

5 Propositional logic

Stephan Falke

November 8th 2006

Contents

5	Propositional logic	1
5.1	Syntax and semantics	1
5.2	Conjunctive normal form	5
5.3	Horn formulas	6
5.4	Resolution	8
	5.4.1 Unit-resolution for Horn formulas	11
5.5	Digression: Trees	13
5.6	Sequent calculus	14

Chapter 5

Propositional logic

Assume we know the following:

1. If Alice enrolls in CS 401 and studies hard, then she passes CS 401.
2. Alice studies hard.
3. Alice does not pass CS 401.

Is this situation possible? Propositional logic deals with problems like this. In propositional logic, we abstract the above statements using propositional variables:

E : Alice enrolls in CS 401
 S : Alice studies hard
 P : Alice passes CS 401

The above statements will then be written as follows:

1. $E \wedge S \rightarrow P$
2. S
3. $\neg P$.

Determining whether the above situation is possible then reduces to checking whether the formula

$$(E \wedge S \rightarrow P) \wedge S \wedge \neg P$$

is satisfiable. We will discuss methods to determine this.

5.1 Syntax and semantics

In propositional logic we examine expressions that are built from propositional variables using propositional connectives. We fix an enumerable set τ of propositional variables. The propositional variables are interpreted using the truth values 0 (FALSE) and 1 (TRUE).

Definition 5.1.1

The set PL of PROPOSITIONAL FORMULAS is inductively defined by

1. $0, 1 \in PL$ (the truth values are formulas)
2. $\tau \subseteq PL$ (every propositional variable is a formula)
3. If $\psi, \varphi \in PL$, then $\neg\psi$, $(\psi \wedge \varphi)$, $(\psi \vee \varphi)$, $(\psi \rightarrow \varphi)$ and $(\psi \leftrightarrow \varphi)$ are formulas in PL as well.

Formulas created using condition 1 or 2 are called *atomic formulas*. In the following, \circ denotes one of \wedge , \vee , \rightarrow and \leftrightarrow .

Example 5.1.2

$\rho = ((X_1 \vee X_3) \wedge (X_2 \vee (X_3 \rightarrow \neg X_1)))$ is a formula.

Remark 5.1.3

We often omit parenthesis that are not needed. For this, we assume that \neg binds stronger than any other connective, and that \wedge and \vee bind stronger than \rightarrow and \leftrightarrow . Thus, $\psi \wedge \neg\varphi \rightarrow \vartheta$ is shorthand for $((\psi \wedge \neg\varphi) \rightarrow \vartheta)$.

Since PL is defined inductively, we can use well-founded induction to prove properties that hold for all formulas. For this, we first introduce the set of subformulas of a formula.

Definition 5.1.4

Let ψ be a formula. The set $SUB(\psi)$ of SUBFORMULAS of ψ is defined inductively by

1. If $\psi = 0, \psi = 1$ or $\psi \in \tau$, then $SUB(\psi) = \{\psi\}$.
2. If $\psi = \neg\varphi$, then $SUB(\psi) = \{\psi\} \cup SUB(\varphi)$.
3. If $\psi = (\varphi_1 \circ \varphi_2)$, then $SUB(\psi) = \{\psi\} \cup SUB(\varphi_1) \cup SUB(\varphi_2)$.

Example 5.1.5

$SUB(\rho) = \{\rho, (X_1 \vee X_3), (X_2 \vee (X_3 \rightarrow \neg X_1)), (X_3 \rightarrow \neg X_1), \neg X_1, X_1, X_2, X_3\}$

Now we can define a partial order \succsim on PL by $\psi \succsim \varphi$ iff $\varphi \in SUB(\psi)$. It is easy to see that the strict order \succ that is induced by \succsim is well-founded.

Definition 5.1.6

Let ψ be a formula. The SIZE $size(\psi)$ and DEPTH $depth(\psi)$ of ψ is defined inductively as

1. If $\psi = 0, \psi = 1$ or $\psi \in \tau$, then $size(\psi) = 1$ and $depth(\psi) = 0$.
2. If $\psi = \neg\varphi$, then $size(\psi) = 1 + size(\varphi)$ and $depth(\psi) = 1 + depth(\varphi)$.
3. If $\psi = (\varphi_1 \circ \varphi_2)$, then $size(\psi) = 1 + size(\varphi_1) + size(\varphi_2)$ and $depth(\psi) = 1 + \max(depth(\varphi_1), depth(\varphi_2))$.

Example 5.1.7

$size(\rho) = 10$, $depth(\rho) = 4$.

Proposition 5.1.8

Let ψ be a formula. Then $|SUB(\psi)| \leq size(\psi)$.

Proof. We prove the claim by induction on the formula.

base case: In the base case, $\psi = 0$, $\psi = 1$ or $\psi \in \tau$. Then, $|SUB(\psi)| = |\{\psi\}| = 1 = size(\psi)$.

step case 1: In this step case, $\psi = \neg\varphi$. Then, $|SUB(\psi)| = |\{\psi\} \cup SUB(\varphi)| \leq 1 + |SUB(\varphi)| \leq 1 + size(\varphi) = size(\psi)$.

step case 2: In this step case, $\psi = (\varphi_1 \circ \varphi_2)$. Then, $|SUB(\psi)| = |\{\psi\} \cup SUB(\varphi_1) \cup SUB(\varphi_2)| \leq 1 + |SUB(\varphi_1)| + |SUB(\varphi_2)| \leq 1 + size(\varphi_1) + size(\varphi_2) = size(\psi)$. \square

For a formula ψ , we denote by $\tau(\psi)$ the propositional variables occurring in it.

Definition 5.1.9

A (PROPOSITIONAL) ASSIGNMENT is a function $\mathcal{A} : \sigma \rightarrow \{0, 1\}$ for some $\sigma \subseteq \tau$. It is SUITABLE for a formula ψ if $\tau(\psi) \subseteq \sigma$. Every assignment \mathcal{A} that is suitable for ψ defines a truth value $\psi^{\mathcal{A}}$ as follows:

1. $0^{\mathcal{A}} = 0$, $1^{\mathcal{A}} = 1$ and $X^{\mathcal{A}} = \mathcal{A}(X)$ for all $X \in \tau$.
2. $(\neg\varphi)^{\mathcal{A}} = 1 - \varphi^{\mathcal{A}}$.
3. $(\varphi_1 \wedge \varphi_2)^{\mathcal{A}} = \min(\varphi_1^{\mathcal{A}}, \varphi_2^{\mathcal{A}})$.
4. $(\varphi_1 \vee \varphi_2)^{\mathcal{A}} = \max(\varphi_1^{\mathcal{A}}, \varphi_2^{\mathcal{A}})$.
5. $(\varphi_1 \rightarrow \varphi_2)^{\mathcal{A}} = \begin{cases} 0 & \text{if } \varphi_1^{\mathcal{A}} = 1 \text{ and } \varphi_2^{\mathcal{A}} = 0 \\ 1 & \text{otherwise} \end{cases}$
6. $(\varphi_1 \leftrightarrow \varphi_2)^{\mathcal{A}} = \begin{cases} 1 & \text{if } \varphi_1^{\mathcal{A}} = \varphi_2^{\mathcal{A}} \\ 0 & \text{otherwise} \end{cases}$

A MODEL of a formula ψ is an assignment \mathcal{A} such that $\psi^{\mathcal{A}} = 1$. This is also written $\mathcal{A} \models \psi$ (“ \mathcal{A} satisfies ψ ”).

