10

Deciding a Combination of Theories

10.1 Introduction

The decision procedures that we have studied so far focus on one specific theory. Verification conditions that arise in practice, however, frequently mix expressions from several theories. Consider the following examples:

- A combination of linear arithmetic and uninterpreted functions:

$$(x_2 \geq x_1) \wedge (x_1 - x_3 \geq x_2) \wedge (x_3 \geq 0) \wedge f(f(x_1) - f(x_2)) \neq f(x_3) \quad (10.1)$$

- A combination of bit vectors and uninterpreted functions:

$$f(a[32], b[1]) = f(b[32], a[1]) \wedge a[32] = b[32] \quad (10.2)$$

- A combination of arrays and linear arithmetic:

$$x = v\{i \longleftarrow e\}[j] \wedge y = v[j] \wedge x > e \wedge x > y \quad (10.3)$$

In this chapter, we cover the popular **Nelson–Oppen** combination method. This method assumes that we have a decision procedure for each of the theories involved. The Nelson–Oppen combination method permits the decision procedures to communicate information with one another in a way that guarantees a sound and complete decision procedure for the combined theory.

10.2 Preliminaries

Let us recall several basic definitions and conventions that should be covered in any basic course on mathematical logic (see also Sect. 1.4). We assume a basic familiarity with first-order logic here.

First-order logic is a baseline for defining various restrictions thereof, which are called **theories**. It includes:

D. Kroening and O. Strichman, *Decision Procedures*,
Texts in Theoretical Computer Science. An EATCS Series,
DOI 10.1007/978-3-662-50497-0_10

- Variables
- **Logical symbols** that are shared by all theories, such as the Boolean operators ($\wedge$, $\vee$, ...), quantifiers ($\forall$, $\exists$), and parentheses
- **Nonlogical symbols**, namely function and predicate symbols, that are uniquely specified for each theory
- Syntax

It is common to consider the equality sign as a logical symbol rather than a predicate that is specific to a theory, since first-order theories without this symbol are rarely considered. We follow this convention in this chapter.

A first-order theory is defined by a set of sentences (first-order formulas in which all variables are quantified). It is common to represent such sets by a set of axioms, with the implicit meaning that the theory is the set of sentences that are derivable from these axioms. In such a case, we can talk about the "axioms of the theory". Axioms that define a theory are called the **nonlogical axioms**, and they come in addition to the axioms that define the logical symbols, which, correspondingly, are called the **logical axioms**.

Σ A theory is defined over a signature Σ, which is a set of nonlogical symbols (i.e., function and predicate symbols). If T is such a theory, we say it is a Σ-theory. Let T be a Σ-theory. A Σ-formula φ is **T-satisfiable** if there exists an interpretation that satisfies both φ and T. A Σ-formula φ is **T-valid**, denoted $T \models \varphi$ $T \models \varphi$, if all interpretations that satisfy T also satisfy φ. In other words, such a formula is T-valid if it can be derived from the T axioms and the logical axioms.

$\oplus$ **Definition 10.1 (theory combination).** *Given two theories T_1 and T_2 with signatures Σ_1 and Σ_2, respectively, the* theory combination $T_1 \oplus T_2$ *is a $(\Sigma_1 \cup \Sigma_2)$-theory defined by the axiom set $T_1 \cup T_2$.*

The generalization of this definition to n theories rather than two theories is straightforward.

Definition 10.2 (the theory combination problem). *Let φ be a $\Sigma_1 \cup \Sigma_2$ formula. The* theory combination *problem is to decide whether φ is $T_1 \oplus T_2$-valid. Equivalently, the problem is to decide whether the following holds:*

$$T_1 \oplus T_2 \models \varphi \,. \tag{10.4}$$

The theory combination problem is undecidable for arbitrary theories T_1 and T_2, even if T_1 and T_2 themselves are decidable. Under certain restrictions on the combined theories, however, the problem becomes decidable. We discuss these restrictions later on.

An important notion required in this chapter is that of a convex theory.

Definition 10.3 (convex theory). *A Σ-theory T is* convex *if for every conjunctive Σ-formula φ*

$$\begin{array}{l}(\varphi \implies \bigvee_{i=1}^{n} x_i = y_i) \text{ is } T\text{-valid for some finite } n > 1 \implies \\ (\varphi \implies x_i = y_i) \text{ is } T\text{-valid for some } i \in \{1, \ldots, n\}\ ,\end{array} \tag{10.5}$$

where x_i, y_i, for $i \in \{1, \ldots, n\}$, are some variables.

In other words, in a convex theory T, if a formula T-implies a disjunction of equalities, it also T-implies at least one of these equalities separately.

