UNIVERSITÀ DEGLI STUDI DI MILANO
FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI

Dipartimento di Matematica

DOTTORATO DI RICERCA IN MATEMATICA
Settore scientifico-disciplinare MAT/01

XIX CICLO

COMBINED DECISION PROCEDURES FOR CONSTRAINT SATISFIABILITY

Advisor: Prof. Silvio Ghilardi
PhD Coordinator: Prof. Antonio Lanteri

PhD Student:
Enrica Nicolini

ANNO ACCADEMICO 2005/2006

Contents

Introduction

This thesis is devoted to the study of the decidability of fragments of different logical languages and to the transfer of the decidability to their combination. The interest in such topics relies on the deep impact of such results to the applications: it is noteworthy that, in this context, attention is paid more to the development of fast and efficient, even if *ad hoc*, algorithms for the decidability than of logical calculi.

This fact has led to the use of a huge variety of heterogeneous methods for deciding fragments, hence the need of re-using such methods in a modular way to solve more complex problems arises. The natural answer to these needs is a combination schema that should be as general and flexible as possible, and, equipped with suitable hypothesis, should be capable to transfer the decidability from fragments to their combinations.

Overview

Chapter 1 presents some examples to the aim of justifying the need of re-using and integrating already existing procedures, and is divided into two parts. Section 1.2 presents two well-known decidability results: one, due to Presburger, concerns a fragment of the theory of arithmetic, the other concerns the satisfiability problem in the description language $\mathcal{ALC}$. The latter part of the chapter (Section 1.3) describes a result of ours about the decidability of the universal fragment of the theory of arrays with dimension.

Our contribution in the remaining part of the thesis regards the combination of decision procedures, focusing our attention on the decidability of the universal fragment. Deciding the universal fragment of a given first-order theory T on a signature Σ is equivalent to deciding the *constraint satisfiability problem* for T. More in detail, the constraint satisfiability problem for T is the problem of deciding whether the conjunction of a finite set of Σ-literals is satisfiable in a model of T. The combination problem arises when we consider two first-order theories T_1 and T_2 over the signatures Σ_1 and Σ_2 respectively (notice that it may happen that the signatures Σ_1 and Σ_2 are non-disjoint). If we are able to decide the constraint satisfiability problem in both T_1 and T_2, we wonder whether it is possible to solve the same problem in $T_1 \cup T_2$. One of the simplest methodologies

for the combination of decision procedures is represented by the *Nelson-Oppen procedure* (see [69]), which was originally designed only for the disjoint signatures case and which is guaranteed to be terminating and complete under the following assumptions: (i) Σ_1 and Σ_2 are disjoint; (ii) the theories T_1 and T_2 are stably infinite, i.e. T_i is such that any quantifier-free Σ-formula φ which is satisfiable in a model of T_i is satisfiable in a model of T_i whose domain is infinite.

In **Chapter 2** assumption (ii) is analyzed. If we drop this hypothesis, we prove that it is possible to incur in undecidability. In fact, in Section 2.2.1 we show an example of a theory whose constraint satisfiability problem is decidable, but it is not decidable if a constraint is satisfiable in an infinite model (obviously this is not the case of stably infinite theories). Relying on such a result, we prove that there exist theories over finite and disjoint signatures, having decidable constraint satisfiability problem, and such that their union has undecidable constraint satisfiability problem. On the other hand, if we are able to decide whether a constraint is satisfiable in an infinite model, we are close to get a decidability result. Indeed, if we consider *strongly* $\exists_\infty$*-decidable theories* over disjoint signatures (i.e., theories satisfying the following condition: (i) the constraint satisfiability problem is decidable; (ii) it is decidable if a constraint is satisfiable in an infinite model; (iii) it is decidable whether a finite structure is a model of the theory), we can transfer the decidability to the union of the theories.

On the other side, in **Chapter 3** we consider the assumption (i) about the disjointness of the signatures. A first attempt to drop it can be found in [44], where the decidability of the constraint satisfiability problem for $T_1 \cup T_2$ is obtained under the following assumptions: (a) the constraint satisfiability problems for T_1 and T_2 are decidable; (b) the theory T_1 and T_2 are both compatible with respect to a Σ_0-theory T_0, where Σ_0 is the shared signature $\Sigma_1 \cap \Sigma_2$; (c) the theory T_0 is effectively locally finite, i.e. the signature Σ_0 is finite and, given a finite set of constants $\underline{a}$, there exists only a finite (and computable) set of ground $\Sigma_0^{\underline{a}}$-terms which are "representative" modulo T_0. After introducing a suitable notions of *noetherian theory* and T_i*-basis enumerator*, our main contribution lies in proving that the assumption (c) can be weakened by requiring that the common subtheory T_0 is noetherian and there exist T_i-basis enumerators. This result is a proper extension of the result presented in [44] as it offers, for example, the opportunity of guaranteeing the decidability of the constraint satisfiability problem for the combination of theories coming from the field of computer algebra.

In **Chapter 4** it is shown how to push Nelson-Oppen methodology even further, in order to solve in a uniform way as many problems as possible. In fact, when it is used in conjunction with model-theoretic results, it succeeds in dealing with various classes of

combination problems, often quite far from the originally intended application domain. In this respect, the Nelson-Oppen schema has been recast into an higher-order framework by adopting type-theoretic signatures in Church's style.

The interest of this approach relies on the existence of tractable fragments of general type theory whose "combination" often turns out to be tractable. In order to plug the Nelson-Oppen procedure into a higher-order context, a suitable notion of algebraic fragment is needed (Section 4.2). In this framework, a constraint satisfiability problem for the algebraic fragment Φ is reformulated as the problem of deciding whether a Φ-constraint is satisfiable in some structure $\mathcal{M}$ belonging to the class of structures where Φ is interpreted. Our definition of a fragment is sufficient to naturally reproduce Nelson-Oppen steps, but we still have to face termination and completeness issues: for these, additional hypotheses are needed.

In order to ensure termination, we modify the noetherianity notion previously introduced by adapting it to the new context. To guarantee the completeness of the combination procedure in our context, we rely on powerful model-theoretic and semantically driven tools, the so-called *structural operations* which, in order to be useful, are required to admit an *isomorphism theorem*. An example of structural operations admitting an isomorphism theorem (for fragments consisting of all first-order formulae interpreted in an elementary class) are ultrapowers: *Keisler-Shelah isomorphism theorem* (see [25]) proves that two $\mathcal{L}(A)$-models $\mathcal{M}$ and $\mathcal{N}$ are elementarily equivalent if and only if there is an ultrafilter $\mathcal{U}$ such that the ultrapowers $\prod_{\mathcal{U}} \mathcal{M}$ and $\prod_{\mathcal{U}} \mathcal{N}$ are $\mathcal{L}(A)$-isomorphic. Another example of isomorphism theorem, operating on certain monadic first-order fragments, is given by disjoint unions (this is the isomorphism theorem useful to get fusion decidability transfer results in modal logic).

Equipped with such tools, we are able to prove a general decidability transfer result that covers as special cases, besides new applications, the aforementioned extension of Nelson-Oppen procedure to non-disjoint signatures, the fusion transfer for decidability of global consequence relation in modal logic, and the fusion transfer of decidability of A-Boxes with respect to T-Boxes axioms in local abstract description systems.

Finally, **Chapter 5** contains a non trivial result of combination which fully exploits the framework developed in the previous chapter. After introducing a proper definition of monadically suitable fragment, the key theorem of the chapter states the transfer decidability result for the monodic fusion of the one variable modal fragment and the monadically suitable one. As a particular case, this theorem allows to reduce decidability of modal and temporal monodic fragments to their extensional (i.e. non modal) and one-variable components, thus recovering and properly extending the results in [94].

Motivations

With the birth of the modern symbolic logic, the idea that all the mathematical theorems can be derived in a somehow "mechanizable" and "automatizable" way arose. In the thirties the results by Gödel strongly limited the validity of that idea, showing that it is not always possible to state, given a sentence in the language of a certain theory, whether or not it is true in all the models of the theory. Notwithstanding Gödel's results, the idea of mechanizing part of the human reasoning has been developed.

More recently, the interplay between logic and computer science has been so rich, active and dynamic to gradually get recognized by itself as a field of research, the so-called field of *computational logic*, which essentially consists of all uses of logic in computer science. Indeed, logic continues to play an important role in computer science and has permeated several of its areas, including artificial intelligence, computational complexity, program logics, programming languages and program verification.

In this context the area of *automated deduction* offers not only a novel use of logic in computer science, but gives also an opportunity for the study and development of new logical instruments, which can be advantageously used to cope with problems coming from concrete applications. This field is young, as suggested by the word "automated", but its basis is ancient as suggested by the word "deduction". Some trace the origins of automated deduction to the Cornell summer meeting in 1957 that brought together a large number of logicians and computer scientists, and where Abraham Robinson outlined a proof procedure for the first-order predicate calculus based on searching for models in the Herbrand universe. Others say that it began before that, with the 1955 Logic Theorist program of Newell, Shaw and Simon, or with Martin Davis 1954 implementation of Presburger's decision procedure. Nowadays, two of the most fruitful application fields of automated deduction are *software verification* and *knowledge representation*.

Software Verification

Software verification is a broad discipline of software engineering whose goal is to assure that a software fully satisfies certain expected requirements. Usually, this requirements are focused on guaranteeing the absence of incorrect behaviors (i.e., "bugs"). Catching bugs in programs is difficult and time-consuming, however the efforts are justified because the use of faulty software has a huge negative impact on the balance of the companies. Indeed, according to [74], costs of inadequate software testing infrastructure on the economy of the United States are estimated in 59 billions US dollars, whereas the potential cost reduction from feasible infrastructure improvements is quantified around 22 billions US dollars. These estimates do not reflect "costs" associated with mission critical software, where failure can lead to extremely high costs such as loss of life or catastrophic failure.

Examples of that kind are the breakdown of a SCUD missile which killed 28 people at Dhahran in 1991, the explosion of the ESA rocket Ariane 5 in 1996, and even the well-known "pentium-bug" that forced Intel to replace all flawed Pentium microprocessor in 1994.

The effort of debugging and proving correct even small units of code can surpass the effort of programming. Bugs inserted while "programming in the small" can have dramatic consequences for the consistency of a whole software system as shown, e.g., by viruses which can spread by exploiting buffer overflows, a bug which typically arises while coding a small portion of code. To detect this kind of errors, many verification techniques have been put forward. There are two main approaches to verification, the so-called *dynamic* and *static verification*. The former, also called *testing*, is the most widespread technique and consists in the process of executing a program or an application with the aim of finding errors. There are many approaches to software testing, depending on the availability of the source code (white- and black-box approaches), on the stage of the development of the product (alpha, beta and gamma testing), on the kind of product to be tested, on the required degree of automation, and so on. However, effective testing of complex products is essentially a process of investigation. Testing could mean "the process of questioning a software in order to evaluate it", where the "questions" are things the tester tries to do with the software, and the product answers with its behavior in reaction to the probing of the tester. With that in mind, it is clear that testing can never completely establish the correctness of an arbitrary computer software.

Here in the following, we shall be mainly interested in static verification (also called *static analysis* or *formal verification*), which differs from testing because it aims to guarantee that a system does not have a certain defect or does have a certain property. Formal verification can be used, for example, for systems such as cryptographic protocols, combinatorial circuits, digital circuits with internal memory, and software expressed as source code. The verification of these systems is done by providing a formal proof on an abstract formal model of the system, the correspondence between the formal model and the nature of the system being otherwise known by construction.

We can distinguish two main approaches to formal verification. The first approach is called *model checking* (see, e.g., [26]): roughly speaking, it consists of a systematically exhaustive exploration of the mathematical model. This technique is commonly used to verify finite-state systems, but it sometimes be applied also to the infinite-state case.

The second approach consists of using a formal version of logical reasoning about the system, usually using software such as the PVS, COQ, HOL, Isabelle, and Nqthm theorem provers. This process is generally only partially automated and is driven by the user's understanding of the system to validate, but relies on fully automated procedures (from now on called *decision procedures*) for some subproblems. Indeed, in verification with

proof assistants, decision procedures are typically used for eliminating trivial subgoals represented, for instance, as sequents modulo a background theory which usually axiomatize standard data-types commonly used in programs such as arrays, lists, bit-vectors and so on.

Formal verification has been used to automatically detect common bugs such as out-of-range array subscripts and variables used before initialization. More recently, in the static analysis community, there seems to be a growing demand for a more declarative approach. A declarative approach seems mandatory to enable the programmer to express the properties to be checked so that the results of the analysis can be confronted against his expectations. In this direction, some tools (e.g. [50, 38]) based on (extensions of) first-order logic have been developed. These tools take a program with some annotations written in (an extension of) first-order logic and produce a set of formulae of (a fragment of) first-order logic whose satisfiability implies that a bug is present in the code. In order to check for satisfiability, a procedure capable of handling the generated proof obligations must be available.

Discharging the proof obligations arising in software verification and eliminating subgoals in verification with proof assistants reduce to the problem of proving the unsatisfiability of a quantifier-free sentence with a complex Boolean structure modulo a background theory T. This is the main reason to study the constraint satisfiability problem and to develop decidability results for *fragments* of first-order theories. Moreover, since problems deriving from software verification involve heterogeneous domains which are axiomatized by different theories, modularity in combining and re-using algorithms and concrete implementation of already developed decision procedures becomes crucial. In any involved area, the combination and integration of existing decision procedures are non trivial tasks mainly because of the heterogeneity of the techniques used by the component decision procedures. If we consider the theories which are suitable for software verification, decision procedures are obtained in many different ways: sometimes (e.g., when T is the empty theory, the theory of lists or the theory of arrays) the superposition calculus decides constraint satisfiability (see [5]), but in many other cases (for example whenever arithmetic constraints are involved) ad hoc procedures are needed. In this context the problem of combining decision procedures, which is the main topic considered in this thesis, naturally arises.

A Toy Example In order to give an idea of the properties that are usually involved in practical applications, we present a prototypical example of the properties one would check in software verification.

Consider the following two program fragments written in C language:

```
for (k=1; k<=n; k++)              for (k=1; k<=n; k++)
   a[i+k] = a[i]+k;                  a[i+n-k] = a[i+n]-k;
```

Notice that, if the execution of either fragment produces the same result in the array `a`, then `a[i+n]==a[i]+n` must hold initially for any value of `i` and `n`. Fixed an integer n, it is possible to automatically prove the above property. First of all, we need to give a representation of the property using the formal language of first-order logic. We rely on the following two theories T_1 and T_2, which will be presented more in detail in Sections 1.2.1 and 1.3:

(T_1) The *theory of arrays* is used to express the formal properties of this common data-type. Its signature consists of two function symbols (select and store) used to represent the action of recovering an element from an array and of writing an element into an array;

(T_2) A fragment of the *theory of Presburger Arithmetic* is used to express the formal properties of the additive group of the integers we are interested in.

For $n = 2$, we can represent the property with the following formula:

$$\begin{aligned}\text{store}(\text{store}(a, i+1, \text{select}(a,i)+1), i+2, \text{select}(a,i)+2) = \\ \text{store}(\text{store}(a, i+1, \text{select}(a,i+2)-1), i, \text{select}(a,i+2)-2) \\ \rightarrow \text{select}(a,i+2) = \text{select}(a,i)+2\end{aligned}$$

where i is an implicitly universally quantified variable.

In general, for $n \geq 0$, the required formula is more complicated and has the following form:

$$L_n^n = R_n^n \rightarrow \text{select}(a, i+n) = \text{select}(a,i) + n \tag{1}$$

where

$$\begin{aligned} L_0^n = R_0^n &= a \\ L_k^n &= \text{store}(L_{k-1}^n, i+k, \text{select}(a,i)+k) \qquad \text{for } k = 1, \ldots, n \\ R_k^n &= \text{store}(R_{k-1}^n, i+n-k, \text{select}(a,i+n)-k) \qquad \text{for } k = 1, \ldots, n \end{aligned}$$

Both of the involved theories admit decision procedures able to check the satisfiability of a constraint, i.e. a conjunction of ground literals. That is the reason why we proceed by refutation: indeed, checking the validity of (1) is equivalent to checking the satisfiability of its negation which is, once skolemized, a conjunction Γ of ground literals. Discussing this example in full details is beyond the aim of this introductory section; the interested reader can recover it in [4], where the decision procedure is completely described.

The presented example is a typical *combination problem*, since we have at our disposal the decision procedures for the two theories, but we are interested in deciding the satisfiability in the union of the theories. The first step is to purify the constraint Γ into $\Gamma_1 \cup \Gamma_2$ in such a way that Γ_1 is over the signature of T_1 and Γ_2 is over the signature of T_2 (this can be trivially done by adding new equations $c = t$, for new constants c and alien subterms t). However, some extra work is needed in order to guarantee that the satisfiability of Γ_i w.r.t. T_i implies the satisfiability of Γ w.r.t. $T_1 \cup T_2$.