Example 5.1.10

Consider the assignment \mathcal{A} such that $\mathcal{A}(X_1) = 0$, $\mathcal{A}(X_2) = 0$, $\mathcal{A}(X_3) = 1$. Then $\rho^{\mathcal{A}} = 1$, i.e., $\mathcal{A} \models \rho$.

Definition 5.1.11

Let ψ be a formula. If ψ has a model, then ψ is SATISFIABLE, otherwise ψ is UNSATISFIABLE. The formula ψ is VALID if every assignment that is suitable for ψ is a model of ψ .

Proposition 5.1.12

Let ψ be a formula. Then the following are equivalent:

1. ψ is valid
2. $\neg\psi$ is unsatisfiable

Proof. ψ is valid iff $\mathcal{A} \models \psi$ for every \mathcal{A} that is suitable for ψ . This is equivalent to $\mathcal{A} \not\models \neg\psi$ for every \mathcal{A} that is suitable for ψ , which in turn is equivalent to $\neg\psi$ being unsatisfiable. \square

Definition 5.1.13

Two formulas ψ and φ are EQUIVALENT, written $\psi \equiv \varphi$, iff $\psi^{\mathcal{A}} = \varphi^{\mathcal{A}}$ for any assignment \mathcal{A} that is suitable for both ψ and φ .

The following proposition lists some commonly used equivalences. They are an immediate consequence of the definitions.

Proposition 5.1.14

Let ψ, φ, ϑ be formulas.

1. $\neg\neg\psi \equiv \psi$ (double negation)
2. $\neg(\psi \wedge \varphi) \equiv (\neg\psi \vee \neg\varphi)$
 $\neg(\psi \vee \varphi) \equiv (\neg\psi \wedge \neg\varphi)$ (de Morgan laws)
3. $\psi \wedge (\varphi \vee \vartheta) \equiv (\psi \wedge \varphi) \vee (\psi \wedge \vartheta)$
 $\psi \vee (\varphi \wedge \vartheta) \equiv (\psi \vee \varphi) \wedge (\psi \vee \vartheta)$ (distributivity)
4. $\psi \rightarrow \varphi \equiv \neg\varphi \rightarrow \neg\psi$ (contrapositive)
5. $\psi \wedge (\psi \vee \varphi) \equiv \psi \vee (\psi \wedge \varphi) \equiv \psi$ (absorption)
6. $\psi \wedge \psi \equiv \psi \vee \psi \equiv \psi$ (idempotency)
7. $\psi \wedge \varphi \equiv \varphi \wedge \psi$
 $\psi \vee \varphi \equiv \varphi \vee \psi$ (commutativity)
8. $\psi \wedge (\varphi \wedge \vartheta) \equiv (\psi \wedge \varphi) \wedge \vartheta$
 $\psi \vee (\varphi \vee \vartheta) \equiv (\psi \vee \varphi) \vee \vartheta$ (associativity)

Definition 5.1.15

Let Φ be a set of formulas. A MODEL of Φ is an assignment \mathcal{A} such that $\mathcal{A} \models \varphi$ for all $\varphi \in \Phi$. The set Φ is SATISFIABLE iff Φ has a model. Otherwise, Φ is UNSATISFIABLE.

Definition 5.1.16

Let Φ be a set of formulas, let ψ be a formula. Then ψ is a SEMANTIC CONSEQUENCE of Φ , written $\Phi \models \psi$, iff $\mathcal{A} \models \psi$ for every assignment \mathcal{A} that is suitable for $\Phi \cup \{\psi\}$ and for which $\mathcal{A} \models \Phi$. If $\Phi = \{\varphi\}$, we also write $\varphi \models \psi$ instead of $\{\varphi\} \models \psi$.

Example 5.1.17

1. $\{\psi, \varphi\} \models \psi \wedge \varphi$
2. If $\Phi \cup \{\psi\} \models \varphi$ and $\Phi \cup \{\neg\psi\} \models \varphi$, then already $\Phi \models \varphi$.
3. $\Phi \cup \{\psi\} \models \varphi$ iff $\Phi \models \psi \rightarrow \varphi$
4. $\varphi \equiv \psi$ iff $\varphi \models \psi$ and $\psi \models \varphi$
5. $\Phi \cup \{\psi\}$ is unsatisfiable iff $\Phi \models \neg\psi$
6. If $\Phi \models \psi$ and $\Phi \models \neg\psi$ for some $\psi \in \text{PL}$, then Φ is unsatisfiable.
7. If Φ is unsatisfiable, then $\Phi \models \psi$ for all $\psi \in \text{PL}$.

5.2 Conjunctive normal form

Definition 5.2.1

Let ψ be a formula. Then ψ is in CONJUNCTIVE NORMAL FORM (CNF) iff ψ has the form $C_1 \wedge C_2 \wedge \dots \wedge C_n$, where each C_i has the form $l_1 \vee l_2 \vee \dots \vee l_m$ such that each l_j is either a propositional variable X or its negation $\neg X$. The C_i are called CLAUSES, the l_j are called LITERALS.

Proposition 5.2.2

Let ψ be a formula. Then there exists a formula ψ' such that

1. $\psi' \equiv \psi$, and
2. ψ' is in CNF.

Proof. We give a method that transforms ψ into an equivalent formula in CNF. In a first step, we eliminate \leftrightarrow and \rightarrow using the equivalences $\psi \leftrightarrow \varphi \equiv (\psi \rightarrow \varphi) \wedge (\varphi \rightarrow \psi)$ and $\psi \rightarrow \varphi \equiv \neg\psi \vee \varphi$.

Next, we push \neg in front of the propositional variables. This is done using the de Morgan laws and double negation.

Finally, we distribute \vee over \wedge by applying the equivalence $\psi \vee (\varphi \wedge \vartheta) \equiv (\psi \vee \varphi) \wedge (\psi \vee \vartheta)$ together with the commutativity of \vee .