Example 10.4. Examples of convex and nonconvex theories include:

- Linear arithmetic over $\mathbb{R}$ is convex. A conjunction of linear arithmetic predicates defines a set of values which can be empty, a singleton, as in

$$x \leq 3 \wedge x \geq 3 \implies x = 3\ , \tag{10.6}$$

  or infinitely large, and hence it implies an infinite disjunction. In all three cases, it fits the definition of convexity.
- Linear arithmetic over $\mathbb{Z}$ is not convex. For example, while

$$x_1 = 1 \wedge x_2 = 2 \wedge 1 \leq x_3 \wedge x_3 \leq 2 \implies (x_3 = x_1 \vee x_3 = x_2) \tag{10.7}$$

  holds, neither

$$x_1 = 1 \wedge x_2 = 2 \wedge 1 \leq x_3 \wedge x_3 \leq 2 \implies x_3 = x_1 \tag{10.8}$$

  nor

$$x_1 = 1 \wedge x_2 = 2 \wedge 1 \leq x_3 \wedge x_3 \leq 2 \implies x_3 = x_2 \tag{10.9}$$

  holds.
- The conjunctive fragment of equality logic is convex. A conjunction of equalities and disequalities defines sets of variables that are equal (equality sets) and sets of variables that are different. Hence, it implies any equality between variables in the same equality set separately. Convexity follows.

Many theories used in practice are in fact nonconvex, which, as we shall soon see, makes them computationally harder to combine with other theories.

10.3 The Nelson–Oppen Combination Procedure

10.3.1 Combining Convex Theories

The Nelson–Oppen combination procedure solves the theory combination problem (see Definition 10.2) for theories that comply with several restrictions.

Definition 10.5 (Nelson–Oppen restrictions). *In order for the Nelson–Oppen procedure to be applicable, the theories $T_1, \ldots, T_n$ should comply with the following restrictions:*

1. *$T_1, \ldots, T_n$ are quantifier-free first-order theories with equality.*
2. *There is a decision procedure for each of the theories $T_1, \ldots, T_n$.*
3. *The signatures are disjoint, i.e., for all $1 \leq i < j \leq n$, $\Sigma_i \cap \Sigma_j = \emptyset$.*
4. *$T_1, \ldots, T_n$ are theories that are interpreted over an infinite domain (e.g., linear arithmetic over $\mathbb{R}$, but not the theory of finite-width bit vectors).*

There are extensions to the basic Nelson–Oppen procedure that overcome each of these restrictions, some of which are covered in the bibliographic notes at the end of this chapter.

Algorithm 10.3.1 is the Nelson–Oppen procedure for combinations of convex theories. It accepts a formula φ, which must be a conjunction of literals, as input. In general, adding disjunction to a convex theory makes it nonconvex. Extensions of convex theories with disjunctions can be supported with the extension to nonconvex theories that we present later on or, alternatively, with the methods described in Chap. 3, which are based on combining a decision procedure for the theory with a SAT solver.

The first step of Algorithm 10.3.1 relies on the idea of **purification**. Purification is a satisfiability-preserving transformation of the formula, after which each atom is from a specific theory. In this case, we say that all the atoms are **pure**. More specifically, given a formula φ, purification generates an equisatisfiable formula φ' as follows:

1. Let $\varphi' := \varphi$.
2. For each "alien" subexpression ϕ in φ':
   (a) Replace ϕ with a new auxiliary variable a_ϕ
   (b) Constrain φ' with $a_\phi = \phi$

Example 10.6. Given the formula

$$\varphi := \ x_1 \leq f(x_1) \ , \tag{10.10}$$

which mixes arithmetic and uninterpreted functions, purification results in

$$\varphi' := \ x_1 \leq a \wedge a = f(x_1) \ . \tag{10.11}$$

In φ', all atoms are pure: $x_1 \leq a$ is an arithmetic formula, and $a = f(x_1)$ belongs to the theory of equalities with uninterpreted functions. ∎

F_i

After purification, we are left with a set of pure expressions $F_1, \ldots, F_n$ such that:

1. For all i, F_i belongs to theory T_i and is a conjunction of T_i-literals.
2. Shared variables are allowed, i.e., it is possible that for some i, j, $1 \leq i < j \leq n$, $var(F_i) \cap var(F_j) \neq \emptyset$.
3. The formula φ is satisfiable in the combined theory if and only if $\bigwedge_{i=1}^{n} F_i$ is satisfiable in the combined theory.

