

Chapter 9

Semantic Tableaux

Piles of cards where building (or packing) is usually permitted. Cards may be packed on the exposed cards of each pile, according to rules that vary from game to game. Usually the game begins with a certain number of cards in each pile. When all the cards in a tableau pile have been played elsewhere, there is usually a rule about whether or what kind of cards can be moved to the empty space.

Definition of “tableaux” at www.goodsol.com.

Carved in the limestone of a desert cliff in Egypt is a 5,250-year-old tableau of a victorious ruler, perhaps the so-called King Scorpion – whose exploits, previously the stuff of myth and legend, may have been critical to the founding of Egyptian civilization.

New York Times.

Contents

9.1 Semantic Tableaux	118
9.2 Soundness and Completeness	122
9.3 Another Look at Tableaux	126
9.4 Subformula Property	130
Exercises	132

In this chapter we define semantic tableaux. They can be used for satisfiability-checking and finding models of a formula. The rules for building a tableau for a formula are based on the analysis of operation tables for logical connectives. By applying these rules to the input formula one can build all models of this formula.

In Section 9.1 we define semantic tableaux. In Section 9.2 we discuss the notions of soundness and completeness for logical calculi. In Section 9.3 we give a different tableau calculus that captures deletion of formulas from branches and establish completeness using an interesting proof-theoretic argument called *invertibility*. In Section 9.4 we show that tableau calculi have an important proof-theoretic property called the *subformula property*.

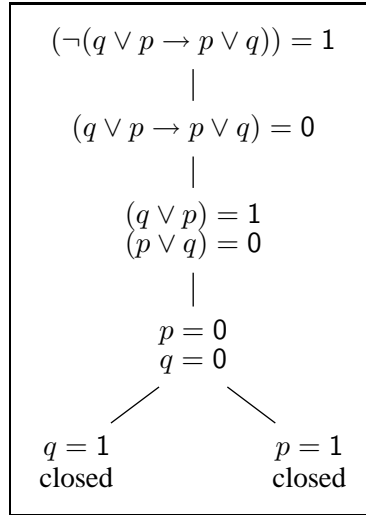


Figure 9.1: Semantic tableau for the signed formula $(\neg(q \vee p \rightarrow p \vee q)) = 1$

As an example, we show that Pure Atom Theorem 4.16 can be proved using proof-theoretic arguments based on the subformula property.

9.1 Semantic Tableaux

Consider the formula $\neg(q \vee p \rightarrow p \vee q)$ and suppose that we want to check if this formula is satisfiable. This formula is satisfiable if and only if the signed formula $(\neg(q \vee p \rightarrow p \vee q)) = 1$ has a model I . By inspecting the operation table for negation \neg , see Figure 3.2 on page 32, one can see that I must be the model of the signed formula $(q \vee p \rightarrow p \vee q) = 0$. Again, by inspecting the operation table for \rightarrow , we see that in order for I to satisfy $(q \vee p \rightarrow p \vee q) = 0$, it must satisfy both $(q \vee p) = 1$ and $(p \vee q) = 0$. For the formula $(p \vee q) = 0$ to be satisfied in I , we must have $p = 0$ and $q = 0$. For the formula $(q \vee p) = 1$ to be satisfied in I we must have either $q = 1$ or $p = 1$. But $q = 1$ contradicts to $q = 0$ and $p = 1$ contradicts to $p = 0$. Therefore, $\neg(q \vee p \rightarrow p \vee q)$ has no model.

The structure of the argument given above can be illustrated by putting the signed formulas used in the argument in a tree shown in Figure 9.1. The branches of this tree correspond to various alternatives in constructing a model of the signed formulas in the nodes of the tree. For example, branching on $q = 1$ and $p = 1$ corresponds to two alternative ways of satisfying $(q \vee p) = 1$.

Note the following: we systematically tried to build a model of a signed formula using the operation tables for the connectives occurring in the formula. The rules applied during the attempt of constructing a model depend on the main connective of the formula under consideration. For example, we know that every model of $(A \vee B) = 0$ must also be a

model of $A = 0$ and $B = 0$.

The tree shown in Figure 9.1 contains one or more signed formulas in its nodes. Such trees formalize search for models and will be called *tableaux*. The branches in tableaux can be of two kinds: *open* and *closed*. Intuitively, a branch is closed if it is established that the set of signed formulas on this branch has no model.

Let us formalize the process of building a tableau as a *one-player game*. Such a game consists of a sequence of *moves*. At each step of the game the player selects a move using the rules of the game. Each move changes the current *configuration*.

In the case of semantic tableaux a configuration is a tableau. To build a *tableau for a signed formula α* we initially create a tableau consisting of a single node labeled by α . Naturally, this *initial tableau for α* contains a single branch. We declare this branch open.

The game for building a tableau for a signed formula α consists of a sequence of moves of the following kind. The player repeatedly does the following:

- (1) Choose an open branch B in the current tableaux and a signed formula α on this branch such that α is not a signed atom.
- (2) Expand the tableau using the rules described below.