Since we only apply equivalences, the resulting formula is equivalent to ψ . Also, it is clearly in CNF. \square

Example 5.2.3

Consider the formula $(E \wedge S \rightarrow P) \wedge S \wedge \neg P$ from the introduction:

$$\begin{aligned}
 & (E \wedge S \rightarrow P) \wedge S \wedge \neg P \\
 & \equiv \text{elimination of } \rightarrow \\
 & (\neg(E \wedge S) \vee P) \wedge S \wedge \neg P \\
 & \equiv \text{de Morgan laws} \\
 & (\neg E \vee \neg S \vee P) \wedge S \wedge \neg P
 \end{aligned}$$

Example 5.2.4

Consider the formula $\psi = (X \rightarrow Y) \leftrightarrow Z$:

$$\begin{aligned}
& (X \rightarrow Y) \leftrightarrow Z \\
& \equiv && \text{elimination of } \leftrightarrow \\
& ((X \rightarrow Y) \rightarrow Z) \wedge (Z \rightarrow (X \rightarrow Y)) \\
& \equiv && \text{elimination of } \rightarrow \\
& (\neg(\neg X \vee Y) \vee Z) \wedge (\neg Z \vee \neg X \vee Y) \\
& \equiv && \text{de Morgan laws} \\
& ((\neg\neg X \wedge \neg Y) \vee Z) \wedge (\neg Z \vee \neg X \vee Y) \\
& \equiv && \text{double negation} \\
& ((X \wedge \neg Y) \vee Z) \wedge (\neg Z \vee \neg X \vee Y) \\
& \equiv && \text{distribute } \vee \text{ over } \wedge \\
& (X \vee Z) \wedge (\neg Y \vee Z) \wedge (\neg Z \vee \neg X \vee Y)
\end{aligned}$$

5.3 Horn formulas

A class of formulas that is highly relevant in practical applications (e.g. in the programming language Prolog) is the class of Horn formulas. In particular, we will give an efficient algorithm that decides whether a Horn formula is satisfiable.

Definition 5.3.1

A HORN FORMULA is a CNF formula of the form

$$\psi = \bigwedge_{i=1}^n \bigvee_{j=1}^{m_i} l_{i,j}$$

such that each disjunction $\bigvee l_{i,j}$ contains at most one positive literal X .

Clearly, a Horn formula is equivalent to a conjunction of implications of the form:

1. $\neg X_1 \vee \dots \vee \neg X_n \vee X \equiv X_1 \wedge \dots \wedge X_n \rightarrow X$
2. $\neg X_1 \vee \dots \vee \neg X_n \equiv X_1 \wedge \dots \wedge X_n \rightarrow 0$

Implications of type 1 with $k = 0$ are written $1 \rightarrow X$.

Clearly, a Horn formula that does not contain an implication of type 1 with $n = 0$ is satisfiable, e.g. by the assignment \mathcal{A} that maps every propositional variable to 0. Also, a Horn formula that does not contain any implication of type 2 is satisfiable, e.g. by the assignment \mathcal{A} that maps every propositional variable to 1.

The following algorithm decides whether a general Horn formula is satisfiable.

Algorithm 5.3.2 (Satisfiability test for Horn formulas)

<p>Input: A Horn formula $\psi = \bigwedge_i C_i$ Output: “unsatisfiable” or an $M \subseteq \tau$ $N := \emptyset$ $M := \{X \in \tau(\psi) \mid \text{some } C_i \text{ is } 1 \rightarrow X\}$ while $N \neq M$ do $N := M$ $M := M \cup \{X \mid \text{some } C_i \text{ is } X_1 \wedge \dots \wedge X_n \rightarrow X$ with $\{X_1, \dots, X_n\} \subseteq M\}$ if some C_i is $X_1 \wedge \dots \wedge X_n \rightarrow 0$ with $\{X_1, \dots, X_n\} \subseteq M$ then return “unsatisfiable” end end return M</p>
--

If the set M is returned it defines an assignment \mathcal{A}_M with $\mathcal{A}_M(X) = 1$ iff $X \in M$.

Proposition 5.3.3

Algorithm 5.3.2 is correct. If ψ is satisfiable, then $\mathcal{A}_M \models \psi$. If ψ contains n propositional variables, then the algorithm terminates after at most $n + 1$ iterations of the while-loop.

Proof. Assume $\mathcal{A} \models \psi$ for some assignment \mathcal{A} . Clearly, $\mathcal{A}(X) = 1$ for all X that are added to M during execution of the algorithm. Furthermore, ψ cannot contain a clause C_i of the form $X_1 \wedge \dots \wedge X_n \rightarrow 0$ with $\{X_1, \dots, X_n\} \subseteq M$. Thus, the algorithm correctly determines ψ 's satisfiability.

If the algorithm outputs the set M , then \mathcal{A}_M is indeed a model of ψ since the final value of M satisfies the following conditions:

1. For all clauses C_i of the form $X_1 \wedge \dots \wedge X_n \rightarrow X$ we have the following: if $\{X_1, \dots, X_n\} \subseteq M$, then $X \in M$ (otherwise the while-loop would not have been left).
2. For all clauses C_i of the form $X_1 \wedge \dots \wedge X_n \rightarrow 0$ we have $\{X_1, \dots, X_n\} \not\subseteq M$ (otherwise the algorithm would have returned “unsatisfiable”).

Since each iteration of the while-loop adds at least one propositional variable to M or concludes that ψ is unsatisfiable, the last claim follows as well. \square

Example 5.3.4

Consider $\psi = (1 \rightarrow X) \wedge (X \rightarrow Y) \wedge (X \wedge Y \rightarrow Z) \wedge (Z \rightarrow 0)$. The algorithm operates as follows:

N	M	return value
\emptyset	$\{X\}$	
$\{X\}$	$\{X, Y\}$	
$\{X, Y\}$	$\{X, Y, Z\}$	“unsatisfiable”

Thus, ψ is unsatisfiable.

Example 5.3.5

We consider the formula from the introduction. As a Horn formula this is written $(E \wedge S \rightarrow P) \wedge (1 \rightarrow S) \wedge (P \rightarrow 0)$. The algorithm operates as follows:

N	M	return value
\emptyset	$\{S\}$	
$\{S\}$	$\{S\}$	$\{S\}$

Thus, the formula is satisfiable. Furthermore, the assignment \mathcal{A} with $\mathcal{A}(S) = 1$ and $\mathcal{A}(E) = \mathcal{A}(P) = 0$ is a model of the formula. Hence, the situation described in the introduction is possible if Alice does not enroll in CS 401.

5.4 Resolution

Resolution is a method that determines whether a formula in CNF is satisfiable. For this, it is convenient to identify a formula in CNF with a set of clauses, where each set is identified with the set of its literals.

Definition 5.4.1

A *CLAUSE* is a finite set of literals. The empty clause is denoted “ \square ”. A formula $\psi = \bigwedge_{i=1}^n \bigvee_{j=1}^{m_i} l_{i,j}$ in CNF is identified with a finite set $K(\psi)$ of clauses as follows: every disjunction $\bigvee_{j=1}^{m_i} l_{i,j}$ is identified with the clause $C_i = \{l_{i,j} \mid 1 \leq j \leq m_i\}$, and $K(\psi) := \{C_1, \dots, C_n\}$.