The key ingredient in such cases is the *Nelson-Oppen method* (see [69, 73, 86]), which was originally designed in order to combine decision procedures for the *universal fragment* of *first-order theories*. The basic feature of the Nelson-Oppen method is quite simple: constraints involving mixed signatures are purified into two equisatisfiable pure constraints and then the specialized reasoners try to share all the information they can acquire concerning constraints in the common subsignature, till an inconsistency is detected or a saturation state is reached. We can illustrate the procedure by the following simple example.

A Toy Example for Combination Suppose that theory T_1 is Presburger arithmetic and that theory T_2 is the theory of two uninterpreted function symbols f, g (f is unary and g is binary). We want to check unsatisfiability modulo $T_1 \cup T_2$ of the constraint

$$\Gamma \ \equiv \ \{x + f(y) = x, \ g(f(y) + z, z) \neq g(z, z)\}.$$

We use two decision procedures for satisfiability of literals modulo T_1 and T_2 as black boxes. In the *first step*, Nelson-Oppen method repeatedly abstracts out alien subterms with fresh variables, till an equisatisfiable finite set of literals $\Gamma_1 \cup \Gamma_2$ is produced, where Γ_1 contains only literals in the signature of T_1 and Γ_2 contains only literals in the signature of T_2. In practice, subterms t are replaced by fresh variables x and new equations $x = t$ are added to the current constraint, till the the desired purified status is reached: in the present example, we get

$$\Gamma_1 \ \equiv \ \{x + w = x, \ u = w + z\}, \qquad \Gamma_2 \ = \ \{w = f(y), \ g(u, z) \neq g(z, z)\}.$$

In the *second step*, information exchange concerning the common subsignature is performed. In our case, for instance, the decision procedure for T_1 realizes that $u = z$ is a logical consequence of $T_1 \cup \Gamma_1$; as soon as the decision procedure for T_2 knows this fact, it reports the inconsistency. Notice that this example is very simple: in general, the exchange of entailed atoms from the common subsignature is not sufficient, one needs to exchange entailed *disjunctions* of atoms (such an exchange of disjunctions of atoms may be implemented for instance by case-split and backtracking).

There are two main problems that must be adequately addressed in this Nelson-Oppen approach, namely termination and completeness of the proposed combined procedure. In fact, Nelson-Oppen method was guaranteed to be complete only for disjoint signatures and stably infinite theories. One of the task of this thesis (Chapters 2, 3 and 4) will be the weakening of the hypothesis under which the Nelson-Oppen combination schema is guaranteed to be terminating and complete.

Knowledge Representation

In this section we provide a very brief introduction to Description Logics, a formal language for representing knowledge and reasoning about it. More detailed information can be found in [7], that is the main source of this paragraph. Since we will present a specific formalism (though expressive enough to cope with the questions arising from the knowledge representation area), this section will be more technical than the previous one; on the other hand, a comparison with the examples sketched in the previous section will give an idea of how different can be the logic formalisms (and, consequently, also the techniques) developed in order to treat problems coming from concrete applications. These are essentially the motivations for looking for schemata for combination of decision procedures as general and flexible as possible.

Description Logics is the most recent name for a family of knowledge representation formalisms that represent the knowledge of an application domain (the "world") by first defining the relevant concepts of the domain (its terminology), and then using these concepts to specify properties of objects and individuals occurring in the domain (the world description). From the beginning Description Logics have been considered general purpose languages for knowledge representation and reasoning, and therefore suited for many applications. In particular, they were considered especially effective for those domains where the knowledge could be easily organized along a hierarchical structure.

The ability to represent and reason about taxonomies in Description Logics has motivated their use as a modeling language in the design and maintenance of large, hierarchically structured bodies of knowledge as well as their adoption as the representation language for formal ontologies (see [91]). Other application domains are natural language processing, because Description Logics can be viewed as a basic representation language; database management, because the Description Logics formalism can be used in a variety of ways in concert with the main technology of the area; software engineering, where Description Logics are used in systems that would support the software developer by helping him or her in finding out information about a large software system; configuration, which includes applications that support the design of complex systems created by combining multiple components; medicine, where one focus has been on the construc-

tion and maintenance of very large ontologies of medical knowledge; and digital libraries and Web-based information systems, where the Description Logic are used to represent bibliographic information and to support classification and retrieval in digital libraries.

A distinguished feature of Description Logics is the emphasis on reasoning, that allows one to infer implicitly represented knowledge from the knowledge that is explicitly contained in the knowledge base by inference patterns that above all exploit classification of concepts and individuals. Classification of concepts determines subconcept/superconcept relationships (called *subsumption relationships*) between the concepts of a given terminology, and thus allows one to structure the terminology in the form of a subsumption hierarchy. This hierarchy provides useful information on the connection between different concepts, and it can be used to speed-up other inference services. Classification of individuals (or objects) determines whether a given individual is always an instance of a certain concept (i.e., whether this instance relationship is implied by the description of the individual and the definition of the concept). Thus, it provides useful information on the properties of an individual. Moreover, instance relationships may trigger the application of rules that insert additional facts into the knowledge base.

The following three ideas, first put forward in [22, 23], have largely shaped the subsequent development of Description Logics:

- The basic syntactic building blocks are atomic concepts (unary predicates), atomic roles (binary predicates), and individuals (constants).
- The expressive power of the language is restricted in that it uses a rather small set of (epistemologically adequate) constructors for building complex concepts and roles.
- Implicit knowledge about concepts and individuals can be inferred automatically with the help of inference procedures. In particular, subsumption relationships between concepts and instance relationships between individuals and concepts play an important role: subsumption relationships and instance relationships are inferred from the definition of the concepts and the properties of the individuals.

By way of a prototypical example, a Description Logic formalism first introduces the formalism for describing concepts (i.e., the description language), and then defines the terminological (T-Box) and the assertional (A-Box) formalisms. A knowledge base comprises two components, the T-Box and the A-Box. The T-Box introduces the terminology, i.e., the vocabulary of an application domain, while the A-Box contains assertions about named individuals in terms of this vocabulary. The vocabulary consists of concepts, which denote sets of individuals, and roles, which denote binary relationships between individuals. In addition to atomic concepts and roles (concept and role names), all Description

Logic systems allow their users to build complex descriptions of concepts and roles.

A Description Logic system not only stores terminologies and assertions, but also offers services that reason about them. Typical reasoning tasks for a terminology are to determine whether a description is satisfiable (i.e. consistent), or whether one description is more general than another one, that is, whether the first subsumes the second. Important problems for an A-Box are to find out whether its set of assertions is consistent, that is, whether it has a model, and whether the assertions in the A-Box entail that a particular individual is an instance of a given concept description. Satisfiability checks of descriptions and consistency checks of sets of assertions are useful to determine whether a knowledge base is meaningful at all. With subsumption tests, one can organize the concepts of a terminology into a hierarchy according to their generality. A concept description can also be conceived as a query, describing a set of objects one is interested in. Thus, with instance tests, one can retrieve the individuals that satisfy the query. Moreover, also for the Description Logic systems combination problems arise.

A Toy Example To introduce a description language (see [7] for more information), we need a set of atomic concepts $\mathrm{x}, \mathrm{y}, \ldots$, a set of role names $R, S, \ldots$ and a set of individual names $a, b, \ldots$; *concepts* are built up from atomic concepts, Boolean operators $\bot, \top, \sqcap, \sqcup, \neg$, and relativized existential quantification $\exists R$ (here R is a role name). Concepts only notationally differ from propositional multimodal formulae (in modal logic the notation for $\exists R$ is the 'possibility' operator $\Diamond_R$), however description logics are richer because they allow to write also assertions. We have three kinds of assertions, namely *concepts assertions* $C(a)$ (here a is an individual name and C is a concept), *role assertions* $R(a, b)$ (here a, b are individual names and R is a role name) and *concept equalities* $C = D$ (here C, D are concepts). A finite set of concept assertions or of role assertions is called an A-Box, whereas a finite set of concept equalities is called a T-Box;[1] a pair given by a T-Box and an A-Box is said to be a *knowledge basis*. The semantic for a description language is rather intuitive: an *interpretation* is a pair $\mathcal{I} = (W^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $W^{\mathcal{I}}$ is a non-empty set (the *domain*) and $\cdot^{\mathcal{I}}$ is the *interpretation function*, assigning to each atomic concept x a subset $\mathrm{x}^{\mathcal{I}} \subseteq W^{\mathcal{I}}$, to each role name R a binary relation $R^{\mathcal{I}} \subseteq W^{\mathcal{I}} \times W^{\mathcal{I}}$, and to every individual name a an element $a^{\mathcal{I}} \in W^{\mathcal{I}}$. The interpretation function is inductively extended to concepts by interpreting the Boolean operators as intersection, union and complement and by interpreting relativized existential quantification as

$$(\exists R.C)^{\mathcal{I}} \ := \ \{w \in W^{\mathcal{I}} \mid \exists v \ (w, v) \in R^{\mathcal{I}} \wedge v \in C^{\mathcal{I}}\}.$$

[1] Sometimes, in the literature, concept inclusions are used instead of concept equalities; the difference is immaterial, as far as concepts are closed under intersection.

An interpretation $\mathcal{I}$ satisfies a concept assertion $C(a)$ iff $a^{\mathcal{I}} \in C^{\mathcal{I}}$, it satisfies a role assertion $R(a,b)$ iff $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$ and it satisfies a concept equality $C = D$ iff $C^{\mathcal{I}} = D^{\mathcal{I}}$; $\mathcal{I}$ is a *model* of a knowledge basis iff it satisfies all assertions from it.

As soon as concepts are closed under all Boolean operators, Description Logics reasoning tasks all reduce to satisfiability problems; satisfiability problems can however be formulated at different levels and it is important to keep such distinctions, for instance, for complexity reasons. For our purposes, it is useful to distinguish three different satisfiability problems: (i) the *global* satisfiability problem is the problem of deciding whether there exists a model $\mathcal{I}$ of a given T-Box in which we have $C^{\mathcal{I}} \neq \emptyset$ for a preassigned concept C; (ii) the *local* satisfiability problem is the problem of deciding whether there exists a model of a given A-Box; (iii) the *full* satisfiability problem is the problem of deciding whether there exists a model of a knowledge basis. Problem (i) is a notational variant of the relativized satisfiability problem in modal logic, whereas problem (ii) is the same as the local satisfiability problem in modal logic, if we restrict to A-Boxes consisting on a single concept assertion.

Suppose our description language contains roles $R, R^*, S, S^\smile$ and that we restrict to interpretations in which R^* is the reflexive-transitive closure of R and $S^\smile$ is the converse relation of S; suppose also that we know how to process satisfiability problems involving reflexive-transitive closures and converse relations separately, but we do not know how to process a problem involving both of them (or that we do not want to re-implement from scratch a device for solving such combined problems). Consider the following global satisfiability problem

$$\mathrm{y} = \exists R.\exists R.\mathrm{x}, \quad (\forall R^*.\neg \mathrm{x}) \sqcap (\exists S.\forall S^\smile.\mathrm{y}) \neq \bot$$

(where $\forall$ is defined as $\neg\exists\neg$). To solve the problem, we first purify it as

$$\begin{aligned} &\mathrm{y} = \exists R.\exists R.\mathrm{x}, \qquad (\forall R^*.\neg \mathrm{x}) \sqcap \mathrm{z} \neq \bot \\ &\mathrm{z} = \exists S.\forall S^\smile.\mathrm{y} \end{aligned}$$

and then we begin constraint propagation. In fact, the 'converse relation' procedure discovers the entailed equality $\mathrm{z} \sqcap \mathrm{y} = \mathrm{z}$ and, using this information, the 'reflexive-transitive closure' procedure can report the unsatisfiability of the problem.

There is a parallelism between the procedures described in the last example of Section *Software Verification* on page (vi) and this one: in both cases, abstraction equations are added to the current problem and propagation of shared information is used to detect inconsistency. The development of a general framework for combination which is able to

cope with problems arising from disparate areas and involving different formalisms is one of the main aims of this thesis.

Chapter 1

Deciding First-Order Fragments

Symbolic logic offers a great variety of formalisms able to treat decision problems (modulo a particular theory) for relevant fragments of a logical language, or to treat satisfiability problems for various kinds of constraints. We can briefly mention some of the many different kind of techniques commonly used: congruence closure, rewrite-based methods, elimination of quantifiers, automata techniques, filtrations, and mosaics.

In this chapter we will describe three examples of decision procedures for fragments, illustrating thus how different could be the techniques which lead to decidability results.

1.1 Preliminaries

Let us start fixing some formal preliminaries. A *signature* Σ is a (at most countable) set of sort symbols together with a set of functions and predicate symbols (both equipped with suitable lists of sort symbols as arity). We assume the binary equality predicate symbol '$=_S$' to be always present in any signature Σ for every sort S and we usually omit the subscript S in $=_S$. The signature obtained from Σ by the addition of a set $\mathcal{K}$ of new constants (that is, 0-ary function symbols, each of them again equipped with a sort) is denoted by $\Sigma \cup \mathcal{K}$ or by $\Sigma^{\mathcal{K}}$; when the set of constants is finite, we use letters $\underline{a}, \underline{b}, \underline{c}$, etc. in place of $\mathcal{K}$. A signature Σ' is a *subsignature* of signature Σ (in symbols, $\Sigma' \subseteq \Sigma$) if Σ' is a signature that can be obtained from Σ by removing some of its sorts, function, and predicate symbols. First-order terms and formulae over a signature Σ are defined in the usual way, i.e. they must respect the arities of function and predicate symbols, and the variables occurring in them must also be equipped with sorts (well-sortedness). The notions of Σ-*atom*, *-literal*, *-clause*, *-positive clause*, etc. are the usual ones: e.g., an atom is is a predicate symbol applied to (well-sorted) terms, a literal is an atom or the negation of an atom, a clause is a multiset of literals, a positive clause is a multiset of atoms, etc. As usual, writing $A(x_1, \ldots, x_n)$ means that the variables occurring free in the formula A

are a subset of $\{x_1, \ldots, x_n\}$.

Terms, literals, clauses and formulae are called *ground* whenever variables do not appear. Formulae without free variables are called *sentences*. The universal (resp. existential) closure of a formula φ is the sentence obtained from φ by adding a prefix of universal (resp. existential) quantifiers binding all variables occurring free in φ. A Σ-*theory* T is a set of sentences (called the axioms of T) in the signature Σ. If T is finite, the theory is said to be finitely axiomatized. A *universal* theory is a theory whose axioms are universal closures of quantifier-free sentences.

From the semantic side, we have the standard notion of a Σ-*structure* $\mathcal{A}$: this consists of non-empty and pairwise disjoint domains $S^{\mathcal{A}}$ for every sort S, and interprets each function symbol f and predicate symbol P as functions $f^{\mathcal{A}}$ and relations $P^{\mathcal{A}}$, respectively, according to their arities. If t is a ground term, we also use $t^{\mathcal{A}}$ for the element denoted by t in the structure $\mathcal{A}$. We write A to denote the disjoint union of all domains $S^{\mathcal{A}}$ of a structure $\mathcal{A}$. Let Σ and Σ' be two signatures such that $\Sigma' \subseteq \Sigma$ and let $\mathcal{M}$ be a Σ-structure; the Σ'-*reduct* of $\mathcal{M}$ is the Σ'-structure $\mathcal{M}_{|_{\Sigma'}}$ obtained from $\mathcal{M}$ by forgetting the interpretations of sort, function and predicate symbols of Σ not belonging to Σ'. A Σ-*embedding* between two Σ-structures $\mathcal{M}$ and $\mathcal{N}$ is a mapping $\mu : M \to N$ that satisfies the condition

$$\mathcal{M} \models A(a_1, \ldots, a_n) \quad \text{iff} \quad \mathcal{N} \models A(\mu(a_1), \ldots, \mu(a_n)) \tag{1.1}$$

for all Σ-atoms $A(x_1, \ldots, x_n)$ and elements $a_1, \ldots, a_n$ of M.[1] If the embedding μ is the identity on A, then we say that $\mathcal{M}$ is a *substructure* of $\mathcal{N}$. In case (1.1) holds for all first-order formulae, then μ is said to be an *elementary* embedding. If the elementary embedding μ is the identity on M, then we say that $\mathcal{M}$ is an *elementary substructure* of $\mathcal{N}$ or that $\mathcal{N}$ is an *elementary extension* of $\mathcal{N}$. An *isomorphism* is a surjective embedding.

The *truth* of a Σ-formula in $\mathcal{A}$ is defined in the standard way (so that truth of a formula is equivalent to truth of its *universal* closure). A formula φ is *satisfiable* in $\mathcal{A}$ iff its *existential* closure is true in $\mathcal{A}$. A Σ-structure $\mathcal{A}$ is a *model* of a Σ-theory T (in symbols $\mathcal{A} \models T$) iff all the sentences in T are true in $\mathcal{A}$. For models of a Σ-theory T we shall use the letters $\mathcal{M}, \mathcal{N}, \ldots$ to distinguish them from arbitrary Σ-structures. If φ is a formula, $T \models \varphi$ ('φ is a *logical consequence* of T') means that φ is true in any model of T. A Σ-theory T is *complete* iff for every Σ-sentence φ, either φ or $\neg\varphi$ is a logical consequence of T; finally, T is *consistent* iff it has a model.