Algorithm 10.3.1: NELSON–OPPEN-FOR-CONVEX-THEORIES

Input: A convex formula φ that mixes convex theories, with restrictions as specified in Definition 10.5

Output: "Satisfiable" if φ is satisfiable, and "Unsatisfiable" otherwise

1. *Purification:* Purify φ into $F_1, \ldots, F_n$.
2. Apply the decision procedure for T_i to F_i. If there exists i such that F_i is unsatisfiable in T_i, return "Unsatisfiable".
3. *Equality propagation:* If there exist i, j such that F_i T_i-implies an equality between variables of φ that is not T_j-implied by F_j, add this equality to F_j and go to step 2.
4. Return "Satisfiable".

Example 10.7. Consider the formula

$$\begin{array}{c}(f(x_1,0) \geq x_3) \wedge (f(x_2,0) \leq x_3) \wedge \\ (x_1 \geq x_2) \wedge (x_2 \geq x_1) \wedge \\ (x_3 - f(x_1,0) \geq 1)\ ,\end{array} \tag{10.12}$$

which mixes linear arithmetic and uninterpreted functions. Purification results in

$$\begin{array}{l}(a_1 \geq x_3) \wedge (a_2 \leq x_3) \wedge (x_1 \geq x_2) \wedge (x_2 \geq x_1) \wedge (x_3 - a_1 \geq 1) \wedge \\ (a_0 = 0) \wedge \\ (a_1 = f(x_1, a_0)) \wedge \\ (a_2 = f(x_2, a_0))\ .\end{array} \tag{10.13}$$

In fact, we applied a small optimization here, assigning both instances of the constant "0" to the same auxiliary variable a_0. Similarly, both instances of the term $f(x_1, 0)$ have been mapped to a_1 (purification, as described earlier, assigns them to separate auxiliary variables).

The top part of Table 10.1 shows the formula (10.13) divided into the two pure formulas F_1 and F_2. The first is a linear arithmetic formula, whereas the second is a formula in the theory of equalities with uninterpreted functions (EUF). Neither F_1 nor F_2 is independently contradictory, and hence we proceed to step 3. With a decision procedure for linear arithmetic over the reals, we infer $x_1 = x_2$ from F_1, and propagate this fact to the other theory (i.e., we add this equality to F_2). We can now deduce $a_1 = a_2$ in T_2, and propagate this equality to F_1. From this equality, we conclude $a_1 = x_3$ in T_1, which is a contradiction to $x_3 - a_1 \geq 1$ in T_1. ▟

Example 10.8. Consider the following formula, which mixes linear arithmetic and uninterpreted functions:

F_1 (arithmetic over $\mathbb{R}$)	F_2 (EUF)
$a_1 \geq x_3$	$a_1 = f(x_1, a_0)$
$a_2 \leq x_3$	$a_2 = f(x_2, a_0)$
$x_1 \geq x_2$	
$x_2 \geq x_1$	
$x_3 - a_1 \geq 1$	
$a_0 = 0$	
$\star\ x_1 = x_2$	$x_1 = x_2$
$a_1 = a_2$	$\star\ a_1 = a_2$
$\star\ a_1 = x_3$	
$\star$ FALSE	

Table 10.1. Progress of the Nelson–Oppen combination procedure starting from the purified formula (10.13). The equalities beneath the middle horizontal line result from step 3 of Algorithm 10.3.1. An equality is marked with a "$\star$" if it was inferred within the respective theory

$$(x_2 \geq x_1) \wedge (x_1 - x_3 \geq x_2) \wedge (x_3 \geq 0) \wedge (f(f(x_1) - f(x_2)) \neq f(x_3)) \ . \quad (10.14)$$

Purification results in

$$\begin{array}{l} (x_2 \geq x_1) \ \wedge \ (x_1 - x_3 \geq x_2) \ \wedge \ (x_3 \geq 0) \ \wedge \ (f(a_1) \neq f(x_3)) \wedge \\ (a_1 = a_2 - a_3) \wedge \\ (a_2 = f(x_1)) \wedge \\ (a_3 = f(x_2)) \ . \end{array} \quad (10.15)$$

The progress of the equality propagation step, until the detection of a contradiction, is shown in Table 10.2.

10.3.2 Combining Nonconvex Theories

Next, we consider the combination of nonconvex theories (or of convex theories together with theories that are nonconvex). First, consider the following example, which illustrates that Algorithm 10.3.1 may fail if one of the theories is not convex:

$$(1 \leq x) \ \wedge \ (x \leq 2) \ \wedge \ p(x) \wedge \neg p(1) \wedge \neg p(2) \ , \quad (10.16)$$

where $x \in \mathbb{Z}$.