Example 5.4.2

The formula $(X_1 \vee \neg X_2) \wedge X_3$ is identified with the set $\{\{X_1, \neg X_2\}, \{X_3\}\}$.

On the other hand, a clause C can be identified with the formula $\bigvee_{l \in C} l$. Then, a finite set of clauses K can be identified with the formula $\bigwedge_{C \in K} \bigvee_{l \in C} l$. Hence, we can talk about satisfiability, equivalence, etc., of a finite set of clauses. In particular, a finite set of clauses K is satisfiable iff there exists an assignment \mathcal{A} such that every clause $C \in K$ contains a literal l such that $l^{\mathcal{A}} = 1$. According to this, the empty set of clauses is satisfiable, and K is unsatisfiable if $\square \in K$.

Definition 5.4.3

Let l be a literal. The *COMPLEMENTARY LITERAL* \bar{l} of l is defined as

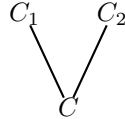
$$\bar{l} := \begin{cases} \neg X & \text{if } l = X \\ X & \text{if } l = \neg X \end{cases}$$

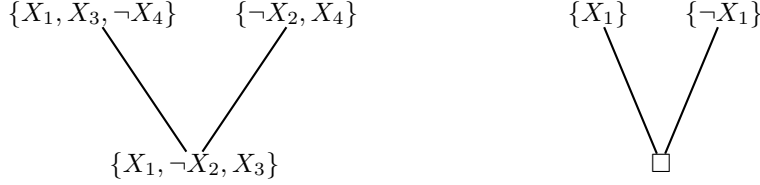
Definition 5.4.4

Let C, C_1, C_2 be clauses. Then C is a *RESOLVENT* of C_1 and C_2 iff there is a literal l such that $l \in C_1$, $\bar{l} \in C_2$, and $C = (C_1 - \{l\}) \cup (C_2 - \{\bar{l}\})$.

Remark 5.4.5

If C is a resolvent of C_1 and C_2 , this is denoted



Example 5.4.6**Proposition 5.4.7**

Let K be a set of clauses with $C_1, C_2 \in K$. Let C be a resolvent of C_1 and C_2 . Then K and $K \cup \{C\}$ are equivalent.

Proof. If $(K \cup \{C\})^{\mathcal{A}} = 1$, then obviously $K^{\mathcal{A}} = 1$ as well.

For the converse, assume $K^{\mathcal{A}} = 1$ and $C = (C_1 - \{l\}) \cup (C_2 - \{\bar{l}\})$.

1. If $l^{\mathcal{A}} = 1$, then $(C_2 - \{\bar{l}\})^{\mathcal{A}} = 1$ since otherwise $C_2^{\mathcal{A}} = 0$. Thus, $C^{\mathcal{A}} = 1$.
2. If $l^{\mathcal{A}} = 0$, then $(C_1 - \{l\})^{\mathcal{A}} = 1$ since otherwise $C_1^{\mathcal{A}} = 0$. Thus, $C^{\mathcal{A}} = 1$ as well.

Hence, $(K \cup \{C\})^{\mathcal{A}} = 1$. □

Definition 5.4.8

Let K be a set of clauses.

1. $\mathcal{RES}(K) := K \cup \{C \mid C \text{ is a resolvent of two clauses in } K\}$
2. $\mathcal{RES}^0(K) := K$
 $\mathcal{RES}^{n+1}(K) := \mathcal{RES}(\mathcal{RES}^n(K))$ for $n \geq 0$
3. $\mathcal{RES}^*(K) := \bigcup_{n \in \mathbb{N}} \mathcal{RES}^n(K)$

A proof system is **CORRECT** if no false proposition can be proved in it. It is **COMPLETE** if all true propositions can be proved in it. Resolution will be method that determines unsatisfiability of sets of clauses. Its correctness and completeness are stated in the following proposition.

Proposition 5.4.9

A finite set of clauses K is unsatisfiable iff $\square \in \mathcal{RES}^*(K)$.

Proof.

Correctness: By the previous Proposition, $K \equiv \mathcal{RES}(K)$. By induction on n we get $K \equiv \mathcal{RES}^n(K)$. Since K is finite, it contains only finitely many propositional variables. Hence, $\mathcal{RES}^*(K)$ is also finite since resolution does not introduce new propositional variables and there are only finitely many clauses that can be built out of finitely many propositional variables. But this means that $\mathcal{RES}^*(K) = \mathcal{RES}^m(K)$ for some $m \in \mathbb{N}$. Thus, $K \equiv \mathcal{RES}^*(K)$ as well. Now, if $\square \in \mathcal{RES}^*(K)$, then $\mathcal{RES}^*(K)$ is unsatisfiable. But then K is unsatisfiable, too.

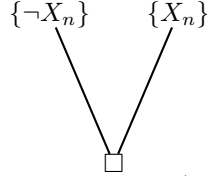
Completeness: Since K is finite, there is an $n \in \mathbb{N}$ such that K contains exactly n propositional variables. We show $\square \in \mathcal{RES}^*(K)$ by induction on n .

Let $n = 0$. There are only two sets of clauses that don't contain any propositional variables, namely \emptyset and $\{\square\}$. Since the empty set of clauses is satisfiable, we get $K = \{\square\}$. Thus, $\square \in \mathcal{RES}^*(K)$.

For the step case, we assume that K is built using the propositional variables X_0, \dots, X_n . We construct two sets of clauses K^+ and K^- that don't contain the propositional variable X_n . For this, $K^+ := \{C - \{\neg X_n\} \mid X_n \notin C, C \in K\}$, i.e., we remove all clauses that contain X_n from K , and then remove $\neg X_n$ from the remaining clauses. Similarly, $K^- := \{C - \{X_n\} \mid \neg X_n \notin C, C \in K\}$.

Then K^+ and K^- are unsatisfiable. Otherwise, there would exist an assignment $\mathcal{A} : \{X_0, \dots, X_{n-1}\} \rightarrow \{0, 1\}$ such that $(K^+)^{\mathcal{A}} = 1$ or $(K^-)^{\mathcal{A}} = 1$. If $(K^+)^{\mathcal{A}} = 1$, extend \mathcal{A} with $\mathcal{A}(X_n) := 1$. Then, $K^{\mathcal{A}} = 1$, contradicting the unsatisfiability of K . Similarly, if $(K^-)^{\mathcal{A}} = 1$, then extending \mathcal{A} by $\mathcal{A}(X_n) := 0$ yields a contradiction.

