $P_F \wedge (P_F \leftrightarrow (P_{Q_1 \wedge Q_2} \vee P_{R_1 \wedge R_2})) \wedge (P_{Q_1 \wedge Q_2} \leftrightarrow (Q_1 \wedge Q_2))$
 $\wedge (P_{R_1 \wedge R_2} \leftrightarrow (R_1 \wedge R_2)) \models \perp$
- $P_F \wedge (P_F \rightarrow (P_{Q_1 \wedge Q_2} \vee P_{R_1 \wedge R_2})) \wedge (P_{Q_1 \wedge Q_2} \rightarrow (Q_1 \wedge Q_2))$
 $\wedge (P_{R_1 \wedge R_2} \rightarrow (R_1 \wedge R_2)) \models \perp$
- $P_F \wedge (\neg P_F \vee P_{Q_1 \wedge Q_2} \vee P_{R_1 \wedge R_2}) \wedge (\neg P_{Q_1 \wedge Q_2} \vee Q_1) \wedge (\neg P_{Q_1 \wedge Q_2} \vee Q_2)$
 $\wedge (\neg P_{R_1 \wedge R_2} \vee R_1) \wedge (\neg P_{R_1 \wedge R_2} \vee R_2) \models \perp$

Decision Procedures for Satisfiability

- Simple Decision Procedures
truth table method
- The Resolution Procedure
- The Davis-Putnam-Logemann-Loveland Algorithm

1.5 Inference Systems and Proofs

Inference systems Γ (proof calculi) are sets of tuples

$$(F_1, \dots, F_n, F_{n+1}), \quad n \geq 0,$$

called inferences or inference rules, and written

$$\frac{\overbrace{F_1 \dots F_n}^{\text{premises}}}{\underbrace{F_{n+1}}_{\text{conclusion}}} .$$

Clausal inference system: premises and conclusions are clauses. One also considers inference systems over other data structures.

Proofs

A **proof** in Γ of a formula F from a set of formulas N (called **assumptions**) is a sequence F_1, \dots, F_k of formulas where

- (i) $F_k = F$,
- (ii) for all $1 \leq i \leq k$: $F_i \in N$, or else there exists an inference $(F_{i_1}, \dots, F_{i_{n_i}}, F_i)$ in Γ , such that $0 \leq i_j < i$, for $1 \leq j \leq n_i$.

Soundness and Completeness

Provability \vdash_{Γ} of F from N in Γ :

$N \vdash_{\Gamma} F \iff$ there exists a proof Γ of F from N .

Γ is called **sound** \iff

$$\frac{F_1 \dots F_n}{F} \in \Gamma \Rightarrow F_1, \dots, F_n \models F$$

Γ is called **complete** \iff

$$N \models F \Rightarrow N \vdash_{\Gamma} F$$

Γ is called **refutationally complete** \iff

$$N \models \perp \Rightarrow N \vdash_{\Gamma} \perp$$

1.6 The Propositional Resolution Calculus

Resolution inference rule:

$$\frac{C \vee A \quad \neg A \vee D}{C \vee D}$$

Terminology: $C \vee D$: resolvent; A : resolved atom

(Positive) factorisation inference rule:

$$\frac{C \vee A \vee A}{C \vee A}$$

C, D : clauses

A atom (propositional variable)

The Resolution Calculus *Res*

These are **schematic inference rules**; for each substitution of the **schematic variables** C , D , and A , respectively, by propositional clauses and atoms we obtain an inference rule.

As “ \vee ” is considered associative and commutative, we assume that A and $\neg A$ can occur anywhere in their respective clauses.

Sample Refutation

1. $\neg P \vee \neg P \vee Q$ (given)
2. $P \vee Q$ (given)
3. $\neg R \vee \neg Q$ (given)
4. R (given)
5. $\neg P \vee Q \vee Q$ (Res. 2. into 1.)
6. $\neg P \vee Q$ (Fact. 5.)
7. $Q \vee Q$ (Res. 2. into 6.)
8. Q (Fact. 7.)
9. $\neg R$ (Res. 8. into 3.)
10. \perp (Res. 4. into 9.)