The tableau expansion rules add one or more formulas to the tableau. For some of the rules, the formulas are put on the branch B . Other rules add several new nodes below the branch, thus creating new branches. Suppose that the chosen branch B contains a formula β in the leaf. The rules are as follows. For simplicity, in the rules we use β also to denote the leaf node containing the formula β .

- (1) α has the form $(A_1 \wedge \dots \wedge A_n) = 0$. Add to β as children n nodes $A_1 = 0, \dots, A_n = 0$.
- (2) α has the form $(A_1 \wedge \dots \wedge A_n) = 1$. Add to β as a child a single node containing n formulas $A_1 = 1, \dots, A_n = 1$.
- (3) α has the form $(A_1 \vee \dots \vee A_n) = 0$. Add to β as a child a single node containing n formulas $A_1 = 0, \dots, A_n = 0$.
- (4) α has the form $(A_1 \vee \dots \vee A_n) = 1$. Add to β as children n nodes $A_1 = 1, \dots, A_n = 1$.
- (5) α has the form $(A_1 \rightarrow A_2) = 0$. Add to β as a child a single node containing two formulas $A_1 = 1, A_2 = 0$.
- (6) α has the form $(A_1 \rightarrow A_2) = 1$. Add to β as children two nodes $A_1 = 0$ and $A_2 = 1$.
- (7) α has the form $(\neg A_1) = 0$. Add to β as a child a single node containing the formula $A_1 = 1$.

$(A_1 \wedge \dots \wedge A_n) = 0$	\rightsquigarrow	$A_1 = 0 \mid \dots \mid A_n = 0$
$(A_1 \wedge \dots \wedge A_n) = 1$	\rightsquigarrow	$A_1 = 1, \dots, A_n = 1$
$(A_1 \vee \dots \vee A_n) = 0$	\rightsquigarrow	$A_1 = 0, \dots, A_n = 0$
$(A_1 \vee \dots \vee A_n) = 1$	\rightsquigarrow	$A_1 = 1 \mid \dots \mid A_n = 1$
$(A_1 \rightarrow A_2) = 0$	\rightsquigarrow	$A_1 = 1, A_2 = 0$
$(A_1 \rightarrow A_2) = 1$	\rightsquigarrow	$A_1 = 0 \mid A_2 = 1$
$(\neg A_1) = 0$	\rightsquigarrow	$A_1 = 1$
$(\neg A_1) = 1$	\rightsquigarrow	$A_1 = 0$
$(A_1 \leftrightarrow A_2) = 0$	\rightsquigarrow	$A_1 = 0, A_2 = 1 \mid A_1 = 1, A_2 = 0$
$(A_1 \leftrightarrow A_2) = 1$	\rightsquigarrow	$A_1 = 0, A_2 = 0 \mid A_1 = 1, A_2 = 1$

Figure 9.2: Branch expansion rules

- (8) α has the form $(\neg A_1) = 1$. Add to β as a child a single node containing the formula $A_1 = 0$.
- (9) α has the form $(A_1 \leftrightarrow A_2) = 0$. Add to β two children, the first containing $A_1 = 0, A_2 = 1$, the second containing $A_1 = 1, A_2 = 0$.
- (10) α has the form $(A_1 \leftrightarrow A_2) = 1$. Add to β two children, the first containing $A_1 = 0, A_2 = 0$, the second containing $A_1 = 1, A_2 = 1$.

These rules are summarized in Figure 9.2. The signed formula α is shown on the left of the \rightsquigarrow symbol. The children of the leaf node β are shown on the right of \rightsquigarrow , divided by \mid . We will refer to these rules of Figure 9.2 as *branch expansion rules* since they expand the selected branch of the tableau.

In addition to the branch expansion rules there are two *branch closure rules* formulated as follows:

- (1) If a branch of the tableau contains a pair of signed atoms $p = 0$ and $p = 1$. then this branch is marked as closed.
- (2) If a branch of the tableau contains a signed formula $\top = 0$ or $\perp = 1$, then this branch is marked as closed.

The tableau building game is non-deterministic, because different branches and formulas can be selected at every step. The player can choose different *strategies* for expanding branches. In principle, the game can be infinite because, for example, the player can repeatedly select the top node. To make any game finite, we impose the following restriction on branch expansion rules: it is not allowed to select a signed formula on a branch if this signed formula has previously been selected on that branch. Moreover, we do not want to expand closed branches. Thus the branch expansion rules are modified so that

- (1) the player chooses an open branch;
- (2) he selects a formula not previously been selected on this branch.

It is not hard to argue that with this restriction any game is finite, because the signed formulas added after every move have a smaller size than the chosen signed formula.

We have two possible outcomes for every game:

- (1) There are no possible moves because all branches are closed.
- (2) There is an open branch but on this branch every signed formula has already been selected.

We claim that in the first case the initial signed formula is unsatisfiable, while in the second case it is satisfiable. The proof of this fact will be given in the next section. Moreover, we will show how, in the second case, one can build a model of the formula.