Equation (10.16) mixes linear arithmetic over the integers and equalities with uninterpreted predicates. Linear arithmetic over the integers, as demonstrated in Example 10.4, is not convex. Purification results in

$$\begin{array}{l} 1 \leq x \ \wedge \ x \leq 2 \ \wedge \ p(x) \ \wedge \ \neg p(a_1) \ \wedge \ \neg p(a_2) \wedge \\ a_1 = 1 \wedge \\ a_2 = 2 \end{array} \quad (10.17)$$

F_1 (arithmetic over $\mathbb{R}$)	F_2 (EUF)
$x_2 \geq x_1$	$f(a_1) \neq f(x_3)$
$x_1 - x_3 \geq x_2$	$a_2 = f(x_1)$
$x_3 \geq 0$	$a_3 = f(x_2)$
$a_1 = a_2 - a_3$	
$\star\ x_3 = 0$	
$\star\ x_1 = x_2$	$x_1 = x_2$
$a_2 = a_3$	$\star\ a_2 = a_3$
$\star\ a_1 = 0$	
$\star\ a_1 = x_3$	$a_1 = x_3$
	FALSE

Table 10.2. Progress of the Nelson–Oppen combination procedure starting from the purified formula (10.15)

F_1 (arithmetic over $\mathbb{Z}$)	F_2 (EUF)
$1 \leq x$	$p(x)$
$x \leq 2$	$\neg p(a_1)$
$a_1 = 1$	$\neg p(a_2)$
$a_2 = 2$	

Table 10.3. The two pure formulas corresponding to (10.16) are independently satisfiable and do not imply any equalities. Hence, Algorithm 10.3.1 returns "Satisfiable"

Table 10.3 shows the partitioning of the predicates in the formula (10.17) into the two pure formulas F_1 and F_2. Note that both F_1 and F_2 are individually satisfiable, and neither implies any equalities in its respective theory. Hence, Algorithm 10.3.1 returns "Satisfiable" even though the original formula is unsatisfiable in the combined theory.

The remedy to this problem is to consider not only implied equalities, but also implied *disjunctions* of equalities. Recall that there is a finite number of variables, and hence of equalities and disjunctions of equalities, which means that computing these implications is feasible. Given such a disjunction, the problem is split into as many parts as there are disjuncts, and the procedure is called recursively. For example, in the case of the formula (10.16), F_1 implies $x = 1 \vee x = 2$. We can therefore split the problem into two, considering separately the case in which $x = 1$ and the case in which $x = 2$. Algorithm 10.3.2 merely adds one step (step 4) to Algorithm 10.3.1: the step that performs this split.

Algorithm 10.3.2: NELSON–OPPEN

Input: A formula φ that mixes theories, with restrictions as specified in Definition 10.5
Output: "Satisfiable" if φ is satisfiable, and "Unsatisfiable" otherwise

1. *Purification:* Purify φ into $\varphi' := F_1, \ldots, F_n$.
2. Apply the decision procedure for T_i to F_i. If there exists i such that F_i is unsatisfiable, return "Unsatisfiable".
3. *Equality propagation:* If there exist i, j such that F_i T_i-implies an equality between variables of φ that is not T_j-implied by F_j, add this equality to F_j and go to step 2.
4. *Splitting:* If there exists i such that
   - $F_i \implies (x_1 = y_1 \vee \cdots \vee x_k = y_k)$ and
   - $\forall j \in \{1, \ldots, k\}.\ F_i \not\Rightarrow x_j = y_j$,

   then apply NELSON–OPPEN recursively to

   $$\varphi' \wedge x_1 = y_1, \ldots, \varphi' \wedge x_k = y_k \ .$$

   If any of these subproblems is satisfiable, return "Satisfiable". Otherwise, return "Unsatisfiable".
5. Return "Satisfiable".

F_1 (arithmetic over $\mathbb{Z}$)	F_2 (EUF)
$1 \leq x$	$p(x)$
$x \leq 2$	$\neg p(a_1)$
$a_1 = 1$	$\neg p(a_2)$
$a_2 = 2$	
$\star\ x = 1 \vee x = 2$	

Table 10.4. The disjunction of equalities $x = a_1 \vee x = a_2$ is implied by F_1. Algorithm 10.3.2 splits the problem into the subproblems described in Tables 10.5 and 10.6, both of which return "Unsatisfiable"

Example 10.9. Consider the formula (10.16) again. Algorithm 10.3.2 infers $(x = 1 \vee x = 2)$ from F_1, and splits the problem into two subproblems, as illustrated in Tables 10.4–10.6. ⌟