The main problems we deal with are *decidability of fragments*; more precisely, given a Σ-theory T and a recursive set of Σ-formulae Φ, we want to decide whether $T \models \varphi$ holds for a formula φ in the fragment Φ. In particular,

[1] As usual, there are some implicit (but obvious) notation conventions in the formulation of (1.1): the signature Σ is expanded to Σ^M, $\mathcal{M}$ is seen as a Σ^M-structure by interpreting $a \in M$ into a, $\mathcal{N}$ is seen as a Σ^M-structure by interpreting $a \in M$ into $\mu(a)$.

– if Φ consists of atomic Σ-formulae of the kind $t = u$, then it is called *equational fragment*;

– if Φ consists of Σ-clauses containing at most one atom, then it is called *universal Horn fragment*;

– if Φ consists of Σ-clauses, then it is called *universal fragment*;

– if Φ consists of all the Σ-formulae, then it is called *elementary fragment*.

The constraint satisfiability problem for the theory T is the problem of deciding whether (the conjunction of) a finite set of Σ-literals is satisfiable in a model of T. The complementary constraint unsatisfiability problem (i.e. the problem of deciding whether a finite set of Σ-literals is unsatisfiable in all the models of T) is easily reduced to the decidability of the universal fragment: notice in fact that T-unsatisfiability of $A_1 \wedge \cdots \wedge A_n$ is the same as the relation $T \models \neg\exists \underline{x}(A_1 \wedge \cdots \wedge A_n)$ (for the appropriate existential closure $\exists \underline{x}$), i.e. as the relation $T \models \forall \underline{x}(\neg A_1 \vee \cdots \vee \neg A_n)$. Vice versa, $T \models C$ (where C is the clause $B_1 \vee \cdots \vee B_m$) is equivalent to $T \models \forall \underline{x} C$ and hence to the T-unsatisfiability of $\neg B1 \wedge \cdots \wedge \neg B_m$. In the following, we shall prefer to use free constants instead of variables in constraint satisfiability problems, so that we (equivalently) redefine a Σ-*constraint* in a signature Σ as a finite set of $\Sigma^{\underline{a}}$-literals (where $\underline{a}$ is a finite set of new free constants); *constraint satisfiability problem* for the theory T becomes the problem of establishing the consistency of $T \cup \Gamma$ for a finite set Γ of ground $\Sigma^{\underline{a}}$-literals (where $\underline{a}$ is a finite set of new constants).

1.2 Examples from the Literature

We propose now two well-known decidability results: in Section 1.2.1 we describe the classical decision procedure for the Presburger Arithmetic, whereas in Section 1.2.2 we present an algorithm for the satisfiability of a formula in some model for the description language $\mathcal{ALC}$.

1.2.1 Presburger Arithmetic

Let us start recalling the classical decidability result for the theory of Presburger Arithmetic. A possible formalization is the following: let us consider the signature $\Sigma_{\mathcal{P}} = (0, \mathrm{s}, <, +)$, where 0 is a constant, s is a unary function symbol, $<$ is a binary predicate symbol and $+$ is a binary function symbol. Naturally, a $\Sigma_{\mathcal{P}}$-structure $\mathcal{N}$ is the set of natural number, where the symbols $(0, \mathrm{s}, <, +)$ are interpreted in the standard way. *Presburger Arithmetic* $\mathcal{P}$ is the theory whose axioms are the set of all the $\Sigma_{\mathcal{P}}$-sentences which are true in $\mathbb{N}$. The elementary fragment of $\mathcal{P}$ is decidable: in fact, using a process of quantifier

elimination it is possible to decide whether a generic $\Sigma_{\mathcal{P}}$-formula is entailed by the axioms of $\mathcal{P}$. In the following we will refer to this property simply saying that $\mathcal{P}$ is decidable.

We recall that a theory T over the signature Σ admits *quantifier elimination* if and only if for each Σ-formula $A(\underline{x})$ there exists a quantifier-free Σ-formula $B(\underline{x})$ such that $T \models A(\underline{x}) \leftrightarrow B(\underline{x})$. Moreover, if we define a *simply primitive* formula $A(\underline{x})$ a formula of the kind $\exists y(B_1 \wedge \cdots \wedge B_m)$, where B_i is an atom or a negation of an atom, it is possible to show that T admits quantifier elimination if and only if for every simply primitive formula $A(\underline{x})$ there exists a quantifier-free formula $C(\underline{x})$ such that $T \models A(\underline{x}) \leftrightarrow C(\underline{x})$.

Theorem 1.2.1 (Presburger, 1929)**.** *The theory $\mathcal{P}$ is decidable.*

Proof. First of all, we enlarge the language adding the infinite supply of binary predicate symbols $\equiv_2, \equiv_3, \equiv_4, \ldots$, which are interpreted, respectively, as the congruence relation modulo 2, modulo 3, modulo 4 etc. This will be useful in order to deal with the formulae of the kind $\exists y\, v_1 = y + y + \cdots + y$ that cannot be replaced in the original signature by a quantifier-free formula.

For a term t and a natural number n, let nt be the term $t+t+\cdots+t$ with n summands; it it easy to notice that every term can be expanded to one of the form $\mathrm{s}^{n_0}(0) + n_1x_1 + \cdots + n_kx_k$, where the numeral n_0 is used to abbreviate the term $\underbrace{s(0) + \cdots + s(0)}_{n_0-\text{times}}$.

Let us consider a simply primitive formula $\exists y(B_1 \wedge \cdots \wedge B_m)$. We start eliminating negations: we replace $\neg(t_1 = t_2)$ by $(t_1 < t_2 \vee t_2 < t_1)$, $\neg(t_1 < t_2)$ by $(t_1 = t_2 \vee t_2 < t_1)$ and $\neg(t_1 \equiv_n t_2)$ by $(t_1 \equiv_n t_2 + \mathrm{s}^1(0) \vee \cdots \vee t_1 \equiv_n t_2 + \mathrm{s}^{n-1}(0))$. Then we regroup into a disjunction of formulae of the form $\exists y(A_1 \wedge \cdots \wedge A_l)$, where each A_i is atomic; we may further suppose that y occurs in each A_i, and that each A_i has one of the following forms: $ny + t = u$ or $ny + t \equiv_n u$ or $ny + t < u$ or $u < ny + t$, where u and t do not contain y.

We uniform the coefficients of y. Let p be the least common multiple of the coefficient of y; each atomic formula can be converted to one in which the coefficient of y is p, by "multiplying through" by the appropriate factor. This is legitimate, recalling that two natural numbers a and b are congruent modulo n if and only if ka and kb are congruent modulo kn. Now we eliminate the coefficient of y replacing py by x and adding the new conjunct $x \equiv_p 0$. If one of the atoms in the formula $\exists x\vartheta$ obtained by our manipulations has the form $x + t = u$, we can substitute x by $u - t$. More precisely, we should transpose terms to compensate for the absence of subtraction: for example the atom $(x < v)[u-t/x]$ should be rewritten into $u < v+t$. In the following it is convenient to write these formulae with a subtraction symbol, using it only as a device for the sake of readability. Thus the formula $\exists x\vartheta$ is equivalent in the theory of $\mathcal{P}$ to the formula $\vartheta[u - t/x] \wedge t \leq u$, which is quantifier-free.

We may assume henceforth that no equality occurs between terms. So we consider

formulae of the kind

$$\begin{aligned}\exists x[&r_0 - t_0 < x \wedge \cdots \wedge r_{h-1} - t_{h-1} < x\\ &x < u_0 - v_0 \wedge \cdots \wedge x < u_{k-1} - v_{k-1}\\ &x \equiv_{m_0} w_0 - q_0 \wedge \cdots \wedge x \equiv_{m_{n-1}} w_{n-1} - q_{n-1}],\end{aligned}$$

where r_i, t_i, u_i, v_i, w_i and q_i are terms not containing x. We can shorten our formula into

$$\exists x[\bigwedge_{j<h} r_j - t_j < x \wedge \bigwedge_{i<k} x < u_i - v_i \wedge \bigwedge_{l<n} x \equiv_{m_l} w_l - q_l]. \tag{1.2}$$

If there are no congruences, then (1.2) asserts that there is a nonnegative space between the lower bound given by the maximum of the $(r_j - t_j)$'s and the upper bound given by the minimum of the $(u_i - v_i)$'s. So the formula can be replaced by the following quantifier-free one:

$$\bigwedge_{i<k} \bigwedge_{j<h} (r_j - t_j) + \mathrm{s}(0) < u_i - v_i \wedge \bigwedge_{i<k} 0 < u_i - v_i.$$

Otherwise, let M be the least common multiple of the moduli $m_0, \ldots, m_{n-1}$. (1.2) asserts the existence of a nonnegative number which is no less than a certain lower bound L, given by the maximum of the $(r_j - t_j)$'s, and which satisfies certain congruences and certain upper bounds. Because of the requirement of a non negative solution, (1.2) is equivalent to

$$\exists x[0 < x \wedge \bigwedge_{j<h} r_j - t_j < x \wedge \bigwedge_{i<k} x < u_i - v_i \wedge \bigwedge_{l<n} x \equiv_{m_l} w_l - q_l]. \tag{1.3}$$

In this way we can always assume that L is non negative or, equivalently, that r_0 is 0 and t_0 is 0 too. Moreover, if such a solution exists, then one of the following is a solution: $L, L+1, L+2, \ldots, L+M-1$. (1.3) can now be replaced by a quantifier-free disjunction that asserts that one of the numbers of the kind $(r_j - t_j) + D$, where D is in $\{0, \ldots, M-1\}$, is a nonnegative solution:

$$\bigvee_{j \le l} \bigvee_{d<M} [\bigwedge_{i<l} r_i - t_i < (r_j - t_j) + \mathrm{s}^d(0) \wedge \bigwedge_{i<k} (r_j - t_j) + \mathrm{s}^d(0) < u_i - v_i \wedge \bigwedge_{l<n} (r_j - t_j) + \mathrm{s}^d(0) \equiv_{l_i} w_l - q_l].$$

What shown above implies that every formula is equivalent with respect to the theory $\mathcal{P}$ to a quantifier-free formula over a language augmented with the symbols for the congruence relations modulo n. If we are now given a sentence φ, we can find a ground sentence ψ which is true in the intended structure $\mathbb{N}$ if and only if φ is. Deciding the truth of ψ is straightforward: we have to look at the atomic sentences and, as every term is in the form $\mathrm{s}^n(0)$, we have to check atoms of the kind $\mathrm{s}^h(0) = \mathrm{s}^k(0)$, or $\mathrm{s}^h(0) < \mathrm{s}^k(0)$, or $\mathrm{s}^h(0) \equiv_m \mathrm{s}^k(0)$, which are true, respectively, if and only if $h = k$, or $h < k$, or $h = k$

modulo m. □

There are many results in the literature about the complexity of the decision problem for (fragments of) Presburger Arithmetic. A lower bound for proving the validity of a prenex closed formula in the domain of integers can be found in [37], where it is proved that there is a series $\{\varphi_n\}$ of closed formulae of length linear in n such that the validity of φ_n cannot be algorithmically decided in non-deterministic time 2^{2^n}.

An upper bound for the worst-case complexity of the quantifier elimination problem in Presburger Arithmetic is in [89]. Let l be the length of the input formula φ, n the number of quantified variables in φ, and b the number of the quantifier-blocks in φ. The bound is of the form $\exp(l^{(cn)^b})$ for some constant c, where $\exp(x) = 2^x$. In [90] it is shown that this upper bound is essentially tight; so the quantifier elimination problem is inherently triply exponential.

These results seems to preclude any practical applications of algorithms involving decidability procedures for formulae of Presburger Arithmetic. However, if we consider the fragment of Presburger Arithmetic containing only clauses, we have at our disposal algorithms of lower complexity. In fact the validity problem for this fragment is the dual of checking a conjunction of ground literals for satisfiability. The latter reduces to the well-known Integer Linear Programming problem which is NP-complete, as shown in [78].

1.2.2 The Description Logic $\mathcal{ALC}$

In this section we deal with the the family of description languages $\mathcal{ALC}$. As already hinted in Section *Knowledge Representation* on page (xi), in order to introduce a description language we consider a set of *atomic concepts* $\mathrm{x}, \mathrm{y}, \ldots$ and a set of *role names* $R, S, \ldots$; *concepts* are built up from atomic concepts, Boolean operators $\bot, \top, \sqcap, \sqcup, \neg$, and relativized universal quantification $\forall R$ (here R is a role name). It is easy to see that concepts are a notational variant of propositional multimodal formulae (in modal logic the notation for $\forall R$ is the 'necessity' operator $\Box_R$), as pointed out by [76]. Moreover, it is usual to shorten the concept $\neg\forall R.\neg C$ with $\exists R.C$.

From the semantic side, an *interpretation* is a pair $\mathcal{I} = (W^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $W^{\mathcal{I}}$ is a non-empty set (the *domain*) and $\cdot^{\mathcal{I}}$ is the *interpretation function*, assigning to each atomic concept x a subset $\mathrm{x}^{\mathcal{I}} \subseteq W^{\mathcal{I}}$ and to each role name R a binary relation $R^{\mathcal{I}} \subseteq W^{\mathcal{I}} \times W^{\mathcal{I}}$. The interpretation function is inductively extended to concepts by interpreting the

Boolean operators and relativized universal quantification as follows:

$$\begin{aligned}
(\bot)^{\mathcal{I}} &:= \emptyset \\
(\top)^{\mathcal{I}} &:= W^{\mathcal{I}} \\
(C \sqcap D)^{\mathcal{I}} &:= C^{\mathcal{I}} \cap D^{\mathcal{I}} \\
(C \sqcup D)^{\mathcal{I}} &:= C^{\mathcal{I}} \cup D^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} &:= W^{\mathcal{I}} \setminus C^{\mathcal{I}} \\
(\forall R.C)^{\mathcal{I}} &:= \{w \in W^{\mathcal{I}} \mid \forall v \ (w,v) \in R^{\mathcal{I}} \rightarrow v \in C^{\mathcal{I}}\}.
\end{aligned}$$

Moreover, description languages give the opportunity to express inclusion between concepts, $C \sqsubseteq D$, and equality between concept, $C \equiv D$; naturally, an interpretation $\mathcal{I} = (W^{\mathcal{I}}, \cdot^{\mathcal{I}})$ satisfies the inclusion $C \sqsubseteq D$ if and only if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$, and satisfies the equality $C \equiv D$ if and only if $C^{\mathcal{I}} = D^{\mathcal{I}}$. A finite set $\mathcal{T}$ of concept equalities is called a *T-Box*,[2] and a *model* for $\mathcal{T}$ is an interpretation that satisfies all the equality of $\mathcal{T}$.

There are four important reasoning tasks involving concepts: given a T-Box $\mathcal{T}$, we can be interested in one of the following problems.

Satisfiability: a concept C is satisfiable with respect to $\mathcal{T}$ if there exists a model $\mathcal{I}$ of $\mathcal{T}$ such that $C^{\mathcal{I}}$ is nonempty;

Subsumption: a concept C is subsumed by a concept D with respect to $\mathcal{T}$ if $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{T}$;

Equivalence: two concepts C and D are equivalent with respect to $\mathcal{T}$ if $C^{\mathcal{I}} = D^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{T}$;

Disjointness: two concepts C and D are disjoint with respect to $\mathcal{T}$ if $C^{\mathcal{I}} \cup D^{\mathcal{I}} = \emptyset$ for every model $\mathcal{I}$ of $\mathcal{T}$.

Actually, the last three problems can be easily reduced to satisfiability tests: in fact, given a T-Box, (a) C is subsumed by D iff $C \sqcap \neg D$ is unsatisfiable; (b) C and D are equivalent iff both $C \sqcap \neg D$ and $\neg C \sqcap D$ are unsatisfiable; (c) C and D are disjoint iff $C \sqcap D$ is unsatisfiable.

Description Logics formalism allows also the reasoning about individuals. We enrich the language introducing *individual names*: $a, b, c, \ldots$; every interpretation $\mathcal{I} = (W^{\mathcal{I}}, \cdot^{\mathcal{I}})$ will associate to any individual name a an element of the set $w^{\mathcal{I}}$. Using concepts C one can make *concept assertion* $C(a)$, whereas using roles R one can make *role assertions* $R(a, b)$. A finite set of concept assertions and role assertions is said *A-Box*, and it is

[2]Notice that inclusions between concepts can be expressed as concept equalities; for example, $C \sqsubseteq D$ is equivalent to $C \equiv \overline{C} \sqcap D$.

usually denoted with $\mathcal{A}$. The interpretation $\mathcal{I} = (W^{\mathcal{I}}, \cdot^{\mathcal{I}})$ satisfies the concept assertion $C(a)$ if $a^{\mathcal{I}} \in C^{\mathcal{I}}$, and it satisfies the role assertion $R(a, b)$ if $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$; moreover, it satisfies an A-Box A if it satisfies each assertion in A. Finally, $\mathcal{I}$ satisfies an A-Box $\mathcal{A}$ with respect to a T-Box $\mathcal{T}$ if it satisfies both.