EXAMPLE 9.1 Consider again the formula $\neg((p \rightarrow q) \wedge (p \wedge q \rightarrow r) \rightarrow (p \rightarrow r))$. To check if this formula is satisfiable, we try to build a tableau for the signed formula $(\neg((p \rightarrow q) \wedge (p \wedge q \rightarrow r) \rightarrow (p \rightarrow r))) = 1$. Such a tableau is shown in Figure 9.3. To make it clear which signed formula is selected at each node, we denote the signed formulas by (a)–(g) on the right of the figure and label each arc by one of (a)–(g) as well. Since all branches of the tableau are closed, the formula is unsatisfiable. \square

Suppose that a game terminated and the tableau has an open branch. Let us explain how one can build a model of the signed formula at the root in this case. Let B be the set of all signed formulas on the open branch. Consider any interpretation I such that for every boolean variable p we have

$$I(p) \stackrel{\text{def}}{=} \begin{cases} 0, & \text{if } (p = 0) \in B; \\ 1, & \text{if } (p = 1) \in B. \end{cases}$$

If neither $p = 0$ nor $p = 1$ lie on the branch, we can define $I(p)$ in an arbitrary way. It is not hard to argue that at least one such model exists since B may contain at most one of the signed formulas $p = 0$ and $p = 1$. Note that the construction of I uses only signed atomic formulas in B . We claim that I is a model of the signed formula at the root of the tableau. The proof of this fact will be given in the next section. Moreover, it will be seen from the proof that I is also a model of *every* signed formula on the branch.

EXAMPLE 9.2 Consider again the formula $\neg((p \rightarrow q) \wedge (p \wedge q \rightarrow r) \rightarrow (\neg p \rightarrow r))$. To check if this formula is satisfiable, we try to build a tableau for the signed formula $(\neg((p \rightarrow q) \wedge (p \wedge q \rightarrow r) \rightarrow (\neg p \rightarrow r))) = 1$. A part of a tableau for this formula is shown in Figure 9.4. It contains two open branches to which all possible rules have been applied. The leftmost branch contains two signed atoms $p = 0$ and $r = 0$. This means that the interpretations $\{p \mapsto 0, r \mapsto 0, q \mapsto b\}$, where b is an arbitrary boolean value, are models of the formula. The branch next to it also gives us a model $\{p \mapsto 0, q \mapsto 0, r \mapsto 0\}$. \square

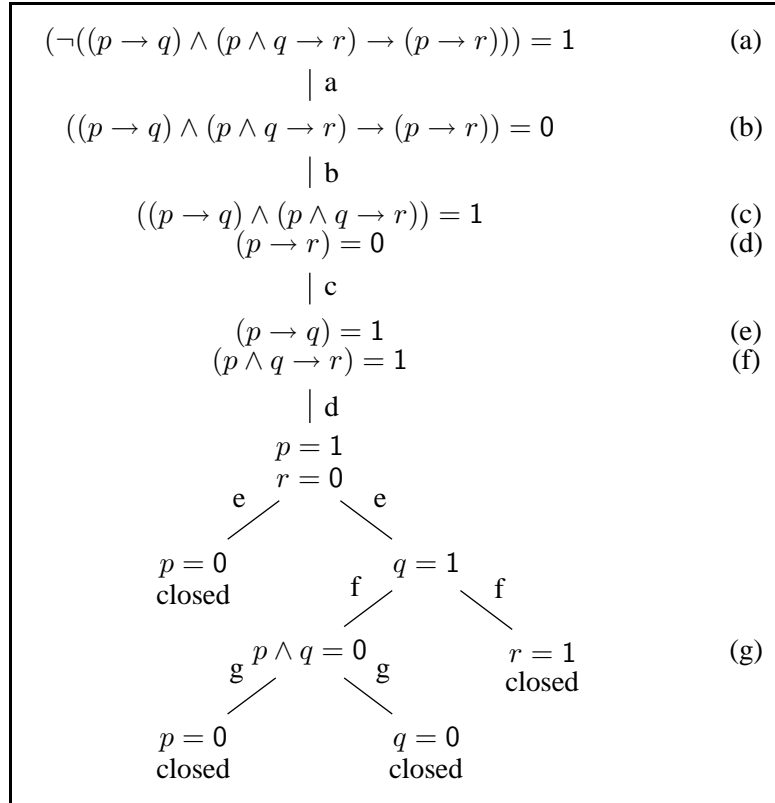


Figure 9.3: Semantic tableau for the formula $\neg((p \rightarrow q) \wedge (p \wedge q \rightarrow r) \rightarrow (p \rightarrow r))$

9.2 Soundness and Completeness

Logics are usually defined in a semantical way, using a suitable notion of model. Logical calculi, such as the resolution calculus, of tableaux, are defined using a notion of proof or derivation. One has to show that model-theoretic notions are consistent with proof-theoretic ones. There are two key properties of logical calculi reflecting their adequacy to model-theoretic notions: *soundness* and *completeness*. Intuitively, *soundness* means that formulas having proofs are also semantically sound (for example, valid or satisfiable). *Completeness* is the inverse property: every semantically sound formula has a proof. It is usually relatively easy to show that a calculus is sound by showing that every inference of the calculus preserves soundness. Showing completeness is usually more involved.