Soundness of Resolution

Theorem 1.10. Propositional resolution is sound.

Completeness of Resolution

How to show refutational completeness of propositional resolution:

- We have to show: $N \models \perp \Rightarrow N \vdash_{Res} \perp$,
or equivalently: If $N \not\vdash_{Res} \perp$, then N has a model.
- **Idea:** Suppose that we have computed sufficiently many inferences (and not derived \perp).

Now order the clauses in N according to some appropriate ordering, inspect the clauses in ascending order, and construct a series of valuations.

- The limit valuation can be shown to be a model of N .

Clause Orderings

1. We assume that \succ is any fixed ordering on propositional variables that is *total* and well-founded.
2. Extend \succ to an ordering \succ_L on literals:

$$\begin{array}{l} [\neg]P \succ_L [\neg]Q \quad , \text{ if } P \succ Q \\ \neg P \succ_L P \end{array}$$

3. Extend \succ_L to an ordering \succ_C on clauses:
 $\succ_C = (\succ_L)_{\text{mul}}$, the multi-set extension of \succ_L .

Notation: \succ also for \succ_L and \succ_C .

Multi-Set Orderings

Let (M, \succ) be a partial ordering. The **multi-set extension** of \succ to multi-sets over M is defined by

$$S_1 \succ_{\text{mul}} S_2 :\Leftrightarrow S_1 \neq S_2$$

$$\text{and } \forall m \in M : [S_2(m) > S_1(m)]$$

$$\Rightarrow \exists m' \in M : (m' \succ m \text{ and } S_1(m') > S_2(m'))]$$

Theorem 1.11:

- a) \succ_{mul} is a partial ordering.
- b) \succ well-founded $\Rightarrow \succ_{\text{mul}}$ well-founded
- c) \succ total $\Rightarrow \succ_{\text{mul}}$ total

Proof:

see Baader and Nipkow, page 22–24.

Example

Suppose $P_5 \succ P_4 \succ P_3 \succ P_2 \succ P_1 \succ P_0$. Then:

$$\begin{aligned} & P_0 \vee P_1 \\ \prec & P_1 \vee P_2 \\ \prec & \neg P_1 \vee P_2 \\ \prec & \neg P_1 \vee P_4 \vee P_3 \\ \prec & \neg P_1 \vee \neg P_4 \vee P_3 \\ \prec & \neg P_5 \vee P_5 \end{aligned}$$

Stratified Structure of Clause Sets

Let $A \succ B$. Clause sets are then stratified in this form:

Closure of Clause Sets under Res

$$Res(N) = \{C \mid C \text{ is concl. of a rule in } Res \text{ w/ premises in } N\}$$

$$Res^0(N) = N$$

$$Res^{n+1}(N) = Res(Res^n(N)) \cup Res^n(N), \text{ for } n \geq 0$$

$$Res^*(N) = \bigcup_{n \geq 0} Res^n(N)$$

N is called **saturated** (wrt. resolution), if $Res(N) \subseteq N$.

Proposition 1.12

- (i) $Res^*(N)$ is saturated.
- (ii) Res is refutationally complete, iff for each set N of ground clauses:
clauses:

$$N \models \perp \Leftrightarrow \perp \in Res^*(N)$$

Construction of Interpretations

Given: set N of clauses, atom ordering \succ .

Wanted: Valuation \mathcal{A} such that

- “many” clauses from N are valid in \mathcal{A} ;
- $\mathcal{A} \models N$, if N is saturated and $\perp \notin N$.

Construction according to \succ , starting with the minimal clause.

Main Ideas of the Construction

- Clauses are considered in the order given by \prec . We construct a model for N incrementally.
- When considering C , one already has a partial interpretation I_C (initially $I_C = \emptyset$) available.