F_1 (arithmetic over $\mathbb{Z}$)	F_2 (EUF)
$1 \leq x$ $x \leq 2$ $a_1 = 1$ $a_2 = 2$	$p(x)$ $\neg p(a_1)$ $\neg p(a_2)$
$x = 1$ $\star\ x = a_1$	 $x = a_1$ FALSE

Table 10.5. The case $x = a_1$ after the splitting of the problem in Table 10.4

F_1 (arithmetic over $\mathbb{Z}$)	F_2 (EUF)
$1 \leq x$ $x \leq 2$ $a_1 = 1$ $a_2 = 2$	$p(x)$ $\neg p(a_1)$ $\neg p(a_2)$
$x = 2$ $\star\ x = a_2$	 $x = a_2$ FALSE

Table 10.6. The case $x = a_2$ after the splitting of the problem in Table 10.4

10.3.3 Proof of Correctness of the Nelson–Oppen Procedure

We now prove the correctness of Algorithm 10.3.1 for convex theories and for conjunctions of theory literals. The generalization to Algorithm 10.3.2 is not hard. Without proof, we rely on the fact that $\bigwedge_i F_i$ is equisatisfiable with φ.

Theorem 10.10. *Algorithm 10.3.1 returns "Unsatisfiable" if and only if its input formula φ is unsatisfiable in the combined theory.*

Proof. Without loss of generality, we can restrict the proof to the combination of two theories T_1 and T_2.

($\Rightarrow$, Soundness) Assume that φ is satisfiable in the combined theory. We are going to show that this contradicts the possibility that Algorithm 10.3.2 returns "Unsatisfiable". Let α be a satisfying assignment of φ. Let A be the set of auxiliary variables added as a result of the purification step (step 1). As $\bigwedge_i F_i$ and φ are equisatisfiable in the combined theory, we can extend α to an assignment α' that includes also the variables A.

Lemma 10.11. *Let φ be satisfiable. After each loop iteration, $\bigwedge_i F_i$ is satisfiable in the combined theory.*

Proof. The proof is by induction on the number of loop iterations. Denote by F_i^j the formula F_i after iteration j.

Base case. For $j = 0$, we have $F_i^j = F_i$, and, thus, a satisfying assignment can be constructed as described above.

Induction step. Assume that the claim holds up to iteration j. We shall show the correctness of the claim for iteration $j + 1$. For any equality $x = y$ that is added in step 3, there exists an i such that $F_i^j \implies x = y$ in T_i. Since $\alpha' \models F_i^j$ in T_i by the hypothesis, clearly, $\alpha' \models x = y$ in T_i. Since for all i it holds that $\alpha' \models F_i^j$ in T_i, then for all i it holds that $\alpha' \models F_i \wedge x = y$ in T_i. Hence, in step 2, the algorithm will *not* return "Unsatisfiable". ⌟

($\Leftarrow$, Completeness) First, observe that Algorithm 10.3.1 always terminates, as there are only finitely many equalities over the variables in the formula. It is left to show that the algorithm gives the answer "Unsatisfiable". We now record a few observations about Algorithm 10.3.1. The following observation is simple to see:

F_i'

Lemma 10.12. *Let F_i' denote the formula F_i upon termination of Algorithm 10.3.1. Upon termination with the answer "Satisfiable", any equality between φ's variables that is implied by any of the F_i' is also implied by all F_j' for any j.*

We need to show that, if φ is unsatisfiable, Algorithm 10.3.1 returns "Unsatisfiable". Assume falsely that it returns "Satisfiable".

Let $E_1, \ldots, E_m$ be a set of equivalence classes of the variables in φ such that x and y are in the same class if and only if F_1' implies $x = y$ in T_1. Owing to Lemma 10.12, $x, y \in E_i$ for some i if and only if $x = y$ is T_2-implied by F_2'.

Δ

For $i \in \{1, \ldots, m\}$, let r_i be an element of E_i (a *representative* of that set). We now define a constraint Δ that forces all variables that are not implied to be equal to be different:

$$\Delta \doteq \bigwedge_{i \neq j} r_i \neq r_j \,. \tag{10.18}$$

Lemma 10.13. *Given that both T_1 and T_2 have an infinite domain and are convex, Δ is T_1-consistent with F_1' and T_2-consistent with F_2'.*

Informally, this lemma can be shown to be correct as follows: Let x and y be two variables that are not implied to be equal. Owing to convexity, they do not have to be equal to satisfy F_i'. As the domain is infinite, there are always values left in the domain that we can choose in order to make x and y different.