The main reasoning task involving A-Boxes is a consistency check: an A-Box $\mathcal{A}$ is *consistent* with respect to a T-Box $\mathcal{T}$ if there is an interpretation that satisfies $\mathcal{A}$ with respect to $\mathcal{T}$. The problem of the consistency of an A-Box with respect a T-Box is called *full satisfiability problem*; in case the T-Box is empty it is called *local satisfiability problem*.

In the following, for the sake of simplicity, we describe a decision procedure for the local satisfiability problem in the special case the description language contains a unique role name R, and the A-Box consists only on a single concept assertion $C(a)$. Notice that this problem is equivalent to concept satisfiability, because the concept C is satisfiable if and only if $C(a)$ is consistent, where a is an arbitrarily chosen individual name.

This kind of problem is indeed a problem of decidability of fragments: in fact, relying on the so-called *standard translation*, it is easy to translate $\mathcal{ALC}$ concepts into formulae of a first-order fragment. First of all, we choose a language containing infinite unary predicate symbols and a binary predicate symbol $R(x, y)$, and we fix a bijective correspondence $\mathrm{x} \mapsto X$ between the atomic concepts and the unary predicate symbols. Now it is sufficient to associate to every concept C a formula $ST(C, x)$ as follows:

$$\begin{aligned}
ST(\bot, x) &= \bot \\
ST(\top, x) &= \top \\
ST(\mathrm{x}, x) &= X(x) \\
ST(C_1 \sqcup C_2, x) &= ST(C_1, x) \vee ST(C_2, x) \\
ST(C_1 \sqcap C_2, x) &= ST(C_1, x) \wedge ST(C_2, x) \\
ST(\neg C, x) &= \neg ST(C, x) \\
ST(\forall R.C, x) &= \forall y(R(x, y) \rightarrow ST(C, y)).
\end{aligned}$$

It is straightforward to notice that a concept C is unsatisfiable iff the formula $\forall x\ ST(\neg C, x)$ is valid; in this way to solve the problem of concept satisfiability is equivalent to decide the validity of the formulae in the fragment $\Phi = \{\forall x\ ST(C, x)\}$.

A Decision Procedure for the Satisfiability Problem in $\mathcal{ALC}$

Consider the concept C. We can freely suppose that the only non boolean connective occurring in is $\forall R.$ since $\exists R.A$ has been introduced as a shortening for $\neg\forall R.\neg A$. Let us call $\mathcal{ALC}$ *atom* any concept whose main connective is not boolean. Examples of $\mathcal{ALC}$ atoms are x, $\forall R.(\mathrm{x} \sqcap \mathrm{y})$, and $\forall R.(\neg \mathrm{x} \sqcup \forall R.\mathrm{y})$. We write $C(\overrightarrow{\mathrm{x}}, \overrightarrow{\forall R.\mathrm{X}})$ to indicate that the

$\mathcal{ALC}$ atoms of C not occurring under the scope of the connective $\forall R.$ are $\overrightarrow{\mathrm{x}}$ and $\overrightarrow{\forall R.\mathrm{X}}$, where $\overrightarrow{\mathrm{x}}$ are the $\mathcal{ALC}$ atoms which are atomic concepts and $\overrightarrow{\forall R.\mathrm{X}}$ are the remaining ones. For example, if C is the concept

$$\neg\forall R.(\mathrm{x} \sqcup \neg\mathrm{y}) \sqcap \forall R.(\neg\forall R.(\neg\forall R.\mathrm{z})) \sqcap (\mathrm{x} \sqcup \neg\mathrm{w} \sqcup \forall R.\mathrm{w}),$$

the $\mathcal{ALC}$ atoms in $\overrightarrow{\mathrm{x}}$ are x and w, whereas the $\mathcal{ALC}$ atoms in $\overrightarrow{\forall R.\mathrm{X}}$ are $\forall R.(\mathrm{x} \sqcup \neg\mathrm{y})$, $\forall R.(\neg\forall R.(\neg\forall R.\mathrm{z}))$, and $\forall R.\mathrm{w}$. On the other hand, y is an $\mathcal{ALC}$ atoms occurring in C which is not member of $\overrightarrow{\mathrm{x}}$, whereas $\forall R.(\neg\forall R.\mathrm{z})$ and $\forall R.\mathrm{z}$ are $\mathcal{ALC}$ atoms occurring in C which are not members of $\overrightarrow{\forall R.\mathrm{X}}$.

We suppose now to have at our disposal a propositional decision procedure which is capable to enumerate all the possible assignments satisfying a propositional formula β, i.e. a formula built up from a set of propositional variables $p, q, r, \ldots$ using the boolean connectives $\vee, \wedge$ and $\neg$. This is not an heavy assumption at all: for example the DPLL procedure (see [29, 30] for the original algorithm) performs very well to this aim. We are now ready to describe our procedure $\textsc{Sat}_{\mathcal{ALC}}$.

Algorithm 1 The function $\textsc{Sat}_{\mathcal{ALC}}$

function $\textsc{Sat}_{\mathcal{ALC}}(C(\overrightarrow{\mathrm{x}}, \overrightarrow{\forall R.\mathrm{X}}))$
  for all $\alpha \in \{\alpha \mid \alpha \text{ is an assignment and } \alpha \models C\}$ **do**
    if $\{\forall R.D \in \overrightarrow{\forall R.\mathrm{X}} \mid \alpha(\forall R.D) = 0\} = \emptyset$ **then**
      return *"satisfiable"*
    else
      $\mathcal{B} \leftarrow \{B \mid \forall R.B \in \overrightarrow{\forall R.\mathrm{X}} \text{ and } \alpha(\forall R.B) = 1\}$
      $\mathcal{D} \leftarrow \{D \mid \forall R.D \in \overrightarrow{\forall R.\mathrm{X}} \text{ and } \alpha(\forall R.D) = 0\}$
      if $\bigwedge_{D \in \mathcal{D}}(\textsc{Sat}_{\mathcal{ALC}}(\neg D \sqcap \bigsqcap \mathcal{B}) =$ *"satisfiable"*$)$ **then return** *"satisfiable"*
    end if
  end for
  return *"unsatisfiable"*
end function

Remark 1.2.2 (Algorithm 1)**.** The sentence "α is an assignment and $\alpha \models C$" means that we consider C as a propositional formula whose propositional variables are the $\mathcal{ALC}$ atoms of C, and we make the obvious identification of the connectives $\sqcap, \sqcup$ with, respectively, $\wedge$ and $\vee$. More in detail, each of the $\mathcal{ALC}$ atoms in $\overrightarrow{\forall R.\mathrm{X}}$ plays the role of a propositional variable: thus, it makes sense to ask for an assignment to a SAT-solver.

To the procedure $\textsc{Sat}_{\mathcal{ALC}}$ we associate the procedure $\textsc{Tree}_{\mathcal{ALC}}$ which, given a concept C, returns, if possible, a labeled tree. A *tree* is a set W endowed with an antisymmetric binary relation R satisfying the following conditions: there exists w_0 such that (a) for every $w \in (W \setminus \{w_0\})$ there exists $n > 0$ such that $w_0 R w_1 R \cdots R w_n = w$; moreover (b)

the aforesaid path $w_0, w_1, \ldots, w_n$ is unique (the element w_0 is said the *root* of the tree). In this context, a *labeling function* is a function ν from W to the powerset of the atomic concept of a given $\mathcal{ALC}$ language; a *labeled tree* is a tree endowed of such a labeling function.

Algorithm 2 An algorithm generating labeled trees for $\textsc{Sat}_{\mathcal{ALC}}$

function $\textsc{Tree}_{\mathcal{ALC}}(C(\overrightarrow{\mathrm{x}}, \overrightarrow{\forall R.\mathrm{X}}))$
  for all $\alpha \in \{\alpha \mid \alpha \text{ is an assignment and } \alpha \models C\}$ **do**
    if $\{\forall R.D \in \overrightarrow{\forall R.\mathrm{X}} \mid \alpha(\forall R.D) = 0\} = \emptyset$ **then**
      $M_C \leftarrow \langle \{\rho_C\}, \emptyset, \nu \rangle$ (where $\nu(\rho_C) = \{\mathrm{x} \in \overrightarrow{\mathrm{x}} \mid \alpha(\mathrm{x}) = 1\}$)
      return M_C
    else
      $\mathcal{B} \leftarrow \{B \mid \forall R.B \in \overrightarrow{\forall R.\mathrm{X}} \text{ and } \alpha(\forall R.B) = 1\}$
      $\mathcal{D} \leftarrow \{D \mid \forall R.D \in \overrightarrow{\forall R.\mathrm{X}} \text{ and } \alpha(\forall R.D) = 0\}$
      $M \leftarrow \bigcup_{D \in \mathcal{D}} \textsc{Tree}_{\mathcal{ALC}}(\neg D \sqcap \bigsqcap \mathcal{B})$
      if $\emptyset \notin M_D$ **then**
        let M_C be the tree obtained from the disjoint union $\sum_D M_D$ of the trees M_D in M by: *(i)* adjoining a new point ρ_C, *(ii)* stating the relation $\rho_C \, R \, \rho_D$, where ρ_D is the root of the tree M_D for each D, and *(iii)* extending the function ν by $\nu(\rho_C) = \{\mathrm{x} \in \overrightarrow{\mathrm{x}} \mid \alpha(\mathrm{x}) = 1\}$
        return M_C
      end if
  end for
  return $\emptyset$
end function

Remark 1.2.3 (Algorithm 2)**.** Without loss of generality, we can always suppose that the sets $\{W_D\}_D$ of the trees $\{M_D\}_D$ are pairwise disjoint (if not, a renaming is sufficient). The disjoint union of the structures $M_D = \langle W_D, R_D, \nu_D \rangle$ is the structure

$$\sum_D M_D = \langle \bigcup_D W_D, \bigcup_D R_D, \bigcup_D \nu_D \rangle$$

(notice that, being the domain of ν_D's pairwise disjoint, $\bigcup_D \nu_D$ is still a function).

The labeled tree $\langle W, R, \nu \rangle$ returned by $\textsc{Tree}_{\mathcal{ALC}}(C)$ is naturally an interpretation $\mathcal{I}$: the unique role name of the language is interpreted in the binary relation R, and the labeling function ν determines the interpretation of the concepts. In fact, it is sufficient to set, for each atomic concept x, $\mathrm{x}^{\mathcal{I}} = \{w \in W \mid \mathrm{x} \in \nu(w)\}$. In general, every set W endowed with a binary relation R and a function ν from W to the powerset of the atomic concept of an $\mathcal{ALC}$ language containing a unique role name determines in a unique way an interpretation for that language. Also the converse is (obviously) true: so we shall use indifferently interpretations $\mathcal{I} = (W^{\mathcal{I}}, \cdot^{\mathcal{I}})$ or structures $M = \langle W, R, \nu \rangle$ for a given $\mathcal{ALC}$

language containing a unique role symbol.

We say that a concept C is satisfied in a point w of the interpretation $\mathcal{I}$ determined by the structure $M = \langle W, R, \nu \rangle$ if $w \in C^{\mathcal{I}}$; in symbols, we write $M \models_w C$.

Lemma 1.2.4. *Let C be a concept such that $\textsc{Sat}_{\mathcal{ALC}}(C)$ = "satisfiable". Then the tree M_C returned by $\textsc{Tree}_{\mathcal{ALC}}(C)$ satisfies C in its root.*

Proof. It is straightforward to notice that if $\textsc{Sat}_{\mathcal{ALC}}(C)$ = *"satisfiable"*, then $\textsc{Tree}_{\mathcal{ALC}}(C) \neq \emptyset$. The proof of the lemma is by induction on the depth of M_C. Let us start analyzing the case M_C has depth 0, i.e. it consists only of a root node named ρ_C. We have to show that $M_C \models_{\rho_C} C$. We notice that for each atom $\forall R.D$ in $\overrightarrow{\forall R.\mathrm{X}}$, the propositional assignment α considered by the procedure $\textsc{Tree}_{\mathcal{ALC}}$ to return a tree of depth 0 is such that $\alpha(\forall R.D) = 1$, otherwise $\textsc{Tree}_{\mathcal{ALC}}(C)$ would have returned a structure different from $\langle \{\rho_C\}, \emptyset, \nu \rangle$. Observing that (i) $M_C \models_{\rho_C} \forall R.A$ for each concept A; (ii) for each atomic concept x, $M_C \models_{\rho_C} \mathrm{x}$ if and only if $\alpha(\mathrm{x}) = 1$, and (iii) α is an assignment which satisfies C, it follows the thesis.

Suppose now that M_C has depth $n > 0$ and consider the assignment α used by the procedure $\textsc{Tree}_{\mathcal{ALC}}$. Given the set of atoms $\mathcal{B} = \{B \mid \forall R.B \in \overrightarrow{\forall R.\mathrm{X}} \text{ and } \alpha(\forall R.B) = 1\}$, for each $\forall R.D$ in $\overrightarrow{\forall R.\mathrm{X}}$ such that $\alpha(\forall R.D) = 0$ we have at our disposal the tree M_D such that $M_D \models_{\rho_D} (\neg D \sqcap \bigsqcap \mathcal{B})$ by inductive hypothesis (since the depth of M_D is less then the depth of M_C). We have to show that $M_C \models_{\rho_C} C$. Since, for each D as above, $M_D \models_{\rho_D} (\neg D \sqcap \bigsqcap \mathcal{B})$, then $M_D \models_{\rho_D} B$ for each $B \in \mathcal{B}$. Thus, by construction of M_C, we have $M_C \models_{\rho_C} \forall R.B$ for each $B \in \mathcal{B}$. Analogously, since $M_D \models_{\rho_D} \neg D$, $M_C \models_{\rho_C} \neg \forall R.D$ for each D as above. Finally, observing that, for each atomic concept x, $M_C \models_{\rho_C} \mathrm{x}$ if and only if $\alpha(\mathrm{x}) = 1$, and that α is an assignment which satisfies C, it follows that $M_C \models_{\rho_C} C$. □

Lemma 1.2.5. *Let C be a satisfiable concept. Then $\textsc{Sat}_{\mathcal{ALC}}(C)$ returns "satisfiable".*

Proof. Let us define the *degree* of a concept as follows:

- $deg(\mathrm{x}) = 0$ if x is a atomic concept;
- $deg(D * B) = max(deg(D), deg(B))$ if $*$ is the connective $\sqcup$ or $\sqcap$;
- $deg(\neg D) = deg(D)$;
- $deg(\forall R.D) = deg(D) + 1$.

The proof of the statement is by induction on the degree of C. By hypothesis, there exists an interpretation $\mathcal{I}$ such that $C^{\mathcal{I}}$ contains at least an element, say ρ_C. Let $M = \langle W, R, \nu \rangle$ be the structure determined by $\mathcal{I}$. We can consider the propositional assignment $\overline{\alpha}$ defined as follows:

– for every atomic concept x in $\overrightarrow{\mathrm{x}}$, $\overline{\alpha}(\mathrm{x}) = 1$ iff $M \models_{\rho_C} \mathrm{x}$;

– for every atom $\forall R.A$ in $\overrightarrow{\forall R.\mathrm{X}}$, $\overline{\alpha}(\forall R.A) = 1$ iff $M \models_{\rho_C} \forall R.A$.

If $deg(C) = 0$, $\textsc{Sat}_{\mathcal{ALC}}(C)$ returns *"satisfiable"* because our assumption on the availability of a procedure able to find all the propositional assignment for C implies that $\overline{\alpha}$ will be eventually analyzed in $\textsc{Sat}_{\mathcal{ALC}}(C)$.

On the other hand, if $deg(C) > 0$, suppose by inductive hypothesis that $\textsc{Sat}_{\mathcal{ALC}}(D)$ returns *"satisfiable"* for each satisfiable concept D such that $deg(D) < deg(C)$. For each atom $\forall R.D$ such that $\overline{\alpha}(\forall R.D) = 0$, there exists an element ρ_D in W such that $\rho_C R \rho_D$ and such that $M \models_{\rho_D} \neg D \sqcap \bigsqcap \mathcal{B}$ where $\mathcal{B} = \{B \mid \forall R.B \in \overrightarrow{\forall R.\mathrm{X}} \text{ and } \overline{\alpha}(\forall R.B) = 1\}$. Since $\forall R.D$ and $\forall R.B$ (for each $B \in \mathcal{B}$) are proper subconcepts of C, it is easy to check that the degree of $\neg D \sqcap \bigsqcap \mathcal{B}$ is less than the degree of C. Thus the inductive hypothesis applies, guaranteeing that $\textsc{Sat}_{\mathcal{ALC}}(\neg D \sqcap \bigsqcap \mathcal{B})$ is *"satisfiable"* for every D as above. Now it is clear that $\textsc{Sat}_{\mathcal{ALC}}(C)$ returns *"satisfiable"*. □

Some few words about complexity issue. A brief inspection of Algorithm 1 makes it clear that the proposed procedure is a PSPACE one; on the other hand, the satisfiability of $\mathcal{ALC}$-concept description (without any condition on the number of roles present in the language) is a problem which is PSPACE-complete (see [7, 77]).