In this section we prove that the tableau method is sound and complete. That is, a signed formula α is satisfiable if and only if every terminated game for α has an open branch. Likewise, α is unsatisfiable if and only if every terminated game for α results in a tree with all branches closed. The proof of the soundness and completeness is very interesting and based on arguments which will be used in this book over and over again. It emphasizes

the initial tableau and the claim is straightforward. So, we assume that the lemma holds for some tableau T and show that also holds for every tableau T' obtained from T by a branch expansion rule. Suppose that B is the branch chosen in T and α is the selected formula of this branch. We will consider only one case, when α has the form $(A_1 \wedge \dots \wedge A_n) = 0$, all other cases will be similar. Let B, B_1, \dots, B_m be all branches of T . Then the branches of T' are $B_1, \dots, B_m, B \cup \{A_1 = 0\}, \dots, B \cup \{A_n = 0\}$. By the induction hypothesis, I is a model of γ if and only if it is a model of some branch in T . So it remains to prove that I is model of some branch in T if and only if it is a model of some branch in T' . Note that this statement is trivial if I is a model of at least one of B_1, \dots, B_m , so we assume that I is not a model of any of the B_1, \dots, B_m .

Suppose that I is a model of B . Then I is a model of $(A_1 \wedge \dots \wedge A_n) = 0$, since this signed formula lies on B . But then I is also a model of some $A_i = 0$, and so a model of $B \cup \{A_i = 0\}$.

Conversely, suppose that I is a model of some branch $B \cup \{A_i = 0\}$. Obviously, it is also a model of B . \square

Let us call a set of signed formulas *trivially unsatisfiable* if it either contains a pair of signed atoms $p = 0$ and $p = 1$, or it contains at least one of the signed formulas $\top = 0$ and $\perp = 1$. Obviously, any trivially unsatisfiable set is unsatisfiable. Moreover, we have the following lemma.

LEMMA 9.4 *A tableau branch is closed if and only if it is trivially unsatisfiable.* \square

Lemma 9.3 implies one part of the soundness and completeness theorem given in the following lemma.

LEMMA 9.5 *Let T be a tableau for γ . If all branches in T are closed, then γ is unsatisfiable.*

PROOF. Suppose, by contradiction, that γ has a model I . Then, by Lemma 9.3, I is also a model of some branch in T . But every closed branch is trivially unsatisfiable, and hence cannot have a model. We obtain a contradiction. \square

Let us now show that an open branch in a terminated tableau has a model. To this end, let us introduce an auxiliary notion of *saturated set*.¹

DEFINITION 9.6 (Saturated Set) A set S of signed formulas is called *saturated* if it satisfies the following properties:

- (1) If S contains a signed formula $(A_1 \wedge \dots \wedge A_n) = 0$, then S also contains at least one of the signed formulas $A_1 = 0, \dots, A_n = 0$.
- (2) If S contains a signed formula $(A_1 \wedge \dots \wedge A_n) = 1$, then S also contains all of the signed formulas $A_1 = 1, \dots, A_n = 1$.

¹The notion of saturated set of signed formulas here should not be mixed with the notion of \mathbb{I} -saturated set of formulas defined in the context of inference systems. For inference systems, a set is saturated with respect to inferences in this system. ‘‘Saturated’’ in this chapter is a semantic notion. It is sometimes called ‘‘downward saturated’’ because it imposes some conditions on subformulas of formulas in this set.

- (3) If S contains a signed formula $(A_1 \vee \dots \vee A_n) = 0$, then S also contains all of the signed formulas $A_1 = 0, \dots, A_n = 0$.
- (4) If S contains a signed formula $(A_1 \vee \dots \vee A_n) = 1$, then S also contains at least one of the signed formulas $A_1 = 1, \dots, A_n = 1$.
- (5) If S contains a signed formula $(A_1 \rightarrow A_2) = 0$, then S also contains both $A_1 = 1$ and $A_2 = 0$.
- (6) If S contains a signed formula $(A_1 \rightarrow A_2) = 1$, then S also contains either $A_1 = 0$ or $A_2 = 1$.
- (7) If S contains a signed formula $(\neg A_1) = 0$, then S also contains $A_1 = 1$.
- (8) If S contains a signed formula $(\neg A_1) = 1$, then S also contains $A_1 = 0$.
- (9) If S contains a signed formula $(A_1 \leftrightarrow A_2) = 0$, then S also either contains both of the signed formulas $A_1 = 0$ and $A_2 = 1$, or both of the signed formulas $A_1 = 1$ and $A_2 = 0$.
- (10) If S contains a signed formula $(A_1 \leftrightarrow A_2) = 1$, then S also either contains both of the signed formulas $A_1 = 0$ and $A_2 = 0$, or both of the signed formulas $A_1 = 1$ and $A_2 = 1$. \square

It is easy to observe the following fact.