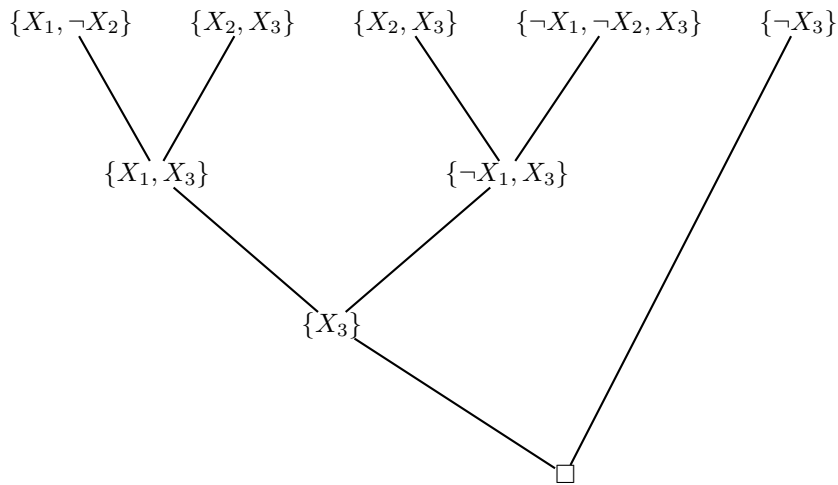
The inductive hypothesis now implies $\square \in \mathcal{RES}^*(K^+)$ and $\square \in \mathcal{RES}^*(K^-)$. Thus, there are clauses C_1, \dots, C_m such that $C_m = \square$ and for all $1 \leq i \leq m$ we have that $C_i \in K^+$ or that C_i is the resolvent of some C_j, C_k with $j, k < i$. Some of the clauses C_i might be obtained from clauses in K by removing $\neg X_n$. If not, then C_1, \dots, C_m are in $\mathcal{RES}^*(K)$ as well, thus $\square \in \mathcal{RES}^*(K)$. Otherwise, adding $\neg X_n$ to the clauses of C_1, \dots, C_m where it was removed yields clauses C'_1, \dots, C'_m which show that $\{\neg X_n\} \in \mathcal{RES}^*(K)$ using the "same" resolution steps. Analogously, $\square \in \mathcal{RES}^*(K^-)$ implies $\square \in \mathcal{RES}^*(K)$ or $\{X_n\} \in \mathcal{RES}^*(K)$. Using



we have $\square \in \mathcal{RES}^*(K)$ in either case. □

Example 5.4.10

Consider $\psi = (X_1 \vee \neg X_2) \wedge \neg X_3 \wedge (\neg X_1 \vee \neg X_2 \neg X_3) \wedge (X_2 \vee X_3)$. Then the empty clause can be derived from $K(\psi)$ as follows:



Thus, ψ is unsatisfiable.

Algorithm 5.4.11 (Satisfiability test using resolution)

<p>Input: A finite set of clauses K Output: “satisfiable” or “unsatisfiable” $R := K$ $S := \mathcal{RES}(K)$ while $R \neq S$ do $R := S$ $S := \mathcal{RES}(R)$ if $\square \in S$ then return “unsatisfiable” end end return “satisfiable”</p>
--

The algorithm has exponential worst-case complexity. We cannot expect to have a more efficient algorithm for the problem since it is known to be “hard”¹.

5.4.1 Unit-resolution for Horn formulas

If ψ is a Horn formula, then $K(\psi)$ consists of clauses of the form $\{\neg X_1, \dots, \neg X_n\}$ (just negative literals) or $\{\neg X_1, \dots, \neg X_n, X\}$ (one positive literal). Clauses of this form are called HORN CLAUSES. With $n = 0$ we see that \square and $\{X\}$ are Horn clauses as well. Next, we will present a restriction of resolution that is complete for Horn clauses.

Definition 5.4.12

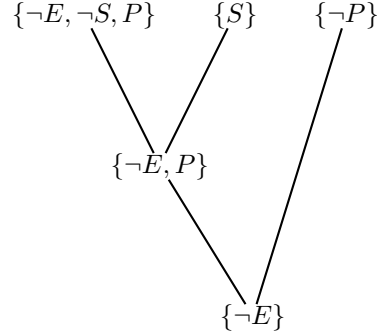
Let C, C_1, C_2 be clauses. Then C is a UNIT RESOLVENT of C_1 and C_2 iff C is a resolvent of C_1 and C_2 and either $|C_1| = 1$ or $|C_2| = 1$.

¹This will be made precise in CS 500.

Thus, at least one of the clauses C_1 or C_2 consists of exactly one literal.

Example 5.4.13

Consider the formula $(E \wedge S \rightarrow P) \wedge S \wedge \neg P$ from the introduction. The clauses obtained from this formula are $\{\neg E, \neg S, P\}, \{S\}, \{\neg P\}$. Then the following are unit resolution steps:



We see that $\neg E$ is implied by the formula, i.e., Alice did not enroll in CS 401.

Proposition 5.4.14

Let ψ be a Horn formula. Then ψ is unsatisfiable iff \square can be derived from $K(\psi)$ by unit resolution.

Proof. Clearly, ψ is unsatisfiable if \square can be derived from $K(\psi)$ by unit resolution since unit resolution is a restriction of resolution and resolution is correct.

For the converse, consider the satisfiability test for Horn formulas (Algorithm 5.3.2). Define

$$\begin{aligned}
 M^0 &:= \{X \mid K(\psi) \text{ contains the clause } \{X\}\} \\
 M^{n+1} &:= M^n \cup \{X \mid \text{there are } X_1, \dots, X_k \in M^n \text{ such that } K(\psi) \\
 &\quad \text{contains the clause } \{\neg X_1, \dots, \neg X_k, X\}\} \\
 M^* &:= \bigcup_{n \in \mathbb{N}} M^n
 \end{aligned}$$

The correctness of Algorithm 5.3.2 yields that ψ is unsatisfiable iff there exist $X_1, \dots, X_k \in M^*$ such that $\{\neg X_1, \dots, \neg X_k\} \in K(\psi)$. We show that $\{X\}$ can be derived from $K(\psi)$ by unit resolution if $X \in M^*$.

If $X \in M^0$, then this is clear. If $X \in M^{n+1}$, then either $X \in M^n$ or there exist $X_1, \dots, X_k \in M^n$ such that $\{\neg X_1, \dots, \neg X_k, X\} \in K(\psi)$. In the first case, the inductive hypothesis implies that $\{X\}$ can be derived from $K(\psi)$ by unit resolution. In the second case, the inductive hypothesis implies that all of $\{X_1\}, \dots, \{X_k\}$ can be derived from $K(\psi)$ by unit resolution. Then $\{X\}$ can be derived by unit resolution as follows:

1. There is no $w \in V$ such that $v E w$.
2. There is a unique path from v to w for each $w \in V$.

The CHILDREN of a node $w \in V$ are the nodes $w' \in V$ such that $w E w'$. A node $w \in V$ is a LEAF if it has no children. Otherwise, it is an INTERNAL NODE.