Using Lemma 10.13, we argue that there are satisfying assignments α_1 and α_2 for $F_1' \wedge \Delta$ and $F_2' \wedge \Delta$ in T_1 and T_2, respectively. These assignments are **maximally diverse**, i.e., any two variables that are assigned equal values by either α_1 or α_2 *must* be equal.

Given this property, it is easy to build a mapping M (an isomorphism) from domain elements to domain elements such that $\alpha_2(x)$ is mapped to $\alpha_1(x)$ for any variable x (this is not necessarily possible unless the assignments are maximally diverse).

As an example, let F_1 be $x = y$ and F_2 be $F(x) = G(y)$. The only equality implied is $x = y$, by F_1. This equality is propagated to T_2, and thus both F_1' and F_2' imply this equality. Possible variable assignments for $F_1' \wedge \Delta$ and $F_2' \wedge \Delta$ are

$$\begin{aligned} \alpha_1 &= \{x \mapsto \mathcal{D}_1, y \mapsto \mathcal{D}_1\} \ , \\ \alpha_2 &= \{x \mapsto \mathcal{D}_2, y \mapsto \mathcal{D}_2\} \ , \end{aligned} \tag{10.19}$$

where $\mathcal{D}_1$ and $\mathcal{D}_2$ are some elements from the domain. This results in an isomorphism M such that $M(\mathcal{D}_1) = \mathcal{D}_2$.

Using the mapping M, we can obtain a model α' for $F_1' \wedge F_2'$ in the combined theory by adjusting the interpretation of the symbols in F_2' appropriately. This is always possible, as T_1 and T_2 do not share any nonlogical symbols.

Continuing our example, we construct the following interpretation for the nonlogical symbols F and G:

$$F(\mathcal{D}_1) = \mathcal{D}_3 \ , \quad G(\mathcal{D}_1) = \mathcal{D}_3 \ . \tag{10.20}$$

As F_i' implies F_i in T_i, α' is also a model for $F_1 \wedge F_2$ in the combined theory, which contradicts our assumption that φ is unsatisfiable. ∎

Note that, without the restriction to infinite domains, Algorithm 10.3.1 may fail. The original description of the algorithm lacked such a restriction. The algorithm was later amended by adding the requirement that the theories are *stably infinite*, which is a generalization of the requirement in our presentation. The following example, given by Tinelli and Zarba in [276], demonstrates why this restriction is important.

Example 10.14. Let T_1 be a theory over signature $\Sigma_1 = \{f\}$, where f is a function symbol, and axioms that enforce solutions with no more than two distinct values. Let T_2 be a theory over signature $\Sigma_2 = \{g\}$, where g is a function symbol.

Recall that the combined theory $T_1 \oplus T_2$ contains the union of the axioms. Hence, the solution to any formula $\varphi \in T_1 \oplus T_2$ cannot have more than two distinct values.

Now, consider the following formula:

$$f(x_1) \neq f(x_2) \wedge g(x_1) \neq g(x_3) \wedge g(x_2) \neq g(x_3) \ . \tag{10.21}$$

This formula is unsatisfiable in $T_1 \oplus T_2$ because any assignment satisfying it must use three different values for x_1, x_2, and x_3. However, this fact is not revealed by Algorithm 10.3.2, as illustrated in Table 10.7. ∎

F_1 (a Σ_1-formula)	F_2 (a Σ_2-formula)
$f(x_1) \neq f(x_2)$	$g(x_1) \neq g(x_3)$ $g(x_2) \neq g(x_3)$

Table 10.7. No equalities are propagated by Algorithm 10.3.2 when checking the formula (10.21). This results in an error: although $F_1 \wedge F_2$ is unsatisfiable, both F_1 and F_2 are satisfiable in their respective theories

An extension to the Nelson–Oppen combination procedure for nonstably infinite theories was given in [276], although the details of the procedure are beyond the scope of this book. The main idea is to compute, for each nonstably infinite theory T_i, a lower bound N_i on the size of the domain in which satisfiable formulas in this theory must be satisfied (it is not always possible to compute this bound). Then, the algorithm propagates this information between the theories along with the equalities. When it checks for consistency of an individual theory, it does so under the restrictions on the domain defined by the other theories. F_j is declared unsatisfiable if it does not have a solution within the bound N_i for all i.

10.4 Problems

Problem 10.1 (using the Nelson–Oppen procedure). Prove that the following formula is unsatisfiable using the Nelson–Oppen procedure, where the variables are interpreted over the integers:

$$g(f(x_1 - 2)) = x_1 + 2 \wedge g(f(x_2)) = x_2 - 2 \wedge (x_2 + 1 = x_1 - 1) \; .$$

Problem 10.2 (an improvement to the Nelson–Oppen procedure). A simple improvement to Algorithm 10.3.1 is to restrict the propagation of equalities in step 3 as follows. We call a variable *local* if it appears only in a single theory. Then, if an equality $v_i = v_j$ is implied by F_i and not by F_j, we propagate it to F_j only if v_i, v_j are not local to F_i. Prove the correctness of this improvement.