LEMMA 9.7 *Let a game for γ terminate at a tableau T . Then every open branch in γ is a saturated set.* \square

The following lemma is a key result for the second part of the soundness and completeness theorem. The proof of this lemma also gives us a way of finding a model of an open tableau branch.

LEMMA 9.8 *Let B be a saturated set of formulas. Then B is satisfiable if and only if it is not trivially unsatisfiable.*

PROOF. Evidently, if B is trivially unsatisfiable, then it is unsatisfiable. So we assume that B is not trivially unsatisfiable and show how to build a model of B . Consider any interpretation I defined as follows: for every boolean variable p we have

$$I(p) \stackrel{\text{def}}{=} \begin{cases} 0, & \text{if } (p = 0) \in B; \\ 1, & \text{if } (p = 1) \in B. \end{cases}$$

The value $I(q)$ for boolean variables q such that neither $q = 0$ nor $q = 1$ belong to B can be arbitrary. Note that I is well-defined: since B is not trivially unsatisfiable, at most one of the formulas $p = 0$ or $p = 1$ belongs to B . We claim that I is model of B , that is, a model of every signed formula $\alpha \in B$. The proof is by induction on the size of α .

When α has the form $p = 0$ or $p = 1$, where p is a boolean variable, we have $I \models \alpha$ by the construction of I . Among all other possible cases for α we consider only one, namely, when α has the form $(A_1 \wedge \dots \wedge A_n) = 0$. By the definition of saturated set, B also contains a signed formula $A_i = 0$, for some $i \in \{1, \dots, n\}$. Note that A_i has a smaller size than α . By the induction hypothesis we have $I \models (A_i = 0)$. But then obviously $I \models \alpha$. \square

Now we are ready to prove the main theorem of this section.

THEOREM 9.9 (Soundness and Completeness) *Let T be a tableau resulting from a terminated game for a signed formula γ . Then γ is unsatisfiable if and only if all branches in T are closed.*

PROOF. If all branches in T are closed, then by Lemma 9.5 γ is unsatisfiable. If there exists an open branch B , then by Lemma 9.7, B is saturated, so by Lemma 9.8 B has a model I . But every branch in a tree contains the root, hence $\gamma \in B$, so I is also a model of γ . \square

Note that this theorem is formulated independently on the strategy used for choosing tableau branches and formulas on these branches.

Let us show some applications of Soundness and Completeness Theorem 9.9 for the problems of checking satisfiability, validity, and equivalence.

COROLLARY 9.10 *The following statements hold.*

- (1) *A formula F is satisfiable if and only if every (some) terminated tableau for $F = 1$ has an open branch.*
- (2) *A formula F is valid if and only if in every (some) terminated tableau for $F = 0$ all branches are closed.*
- (3) *Formulas F_1 and F_2 are equivalent if and only if in every (some) terminated tableau for $(F_1 \leftrightarrow F_2) = 0$ all branches are closed.*

PROOF. (1) is immediate by Completeness Theorem 9.9. Consider (2). Let T be any terminated tableau for $F = 0$. If all branches in T are closed, then by Theorem 9.9 the signed formula $F = 0$ is false in every interpretation. But then $F = 1$ is true in every interpretation, hence F is valid. (3) is analogous to (2). \square

9.3 Another Look at Tableaux

In this section we give an alternative formalization of semantic tableaux. A formalization of tableaux as trees is very intuitive and convenient for proving completeness but does not capture an important point about the tableau construction. Namely, when a formula is selected on a branch, this formula should not be selected again. This can be formalized as

deletion of the formula from the branch, but we cannot properly display deletion from a branch in a tree: if we delete a formula from a tree, then it is automatically deleted from *all* branches containing this formula. Another not very elegant point about our formalization is that we had to formalize branch close rules in a way different from branch extension rules.

This gives us an idea of dealing with a collection of branches instead of a tree. In this section we give an alternative formalization of a tableau as a collection of branches.

DEFINITION 9.11 (Branch, Tableau) A *branch* is a finite non-empty set of signed formulas. A *tableau* is a finite set $\{B_1, \dots, B_n\}$ of branches, denoted by $B_1 \mid \dots \mid B_n$. When $n = 0$, we speak of the *empty tableau*, and denote it by \boxplus . An interpretation I is called a *model* of a tableau T if it is a model of at least one branch in this tableau. \square

Note that by this definition the empty tableau has no models. We will use the words “satisfiable” or “satisfy” for tableaux in the same way as we did for formulas. Note that a tableau $\alpha_1, \dots, \alpha_m$ consisting of a single branch is satisfiable if and only if the set of signed formulas $\alpha_1, \dots, \alpha_m$ is satisfiable.

Let us now introduce a tableau calculus for propositional logic. In this calculus derivable objects are tableaux.

DEFINITION 9.12 (Tableau Calculus) The *tableau calculus* for propositional logic consists of the following inference rules.