1.3 Arrays with Dimension

The results about the decidability of fragments described in Section 1.2.1 and Section 1.2.2 are quite old and well known; in this section we will present a newer one. We will show that the constraint satisfiability problem is decidable for the theory of finite arrays, that is the theory of the sequences over a certain set of elements which are eventually equal to a fixed element, say $\perp$, and which features two operations: one for reading the value stored at a certain position i in an array a (in symbols, $a[i]$), and one for the pointwise modification of an array a at position i with the value e (in symbols, $a[i \mapsto e]$). As we have required that our sequences are eventually equal to $\perp$, for every array a there exists a smallest index n such that $a[i] = \perp$ for $i \geq n$: this index is the *dimension* of a. The reason to use the word "dimension" rather than "length" lies in the fact that we do not require that $a[k] \neq \perp$ for each index k less than the dimension of a. There is just one array whose dimension is zero: we indicate it by ε, and call it the empty array. Moreover we require the theory of Presburger Arithmetic over indexes; this is a natural choice since many applications of verification require to reason about arithmetic expressions over the indexes of arrays.

We formally introduce the theory of array as a union of subtheories over non-disjoint signatures. This will help us in the development of the decision procedure for the constraint satisfiability problem.

$\boxed{\mathcal{T}_0}$ has just one sort symbol INDEX, the following function and predicate symbols: $0 :$ INDEX, $\mathrm{s} :$ INDEX $\rightarrow$ INDEX, and $<:$ INDEX $\times$ INDEX. It is axiomatized by the following sentences:

$$y \neq 0 \rightarrow \exists z(y = \mathrm{s}(z)) \tag{1.4}$$

$$x < \mathrm{s}(y) \leftrightarrow (x < y \vee x = y) \tag{1.5}$$

$$\neg(x < 0) \tag{1.6}$$

$$x < y \vee x = y \vee y < x \tag{1.7}$$

$$x < y \rightarrow \neg(y < x) \tag{1.8}$$

$$x < y \rightarrow (y < z \rightarrow x < z) \tag{1.9}$$

where x and y are implicitly universally quantified variables of sort INDEX.[3] This theory admit elimination of quantifiers and it is complete (see [34] for details): its intended structure is the set of natural numbers endowed with zero and successor. As usual, below, we will write $x \leq y$ as an abbreviation of $x < y \vee x = y$.

$\boxed{\mathcal{P}}$ is Presburger arithmetic over indexes. The signature is that of $\mathcal{T}_0$ extended with the function symbol for addition $+ :$ INDEX $\times$ INDEX $\rightarrow$ INDEX, written infix. As already done in Section 1.2.1, we will still use the numeral n to abbreviate the term $\underbrace{s(0) + \cdots s(0)}_{n-\text{times}}$. Clearly, $\mathcal{T}_0 \subset \mathcal{P}$.

$\boxed{\mathcal{A}}$ is the theory of arrays which has the following signature:

- sort symbols: INDEX, ELEM, ARRAY and
- function symbols: select : ARRAY$\times$INDEX $\rightarrow$ ELEM and store : ARRAY$\times$INDEX$\times$ ELEM $\rightarrow$ ARRAY

and it is axiomatized by the following sentences:

$$\mathrm{select}(\mathrm{store}(a, i, e), i) = e \tag{1.10}$$

$$i \neq j \rightarrow \mathrm{select}(\mathrm{store}(a, i, e), j) = \mathrm{select}(a, j) \tag{1.11}$$

[3] In the following, we will often omit the outermost universal quantification as well as sort information for variables for the sake of readability.

$\boxed{\mathcal{A}_e}$ is the theory of arrays with extensionality which has the same signature of $\mathcal{A}$ and it is axiomatized by (1.10), (1.11), and the axiom of extensionality:

$$\forall i(\text{select}(a,i) = \text{select}(b,i)) \rightarrow a = b \tag{1.12}$$

Notice that $\mathcal{A} \subset \mathcal{A}_e$.

$\boxed{\mathcal{A}_{dim}}$ is the simple theory of arrays with dimension whose signature is the union of the signatures of $\mathcal{T}_0$ and $\mathcal{A}_e$ extended with the following three symbols: $\bot$: ELEM, ε : ARRAY, and dim : ARRAY $\rightarrow$ INDEX. It is axiomatized by the axioms in $\mathcal{T}_0$, those in $\mathcal{A}_e$, and the following sentences:

$$\dim(a) \leq i \rightarrow \text{select}(a,i) = \bot \tag{1.13}$$

$$\dim(a) = \text{s}(i) \rightarrow \text{select}(a,i) \neq \bot \tag{1.14}$$

$$\dim(\varepsilon) = 0 \tag{1.15}$$

Notice that $\mathcal{T}_0 \subset \mathcal{A}_{dim}$ and $\mathcal{A}_e \subset \mathcal{A}_{dim}$.

$\boxed{\mathcal{ADP}}$ is the theory of arrays with dimension whose signature is the union of the signatures of $\mathcal{A}_{dim}$ and $\mathcal{P}$ and is axiomatized by the axioms in $\mathcal{A}_{dim}$ and all valid sentences in $\mathcal{P}$.

We notice that $\mathcal{A}_{dim}$ extends $\mathcal{A}_e$ (both in the signature and in the axioms), but is smaller than $\mathcal{ADP}$, because indexes are only endowed with a discrete linear poset structure. In this way, we have that $\mathcal{ADP} = \mathcal{A}_{dim} \cup \mathcal{P}$ and the theories $\mathcal{A}_{dim}$ and $\mathcal{P}$ share the complete theory $\mathcal{T}_0$. The situation is depicted in Figure 1.1, where solid boxes represent theories and the dashed box represents the non-disjoint union of $\mathcal{P}$ and $\mathcal{A}_{dim}$, which share the theory $\mathcal{T}_0$.

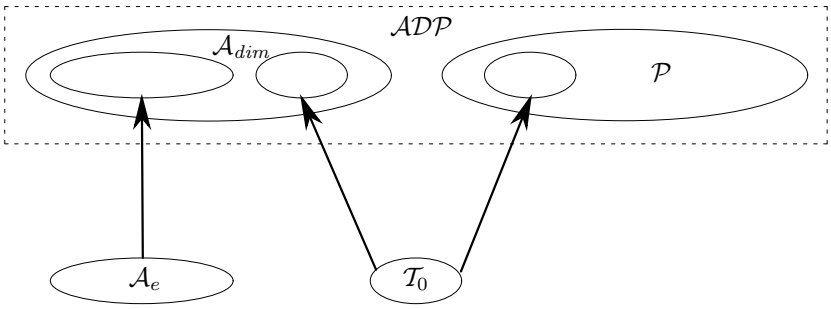

Figure 1.1: $\mathcal{ADP}$ as the non-disjoint combination of $\mathcal{P}$ and $\mathcal{A}_{dim}$ sharing $\mathcal{T}_0$

The theories $\mathcal{T}_0$ is decidable, because it admits quantifier elimination (see, e.g., [34]); this fact leads to the decidability of the theory, by considerations similar to the ones used

to show the decidability of the theory $\mathcal{P}$. $\mathcal{A}$ and $\mathcal{A}_e$ have the universal fragment decidable: for example, these results are obtained in [5] applying the superposition calculus. These considerations allow us to assume the availability of two decision procedures for the constraint satisfiability problems of $\mathcal{P}$ and $\mathcal{A}$.

Standard Models The theories $\mathcal{A}_e$, $\mathcal{A}_{dim}$, and $\mathcal{ADP}$ admit a particular subclass of models, the *standard* ones; they are the structures that give the arrays the meaning of sequences of elements eventually constant. Formally, a standard model is a structure that matches the following requirements.

Definition 1.3.1. Let A be a set and κ be an element of A. The *standard model of $\mathcal{ADP}$ induced by the pair* (A, κ) is the $\Sigma_{\mathcal{ADP}}$-structure $\mathcal{M}$ such that

(i) the sort INDEX is interpreted in $\mathcal{M}$ as $\mathbb{N}$ and the symbols $0, <, s, +$ have their natural meaning;

(ii) the sort ELEM is interpreted in $\mathcal{M}$ as A and the constant $\perp$ is interpreted as κ;

(iii) the sort ARRAY is interpreted in $\mathcal{M}$ as the set of functions $\mathfrak{a} : \mathbb{N} \longrightarrow A$ such that there is some $n_{\mathfrak{a}} \in \mathbb{N}$ for which we have $\mathfrak{a}(m) = \kappa$ whenever $m \geq n_{\mathfrak{a}}$;

(iv) the constant ε is interpreted as the constant function with value κ;

(v) for each function $\mathfrak{a} : \mathbb{N} \longrightarrow A$, $dim^{\mathcal{M}}(\mathfrak{a})$ is the smallest $n \in \mathbb{N}$ such that $\mathfrak{a}(m) = \kappa$ holds for all $m \geq n$;

(vi) finally, for each function $\mathfrak{a}$ and index i, we have $\text{select}^{\mathcal{M}}(\mathfrak{a}, i) = \mathfrak{a}(i)$ and, for each element $e \in A$, $\text{store}^{\mathcal{M}}(\mathfrak{a}, i, e)$ is the following function:

$$\text{store}^{\mathcal{M}}(\mathfrak{a}, i, e)(n) := \begin{cases} \mathfrak{a}(n) & \text{if } n \neq i, \\ e & \text{otherwise.} \end{cases}$$

The standard models of $\mathcal{A}_e$ and $\mathcal{A}_{dim}$ can be defined in a similar way by taking the $\Sigma_{\mathcal{A}_e}$- and $\Sigma_{\mathcal{A}_{dim}}$-reduct (respectively) of $\mathcal{ADP}$-standard models. Notice again that, contrary to finite sequences, we do not require elements at indexes less than the dimension of an array to be different from $\perp$.

In the following we shall prove that a constraint is satisfiable in a model of $\mathcal{ADP}$ iff it is satisfiable in a standard model (see Lemma 1.3.7, below). Notice that this is far from being obvious, because $\mathcal{ADP}$ is not a complete theory: it cannot be so, since the full first-order theory of $\mathcal{ADP}$ is undecidable. This can be reduced to the fact that the full first-order theory of $\mathcal{A}_e$ is undecidable (see, e.g., [83]). As a preliminary step, we state and prove a simple relationship between arbitrary and standard models of $\mathcal{ADP}$.

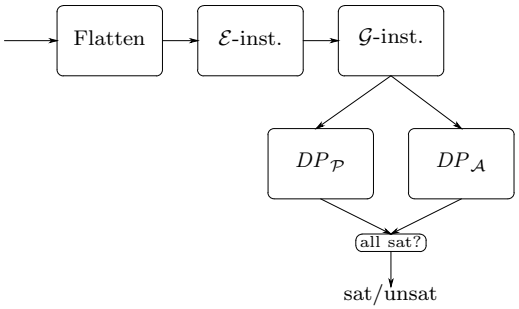

Figure 1.2: The architecture of the decision procedure for $\mathcal{ADP}$

Proposition 1.3.2. *Let $\mathcal{M}$ be a model of $\mathcal{A}_e$, $\mathcal{A}_{dim}$, or $\mathcal{ADP}$. Then, there exists a substructure of $\mathcal{M}$ which is a standard model of $\mathcal{A}_e$, $\mathcal{A}_{dim}$, or $\mathcal{ADP}$, respectively.*

Proof. Call an element of $\textsc{index}^{\mathcal{M}}$ standard iff it is of the kind $n^{\mathcal{M}}$, where $n := \underbrace{\mathrm{s}(\cdots \mathrm{s}}_{n-\text{times}}(0)\cdots)$ is a numeral. Similarly, call an element of sort $\textsc{array}^{\mathcal{M}}$ standard iff it is of the kind $\mathrm{store}^k(\varepsilon, \underline{n}, \underline{e})^{\mathcal{M}}$, where $\mathrm{store}^k(\varepsilon, \underline{n}, \underline{e})$ is a term denoting the result of writing the k-elements $\underline{e} \subseteq \textsc{elem}^{\mathcal{M}}$ in the numeral positions $\underline{n}$ on the empty array. Standard indexes, standard arrays and the whole set $\textsc{elem}^{\mathcal{M}}$ are closed under the operations in the signature $\Sigma_{\mathcal{ADP}}$ and form a substructure of $\mathcal{M}$ which is isomorphic to the standard model built up from $\textsc{elem}^{\mathcal{M}}$ and $\bot^{\mathcal{M}}$. □

1.3.1 A Decision Procedure for Arrays with Dimension

As already said, we assume the availability of two decision procedures solving the $\mathcal{A}$- and $\mathcal{P}$-satisfiability problems. We will see how to reduce the $\mathcal{ADP}$-satisfiability problems to two constraint satisfiability problems, one in $\mathcal{A}$ and one in $\mathcal{P}$.

1.3.2 The Architecture

The overall schema of the decision procedure for $\mathcal{ADP}$ is depicted in Figure 1.2. The module Flatten pre-process the literals in the input constraint so to make them flat and easily recognizable as belonging to one theory among those used to define $\mathcal{ADP}$, i.e. $\mathcal{T}_0$, $\mathcal{P}$, $\mathcal{A}_e$, and $\mathcal{A}_{dim}$. The module $\mathcal{E}$-instantiation produces suitable instances of the extensionality axiom (1.12) so that a simpler decision procedure for the satisfiability problem for $\mathcal{A}$ (with respect to one for $\mathcal{A}_e$) is assumed available. The module $\mathcal{G}$-instantiation is non-deterministic and guesses sufficiently many facts which are potentially entailed in $\mathcal{P}$ and $\mathcal{A}_{dim}$ by the constraints. The modules $DP_{\mathcal{P}}$ and $DP_{\mathcal{A}}$ implement the decision procedures for Presburger Arithmetic and the theory of arrays (without extensionality). The module 'all sat?' returns *"satisfiable"* if both decision procedures for $\mathcal{P}$ and $\mathcal{A}$ returned *"satisfiable"*; otherwise, returns *"unsatisfiable"*.

Now, we are ready to describe the internal working of each module in full detail.

Flattening

Without loss of generality, when considering a set S of ground literals to be checked for satisfiability, we may assume that each literal ℓ in S is *flat*, i.e. ℓ is required to be either of the form $a = f(a_1, \ldots, a_n)$, $P(a_1, \ldots, a_n)$, or $\neg P(a_1, \ldots, a_n)$, where $a, a_1, \ldots, a_n$ are constants (of appropriate sort), f is a function symbol, and P is a predicate symbol (possibly the equality symbol). It is well-known (see, e.g., [5]) that, given a Σ_T-constraint φ, there exists a linear time algorithm which returns a T-equisatisfiable flat $\Sigma_T^{\underline{a}}$-constraint φ' by introducing "fresh" constants to name all the subterms occurring in φ.

In our case, we assume that the module Flatten in Figure 1.2 transforms (in linear time) a set of arbitrary literals over the signature $\Sigma_{\mathcal{ADP}}^{\underline{a}}$[4] into an equisatisfiable set of flat literals on the signature $\Sigma_{\mathcal{ADP}}^{\underline{c}}$, for some set $\underline{c} \supseteq \underline{a}$ of constants (the constants in $\underline{c} \setminus \underline{a}$ are said to be fresh). For the theory $\mathcal{ADP}$, we assume that flat literals can be only of the following two forms:

- $c_1 \bowtie^\star c_2$, where c_1, c_2 are constants and $\bowtie^\star \in \{=, \neq, <, \not<\}$;
- $f(c_1, \ldots, c_n) = c_{n+1}$ where c_i are constants and $f \in \{\text{select}, \text{store}, \text{dim}, +\}$;

It will be useful to regard a set L of flat literals over $\Sigma_{\mathcal{ADP}}^{\underline{c}}$ as the union of four disjoint subsets, i.e.

$$L := L_{\mathcal{A}_e} \cup L_d \cup L_{\mathcal{T}_0} \cup L_+$$

where the literals in $L_{\mathcal{A}_e}$ are of the following forms:

$$\text{select}(a, i) = e \qquad \text{store}(a, i, e) = b \qquad a \bowtie b \qquad e \bowtie e',$$

with $a, b : \textsc{array}, i : \textsc{index}$, and $e, e' : \textsc{elem}$, the literals in L_d are of the form

$$\dim(a) = i,$$

with $a : \textsc{array}, i : \textsc{index}$, the set $L_{\mathcal{T}_0}$ contains literals of the forms

$$i_1 \bowtie^\star i_2 \qquad \text{s}(i_1) = i_2,$$

with $i_1, i_2 : \textsc{index}$; and finally, the set L_+ contains literals of the form

$$i_1 + i_2 = i_3,$$

[4]Recall from our conventions of Section 1.1 that $\Sigma_{\mathcal{ADP}}^{\underline{a}}$ is the signature of the theory $\mathcal{ADP}$ expanded with the free constants $\underline{a}$.

with $i_1, i_2, i_3 : \text{INDEX}$.

Below, w.l.o.g., we assume all sets of literals to be flat. Moreover, we will abbreviate $L_{\mathcal{T}_0} \cup L_+$ as $L_{\mathcal{P}}$.