In what follows, instead of referring to **partial valuations** \mathcal{A}_C we will refer to **partial interpretations** I_C (the set of atoms which are true in the valuation \mathcal{A}_C).

- If C is true in the partial interpretation I_C , nothing is done. ($\Delta_C = \emptyset$).
- If C is false, one would like to change I_C such that C becomes true.

Example

Let $P_5 \succ P_4 \succ P_3 \succ P_2 \succ P_1 \succ P_0$ (max. literals in red)

Construction of I :

	clauses C	I_C	Δ_C	Remarks
1	$\neg P_0$	\emptyset	\emptyset	true in \mathcal{A}_C
2	$P_0 \vee P_1$	\emptyset	$\{P_1\}$	
3	$P_1 \vee P_2$	$\{P_1\}$	\emptyset	true in \mathcal{A}_C
4	$\neg P_1 \vee P_2$	$\{P_1\}$	$\{P_2\}$	
5	$\neg P_1 \vee \neg P_1 \vee P_3 \vee P_0$	$\{P_1, P_2\}$	$\{P_3\}$	
6	$\neg P_1 \vee \neg P_1 \vee P_3 \vee P_3 \vee P_0$	$\{P_1, P_2, P_3\}$	\emptyset	true in \mathcal{A}_C
7	$\neg P_1 \vee P_4 \vee P_3 \vee P_0$	$\{P_1, P_2, P_3\}$	\emptyset	true in \mathcal{A}_C
8	$\neg P_1 \vee \neg P_4 \vee P_3$	$\{P_1, P_2, P_3\}$	\emptyset	true in \mathcal{A}_C
9	$\neg P_3 \vee P_5$	$\{P_1, P_2, P_3\}$	$\{P_5\}$	

The resulting $I = \{P_1, P_2, P_3, P_5\}$ is a model of the clause set.

Example

Let $P_5 \succ P_4 \succ P_3 \succ P_2 \succ P_1 \succ P_0$ (max. literals in red)

	clauses C	$I_C = \mathcal{A}_C^{-1}(1)$	Δ_C	Remarks
1	$\neg P_0$			
2	$P_0 \vee P_1$			
3	$P_1 \vee P_2$			
4	$\neg P_1 \vee P_2$			
5	$\neg P_1 \vee P_4 \vee P_3 \vee P_0$			
6	$\neg P_1 \vee \neg P_4 \vee P_3$			
7	$\neg P_1 \vee P_5$			

Example

Let $P_5 \succ P_4 \succ P_3 \succ P_2 \succ P_1 \succ P_0$ (max. literals in red)

	clauses C	$I_C = \mathcal{A}_C^{-1}(1)$	Δ_C	Remarks
1	$\neg P_0$	\emptyset	\emptyset	true in \mathcal{A}_C
2	$P_0 \vee P_1$			
3	$P_1 \vee P_2$			
4	$\neg P_1 \vee P_2$			
5	$\neg P_1 \vee P_4 \vee P_3 \vee P_0$			
6	$\neg P_1 \vee \neg P_4 \vee P_3$			
7	$\neg P_1 \vee P_5$			

Example

Let $P_5 \succ P_4 \succ P_3 \succ P_2 \succ P_1 \succ P_0$ (max. literals in red)

	clauses C	$I_C = \mathcal{A}_C^{-1}(1)$	Δ_C	Remarks
1	$\neg P_0$	\emptyset	\emptyset	true in \mathcal{A}_C
2	$P_0 \vee P_1$	\emptyset	$\{P_1\}$	P_1 maximal
3	$P_1 \vee P_2$			
4	$\neg P_1 \vee P_2$			
5	$\neg P_1 \vee P_4 \vee P_3 \vee P_0$			
6	$\neg P_1 \vee \neg P_4 \vee P_3$			
7	$\neg P_1 \vee P_5$			