Problem 10.3 (proof of correctness of Algorithm 10.3.2 for the Nelson–Oppen procedure). Prove the correctness of Algorithm 10.3.2 by generalizing the proof of Algorithm 10.3.1 given in Sect. 10.3.3.

10.5 Bibliographic Notes

The theory combination problem (Definition 10.2) was shown to be undecidable in [40], hence combination methods must impose restrictions on the

Aside: An Abstract Version of the Nelson–Oppen Procedure
Let V be the set of variables used in $F_1, \ldots, F_n$. A partition P of V induces equivalence classes, in which variables are in the same class if and only if they are in the same partition as defined by P. (Every assignment to V's variables induces such a partition.) Denote by R the equivalence relation corresponding to these classes. The **arrangement** corresponding to P is defined by

$$ar(P) \doteq \bigwedge_{v_i \, R \, v_j, i<j} v_i = v_j \wedge \bigwedge_{\neg(v_i \, R \, v_j), i<j} v_i \neq v_j \; . \tag{10.22}$$

In words, the arrangement $ar(P)$ is a conjunction of all equalities and disequalities corresponding to P, modulo reflexivity and symmetry. For example, if $V := \{x_1, x_2, x_3\}$ and $P := \{\{x_1, x_2\}, \{x_3\}\}$, then

$$ar(P) := \; x_1 = x_2 \wedge x_1 \neq x_3 \wedge x_2 \neq x_3 \; . \tag{10.23}$$

Now, consider the following abstract version of the Nelson–Oppen procedure:

1. Choose nondeterministically a partition P of V's variables.
2. If one of $F_i \wedge ar(P)$ with $i \in \{1, \ldots, n\}$ is unsatisfiable, return "Unsatisfiable". Otherwise, return "Satisfiable".

We have:

- *Termination.* The procedure terminates, since there is a finite number of partitions.
- *Soundness and completeness.* If the procedure returns "Unsatisfiable", then the input formula is unsatisfiable. Indeed, if there is a satisfying assignment to the combined theory, this assignment corresponds to some arrangement; testing this arrangement leads to a termination with the result "Satisfiable". Proving the other direction is harder, but also possible. See [275] for more details.

The nondeterministic step can be replaced with a deterministic one, by trying all such partitions possible. Hence, now it is clear that the requirement in the Nelson–Oppen procedure for sharing implied equalities can be understood as an optimization over an exhaustive search, rather than a necessity for correctness.

More generally, **abstract decision procedures** such as the one presented here are quite common in the literature. They are convenient for theoretical reasons, and can even help in designing concrete procedures in a more modular way. Abstracting some implementation details—typically by using nondeterminism—can be helpful for various reasons, such as clarity and generality, simplicity of proving an upper bound on the complexity, and simplicity of the correctness argument, as demonstrated above.

theories. There is a rich literature on combining decision procedures for first-order theories, starting with seminal papers by Nelson and Oppen [206] and by Shostak [259]. The presentation of the algorithm in this chapter is based on the former. The original presentation in [206] was not entirely correct, however, because it referred to general theories, although it is correct only for theories that are stably infinite. One year later, Oppen fixed this problem by adding this restriction, but without presenting a revised proof [215]. A full, model-theoretic proof was provided only in 1996 by Tinelli and Harandi in [275], which also serves as a basis for the (simplified) proof in Sect. 10.3.3. Several publications since then have extended the basic algorithms in order to combine theories with fewer restrictions. In Sect. 10.3.3, we mentioned Tinelli and Zarba's extension to the combination of nonstably infinite theories [276]. In [238] Ranise, Ringeissen, and Zarba identify a class of theories (called **polite theories**) that can be combined with nonstably infinite theories. Several extensions of these ideas were published by Jovanovic and Barrett [158].

Nelson and Oppen's combination procedure in its original form, as described in this chapter, can be optimized. Several optimizations have been suggested, including a method for avoiding the purification step [20]. There is empirical evidence showing that the computation of the implied equalities can become a bottleneck when one is combining, for example, linear arithmetic on the basis of the Simplex method [96].