$\mathcal{E}$-instantiation Closure

The module $\mathcal{E}$-instantiation finds enough instances of the axiom (1.12) for extensionality of arrays, according to Definition 1.3.3, so that it can be eliminated without compromising the correctness of the decision procedure for $\mathcal{ADP}$.

Definition 1.3.3 ($\mathcal{E}$-instantiation Closed Set of Literals)**.** A set L of ground flat literals is *$\mathcal{E}$-instantiation closed* iff the following condition is satisfied:

1. if $a \neq b \in L$, with $a, b : \text{ARRAY}$, then $\{\text{select}(a, i) = e_1, \text{select}(b, i) = e_2, e_1 \neq e_2\} \subseteq L$ for some constants $i : \text{INDEX}, e_1, e_2 : \text{ELEM}$;

It is not difficult to see that, given a set of ground flat literals L, there exists an $\mathcal{ADP}$-equisatisfiable set $L^{\mathcal{E}} \supseteq L$ which is $\mathcal{E}$-instantiation closed since contains also the skolemization of some logical consequences of $\mathcal{A}_e \cup L$.

Lemma 1.3.4. *There exists a linear time algorithm which takes a set L of flat literals over the signature $\Sigma^{\underline{a}}_{\mathcal{ADP}}$ and returns a $\mathcal{E}$-instantiation closed set $L^{\mathcal{E}}$ of flat literals over the signature $\Sigma^{\underline{c}}_{\mathcal{ADP}}$ such that (i) $L \subseteq L^{\mathcal{E}}$, (ii) L and $L^{\mathcal{E}}$ are $\mathcal{ADP}$-equisatisfiable, and (iii) $\underline{a} \subseteq \underline{c}$.*

The signature $\Sigma^{\underline{c}}_{\mathcal{ADP}}$ of $L^{\mathcal{E}}$ is a proper simple expansion of the signature $\Sigma^{\underline{a}}_{\mathcal{ADP}}$ of L, because Skolem constants must be fresh. If $L^{\mathcal{E}}$ is the set of literals produced by the module $\mathcal{E}$-instantiation when taking as input a set L of n literals, then the number of new literals in $L^{\mathcal{E}}$ is at most $3n$. Since producing a new literal takes constant time, there must exist a linear time algorithm to compute $\mathcal{E}$-instantiation closed sets.

$\mathcal{G}$-instantiation Closure

The module $\mathcal{G}$-instantiation is non-deterministic and it is responsible to produce suitable instances of the axioms about the dimension of arrays, i.e. (1.13) and (1.14), and to guess enough facts entailed by the input constraint w.r.t. $\mathcal{P}$ so to guarantee the correctness of the overall decision procedure for $\mathcal{ADP}$-satisfiability.

Definition 1.3.5 ($\mathcal{G}$-instantiation Closed Set of Literals)**.** A set L of ground flat literals is *$\mathcal{G}$-instantiation closed* iff the following conditions are satisfied:

1. if ε occurs in L, then $\dim(\varepsilon) = 0 \in L$.

2. if $\dim(a) = i \in L$, with a : ARRAY and i : INDEX, then $\{i = 0\} \subseteq L$ or $\{e \neq \bot, \text{select}(a, j) = e, s(j) = i\} \subseteq L$ for some constant j : INDEX;

3. if i, j occur in L, with i, j : INDEX, then $i = j \in L$ or $i \neq j \in L$;

4. if i, j occur in L, with i, j : INDEX and $i \neq j \in L$, then $i < j \in L$ or $j < i \in L$;

5. if $\{\dim(a) = i, i \leq j\} \subseteq L$, with a : ARRAY and i, j : INDEX, then $\{\text{select}(a, j) = \bot\} \subseteq L$ (here $i \leq j$ stands for $i < j$ or $i = j$).

It is not difficult to see that, given a set of literals, it is always possible to compute an equisatisfiable $\mathcal{G}$-instantiation closed set in (non-deterministic) polynomial time.

Lemma 1.3.6. *There exists a non-deterministic polynomial time algorithm which takes as input a set L of ground flat literals over a signature $\Sigma^{\underline{a}}_{\mathcal{ADP}}$ and returns a $\mathcal{G}$-instantiation closed set $L^{\mathcal{G}}$ of flat literals over the signature $\Sigma^{\underline{c}}_{\mathcal{ADP}}$ such that (i) $L \subseteq L^{\mathcal{G}}$, (ii) L and $L^{\mathcal{G}}$ are $\mathcal{ADP}$-equisatisfiable, and (iii) $\underline{a} \subseteq \underline{c}$.*

Proof. Let m be the number of literals in L of the form $\dim(a_k) = d_{a_k}$ where d_{a_k} is a constant of sort INDEX. Let us consider a set $\underline{b} = \{j_1, \ldots, j_m, e_1, \ldots, e_m\}$ of fresh constants, where j_k : INDEX, e_k : ELEM, and $k \in \{1, \ldots, m\}$. A $\mathcal{G}$-instantiation $L^{\mathcal{G}}$ of L can be computed by exhaustively executing the following three steps:

1. for each pair i, j of constants of sort INDEX in $\underline{a} \cup \underline{b} \cup \{0\}$, exactly one of the atoms $i = j$ and $i \neq j$ is added to $L^{\mathcal{G}}$, and in the latter case either $i < j$ or $j < i$ is added too;

2. if the literal $\dim(a_k) = d_{a_k} \in L^{\mathcal{G}}$, then:

   (a) if $0 = d_{a_k} \in L^{\mathcal{G}}$ or $0 \equiv d_{a_k}$, then add $\{j_k = 0, e_k = \bot\}$ to $L^{\mathcal{G}}$;

   (b) if $0 < d_{a_k} \in L^{\mathcal{G}}$, then add $\{\text{s}(j_k) = d_{a_k}, \text{select}(a_k, j_k) = e_k, e_k \neq \bot\}$ to $L^{\mathcal{G}}$.

3. if $\{\dim(a) = i, i \leq j\} \subseteq L^{\mathcal{G}}$, then add $\{\text{select}(a, j) = \bot\}$ to $L^{\mathcal{G}}$.

There are two important observations. First, each new index j_k : INDEX ($k \in \{1, \ldots, m\}$) denotes the predecessor of the dimension of a_k, when a_k is guessed to be different of ε (if dimension of a_k is guessed to be ε, then j_k is also set to zero). Second, each new constant e_k : ELEM ($k \in \{1, \ldots, m\}$) denotes the result of reading the content of array a_k at position j_k.

These two observations together with the fact that the process described above to build $L^{\mathcal{G}}$ closely follows Definition 1.3.5 should make it clear that L is satisfiable iff there exist a set $L^{\mathcal{G}}$ which is $\mathcal{G}$-instantiation closed and $\mathcal{ADP}$-satisfiable too. The non-deterministic polynomial time result is obtained by a straightforward inspection of the process described above to build $L^{\mathcal{G}}$. □

It is straightforward to check that running the $\mathcal{E}$- and $\mathcal{G}$-instantiation modules in this order, one obtains a set of both $\mathcal{E}$- and $\mathcal{G}$-instantiation closed set of literals.

1.3.3 The Algorithm

The (non-deterministic) Algorithm 3 gives a decision procedure to solve the $\mathcal{ADP}$-satisfiability problem. Without loss of generality (see Section 1.3.2), we assume that L contains only flat literals.

Algorithm 3 The (extensible) decision procedure for $\mathcal{ADP}$

$\mathsf{T} \longleftarrow \{\mathcal{A}, \mathcal{P}\}$
function $DP_{\mathcal{ADP}}$ (L: *set of flat literals*)
  $L^{\mathcal{E}} \leftarrow \mathcal{E}$-*instantiation*$(L)$
  for all $L^{\mathcal{G}} \in \{\mathcal{G}\text{-}instantiation(L^{\mathcal{E}})\}$ **do**
    for all $T \in \mathsf{T}$ **do** $\rho_T \leftarrow DP_T(L^{\mathcal{G}}_T)$
    if $\bigwedge_{T \in \mathsf{T}}(\rho_T =$ *"satisfiable"*$)$ **then return** *"satisfiable"*
  end for
  return *"unsatisfiable"*
end function

The function DP_T, for $T \in \{\mathcal{ADP}, \mathcal{A}, \mathcal{P}\}$, denotes the decision procedure to solve the T-satisfiability problem, i.e. DP_T takes a set L of literals over the signature Σ_T and returns *"satisfiable"* when L is T-satisfiable, *"unsatisfiable"* otherwise. If L is a set of flat literals, then

$$L_T := \{\ell \mid \ell \in L \text{ is a } \Sigma_T\text{-literal}\},$$

where $T \in \{\mathcal{A}, \mathcal{P}\}$. So, for example, $L^{\mathcal{G}}_{\mathcal{P}}$ is the subset of the literals in $L^{\mathcal{G}}$ over the signature $\Sigma_{\mathcal{P}}$. The set T in Algorithm 3 contains the names of the theories for which a decision procedure is assumed available.

Let L be a set of flat literals over the signature $\Sigma_{\mathcal{ADP}}$ to be checked for $\mathcal{ADP}$-satisfiability. The decision procedure $DP_{\mathcal{ADP}}$ first computes the $\mathcal{E}$-instantiation $L^{\mathcal{E}}$ of L (recall from Lemma 1.3.4 that this can be done in linear time). Then, it start enumerating all possible $\mathcal{G}$-instantiations (the **for each** loop in Algorithm 3). If it is capable of finding a $\mathcal{G}$-instantiation $L^{\mathcal{G}}$ such that its literals $L^{\mathcal{G}}_{\mathcal{P}}$ over the signature $\Sigma_{\mathcal{P}}$ are $\mathcal{P}$-satisfiable and its literals $L^{\mathcal{G}}_{\mathcal{A}}$ over the signature $\Sigma_{\mathcal{A}}$ are $\mathcal{A}$-satisfiable, then $DP_{\mathcal{ADP}}$ returns the $\mathcal{ADP}$-satisfiability of the input set L of literals. Otherwise, if all possible $\mathcal{G}$-instantiations are enumerated and the test of the conditional in the body of the loop always fails, then $DP_{\mathcal{ADP}}$ returns the $\mathcal{ADP}$-unsatisfiability of the input set L of literals. Notice that, clauses (3) - (4) of Definition 1.3.5 have the effect of automatically synchronizing the decision procedures for $\mathcal{A}$ and $\mathcal{P}$, in the sense that the $\mathcal{G}$-complete set of literals $L^{\mathcal{G}}$ contains $i \neq j$ when it contains, e.g., $i < j$, and $i \neq j$ is passed to $DP_{\mathcal{A}}$ when

considering to $L^{\mathcal{G}}_{\mathcal{A}}$.

1.3.4 Correctness of the Procedure

The termination of $DP_{\mathcal{ADP}}$ is obvious, since the computation of $L^{\mathcal{E}}$ terminates (see the proof of Lemma 1.3.4) and there are only finitely many possible sets $L^{\mathcal{G}}$ to be considered in the **for each** loop of Algorithm 3.

The soundness and completeness of $DP_{\mathcal{ADP}}$ are consequences of the following combination lemma.

Lemma 1.3.7 (Combination)**.** *Let $L := L_{\mathcal{A}} \cup L_d \cup L_{\mathcal{P}}$ be an $\mathcal{E}$- and $\mathcal{G}$-instantiation closed set. Then, the following conditions are equivalent:*

(i) *L is satisfiable in a standard model of $\mathcal{ADP}$;*

(ii) *L is $\mathcal{ADP}$-satisfiable;*

(iii) *$L_{\mathcal{A}}$ is $\mathcal{A}$-satisfiable and $L_{\mathcal{P}}$ is $\mathcal{P}$-satisfiable.*

Proof. Since the implications (i) $\Rightarrow$ (ii) $\Rightarrow$ (iii) are trivial, it is sufficient to show that (iii) $\Rightarrow$ (i) to conclude the proof.

Let $\mathcal{M}'$ be a structure such that $\mathcal{M}' \models \mathcal{A}_e \cup L_{\mathcal{A}_e}$ and $\mathcal{N}$ be a structure such that $\mathcal{N} \models \mathcal{P} \cup L_{\mathcal{P}}$. Since $\mathcal{P}$ is complete, we are entitled to assume that $\mathcal{N}$ is the standard structure of natural numbers $\mathbb{N}$. We are now ready to build a standard model $\mathcal{M}$ for $\mathcal{ADP} \cup L$ out of $\mathcal{M}$ as follows. We take $\textsc{elem}^{\mathcal{M}}$ to be $\textsc{elem}^{\mathcal{M}'}$ and $\bot^{\mathcal{M}}$ to be $\bot^{\mathcal{M}'}$; the free constants occurring in L are interpreted as follows:

(A) for each constant $i : \textsc{index}$ occurring in $L_{\mathcal{P}}$, let $i^{\mathcal{M}} := i^{\mathcal{N}}$;

(B) for each constant $e : \textsc{elem}$ occurring in $L_{\mathcal{A}}$, let $e^{\mathcal{M}} := e^{\mathcal{M}'}$;

(C) for each constant $a : \textsc{array}$ occurring in $L_{\mathcal{A}}$, we define $a^{\mathcal{M}}$ to be the function $f_a : \mathbb{N} \longrightarrow \textsc{elem}^{\mathcal{M}}$ so defined:

$$f_a(n) := \begin{cases} \text{select}(a, i)^{\mathcal{M}'} & \text{if } n = i^{\mathcal{M}} \text{ for some } i \text{ occurring in } L_{\mathcal{P}}, \\ \bot^{\mathcal{M}} & \text{otherwise.} \end{cases}$$

The construction is well-defined since L is $\mathcal{G}$-instantiation closed, if two constants i_1 and i_2 of sort $\textsc{index}$ occurring in $L_{\mathcal{P}}$ are interpreted into the same element in $\mathcal{M}$, then $i_1^{\mathcal{N}} = i_2^{\mathcal{N}}$, the atom $i_1 = i_2$ is in $L_{\mathcal{P}}$ (and hence in $L_{\mathcal{A}}$) and so $\mathcal{M}' \models \text{select}(a, i_1) = \text{select}(a, i_2)$. Now, we show that for each $\ell \in L$, we have $\mathcal{M} \models \ell$. This is obvious for $\ell \in L_{\mathcal{P}}$ and for ℓ of the form $e_1 \bowtie e_2$, with $e_1, e_2 : \textsc{elem}$. We are left to consider the following cases:

(i) if ℓ has the form $\text{select}(a, i) = e$, with $a : \text{ARRAY}, i : \text{INDEX}, e : \text{ELEM}$, then $\mathcal{M} \models \ell$ because of (A), (B), (C);

(ii) if ℓ has the form $a_1 = a_2$, with $a_1, a_2 : \text{ARRAY}$, then $\mathcal{M} \models \ell$ because $a_1^{\mathcal{M}'} = a_2^{\mathcal{M}'}$, so $\text{select}(a_1, i)^{\mathcal{M}'} = \text{select}(a_2, i)^{\mathcal{M}'}$ for each constant $i : \text{INDEX}$ occurring in $L_{\mathcal{P}}$. Hence, $a_1^{\mathcal{M}} = a_2^{\mathcal{M}}$ by (C);

(iii) if ℓ has the form $\text{store}(a_1, i, e) = a_2$, with $a_1, a_2 : \text{ARRAY}, i : \text{INDEX}, e : \text{ELEM}$, then $\mathcal{M} \models \ell$ by considering an argument similar to that used for case (ii);

(iv) if ℓ has the form $a_1 \neq a_2$, with $a_1, a_2 : \text{ARRAY}$, then $\mathcal{M} \models \ell$ because

$$\{\text{select}(a_1, i) = e_1, \text{select}(a_2, i) = e_2, e_1 \neq e_2\} \subseteq L_{\mathcal{A}}$$

by Definition 1.3.3 of $\mathcal{E}$-instantiation closed set of literals and $\mathcal{M}' \models L_{\mathcal{A}}$ and hence $\text{select}(a_1, i)^{\mathcal{M}} \neq \text{select}(a_2, i)^{\mathcal{M}}$ because of (i). As a consequence, we have $a_1^{\mathcal{M}} \neq a_2^{\mathcal{M}}$.

(v) if ℓ has the form $\dim(a) = i$, then we consider two subcases:

- if $\{i = 0\} \subseteq L_{\mathcal{P}}$, then it is sufficient to prove that for each integer n, $f_a(n)$ is equal to $\bot^{\mathcal{M}}$ where $f_a = a^{\mathcal{M}}$. If $n = j^{\mathcal{M}}$ for some constant $j : \text{INDEX}$ such that

$$\{i < j\} \subseteq L_{\mathcal{P}} \quad \text{or} \quad \{i = j\} \subseteq L_{\mathcal{P}},$$

  then, since L is $\mathcal{G}$-instantiation closed, $\{\text{select}(a, j) = \bot\} \subseteq L_{\mathcal{A}_e}$ hence $f_a(n) = \bot^{\mathcal{M}}$ by (C); otherwise, $f_a(n) = \bot^{\mathcal{M}}$ by (C).
- if $\{i \neq 0\} \subseteq L_{\mathcal{P}}$, then for each integer $n \geq i^{\mathcal{M}}$, $f_a(n) = \{\bot^{\mathcal{M}}\}$ by a similar argument to the one used for the previous subcase. In fact, we observe that since L is $\mathcal{G}$-instantiation closed, $\text{s}(j) = i$ is in $L_{\mathcal{P}}$ for some constant $j : \text{INDEX}$, and also $\text{select}(a, j) = e, e \neq \bot$ must be in $L_{\mathcal{A}_e}$, therefore the thesis follows from (B), (C) and (i).