- (1) *Branch expansion rules:* for every rule of Figure 9.2, if the rule has a form $\alpha \rightsquigarrow S_1 \mid \dots \mid S_m$ and a tableau T has a branch B, α containing α , then replace in T this branch by m branches $B \cup S_1, \dots, B \cup S_m$.
- (2) *Branch closure rules:* if a branch contains the signed formula $\top = 0$, or the signed formula $\perp = 1$, or a pair of signed atoms $p = 0$ and $p = 1$, then delete this branch from the tableau. \square

The branch expansion rule can be presented as an inference rule as follows:

$$\frac{B, \alpha \mid B_1 \mid \dots \mid B_n}{B, S_1 \mid \dots \mid B, S_m \mid B_1 \mid \dots \mid B_n} .$$

One of the branch closure rules can be presented as an inference rule as follows:

$$\frac{B, p = 0, p = 1 \mid B_1 \mid \dots \mid B_n}{B_1 \mid \dots \mid B_n} .$$

Note that in the branch expansion rules the signed formula α is *deleted* from the branch B , or more precisely, *replaced* by m signed formulas S_1, \dots, S_m . Also, instead of closing branches, we simply delete them from the tableau.

$$\begin{aligned}
& (\neg((p \rightarrow q) \wedge (p \wedge q \rightarrow r) \rightarrow (p \rightarrow r))) = 1 \Rightarrow \\
& ((p \rightarrow q) \wedge (p \wedge q \rightarrow r) \Rightarrow (p \rightarrow r)) = 0 \Rightarrow \\
& ((p \rightarrow q) \wedge (p \wedge q \rightarrow r)) = 1, (p \rightarrow r) = 0 \Rightarrow \\
& (p \rightarrow q) = 1, (p \wedge q \rightarrow r) = 1, (p \Rightarrow r) = 0 \Rightarrow \\
& (p \Rightarrow q) = 1, (p \wedge q \rightarrow r) = 1, p = 1, r = 0 \Rightarrow \\
& p = 0, (p \wedge q \rightarrow r) = 1, p = 1, r = 0 \mid q = 1, (p \wedge q \rightarrow r) = 1, p = 1, r = 0 \Rightarrow \\
& q = 1, (p \wedge q \Rightarrow r) = 1, p = 1, r = 0 \Rightarrow \\
& q = 1, (p \wedge q) = 0, p = 1, r = 0 \mid q = 1, r = 1, p = 1, r = 0 \Rightarrow \\
& q = 1, (p \wedge q) = 0, p = 1, r = 0 \Rightarrow \\
& q = 1, p = 0, p = 1, r = 0 \mid q = 1, q = 0, p = 1, r = 0 \Rightarrow \\
& q = 1, q = 0, p = 1, r = 0 \Rightarrow \\
& \boxplus
\end{aligned}$$

Figure 9.5: A tableau derivation

EXAMPLE 9.13 Consider an example that shows how the tableau calculus works for the tableau of Figure 9.3. We apply the tableau calculus inference rules to the initial tableau consisting of a single signed formula $(\neg((p \rightarrow q) \wedge (p \wedge q \rightarrow r) \rightarrow (p \rightarrow r))) = 1$. A derivation of the empty tableau from the tableau consisting of this signed formula is illustrated in Figure 9.5. We distinguish the formula to which we apply a branch expansion rule by shading its main connective. For branch closure rules, we shade both signed formulas of the form $p = 0$ and $p = 1$. \square

The following lemma is straightforward.

LEMMA 9.14 *Let T be a tableau and suppose that no inference can be applied to T . Then*

- (1) *the tableau contains only signed atoms and signed formulas $\top = 1, \perp = 0$;*
- (2) *there is no tableau branch that contains a pair of signed formulas $p = 0$ and $p = 1$;*
- (3) *every branch in T is satisfiable;*
- (4) *T is satisfiable if and only if T is non-empty.*

PROOF.

- (1) If T contains any other signed formula $F = 1$ or $F = 0$, then there is a branch expansion rule applicable to this signed formula.
- (2) If a branch contains both $p = 0$ and $p = 1$, then one can apply a branch closure rule to this branch.

- (3) Take any branch B in T . Consider any interpretation I such that for all variables p and boolean values b we have $I(p) = b$ if the branch contains the signed formula $p = b$. By (2) such an interpretation exists. Evidently, it satisfies all signed atoms $p = b$ on the branch and all formulas $\top = 1, \perp = 0$, so I satisfies B .
- (4) If T has at least one branch, then by (3) this branch has a model, so T is satisfiable. If T is empty, then it is unsatisfiable. \square

This Lemma and arguments about *invertibility* of inferences allow one to prove soundness and completeness of the tableau calculus in a rather straightforward way.

DEFINITION 9.15 (Invertible) An inference is called *invertible* if it has the following property: the conclusion of this inference is satisfiable, if and only if some of its premises is satisfiable too. An inference rule is invertible if every inference in it is invertible. \square

Note that every inference in the tableau inference system has exactly one premise and exactly one conclusion.

LEMMA 9.16 *Every inference of the tableau calculus is invertible.*

PROOF. We will prove an even stronger property: the premise and the conclusion of every inference have the same models. We will prove it for two rules, the proof for the remaining rules is similar.