Example

Let $P_5 \succ P_4 \succ P_3 \succ P_2 \succ P_1 \succ P_0$ (max. literals in red)

	clauses C	$I_C = \mathcal{A}_C^{-1}(1)$	Δ_C	Remarks
1	$\neg P_0$	\emptyset	\emptyset	true in \mathcal{A}_C
2	$P_0 \vee P_1$	\emptyset	$\{P_1\}$	P_1 maximal
3	$P_1 \vee P_2$	$\{P_1\}$	\emptyset	true in \mathcal{A}_C
4	$\neg P_1 \vee P_2$			
5	$\neg P_1 \vee P_4 \vee P_3 \vee P_0$			
6	$\neg P_1 \vee \neg P_4 \vee P_3$			
7	$\neg P_1 \vee P_5$			

Example

Let $P_5 \succ P_4 \succ P_3 \succ P_2 \succ P_1 \succ P_0$ (max. literals in red)

	clauses C	$I_C = \mathcal{A}_C^{-1}(1)$	Δ_C	Remarks
1	$\neg P_0$	\emptyset	\emptyset	true in \mathcal{A}_C
2	$P_0 \vee P_1$	\emptyset	$\{P_1\}$	P_1 maximal
3	$P_1 \vee P_2$	$\{P_1\}$	\emptyset	true in \mathcal{A}_C
4	$\neg P_1 \vee P_2$	$\{P_1\}$	$\{P_2\}$	P_2 maximal
5	$\neg P_1 \vee P_4 \vee P_3 \vee P_0$			
6	$\neg P_1 \vee \neg P_4 \vee P_3$			
7	$\neg P_1 \vee P_5$			

Example

Let $P_5 \succ P_4 \succ P_3 \succ P_2 \succ P_1 \succ P_0$ (max. literals in red)

	clauses C	$I_C = \mathcal{A}_C^{-1}(1)$	Δ_C	Remarks
1	$\neg P_0$	\emptyset	\emptyset	true in \mathcal{A}_C
2	$P_0 \vee P_1$	\emptyset	$\{P_1\}$	P_1 maximal
3	$P_1 \vee P_2$	$\{P_1\}$	\emptyset	true in \mathcal{A}_C
4	$\neg P_1 \vee P_2$	$\{P_1\}$	$\{P_2\}$	P_2 maximal
5	$\neg P_1 \vee P_4 \vee P_3 \vee P_0$	$\{P_1, P_2\}$	$\{P_4\}$	P_4 maximal
6	$\neg P_1 \vee \neg P_4 \vee P_3$			
7	$\neg P_1 \vee P_5$			

Example

Let $P_5 \succ P_4 \succ P_3 \succ P_2 \succ P_1 \succ P_0$ (max. literals in red)

	clauses C	$I_C = \mathcal{A}_C^{-1}(1)$	Δ_C	Remarks
1	$\neg P_0$	\emptyset	\emptyset	true in \mathcal{A}_C
2	$P_0 \vee P_1$	\emptyset	$\{P_1\}$	P_1 maximal
3	$P_1 \vee P_2$	$\{P_1\}$	\emptyset	true in \mathcal{A}_C
4	$\neg P_1 \vee P_2$	$\{P_1\}$	$\{P_2\}$	P_2 maximal
5	$\neg P_1 \vee P_4 \vee P_3 \vee P_0$	$\{P_1, P_2\}$	$\{P_4\}$	P_4 maximal
6	$\neg P_1 \vee \neg P_4 \vee P_3$	$\{P_1, P_2, P_4\}$	\emptyset	P_3 not maximal; <i>min. counter-ex.</i>
7	$\neg P_1 \vee P_5$	$\{P_1, P_2, P_4\}$	$\{P_5\}$	

$I = \{P_1, P_2, P_4, P_5\} = \mathcal{A}^{-1}(1)$: \mathcal{A} is not a model of the clause set

\Rightarrow there exists a counterexample.