□

Now, we are able to state and prove the correctness of $DP_{\mathcal{ADP}}$.

Theorem 1.3.8. *$DP_{\mathcal{ADP}}$ is a decision procedure for the $\mathcal{ADP}$-satisfiability problem, i.e. for any set L of flat literals, L is $\mathcal{ADP}$-satisfiable iff $DP_{\mathcal{ADP}}(L)$ returns "satisfiable". Furthermore, $DP_{\mathcal{ADP}}$ decides the satisfiability problem in the standard models of $\mathcal{ADP}$.*

Proof. If L is $\mathcal{ADP}$-satisfiable, then it is obvious that $DP_{\mathcal{ADP}}(L)$ returns *"satisfiable"*. We are left with the task of proving that the converse implication holds. We will prove that when $DP_{\mathcal{ADP}}(L)$ returns *"satisfiable"*, then L is satisfiable in a standard model

of $\mathcal{ADP}$. If $DP_{\mathcal{ADP}}(L)$ returns *"satisfiable"*, then $DP_{\mathcal{ADP}}$ has found a $\mathcal{G}$-instantiation $L^{\mathcal{G}} := L^{\mathcal{G}}_{\mathcal{A}} \cup L^{\mathcal{G}}_{d} \cup L^{\mathcal{G}}_{\mathcal{P}}$ of $L^{\mathcal{E}}$ at some iteration of the **for each** loop in Algorithm 3. The set $L^{\mathcal{G}}$ is such that

$$L^{\mathcal{G}}_{\mathcal{A}} \text{ is } \mathcal{A}\text{-satisfiable and } L^{\mathcal{G}}_{\mathcal{P}} \text{ is } \mathcal{P}\text{-satisfiable.}$$

From these two facts, the existence of a standard $\mathcal{ADP}$-model of $L^{\mathcal{G}}$ immediately follows by using Lemma 1.3.7 above. □

Chapter 2

Combining Theories over Disjoint Signatures

In the previous chapter different methods for deciding fragments of theories were described. The last of the given examples deals with a typical combination problem, since the involved theory can be decomposed as the union of different subtheories and the proposed procedure exploits the availability of two other different decision procedures. This example shows how naturally the need of combining such different techniques arises; the present chapter will be entirely devoted to the problem of combination.

Suppose we are now given two first-order theories T_1 and T_2 over the signatures Σ_1 and Σ_2 respectively (at the moment we do not ask Σ_1 and Σ_2 to be disjoint). If we are able to solve the constraint satisfiability problem for both T_1 and T_2, we wonder when it is possible to solve the same problem for $T_1 \cup T_2$. In order to be able to re-use any existing decision procedure, it is useful to adopt a so-called *black-box approach*. We assume that a prover $\mathcal{P}_1$ solves the problem for the theory T_1 and a prover $\mathcal{P}_2$ solves the problem for the theory T_2. The provers $\mathcal{P}_1$ and $\mathcal{P}_2$ can exchange information only externally, according to a protocol to be specified: in any case, $\mathcal{P}_1$ and $\mathcal{P}_2$ cannot be internally modified.

One of the most simple combination methodology for the satisfiability decision problems compliant with the black-box approach is represented by the *Nelson-Oppen procedure* (see [69]): it was originally designed only for disjoint signatures case (i.e. the equality is the only shared predicate symbol).

2.1 The Nelson-Oppen Combination Schema

The Nelson-Oppen procedure can be summarized essentially into two steps: the first is the *purification* and the second is the *exchange loop*.

Purification The preprocessing step consists in the transformation of the initial finite set Γ of $\Sigma_1 \cup \Sigma_2 \cup A$-ground literals into the set

$$\Gamma_1 \cup \Gamma_2 \tag{2.1}$$

where Γ_1 are $\Sigma_1 \cup A$-ground literals and Γ_2 are $\Sigma_2 \cup A$-ground literals. We remark that this transformation preserves the equisatisfiability; moreover, the purification is linear (equation like $c = t$ for new constants c and alien subterms t are successively added).

Exchange loop Whenever the prover $\mathcal{P}_i$ ($i \in \{1, 2\}$) finds a disjunction C of ground $\Sigma_0 \cup A$-atoms (where $\Sigma_0 := \Sigma_1 \cap \Sigma_2$) such that $\Gamma_i \cup \{\neg C\}$ is unsatisfiable modulo T_i, C is added to Γ_j ($j \in \{1, 2\}, j \neq i$) if not already present.

Alternatively, one can limit the exchange to atoms instead of clauses: obviously case splitting and backtracking mechanisms are required. However, if the theories T_i are Σ_0-convex, the atoms exchange becomes deterministic, as in the original Nelson-Oppen case. Following [85], a theory T on the signature Σ is said to be Σ_0-*convex* ($\Sigma_0 \subseteq \Sigma$) iff whenever $T \cup \Gamma \models A_1 \vee \cdots \vee A_n$ (for $n \geq 1$ and for ground $\Sigma_0 \cup A$-atoms $A_1, \ldots, A_n$), there is $k \in \{1, \ldots, n\}$ such that $T \cup \Gamma \models A_k$.

The exchange loop returns *"unsatisfiable"* if Γ_1 (or Γ_2) eventually becomes unsatisfiable modulo T_1 (or T_2 respectively). Otherwise, it returns *"satisfiable"* if the loop terminates without finding any inconsistency. The deterministic Nelson-Oppen procedure is guaranteed to be terminating and complete under the following assumption: (*i*) Σ_1 and Σ_2 are disjoint; (*ii*) the theories T_1 and T_2 are Σ_0-convex; (*iii*) they admit only non trivial models, i.e. models with cardinality bigger than 1. In the non-deterministic case, we can drop assumption (*iii*) and weaken the convexity hypothesis (*ii*) to the hypothesis that the theories T_1 and T_2 are stably infinite. We say that a theory T over the signature Σ is *stably infinite* iff for any quantifier-free Σ-formula φ satisfiable modulo T, there exists a model of T whose domain is infinite and which satisfies φ.

It is possible to show (see [17]) that a theory which is convex and admits only trivial models is also stably infinite. Briefly, we recall that a set S of literals is convex in a theory T if $T \cup S$ does not entail any disjunction of equalities without entailing one of the equalities itself. A theory is *convex* if every set of literals in the language of the theory is convex.

What discussed above shows that stable infiniteness is a weaker property characterizing the theories that can be combined according to the Nelson-Oppen schema; the key observation about the requirements for the completeness of this scheme seems to be the satisfiability of constraints in infinite models of the component theories. What would hap-

pen in case we drop the assumption about the cardinality of models for the component theories, maintaining the hypothesis of disjoint signatures?

2.2 Undecidability Results

We start introducing the following

Definition 2.2.1. Let T be a Σ-theory.

- T is $\exists$-*decidable* iff the universal fragment of T is decidable; equivalently if the constraint satisfiability problem for T is decidable;[1]
- T is $\exists_\infty$-*decidable* iff it is $\exists$-decidable and moreover it is decidable whether any Σ-constraint Γ is satisfiable in an *infinite* model of T.

Some few remarks about Definition 2.2.1. The requirement for a Σ-theory T to be $\exists$-decidable is precisely the requirement that the constraint satisfiability problem for T is decidable, which is equivalent to the requirement of deciding whether a universal Σ-formula is entailed by the axioms of T. Moreover it is straightforward to see that, for stably infinite theories, $\exists$-decidability is equivalent to $\exists_\infty$-decidability. To illustrate the interest of studying the decidability of satisfiability in the infinite models of a theory, we state the following

Theorem 2.2.2. *Let T_i be a Σ_i-theory (for $i = 1, 2$) and let the signatures Σ_1, Σ_2 be disjoint. If T_1 is $\exists$-decidable but it is not $\exists_\infty$-decidable and if T_2 is consistent, $\exists$-decidable but does not admit finite models, then the constraint satisfiability for $T_1 \cup T_2$ is undecidable.*

Proof. We simply show that a Σ_1-constraint Γ is $T_1 \cup T_2$-satisfiable iff it is satisfiable in an infinite model of T_1. One side is obvious; for the other side, pick infinite models $\mathcal{M}_1$ of $T_1 \cup \Gamma$ and $\mathcal{M}_2$ of T_2 (the latter exists by consistency of T_2). By Löwenheim-Skolem theorem, we can assume that both models are countable, i.e. that they have the same support (up to isomorphism). But then, we can simply put together the interpretations of functions and predicate symbols and get a model of $T_1 \cup T_2 \cup \Gamma$. □

We notice that there are many theories which are $\exists$-decidable and have only infinite models. One such theory is Presburger Arithmetic, another one is the theory of acyclic lists (see [73]). If there exist $\exists$-decidable theories that are not $\exists_\infty$-decidable, Theorem 2.2.2 implies that *there exist theories which are $\exists$-decidable and whose union is not $\exists$-decidable.*

[1] According to our definitions (see Section refsec:preliminaries), a Σ-constraint is a finite conjunction of ground Σ^a literals.

2.2.1 The Theory TM_∞

In this section we want to exhibit a $\exists$-decidable theory that is not $\exists_\infty$-decidable.

Let Σ_{TM_∞} be the signature containing (in addition to the equality predicate) the following (infinite) set of propositional letters $\{P_{(e,n)} \mid e, n \in \mathbb{N}\}$. Consider the propositional letter $P_{(e,n)}$: we regard e as the index (i.e. the code) of a Turing Machine and n as the input to the Turing machine identified by e (this coding is possible because of basic results about Turing machines - see, e.g., [72]). We indicate by $k : \mathbb{N} \times \mathbb{N} \to \mathbb{N} \cup \{\infty\}$ the (non-computable) function associating to each pair (e, n) the number $k(e, n)$ of computation steps of the Turing Machine e on the input n. We write $k(e, n) = \infty$ when the computation does not halt. The axioms of the theory TM_∞ are the universal closures of the following formulae:

$$P_{(e,n)} \rightarrow \bigvee_{i<j\leq m} x_i = x_j, \qquad \text{if } k(e, n) < m. \tag{2.2}$$

Two observations are in order. First, the property "being an axiom of TM_∞" is decidable, because the ternary predicate $k(e, n) < m$ is recursive. Indeed, it is sufficient to run the Turing Machine e on input n and wait at most m computation steps to verify whether e halts. Second, the consequent of implication (2.2) is an *at-most cardinality constraint*, i.e. it is a formula of the form

$$\bigvee_{i\neq j} x_i = x_j \tag{2.3}$$

where x_i, x_j are (implicitly universally quantified) distinct variables for $i, j = 1, \ldots, n$, which constrain the domain of any model to contain at most $n - 1$ elements. Thus, axioms of the form (2.2) tells us that if the Turing Machine e halts in at most m steps, then the cardinality of the domains of a model is bounded by m. These properties allow us to state and prove the following key result:

Proposition 2.2.3. *The theory TM_∞ is $\exists$-decidable but it is not $\exists_\infty$-decidable.*

Proof. To show that the theory is $\exists$-decidable, consider a constraint Γ over the signature $\Sigma^{\underline{a}}_{TM_\infty}$. First, guess an arrangement Γ_0 for the constants $\underline{a}$ and check the set of equations and inequations from $\Gamma \cup \Gamma_0$ for consistency in the pure theory of equality. Then, if the satisfiability check succeeds, Γ_0 explicitly gives the minimum cardinality m for $\Gamma \cup \Gamma_0$ to be satisfied. Clearly, $\Gamma \cup \Gamma_0$ is unsatisfiable if it contains both $P_{(e,n)}$ and $\neg P_{(e,n)}$. If this is not the case, we still have to consider the constraints represented by axiom (2.2), which states that if a literal of the kind $P_{(e,n)}$ is in a Σ_{TM_∞}-constraint, such a constraint can be only satisfied in a model whose cardinality is at most $k(e, n)$. Thus, if $P_{(e,n)} \in \Gamma \cup \Gamma_0$, we only need to check that $m \leq k(e, n)$, which can be effectively done since the ternary predicate $k(e, n) < m$ is recursive.

To see that TM_∞ is not $\exists_\infty$-decidable, notice that the constraint $\{P_{(e,n)}\}$ is TM_∞-satisfiable in an infinite structure iff $k(e,n) = \infty$. In turn, this is equivalent to check whether the computation of the Turing Machine e on the input n does not terminate, which is obviously undecidable, being the complement of the Halting problem. □

The theory TM_∞ is defined on an infinite signature. However, it is possible to introduce two theories TM_ω and $TM_{\forall\omega}$ over finite signatures, with the same characteristics as TM_∞ as far decidability in finite and infinite models is concerned, and such that $TM_{\forall\omega}$ is also universal.

2.2.2 Refined Undecidability Results

Here we refine Proposition 2.2.3 by avoiding the use of an infinite signature like Σ_{TM_∞}.

A Variant of the Theory TM_∞: TM_ω

Consider the signature Σ_{TM_ω} consisting of a constant symbol 0, a unary predicate symbol P, and two binary predicate symbols $<$ and S. The axioms of the theory TM_ω are the universal closures of the following formulae:

$$\neg x < x \tag{2.4}$$

$$x < y \wedge y < z \rightarrow x < z \tag{2.5}$$

$$x < y \vee x = y \vee y < x \tag{2.6}$$

$$0 = x \vee 0 < x \tag{2.7}$$

$$S(x,y) \leftrightarrow (x < y \wedge \neg\exists z(x < z \wedge z < y)) \tag{2.8}$$

$$x < y \rightarrow \exists z(S(x,z) \wedge (z < y \vee z = y)) \tag{2.9}$$

$$P(x_a) \wedge S(0,x_1) \wedge \cdots \wedge S(x_{a-1},x_a) \wedge S(x_a,x_{a+1}) \wedge \cdots \wedge S(x_{a+m-1},x_{a+m}) \rightarrow \bot, \text{ if } a = \langle e,n\rangle \text{ and } k(e,n) < m \tag{2.10}$$

$$P(x) \wedge P(y) \rightarrow x = y \tag{2.11}$$

where $\langle\cdot,\cdot\rangle$ is a primitive recursive coding for pairs, i.e. a computable bijection from $\mathbb{N}\times\mathbb{N}$ to $\mathbb{N}$ (we are guaranteed that the primitive recursive coding function $\langle\cdot,\cdot\rangle$ exists because of basic results about primitive recursive functions, see again [72] for details).

Two remarks are in order. First, because of axioms (2.4)-(2.9), any model of TM_ω is a Σ_{TM_ω}-structure endowed with a strict linear order with minimum element; moreover, every element (except the last one, if any) has an immediate successor. Second, finite models of TM_ω are initial segments of $\mathbb{N}$, whereas infinite models admit $\mathbb{N}$ as an initial segment. It is also worth to consider the last two axioms of TM_ω:

– axiom (2.10) means that, given a Turing Machine e, its input n, and the coding h of the pair (e, n), the atom $P(h)$ can be satisfied only in models of cardinality at most $h + k(e, n) + 1$;

– axiom (2.11) states that there is no model satisfying two atoms $P(a)$ and $P(b)$ if $a \neq b$. This axiom simplifies the technical development below.

Proposition 2.2.4. *The theory TM_ω is $\exists$-decidable but it is not $\exists_\infty$-decidable.*

Proof. Let Γ be a constraint over the signature $\Sigma^{\underline{c}}$. We define a *TM_ω-guessing* $\mathcal{G}$ on Γ as a finite set of ground $\Sigma^{\underline{c}}$-literals such that (i) $\Gamma \subseteq \mathcal{G}$ and (ii) for every pair of distinct constants $a, b \in \underline{c} \cup \{0\}$, either $a < b \in \mathcal{G}$, $b < a \in \mathcal{G}$, or $a = b \in \mathcal{G}$. Clearly, Γ is TM_ω-satisfiable iff some TM_ω-guessing $\mathcal{G}$ on Γ is TM_ω-satisfiable. As a consequence, we consider the problem of deciding the satisfiability of a TM_ω-guessing $\mathcal{G}$.