Shostak's combination procedure [259] was considered to be an alternative to the Nelson–Oppen procedure for many years. However, Ruess and Shankar [246] showed in 2001 that it was in fact flawed in the general case (it was incomplete and not necessarily terminating), but is correct under certain restrictions. At the time several prominent theorem provers were using it. Currently (2015) only the theorem provers PVS and ALT-ERGO use some variation of Shostak's procedure, for cases where it is known to be correct. Here is N. Shankar's description of Shostak's method:

"Shostak's combination method is based on a far-reaching generalization of Gaussian elimination. He showed that many theories actually support a *canonizer* and a *solver*. A *canonizer* is an algorithm that transforms logically equivalent formulas to a syntactically identical representation. Given an equation of the form $a = b$ (where a, b are Σ-terms, Σ being the signature of the theory), a *solver* transforms it into an equivalent form $solve(a = b)$. The operation $solve(a = b)$ returns s, which is equisatisfiable to $a = b$. When the equation is unsolvable $s = \bot$, and otherwise s is a solution of the form $x_1 = e_1, \ldots, x_n = e_n$, where for $1 \leq i \leq n$, x_i is a variable in the equation $(a = b)$ and e_i is a Σ-term.

A canonizer σ for a theory can be used to decide that the equality $c = d$ is valid by applying the σ to c and d, respectively, to see if the canonical forms $\sigma(c)$ and $\sigma(d)$ are identical. A theory with such a solver and canonizer is called a *Shostak theory*.

Shostak's method uses the combination of a solver and canonizer to verify $a_1 = b_1, ..., a_n = b_n \vdash c = d$ by successively placing the antecedent equations

into a solution set S, with $S_0 = \emptyset$ (the empty substitution), $S_i = solve(S(a_i = b_i))$ for $1 \leq i \leq n$, and $S = S_n$. The claim can then be checked by verifying that either some S_i is $\perp$ for $1 \leq i \leq n$, or $\sigma(S(c))$ and $\sigma(S(d))$ are identical canonical forms.

Though Shostak's ideas are deep, his original algorithm and proof had a number of flaws. He incorrectly claimed that solvers and canonizers for disjoint theories could be combined into a solver and canonizer for the union of these theories. The *basic* combination of a single Shostak theory with equality over uninterpreted functions was presented and proved correct in [246]. Ganzinger showed in [119] that a basic combination could be constructed solely with the solver and without needing a canonizer—the use of a canonizer can be seen as an optimization. Shankar and Ruess show in [257] a method for extending the basic combination to disjoint unions of Shostak theories without requiring the combination of solvers and canonizers."

The lazy approach, as described in Chap. 3, opens up new opportunities with regard to implementing the Nelson–Oppen combination procedure. A contribution by Bozzano et al. [41] suggests a technique called **delayed theory combination**. Each pair of shared variables is encoded with a new Boolean variable (resulting in a quadratic increase in the number of variables). After all the other encoding variables have been assigned, the SAT solver begins to assign values (arbitrary at first) to the new variables, and continues as usual, i.e., after every such assignment, the current partial assignment is sent to a theory solver. If any one of the theory solvers "objects" to the arrangement implied by this assignment (i.e., it finds a conflict with the current assignment to the other literals), this leads to a conflict and backtracking. Otherwise, the formula is declared satisfiable. This way, each theory can be solved separately, without passing information about equalities. Empirically, this method is very effective, both because the individual theory solvers need not worry about propagating equalities, and because only a small amount of information has to be shared between the theory solvers in practice—far less, on average, than is passed during the normal execution of the Nelson–Oppen procedure. In [159], Jovanovic and Barrett showed how many pairs of shared variables can be ignored, when they have no effect on the individual theories.

A different approach has been proposed by de Moura and Bjørner [91]. These authors also make the equalities part of the model, but instead of letting the SAT solver decide on their values, they attempt to compute a consistent assignment to the theory variables that is as diverse as possible. The equalities are then decided upon by following the assignment to the theory variables.

10.6 Glossary

The following symbols were used in this chapter:

Symbol	Refers to ...	First used on page ...
Σ	The signature of a theory, i.e., its set of nonlogical predicates and function symbols and their respective arities (i.e., those symbols that are *not* common to all first-order theories)	230
$T \models \varphi$	φ is T-valid	230
$T_1 \oplus T_2$	Denotes the theory obtained from combining the theories T_1 and T_2, i.e., a theory over $\Sigma_1 \cup \Sigma_2$ defined by the set of axioms $T_1 \cup T_2$	230
F_i	The pure (theory-specific) formulas in Algorithm 10.3.1	232
F_i'	The formula F_i upon termination of Algorithm 10.3.1	238
Δ	A constraint that forces all variables that are not implied to be equal to be different	238