Consider the rule for the signed formula $(A_1 \wedge \dots \wedge A_n) = 0$:

$$\frac{(A_1 \wedge \dots \wedge A_n) = 0, \beta \mid \beta_1 \mid \dots \mid \beta_m}{A_1 = 0, \beta \mid \dots \mid A_n = 0, \beta \mid \beta_1 \mid \dots \mid \beta_m} .$$

Suppose that I is a model of the premise. If for some $i = 1, \dots, m$ we have $I \models \beta_i$, then I is also a model of the conclusion. Suppose $I \models (A_1 \wedge \dots \wedge A_n) = 0, \beta$. Then $I \models \beta$ and $I \not\models A_1 \wedge \dots \wedge A_n$. Then for some $j = 1, \dots, n$ we have $I \not\models A_j$, hence $I \models A_j = 0, \beta$ and so I is a model of the conclusion. The proof that every model of the conclusion is also a model of the premise is similar.

Consider now the branch closure rule. Any branch closure inference removes from the tableau a branch with no models. Therefore, it does not change the satisfiability of the tableau. \square

THEOREM 9.17 (Soundness and Completeness) *Let γ be a signed formula and T be any tableau such that (i) there is a derivation of T from γ ; (ii) no inference can be applied to T . Then γ is unsatisfiable if and only if $T = \boxplus$.*

PROOF. Since T is obtained from γ by a sequence of inferences, by Lemma 9.16 γ is satisfiable if and only if T is satisfiable. But by Lemma 9.14 a tableau to which no inference applies is satisfiable if and only if it is non-empty. \square

signed formula	immediate signed subformulas
$(A_1 \wedge \dots \wedge A_n) = 0$	$A_1 = 0 \quad \dots \quad A_n = 0$
$(A_1 \wedge \dots \wedge A_n) = 1$	$A_1 = 1 \quad \dots \quad A_n = 1$
$(A_1 \vee \dots \vee A_n) = 0$	$A_1 = 0 \quad \dots \quad A_n = 0$
$(A_1 \vee \dots \vee A_n) = 1$	$A_1 = 1 \quad \dots \quad A_n = 1$
$(A_1 \rightarrow A_2) = 0$	$A_1 = 1 \quad A_2 = 0$
$(A_1 \rightarrow A_2) = 1$	$A_1 = 0 \quad A_2 = 1$
$(\neg A_1) = 0$	$A_1 = 1$
$(\neg A_1) = 1$	$A_1 = 0$
$(A_1 \leftrightarrow A_2) = 0$	$A_1 = 0 \quad A_1 = 1 \quad A_2 = 0 \quad A_2 = 1$
$(A_1 \leftrightarrow A_2) = 1$	$A_1 = 0 \quad A_1 = 1 \quad A_2 = 0 \quad A_2 = 1$

Figure 9.6: Immediate signed subformula

9.4 Subformula Property

It is clear from the definition of the branch expansion rules that all formulas occurring in a tableau are subformulas of the root formula. This and similar properties of logical calculi are usually referred to as a *subformula property*.

In fact, an even stronger subformula property takes place for the tableau calculus of this chapter. To formulate this subformula property, first we need to extend the notion of subformula to signed formulas.

DEFINITION 9.18 (Signed Subformula) The *immediate signed subformulas* of a signed formula α are defined as in Figure 9.6. The *signed subformulas* of a signed formula are defined as follows:

- (1) Every signed formula α is a signed subformula of itself.
- (2) If α_1 is an immediate signed subformula of α_2 and α_2 is a signed subformula of α_3 , then α_1 is a signed subformula of α_3 .

In other words, the relation “signed subformula” is the reflexive and transitive closure of the relation “immediate signed subformula”. \square

Compare this definition with that of the branch expansion rules on page 119.

Signed subformulas of a signed formula $F = b$ encode not only information about the subformulas of F , but also information about polarities of occurrences of these subformulas. For example, consider the signed formula $(p \wedge q) = 1$. Then $p = 1$ is a signed subformula of this formula but $p = 0$ is not. The reason is that p has only positive occurrences in $p \wedge q$.

LEMMA 9.19 *Let A be a formula and B be the subformula of A at a position π .*

- (1) If $pol(A, \pi) = 1$ then $B = 0$ is a signed subformula of $A = 0$ and $B = 1$ is a signed subformula of $A = 1$.
- (2) If $pol(A, \pi) = -1$ then $B = 0$ is a signed subformula of $A = 1$ and $B = 1$ is a signed subformula of $A = 0$.
- (3) If $pol(A, \pi) = 0$ then both $B = 0$ and $B = 1$ are signed subformulas of $A = 0$ and also signed subformulas of $A = 1$.

PROOF. The proof is by induction on π . When $\pi = \epsilon$, the statement is obvious. When π is non-empty, $\pi = \pi'.n$ for some position π and positive integer n . We will consider only one case, when $A|_{\pi'}$ has the form $B_1 \rightarrow B_2$, other cases are analogous. In this case either $n = 1$ and $B = B_1$ or $n = 2$ and $B = B_2$. Let us consider the three possible values of $pol(A, \pi')$ and show that in all cases the conditions of the lemma are satisfied.