5.6 Sequent calculus

While resolution aims at deciding whether a formula is satisfiable, this section presents a proof calculus that aims at deciding whether a formula is valid or not without using the duality that a formula ψ is valid iff $\neg\psi$ is unsatisfiable.

The calculus operates on pairs of finite sets of formulas, called SEQUENTS. In the following, let Γ and Δ be finite sets of formulas. We write Γ, Δ for $\Gamma \cup \Delta$, and Γ, ψ for $\Gamma \cup \{\psi\}$. The expressions $\bigwedge \Gamma$ and $\bigvee \Gamma$ denote the conjunction and disjunction of the finitely many formulas in Γ .

Definition 5.6.1

A SEQUENT is an expression of the form $\Gamma \Rightarrow \Delta$ for finite sets of formulas Γ and Δ . The sequent $\Gamma \Rightarrow \Delta$ is VALID iff every model of Γ is also a model of at least one formula in Δ , i.e., if $\bigwedge \Gamma \rightarrow \bigvee \Delta$ is valid. Thus, $\Gamma \Rightarrow \Delta$ is not valid iff there exists an assignment \mathcal{A} that makes all formulas in Γ true and all formulas in Δ false. In that case, we say that \mathcal{A} FALSIFIES the sequent $\Gamma \Rightarrow \Delta$.

Example 5.6.2

1. Every sequent $\Gamma \Rightarrow \Delta$ with $\Gamma \cap \Delta \neq \emptyset$ is valid. Those sequents will be the AXIOMS of the sequent calculus.
2. Let Γ and Δ be sets of propositional variables. Then the sequent $\Gamma \Rightarrow \Delta$ is falsifiable iff $\Gamma \cap \Delta = \emptyset$.
3. A sequent of the form $\Gamma \Rightarrow \emptyset$ is valid iff Γ is unsatisfiable.
4. A sequent of the form $\emptyset \Rightarrow \Delta$ is valid iff $\bigvee \Delta$ is valid.

The sequent calculus \mathcal{SC} derives valid sequents from valid sequents using deduction rules.

Definition 5.6.3

The AXIOMS of \mathcal{SC} are all sequents of the form $\Gamma, \psi \Rightarrow \Delta, \psi$. The DEDUCTION RULES of \mathcal{SC} are as follows:

$$\begin{array}{ll}
(\neg \Rightarrow) \frac{\Gamma \Rightarrow \Delta, \psi}{\Gamma, \neg \psi \Rightarrow \Delta} & (\Rightarrow \neg) \frac{\Gamma, \psi \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg \psi} \\
(\vee \Rightarrow) \frac{\Gamma, \psi \Rightarrow \Delta \quad \Gamma, \varphi \Rightarrow \Delta}{\Gamma, \psi \vee \varphi \Rightarrow \Delta} & (\Rightarrow \vee) \frac{\Gamma \Rightarrow \Delta, \psi, \varphi}{\Gamma \Rightarrow \Delta, \psi \vee \varphi} \\
(\wedge \Rightarrow) \frac{\Gamma, \psi, \varphi \Rightarrow \Delta}{\Gamma, \psi \wedge \varphi \Rightarrow \Delta} & (\Rightarrow \wedge) \frac{\Gamma \Rightarrow \Delta, \psi \quad \Gamma \Rightarrow \Delta, \varphi}{\Gamma \Rightarrow \Delta, \psi \wedge \varphi} \\
(\rightarrow \Rightarrow) \frac{\Gamma \Rightarrow \Delta, \psi \quad \Gamma, \varphi \Rightarrow \Delta}{\Gamma, \psi \rightarrow \varphi \Rightarrow \Delta} & (\Rightarrow \rightarrow) \frac{\Gamma, \psi \Rightarrow \Delta, \varphi}{\Gamma \Rightarrow \Delta, \psi \rightarrow \varphi} \\
(\leftrightarrow \Rightarrow) \frac{\Gamma \Rightarrow \Delta, \psi, \varphi \quad \Gamma, \psi, \varphi \Rightarrow \Delta}{\Gamma, \psi \leftrightarrow \varphi \Rightarrow \Delta} & (\Rightarrow \leftrightarrow) \frac{\Gamma, \psi \Rightarrow \Delta, \varphi \quad \Gamma, \varphi \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \psi \leftrightarrow \varphi}
\end{array}$$

For this, Γ and Δ can be arbitrary finite sets of formulas, and ψ and φ can be arbitrary formulas. Each rule consists of one or two sequents on the first line, called PREMISES, and one sequent on the second line, called CONCLUSION.

Definition 5.6.4

The set of DERIVABLE SEQUENTS for \mathcal{SC} is the smallest set which contains all axioms of \mathcal{SC} and is closed under application of the inference rules, i.e., if the set contains instances of the premises of a rule, then it also contains the corresponding conclusion.

A PROOF of the sequent $\Gamma \Rightarrow \Delta$ in \mathcal{SC} is a tree whose nodes are labelled by sequents as follows:

1. The root is labelled by $\Gamma \Rightarrow \Delta$.
2. Each leaf is labelled by an axiom of \mathcal{SC} .
3. Every internal node is labelled by an instance of the conclusion of a deductive rule of \mathcal{SC} . Then, this node has children that are labelled with the corresponding instances of the premises of that rule.

Thus, a sequent is derivable in \mathcal{SC} iff it occurs as a label in some proof in \mathcal{SC} .

Example 5.6.5

The following is a proof of the sequent $\psi, (\varphi \vee \vartheta) \Rightarrow (\psi \wedge \varphi), (\psi \wedge \vartheta)$:

$$\frac{\frac{\psi, \varphi \Rightarrow \psi, (\psi \wedge \vartheta) \quad \psi, \varphi \Rightarrow \varphi, (\psi \wedge \vartheta)}{\psi, \varphi \Rightarrow (\psi \wedge \varphi), (\psi \wedge \vartheta)} \quad \frac{\psi, \vartheta \Rightarrow (\psi \wedge \varphi), \psi \quad \psi, \vartheta \Rightarrow (\psi \wedge \varphi), \vartheta}{\psi, \vartheta \Rightarrow (\psi \wedge \varphi), (\psi \wedge \vartheta)}}{\psi, (\varphi \vee \vartheta) \Rightarrow (\psi \wedge \varphi), (\psi \wedge \vartheta)}$$

As for any other proof calculus, we need to show that \mathcal{SC} satisfies two fundamental properties:

- **Correctness:** All derivable sequents are valid.
- **Completeness:** Every valid sequent is derivable.

Correctness is easy to show.

Proposition 5.6.6

For every deductive rule of \mathcal{SC} and every suitable assignment \mathcal{A} the following holds true: \mathcal{A} falsifies the conclusion of the rule iff \mathcal{A} falsifies at least one of the premises of the rule. Thus, the conclusion is valid iff the premises are valid.