Main Ideas of the Construction

- Clauses are considered in the order given by \prec .
- When considering C , one already has a partial interpretation I_C (initially $I_C = \emptyset$) available.
- If C is true in the partial interpretation I_C , nothing is done. ($\Delta_C = \emptyset$).
- If C is false, one would like to change I_C such that C becomes true.

Main Ideas of the Construction

- Changes should, however, be *monotone*. One never deletes anything from I_C and the truth value of clauses smaller than C should be maintained the way it was in I_C .
- Hence, one chooses $\Delta_C = \{A\}$ if, and only if, C is false in I_C , if A occurs positively in C (*adding A will make C become true*) and if this occurrence in C is strictly maximal in the ordering on literals (*changing the truth value of A has no effect on smaller clauses*).

Resolution Reduces Counterexamples

$$\frac{\neg P_1 \vee P_4 \vee P_3 \vee P_0 \quad \neg P_1 \vee \neg P_4 \vee P_3}{\neg P_1 \vee \neg P_1 \vee P_3 \vee P_3 \vee P_0}$$

Construction of I for the extended clause set:

	clauses C	I_C	Δ_C	Remarks
1	$\neg P_0$	\emptyset	\emptyset	
2	$P_0 \vee P_1$	\emptyset	$\{P_1\}$	
3	$P_1 \vee P_2$	$\{P_1\}$	\emptyset	
4	$\neg P_1 \vee P_2$	$\{P_1\}$	$\{P_2\}$	
8	$\neg P_1 \vee \neg P_1 \vee P_3 \vee P_3 \vee P_0$	$\{P_1, P_2\}$	\emptyset	P_3 occurs twice <i>minimal counter-ex.</i>
5	$\neg P_1 \vee P_4 \vee P_3 \vee P_0$	$\{P_1, P_2\}$	$\{P_4\}$	
6	$\neg P_1 \vee \neg P_4 \vee P_3$	$\{P_1, P_2, P_4\}$	\emptyset	old counterexample
7	$\neg P_1 \vee P_5$	$\{P_1, P_2, P_4\}$	$\{P_5\}$	

The same I , but smaller counterexample, hence some progress was made.

Factorization Reduces Counterexamples

$$\frac{\neg P_1 \vee \neg P_1 \vee P_3 \vee P_3 \vee P_0}{\neg P_1 \vee \neg P_1 \vee P_3 \vee P_0}$$

Construction of I for the extended clause set:

	clauses C	I_C	Δ_C	Remarks
1	$\neg P_0$	\emptyset	\emptyset	
2	$P_0 \vee P_1$	\emptyset	$\{P_1\}$	
3	$P_1 \vee P_2$	$\{P_1\}$	\emptyset	
4	$\neg P_1 \vee P_2$	$\{P_1\}$	$\{P_2\}$	
9	$\neg P_1 \vee \neg P_1 \vee P_3 \vee P_0$	$\{P_1, P_2\}$	$\{P_3\}$	
8	$\neg P_1 \vee \neg P_1 \vee P_3 \vee P_3 \vee P_0$	$\{P_1, P_2, P_3\}$	\emptyset	true in \mathcal{A}_C
5	$\neg P_1 \vee P_4 \vee P_3 \vee P_0$	$\{P_1, P_2, P_3\}$	\emptyset	
6	$\neg P_1 \vee \neg P_4 \vee P_3$	$\{P_1, P_2, P_3\}$	\emptyset	true in \mathcal{A}_C
7	$\neg P_3 \vee P_5$	$\{P_1, P_2, P_3\}$	$\{P_5\}$	

The resulting $I = \{P_1, P_2, P_3, P_5\}$ is a model of the clause set.

Construction of Candidate Models Formally

Let N, \succ be given. We define sets I_C and Δ_C for all ground clauses C over the given signature inductively over \succ :

$$I_C := \bigcup_{C \succ D} \Delta_D$$

$$\Delta_C := \begin{cases} \{A\}, & \text{if } C \in N, C = C' \vee A, A \succ C', I_C \not\models C \\ \emptyset, & \text{otherwise} \end{cases}$$

We say that C **produces** A , if $\Delta_C = \{A\}$.