- (1) Case $pol(A, \pi') = 1$. The induction hypothesis yields that $B_1 \rightarrow B_2 = 0$ is a signed subformula of $A = 0$ and $B_1 \rightarrow B_2 = 1$ is a signed subformula of $A = 1$. Consider only the case $B_1 \rightarrow B_2 = 0$, the other case is analogous. By the definition of signed subformulas we have that $B_1 = 1$ and $B_2 = 0$ are signed subformulas of $A = 0$. By Definition 4.10 of position and polarity we have $A|_{\pi'.1} = B_1$, $pol(A, \pi'.1) = -1$ and $A|_{\pi'.2} = B_2$, $pol(A, \pi'.2) = 1$, so the lemma conditions hold for both $\pi'.1$ and $\pi'.2$.
- (2) Case $pol(A, \pi') = -1$ is symmetric to the previous one.
- (3) Case $pol(A, \pi') = 0$ can be proved by joining together the two previous cases. \square

In a similar way one can prove the converse to Lemma 9.19 given below.

LEMMA 9.20 Let A_1 be a formula, A_2 a subformula of A_1 , and b_1, b_2 be boolean values.

- (1) If A_2 has only positive occurrences in A_1 , then $A_2 = b_2$ is a signed subformula of $A_1 = b_1$ if and only if $b_1 = b_2$.
- (2) If A_2 has only negative occurrences in A_1 , then $A_2 = b_2$ is a signed subformula of $A_1 = b_1$ if and only if $b_1 \neq b_2$.
- (3) In all other cases $A_2 = b_2$ is a signed subformula of $A_1 = b_1$. \square

The following theorem expresses a fundamental property of our tableau calculi.

THEOREM 9.21 (Subformula Property) Let T be a tableau for a signed formula γ . Then every signed formula occurring in T is a signed subformula of γ .

PROOF. By routine inspection of the branch expansion rules. \square

One of possible corollaries of Subformula Property is that an atom A pure in a formula F will never participate in a branch closure rule in a tableau for a signed formula $F = b$, see Exercise 9.9.

Our proof of Completeness Theorem 9.9 based on saturated sets indirectly used the subformula property. Indeed, a saturated set only contains signed subformulas of the root formula.

Exercises

EXERCISE 9.1 Show unsatisfiability of each of the following formulas using semantic tableaux:

$$(p \leftrightarrow q) \leftrightarrow (\neg q \leftrightarrow p);$$

$$\neg((\neg q \rightarrow \neg p) \rightarrow ((\neg q \rightarrow p) \rightarrow q)). \quad \square$$

EXERCISE 9.2 Show satisfiability of each of the following formulas using semantic tableaux:

$$(p \leftrightarrow q) \rightarrow (\neg q \leftrightarrow p);$$

$$\neg(p \vee q \rightarrow ((\neg p \wedge q) \vee p \vee \neg q)). \quad \square$$

EXERCISE 9.3 Show validity of each of the following formulas using semantic tableaux:

$$(p \rightarrow q) \rightarrow ((p \rightarrow \neg q) \rightarrow \neg p);$$

$$(p \rightarrow r) \rightarrow (p \vee q \rightarrow r \vee q). \quad \square$$

EXERCISE 9.4 For each of the following formulas, describe *all* models of this formula using semantic tableaux:

$$(q \rightarrow (p \wedge r)) \wedge \neg(p \vee r \rightarrow q);$$

$$\neg(p \vee q \rightarrow p \wedge q). \quad \square$$

EXERCISE 9.5 Establish the following equivalences using semantic tableaux:

$$(p \rightarrow \neg p) \equiv \neg p;$$

$$(p \rightarrow q) \equiv (\neg q \rightarrow \neg p);$$

$$(p \vee q) \wedge (p \vee \neg q) \equiv p. \quad \square$$

EXERCISE 9.6 Suppose that we extended the set of logical connectives by the *if-then-else* connective whose semantics is defined as follows. For every interpretation I we have

$$I(\text{if } A \text{ then } B \text{ else } C) \stackrel{\text{def}}{=} \begin{cases} I(B), & \text{if } I(A) = 1; \\ I(C), & \text{if } I(A) = 0. \end{cases}$$

Define tableau rules for this connective. \square

EXERCISE 9.7 Consider the following *exclusive or* connective \oplus defined as follows: for every interpretation I we have $I \models A_1 \oplus \dots \oplus A_n$ if and only if the cardinality of the set

$$\{i \mid 1 \leq i \leq n \text{ and } I \models A_i\}$$

is odd. Define tableau rules for \oplus . \square

EXERCISE 9.8 Prove completeness of semantic tableaux in the form of trees by invertibility arguments similar to those used for the tableau calculus (proof of Theorem 9.17). \square

EXERCISE 9.9 Let A be an atom pure in a formula F and b be a boolean value. Then in any tableau for the signed formula $F = b$, A will not be used a branch closure rule. \square