Proof. We prove the claim for the deduction rules $(\leftrightarrow\Rightarrow)$ and $(\Rightarrow\leftrightarrow)$. The proofs for the remaining deduction rules are similar. They are left as an exercise.

$(\leftrightarrow\Rightarrow)$: Assume that \mathcal{A} falsifies $\Gamma, \psi \leftrightarrow \varphi \Rightarrow \Delta$. Then $(\bigwedge \Gamma)^{\mathcal{A}} = (\psi \leftrightarrow \varphi)^{\mathcal{A}} = 1$, but $(\bigvee \Delta)^{\mathcal{A}} = 0$. If $\psi^{\mathcal{A}} = \varphi^{\mathcal{A}} = 1$, then \mathcal{A} falsifies the premise $\Gamma, \psi, \varphi \Rightarrow \Delta$. If $\psi^{\mathcal{A}} = \varphi^{\mathcal{A}} = 0$, then \mathcal{A} falsifies the premise $\Gamma \Rightarrow \Delta, \psi, \varphi$.

For the converse, first assume that \mathcal{A} falsifies $\Gamma \Rightarrow \Delta, \psi, \varphi$. Then $(\bigwedge \Gamma)^{\mathcal{A}} = 1$ and $(\bigvee \Delta)^{\mathcal{A}} = \psi^{\mathcal{A}} = \varphi^{\mathcal{A}} = 0$. But then \mathcal{A} falsifies $\Gamma, \psi \leftrightarrow \varphi \Rightarrow \Delta$ as well. Finally, assume that \mathcal{A} falsifies $\Gamma, \psi, \varphi \Rightarrow \Delta$. Then $(\bigwedge \Gamma)^{\mathcal{A}} = \psi^{\mathcal{A}} = \varphi^{\mathcal{A}} = 1$ and $(\bigvee \Delta)^{\mathcal{A}} = 0$. But then \mathcal{A} falsifies $\Gamma, \psi \leftrightarrow \varphi \Rightarrow \Delta$ as well.

$(\Rightarrow\leftrightarrow)$: Assume that \mathcal{A} falsifies $\Gamma \Rightarrow \Delta, \psi \leftrightarrow \varphi$. Then $(\bigwedge \Gamma)^{\mathcal{A}} = 1$, but $(\bigvee \Delta)^{\mathcal{A}} = (\psi \leftrightarrow \varphi)^{\mathcal{A}} = 0$. If $\psi^{\mathcal{A}} = 1$ and $\varphi^{\mathcal{A}} = 0$, then \mathcal{A} falsifies the premise $\Gamma, \psi \Rightarrow \Delta, \varphi$. If $\psi^{\mathcal{A}} = 0$ and $\varphi^{\mathcal{A}} = 1$, then \mathcal{A} falsifies the premise $\Gamma, \varphi \Rightarrow \Delta, \psi$.

For the converse, first assume that \mathcal{A} falsifies $\Gamma, \psi \Rightarrow \Delta, \varphi$. Then $(\bigwedge \Gamma)^{\mathcal{A}} = \psi^{\mathcal{A}} = 1$ and $(\bigvee \Delta)^{\mathcal{A}} = \varphi^{\mathcal{A}} = 0$. But then \mathcal{A} falsifies $\Gamma \Rightarrow \Delta, \psi \leftrightarrow \varphi$ as well. Finally, assume that \mathcal{A} falsifies $\Gamma, \varphi \Rightarrow \Delta, \psi$. Then $(\bigwedge \Gamma)^{\mathcal{A}} = \varphi^{\mathcal{A}} = 1$ and $(\bigvee \Delta)^{\mathcal{A}} = \psi^{\mathcal{A}} = 0$. But then \mathcal{A} falsifies $\Gamma \Rightarrow \Delta, \psi \leftrightarrow \varphi$ as well. \square

The correctness of \mathcal{SC} is an immediate consequence of this.

Proposition 5.6.7

Every sequent $\Gamma \Rightarrow \Delta$ that is derivable in \mathcal{SC} is valid.

Instead of proving sequents, \mathcal{SC} can also be used to prove formulas.

Definition 5.6.8

Let Ψ be a finite set of formulas. The formula ψ is DERIVABLE from Ψ , written $\Psi \vdash \psi$, iff the sequent $\Psi \Rightarrow \psi$ is derivable in \mathcal{SC} . In particular, ψ is derivable from the empty set of hypotheses, written $\vdash \psi$, iff the sequent $\emptyset \Rightarrow \psi$ is derivable in \mathcal{SC} .

The calculus \mathcal{SC} enables us to systematically search for a proof or falsifying assignment of sequents. We will derive an algorithm which, given the sequent $\Gamma \Rightarrow \Delta$, either constructs a proof of $\Gamma \Rightarrow \Delta$ or an assignment \mathcal{A} that falsifies $\Gamma \Rightarrow \Delta$.

We illustrate the idea of the algorithm using two examples.

Example 5.6.9

1. Consider $\psi = (X \rightarrow Y) \rightarrow (\neg Y \rightarrow \neg X)$. In order to determine whether ψ is valid we search for a proof of the sequent $\emptyset \Rightarrow \psi$. The formula ψ has the form $\varphi \rightarrow \vartheta$. The only deduction rule that can produce a sequent of the form $\emptyset \Rightarrow \varphi \rightarrow \psi$ is $(\Rightarrow \rightarrow)$. But in order to apply this rule, we first need to derive the sequent $\varphi \Rightarrow \vartheta$, i.e., the sequent $(X \rightarrow Y) \Rightarrow (\neg Y \rightarrow \neg X)$. Hence, the first step in constructing a proof is as follows:

$$\frac{(X \rightarrow Y) \Rightarrow (\neg Y \rightarrow \neg X)}{\emptyset \Rightarrow (X \rightarrow Y) \rightarrow (\neg Y \rightarrow X)}$$

Now, in order to derive $(X \rightarrow Y) \Rightarrow (\neg Y \rightarrow \neg X)$, we can either use the deduction rule $(\Rightarrow \rightarrow)$ or the deduction rule $(\Rightarrow \neg)$. The first possibility results in branching the derivation:

$$\frac{\emptyset \Rightarrow (\neg Y \rightarrow \neg X), X \quad Y \Rightarrow (\neg Y \rightarrow \neg X)}{(X \rightarrow Y) \Rightarrow (\neg Y \rightarrow \neg X)} \\ \frac{}{\emptyset \Rightarrow (X \rightarrow Y) \rightarrow (\neg Y \rightarrow X)}$$

The leafs of this tree can now be expanded using $(\Rightarrow \rightarrow)$ and then $(\neg \Rightarrow)$ and $(\Rightarrow \neg)$:

$$\frac{\frac{\neg Y, X \Rightarrow X}{\neg Y \Rightarrow \neg X, X} \quad \frac{Y \Rightarrow \neg X, Y}{Y, \neg Y \Rightarrow X}}{\emptyset \Rightarrow (\neg Y \rightarrow \neg X), X \quad Y \Rightarrow (\neg Y \rightarrow \neg X)} \\ \frac{(X \rightarrow Y) \Rightarrow (\neg Y \rightarrow \neg X)}{\emptyset \Rightarrow (X \rightarrow Y) \rightarrow (\neg Y \rightarrow X)}$$

Since the leafs of this tree are axioms, we have found a proof of the sequent.