The **candidate model** for N (wrt. \succ) is given as $I_N^\succ := \bigcup_C \Delta_C$.

We also simply write I_N , or I , for I_N^\succ if \succ is either irrelevant or known from the context.

Structure of N, \succ

Let $A \succ B$; producing a new atom does not affect smaller clauses.

Some Properties of the Construction

Proposition 1.13:

- (i) $C = \neg A \vee C' \Rightarrow$ no $D \succeq C$ produces A .
- (ii) C productive $\Rightarrow I_C \cup \Delta_C \models C$.
- (iii) Let $D' \succ D \succeq C$. Then

$$I_D \cup \Delta_D \models C \Rightarrow I_{D'} \cup \Delta_{D'} \models C \text{ and } I_N \models C.$$

If, in addition, $C \in N$ or $\max(D) \succ \max(C)$:

$$I_D \cup \Delta_D \not\models C \Rightarrow I_{D'} \cup \Delta_{D'} \not\models C \text{ and } I_N \not\models C.$$

Some Properties of the Construction

(iv) Let $D' \succ D \succ C$. Then

$$I_D \models C \Rightarrow I_{D'} \models C \text{ and } I_N \models C.$$

If, in addition, $C \in N$ or $\max(D) \succ \max(C)$:

$$I_D \not\models C \Rightarrow I_{D'} \not\models C \text{ and } I_N \not\models C.$$

(v) $D = C \vee A$ produces $A \Rightarrow I_N \not\models C$.

Model Existence Theorem

Theorem 1.14 (Bachmair & Ganzinger):

Let \succ be a clause ordering, let N be saturated wrt. Res , and suppose that $\perp \notin N$. Then $I_N^\succ \models N$.

Corollary 1.15:

Let N be saturated wrt. Res . Then $N \models \perp \Leftrightarrow \perp \in N$.

Model Existence Theorem

Proof:

Suppose $\perp \notin N$, but $I_N \not\models N$. Let $C \in N$ minimal (in \succ) such that $I_N \not\models C$. Since C is false in I_N , C is not productive. As $C \neq \perp$ there exists a maximal atom A in C .

Case 1: $C = \neg A \vee C'$ (i.e., the maximal atom occurs negatively)

$\Rightarrow I_N \models A$ and $I_N \not\models C'$

\Rightarrow some $D = D' \vee A \in N$ produces A . As $\frac{D' \vee A}{D' \vee C'} \frac{\neg A \vee C'}{C}$, we infer that $D' \vee C' \in N$, and $C \succ D' \vee C'$ and $I_N \not\models D' \vee C'$

\Rightarrow contradicts minimality of C .

Case 2: $C = C' \vee A \vee A$. Then $\frac{C' \vee A \vee A}{C' \vee A}$ yields a smaller counterexample $C' \vee A \in N$. \Rightarrow contradicts minimality of C .

Ordered Resolution with Selection

Ideas for improvement:

1. In the completeness proof (Model Existence Theorem) one only needs to resolve and factor maximal atoms
 - ⇒ if the calculus is restricted to inferences involving maximal atoms, the proof remains correct
 - ⇒ *order restrictions*
2. In the proof, it does not really matter with which negative literal an inference is performed
 - ⇒ choose a negative literal don't-care-nondeterministically
 - ⇒ *selection*

Selection Functions

A **selection function** is a mapping

$$S : C \mapsto \text{set of occurrences of } \textit{negative} \text{ literals in } C$$

Example of selection with selected literals indicated as \boxed{X} :

$$\boxed{\neg A} \vee \neg A \vee B$$
$$\boxed{\neg B_0} \vee \boxed{\neg B_1} \vee A$$

Ordered resolution

In the completeness proof, we talk about (strictly) maximal literals of clauses.

Resolution Calculus $Res_S^>$

Ordered Resolution with Selection:

$$\frac{C \vee A \quad D \vee \neg A}{C \vee D}$$

- if
- (i) $A \succ C$;
 - (ii) nothing is selected in C by S ;
 - (iii) $\neg A$ is selected in $D \vee \neg A$,
or else nothing is selected in $D \vee \neg A$ and $\neg A \succeq \max(D)$.

Ordered Factoring:

$$\frac{C \vee A \vee A}{(C \vee A)}$$

if A is maximal in C and nothing is selected in C .

Note: For positive literals, $A \succ C$ is the same as $A \succ \max(C)$.

Search Spaces Become Smaller

1	$A \vee B$	
2	$A \vee \boxed{\neg B}$	
3	$\neg A \vee B$	
4	$\neg A \vee \boxed{\neg B}$	
5	$B \vee B$	Res 1, 3
6	B	Fact 5
7	$\neg A$	Res 6, 4
8	A	Res 6, 2
9	\perp	Res 8, 7

we assume $A \succ B$ and S as indicated by \boxed{X} . The maximal literal in a clause is depicted in red.

With this ordering and selection function the refutation proceeds strictly deterministically in this example. Generally, proof search will still be non-deterministic but the search space will be much smaller than with unrestricted resolution.

Res \succ : Construction of Candidate Models

Let N, \succ be given. We define sets I_C and Δ_C for all ground clauses C over the given signature inductively over \succ :

$$I_C := \bigcup_{C \succ D} \Delta_D$$
$$\Delta_C := \begin{cases} \{A\}, & \text{if } C \in N, C = C' \vee A, A \succ C', I_C \not\equiv C \\ & \text{and nothing is selected in } C \\ \emptyset, & \text{otherwise} \end{cases}$$

We say that C **produces** A , if $\Delta_C = \{A\}$.

The **candidate model** for N (wrt. \succ) is given as $I_N^\succ := \bigcup_C \Delta_C$.

We also simply write I_N , or I , for I_N^\succ if \succ is either irrelevant or known from the context.

Model Existence Theorem

Theorem 1.14^s (Bachmair & Ganzinger):

Let \succ be a clause ordering, let N be saturated wrt. Res_S^\succ , and suppose that $\perp \notin N$. Then $I_N^\succ \models N$.

Corollary 1.15^s:

Let N be saturated wrt. Res_S^\succ . Then $N \models \perp \Leftrightarrow \perp \in N$.

Model Existence Theorem

Proof: Suppose $\perp \notin N$, but $I_N^{\succ} \not\models N$. Let $C \in N$ minimal (in \succ) such that $I_N^{\succ} \not\models C$. Since C is false in I_N , C is not productive. As $C \neq \perp$ there exists a maximal atom A in C .

Case 1: $C = \neg A \vee C'$ (i.e., the maximal atom occurs negatively)

$\Rightarrow I_N \models A$ and $I_N \not\models C' \Rightarrow$ some $D = D' \vee A \in N$ produces A .

As $\frac{D' \vee A}{D' \vee C'} \frac{\neg A \vee C'}{\neg A \vee C'}$, we infer that $D' \vee C' \in N$, and $C \succ D' \vee C'$ and $I_N \not\models D' \vee C' \Rightarrow$ contradicts minimality of C .

Case 1': $C = \neg A' \vee C'$ and $\neg A'$ is selected in C

$\Rightarrow I_N \models A'$ and $I_N \not\models C' \Rightarrow$ some $D = D' \vee A' \in N$ produces A' .

As $\frac{D' \vee A'}{D' \vee C'} \frac{\neg A' \vee C'}{\neg A' \vee C'}$, we infer that $D' \vee C' \in N$, and $C \succ D' \vee C'$ and $I_N \not\models D' \vee C' \Rightarrow$ contradicts minimality of C .

Case 2: $C = C' \vee A \vee A$. Then $\frac{C' \vee A \vee A}{C' \vee A}$ yields a smaller counterexample $C' \vee A \in N$. \Rightarrow contradicts minimality of C .