If we had chosen the second possibility after the first expansion, we would have obtained the following proof:

$$\frac{\frac{Y, X \Rightarrow X \quad X \Rightarrow X, Y}{(X \rightarrow Y), X \Rightarrow Y}}{(X \rightarrow Y) \Rightarrow \neg X, Y} \\ \frac{(X \rightarrow Y), \neg Y \Rightarrow \neg X}{(X \rightarrow Y) \Rightarrow (\neg Y \rightarrow \neg X)} \\ \frac{}{\emptyset \Rightarrow (X \rightarrow Y) \rightarrow (\neg Y \rightarrow X)}$$

Thus, there can be more than one proof of a sequent.

2. Consider the sequent $(X \vee Y) \Rightarrow (X \wedge Y)$. Using the rule $(\Rightarrow \wedge)$, we obtain the tree

$$\frac{(X \vee Y) \Rightarrow X \quad (X \vee Y) \Rightarrow Y}{(X \vee Y) \Rightarrow (X \wedge Y)}$$

Applying $(\vee \Rightarrow)$ twice, we next obtain

$$\frac{\frac{X \Rightarrow X \quad Y \Rightarrow X}{(X \vee Y) \Rightarrow X} \quad \frac{X \Rightarrow Y \quad Y \Rightarrow Y}{(X \vee Y) \Rightarrow Y}}{(X \vee Y) \Rightarrow (X \wedge Y)}$$

The leafs contain only sequents between propositional variables, hence no deduction rule applies. Only the outermost leafs are axioms. The leaf $X \Rightarrow Y$ is falsified by \mathcal{A} such that $\mathcal{A}(X) = 1$ and $\mathcal{A}(Y) = 0$. Due to Proposition 5.6.6, the same assignment falsifies the sequent $(X \vee Y) \Rightarrow (X \wedge Y)$ as well.

As shown in the examples, a systematic search for a proof starts with the sequent to be proven and applies the deduction rules from bottom to top. It relies on the fact that for each sequent $\Gamma \Rightarrow \Delta$ and each non-atomic formula ψ occurring in it there is exactly one deduction rule with the conclusion $\Gamma \Rightarrow \Delta$ where ψ does not occur in any of the premises. The algorithm now picks a formula in $\Gamma \Rightarrow \Delta$ and applies that deduction rule backwards. The proof tree is thus extended by adding the premises of that rule. This process continues until either all leafs are labelled by axioms or a falsifiable sequent containing only atomic formulas is encountered.

Definition 5.6.10

A DERIVATION TREE T for a sequent S is a tree whose root is labelled by S such that every internal node is labelled by the bottom line of a deduction rule, while the children of that node are labelled with the sequents occurring on the top line of that deduction rule.

A leaf of T that is labelled by an axiom is called POSITIVE. A leaf is NEGATIVE if it is labelled with a sequent $\Gamma \Rightarrow \Delta$ where Γ and Δ are sets of propositional variables with $\Gamma \cap \Delta = \emptyset$. The tree T is COMPLETE if each of its leafs is either positive or negative.

Thus, a proof is a complete derivation tree containing only positive leafs. A derivation tree containing a negative leaf is a REFUTATION.

Now, we can construct an algorithm which decides whether a sequent is valid or falsifiable by constructing either a proof or a refutation.

Algorithm 5.6.11 (Proof search in \mathcal{SC})

```

Input: A sequent  $\Gamma \Rightarrow \Delta$ 
Output: A proof or a falsifying assignment
 $T := \text{new Tree}$ 
 $T.\text{root} := \Gamma \Rightarrow \Delta$ 
 $T.\text{root}.\text{marked} := \text{false}$ 
while  $T$  contains unmarked leaves do
   $l := \text{pick-unmarked-leaf}(T)$ 
   $\Gamma' \Rightarrow \Delta' := \text{label}(l)$ 
  if  $l$  is negative then
     $\mathcal{A} :=$  “the assignment with  $\mathcal{A}(X) = 1$  iff  $X$  occurs in  $\Gamma'$ ”
    return  $\mathcal{A}$ 
  else
    if  $l$  is positive then
       $l.\text{marked} := \text{true}$ 
    else
       $\psi := \text{pick-non-atomic-formula}(\Gamma' \Rightarrow \Delta')$ 
       $\text{rule} :=$  “the unique deduction rule with conclusion
         $\Gamma \Rightarrow \Delta$  whose premises don't contain  $\psi$ ”
       $T.\text{addchildren}(l, \text{premises}(\text{rule}))$ 
    end
  end
end
return  $T$ 

```

Proposition 5.6.12

Algorithm 5.6.11 terminates for every input sequent $\Gamma \Rightarrow \Delta$. It returns a proof iff $\Gamma \Rightarrow \Delta$ is valid. Otherwise, it returns a falsifying assignment.

Proof. The complexity of a sequent $\Gamma \Rightarrow \Delta$ is defined as the number of occurrences of $\neg, \wedge, \vee, \rightarrow$ and \leftrightarrow in $\Gamma \Rightarrow \Delta$. For each deduction rule, the complexity of the conclusion is strictly greater than the complexity of the premises. Hence, the height of the derivation tree cannot be bigger than the complexity of the input sequent. Thus, the algorithm terminates.

If the algorithm constructs a derivation tree T containing only marked leaves for the input sequent $\Gamma \Rightarrow \Delta$, then T is clearly a proof of $\Gamma \Rightarrow \Delta$. Since \mathcal{SC} is correct, the sequent $\Gamma \Rightarrow \Delta$ is then valid.

Otherwise, the derivation tree T that is constructed contains a negative leaf labelled $\Gamma' \Rightarrow \Delta'$ such that Γ' and Δ' are disjoint sets of propositional variables. The assignment \mathcal{A} that is constructed by the algorithm falsifies the sequent $\Gamma' \Rightarrow \Delta'$. Using Proposition 5.6.6, \mathcal{A} falsifies $\Gamma \Rightarrow \Delta$ as well. \square

The completeness of \mathcal{SC} is an immediate consequence.

Proposition 5.6.13

If $\Gamma \Rightarrow \Delta$ is valid, then it is derivable in \mathcal{SC} .