Given such a TM_ω-guessing $\mathcal{G}$, notice that for $\mathcal{G}$ to be consistent, the equations belonging to $\mathcal{G}$ must induce an equivalence relation on the constants occurring in it. Let us pick a representative constant for each equivalence class (with 0 being the representative for its class). Furthermore, let us replace all terms in $\mathcal{G}$ with the representative constants of their equivalence classes. After this transformation, without loss of generality, we can delete all equalities and inequalities from $\mathcal{G}$. Let us denote the result of such transformations still with $\mathcal{G}$. For each negative literal $\neg S(c_1, c_2)$ in $\mathcal{G}$ such that $c_1 < c_2 \in \mathcal{G}$ and $\{c_1 < a, a < c_2\} \not\subseteq \mathcal{G}$ for some constant a, we add $c_1 < c_3$, $c_3 < c_2$ to $\mathcal{G}$, where c_3 is a fresh constant. After this step, the literals of the form $a < b$ that are in $\mathcal{G}$ should put the constants in $\mathcal{G}$ in a linear order, i.e.

$$c_0 < c_1 < c_2 < \cdots < c_{s-1} < c_s.$$

Here c_0 is 0, c_i and c_j are distinct for $i \neq j$, and only inequalities of the form $c_i < c_j$ $(0 \leq i < j \leq s)$ are in $\mathcal{G}$ (if it is not so, it is because $\mathcal{G}$ contains inconsistencies from the point of view of the theory of strict linear orders with first element). Furthermore, if $S(c_i, c_j) \in \mathcal{G}$, then $j = i + 1$; otherwise, $\mathcal{G}$ is inconsistent. Thus, all literals in $\mathcal{G}$ (not containing P) are satisfied, for instance, in the linearly ordered structure containing s elements. Clearly, $\mathcal{G}$ is inconsistent if it contains a pair of complementary literals, so we suppose this is not the case. Because of axiom (2.11), it can contain at most one positive literal involving the predicate P and, at this point, $\mathcal{G}$ can be unsatisfiable only because of the presence of such a literal. Let this literal be $P(c_a)$; for $m = s - a$, the following inequalities

$$0 < c_1 < c_2 < \cdots < c_{a-1} < c_a < c_{a+1} < \cdots < c_{a+m}$$

are in $\mathcal{G}$. If there is $j < a$ such that $S(c_j, c_{j+1}) \notin \mathcal{G}$, then $\mathcal{G}$ is satisfiable. To see this, consider a non standard model of Arithmetic and interpret $c_{j+1}, \ldots, c_{a+m}$ as elements

greater than all the standard natural numbers: if the predicate P is interpreted as the singleton subset formed by (the interpretation of) c_a, axiom (2.10) is true because the a-th successor of 0 is not in P. On the other hand, if $\{S(0, c_1), S(c_1, c_2), \ldots, S(c_{a-1}, c_a)\} \subseteq \mathcal{G}$, then $\mathcal{G}$ is satisfiable iff $m \leq k(e, n)$ where $a = \langle e, n \rangle$. Since $\langle e, n \rangle$ and the relation $m \leq k(e, n)$ are computable, we have a decision procedure for the constraint satisfiability problem in TM_ω.

To see that TM_ω is not $\exists_\infty$-decidable, notice that the TM_ω-constraint

$$\{S(0, c_1), S(c_1, c_2), \ldots, S(c_{a-1}, c_a), P(c_a)\},$$

(for $a = \langle e, n \rangle$) is TM_ω-satisfiable in an infinite structure iff the computation of the Turing Machine e over the input n diverges, which is obviously undecidable. □

A Variant of the Theory TM_ω: $TM_{\forall\omega}$

Theory TM_ω is not universal. However, it is not difficult to find an alternative axiomatization over a finite signature $\Sigma_{TM_{\forall\omega}}$ so to define a universal theory $TM_{\forall\omega}$ which is $\exists$-decidable but it is not $\exists_\infty$-decidable.

The main ideas used in the definition of $TM_{\forall\omega}$ are the following: (a) we replace the binary predicate symbol S of TM_ω with a unary function symbol s; (b) we re-use axioms (2.4)-(2.7) and (c) we introduce new axioms to constrain the unary symbol s to be such that $s(x) = x$ holds iff the order $<$ has a last element which is precisely x.

In more detail, the signature of the theory $TM_{\forall\omega}$ coincides with the signature of the theory TM_ω with the exception that the binary predicate symbol S is replaced by the unary function symbol s. The axioms for $TM_{\forall\omega}$ are divided into three groups. In the first group we have axioms (2.4)-(2.7) and in the second group the following ones:

$$x = s(x) \vee x < s(x) \tag{2.12}$$

$$\neg(x < y \wedge y < s(x)) \tag{2.13}$$

$$x < y \rightarrow s(x) < y \vee s(x) = y \tag{2.14}$$

$$s(x) = x \wedge x < y \rightarrow \bot \tag{2.15}$$

Axioms (2.12)-(2.15), together with (2.4)-(2.7), state that the function s behaves like a successor function with the exception that fixed points of s are allowed (see (2.12)). Axiom (2.15) however says that the only possible fixed point of the function s is the maximum element with respect to the order $<$.

In addition to the axioms of the first two groups (namely (2.4)-(2.7) and (2.12)-(2.15)),

in the third group, we have axiom (2.11) and the following one (which replaces (2.10)):

$$P(s^a(0)) \wedge s^{a+m-1}(0) < s^{a+m}(0) \rightarrow \bot \qquad \text{if } a = \langle e, n \rangle \text{ and } k(e, n) < m \tag{2.16}$$

Proposition 2.2.5. *The theory $TM_{\forall\omega}$ is $\exists$-decidable but it is not $\exists_\infty$-decidable.*

Proof. The argument is similar to the argument used in the proof of Proposition 2.2.4, with the proviso that the constraint Γ should be flattened. Moreover, once the linear order

$$c_0 < c_1 < \cdots < c_{s-1} < c_s$$

is obtained, we notice that if the literal $c_j = s(c_i)$ belongs to the guessing $\mathcal{G}$, then this is inconsistent if $j \neq i + 1$ or if $j = i \neq s$. The other steps in the proof of Proposition 2.2.4 remain unchanged. □

Now we are able to prove the following

Theorem 2.2.6. *There exist $\exists$-decidable universal theories over finite and disjoint signatures, whose union is not $\exists$-decidable.*

This result follows from Theorem 2.2.2 and the fact that $TM_{\forall\omega}$ is $\exists$-decidable but not $\exists_\infty$-decidable. It is still an open problem to find an $\exists$-decidable, non $\exists_\infty$-decidable theory (in a finite signature), which is universal and *finitely axiomatized*.

2.3 Decidability Results

Notwithstanding the negative result implied by Theorems 2.2.2 and 2.2.6, we observe that, when both T_1 and T_2 are $\exists_\infty$-decidable, we are close to get the decidability of constraint satisfiability in $T_1 \cup T_2$. To understand why, recall the following well-known fact.

Lemma 2.3.1. *Let Λ be a set of first-order sentences. If Λ does not admit infinite models, then there must exist an integer $N > 0$ such that, for each model $\mathcal{M}$ of Λ, the cardinality of the support set of $\mathcal{M}$ is bounded by N.*

For a proof, the interested reader is referred to any introductory textbook about model theory (see, e.g., [87]). The key idea is to apply compactness to infinitely many "at-least-n-elements" constraints (these are the constraints expressed by the formulae $\exists x_1, \ldots, x_n \bigwedge_{i \neq j} x_i \neq x_j$). It is interesting to notice that the above bound on the cardinality of finite models can be effectively computed for $\exists$-decidable theories.

Lemma 2.3.2. *Let T be an $\exists$-decidable Σ-theory; whenever it happens that a given Σ-constraint Γ is not satisfiable in an infinite model, one can compute a natural number N such that all models of $T \cup \Gamma$ have cardinality at most N.*

Proof. For $h = 2, 3, \ldots$, add the following set $\delta_h := \{c_i \neq c_j \mid 1 \leq i < j \leq h\}$ of literals to $T \cup \Gamma$, where the constants $c_1, \ldots, c_h$ are fresh. Actually, the literals in δ_h are simply the Skolemization of the "at-least-h-elements" constraint. Clearly, if $T \cup \Gamma \cup \delta_h$ is unsatisfiable, then we get a bound for the cardinality of the models of $T \cup \Gamma$. Since, by Lemma 2.3.1, such a bound exists, the process eventually terminates. □

It could be useful to notice that there is a subtle point here: Lemma 2.3.2 applies to all $\exists$-decidable theories, but it is really useful only for $\exists_\infty$-decidable theories, because only for these theories the hypothesis "Γ in not satisfiable in an infinite model of T" can be effectively checked.

Definition 2.3.3. An $\exists_\infty$-decidable Σ-theory T is said to be *strongly $\exists_\infty$-decidable* iff for any finite Σ-structure $\mathcal{A}$, it is decidable whether $\mathcal{A}$ is a model of T.

It is not difficult to find strongly $\exists_\infty$-decidable theories. For example, any finitely axiomatizable $\exists_\infty$-decidable Σ-theory with a finite Σ is strongly $\exists_\infty$-decidable, since it is sufficient to check the truth of the axioms for finitely many valuations. Now, we are in the position to state and prove the following modularity property for $\exists_\infty$-decidable theories.

Theorem 2.3.4. *Let T_i be a strongly $\exists_\infty$-decidable Σ_i-theory (for $i = 1, 2$) such that Σ_1, Σ_2 are finite and disjoint. Then the combined theory $T_1 \cup T_2$ is $\exists$-decidable.*

Proof. Let Γ be a finite set of ground $\Sigma_1 \cup \Sigma_2 \cup \underline{a}$-literals (where $\underline{a}$ is a finite set of free constants). By well-known means (see, e.g., [12]), we can obtain an equisatisfiable set $\Gamma_1 \cup \Gamma_2$ such that Γ_i contains only $\Sigma_i^{\underline{a}}$-symbols, for $i = 1, 2$. Let Γ_0 be an *arrangement* of the constants $\underline{a}$, i.e. a finite set of literals such that either $a_i = a_j \in \Gamma_0$ or $a_i \neq a_j \in \Gamma_0$, for $i \neq j$ and $a_i, a_j \in \underline{a}$. Clearly, $\Gamma_1 \cup \Gamma_2$ is satisfiable iff $\Gamma_1 \cup \Gamma_0 \cup \Gamma_2$ is satisfiable for some arrangement Γ_0 of the constants $\underline{a}$. From the fact that theories T_1, T_2 are both $\exists_\infty$-decidable, the following case analysis can be *effectively* performed:

- If $\Gamma_0 \cup \Gamma_i$ is satisfiable in an infinite model of T_i (for both $i = 1, 2$), then $\Gamma_0 \cup \Gamma_1 \cup \Gamma_2$ is satisfiable in an infinite model of $T_1 \cup T_2$ by the standard argument underlying the correctness of the Nelson-Oppen combination schema (see, e.g., [86, 44]).

- If $\Gamma_0 \cup \Gamma_i$ is unsatisfiable in any infinite model of T_i (for either $i = 1$ or $i = 2$), then (by Lemma 2.3.2) we can effectively compute an integer $N > 0$ such that each model $\mathcal{M}$ of $T \cup \Gamma_i \cup \Gamma_0$ has cardinality less than N. Hence, it is sufficient to exhaustively search through $\Sigma_1 \cup \Sigma_2 \cup \underline{a}$-structures up to cardinality N. The number of these structures is finite because Σ_1 and Σ_2 are finite and, by Definition 2.3.3, it is possible to effectively check whether each such a structure is a model of T_1 and T_2, and hence also of $T_1 \cup T_2 \cup \Gamma_0 \cup \Gamma_1 \cup \Gamma_2$. If a model is found, the procedure returns *"satisfiable"*, otherwise another arrangement Γ_0 (if any) is considered.

□

Theorem 2.3.4, which can be easily generalized to the combination of $n > 2$ theories, raises naturally the question if there is a practical sufficient condition for a theory to be strongly $\exists_\infty$-decidable. Clearly, stably infinite $\exists$-decidable theories are $\exists_\infty$-decidable. We are now looking for more interesting examples. In Section 2.4 we will show that, whenever a finitely axiomatized theory T admits a rewrite-based decision procedure for its constraint satisfiability problem [5, 4], T is not only $\exists$-decidable but also strongly $\exists_\infty$-decidable.

2.4 Combination by Superposition

The task of this section rests on showing that (under suitable assumptions) rewrite-based methods give practical sufficient conditions for a theory to be strongly $\exists_\infty$-decidable. First, we need to introduce some technical definitions. In Section 2.4.2, we recall some basic notions underlying the superposition calculus [71] and we introduce superposition modules as suitable abstractions for the subsequent technical development. Then, in Section 2.4.3, we introduce the notion of invariant superposition modules and, in Section 2.4.4, we show that they can generate an "at-most" cardinality constraint (see (2.3) in Section 2.2.1) whenever a theory does not admit infinite models. Last, in Section 2.4.5, we describe how to combine rewrite-based procedures (see, e.g., [4, 5]) with Satisfiability Modulo Theory (SMT) tools, such as [32, 6, 36, 43], in order to obtain automatic methods to solve constraint satisfiability problems involving theories admitting only finite models (e.g., enumerated data-types).

2.4.1 Superposition Calculus: an Overview

From now on, we consider only universal, finitely axiomatized theories, whose signatures are finite. Without loss of generality, we assume that signatures contain only function symbols, because any atom $P(t_1, \ldots, t_n)$ with predicate symbol P other than equality can be written as an equation $p(t_1, \ldots, t_n) = \mathit{true}$, where p is a fresh function symbol and true a fresh constant symbol, and this transformation preserves satisfiability (see, e.g., [71]).

In the following, $=$ denotes equality, $\equiv$ denotes identity, l, r, u, t are terms, v, w, x, y, z are variables, all other lower case letters are constant or function symbols. A fundamental feature of the Superposition Calculus (from now on, $\mathcal{SP}$) is the usage of a *term reduction ordering (TRO)* $\prec$ (see, e.g., [11]) which is total on ground terms. The ordering $\prec$ is extended to literals in such a way that only maximal sides of maximal instances of literals are considered when applying the expansion rules of Figure 2.1. The most commonly

used orderings are the *Knuth-Bendix ordering (KBO)* and the *lexicographic path ordering (LPO)*.

A clause C is *redundant* with respect to a set S of clauses if either $C \in S$ or S can be obtained from $S \cup \{C\}$ by a sequence of application of the contraction rules of Figure 2.2. An inference is *redundant* with respect to a set S of clauses if its conclusion is redundant with respect to S. A set S of clauses is *saturated* with respect to $\mathcal{SP}$ if every inference of $\mathcal{SP}$ with a premise in S is redundant with respect to S. A *derivation* is a sequence $S_0, S_1, \ldots, S_i, \ldots$ of sets of clauses where at each step an inference of $\mathcal{SP}$ is applied to generate and add a clause (see expansion rules in Figure 2.1) or to delete or reduce a clause (see contraction rules in Figure 2.2). A derivation is characterized by its *limit*, defined as the set of persistent clauses $S_\infty = \bigcup_{j \geq 0} \bigcap_{i > j} S_i$. A derivation $S_0, S_1, \ldots, S_i, \ldots$ with limit S_∞ is *fair* with respect to $\mathcal{SP}$ if for every inference in $\mathcal{SP}$ with premises in S_∞, there is some $j \geq 0$ such that the inference is redundant in S_j.

Theorem 2.4.1 (Nieuwenhuis and Rubio [71])**.** *If $S_0, S_1, \ldots$ is a fair derivation of $\mathcal{SP}$, then* (i) *its limit S_∞ is saturated with respect to $\mathcal{SP}$,* (ii) *S_0 is unsatisfiable iff the empty clause is in S_j for some j, and* (iii) *if such a fair derivation is finite, i.e. it is of the form $S_0, \ldots, S_n$, then S_n is saturated and logically equivalent to S_0.*

We say that $\mathcal{SP}$ is *refutationally complete* since it is possible to derive the empty clause with a finite derivation from an unsatisfiable set of clauses (see *(ii)* of Theorem 2.4.1). The proof of this theorem (see [71], but also [13, 14]) relies on the creation of a convergent rewriting system from the set of all the ground instances of a saturated set of clauses. If the empty clause does not belong to the saturation, a model for S_0 can be built from the set of all the ground terms identified by the equivalence relation deriving from the rewriting rules. The clauses of the derivation concurring to the creation of these rewriting rules are called *productive*. The precise definition of productive clause will be given in the next Section 2.4.2, and it will turn to be essential also for the results presented hereafter.

The rewriting based methodology for T-satisfiability consists of two phases:

1. *Flattening:* all ground literals are flattened by introducing new constants, yielding an equisatisfiable set of *flat* literals.

2. *Ordering selection and termination:* any fair derivation of $\mathcal{SP}$ is shown to be finite when applied to the union of the set of flat literals together and the set of axioms of T, provided that the TRO $\prec$ satisfies a few properties depending on T.

If T is a theory to which the rewriting-based approach can be applied, a T-satisfiability procedure can be built by implementing flattening (this can be done once and for all), and by using a prover mechanizing $\mathcal{SP}$ with a suitable TRO $\prec$. If the final set of clauses

Superposition	$\dfrac{\Gamma \Rightarrow \Delta, l[u'] = r \quad \Pi \Rightarrow \Sigma, u = t}{(\Gamma, \Pi \Rightarrow \Delta, \Sigma, l[t] = r)\sigma}$	$(i), (ii), (iii), (iv)$
Paramodulation	$\dfrac{\Gamma, l[u'] = r \Rightarrow \Delta \quad \Pi \Rightarrow \Sigma, u = t}{(l[t] = r, \Gamma, \Pi \Rightarrow \Delta, \Sigma)\sigma}$	$(i), (ii), (iii), (iv)$
Reflection	$\dfrac{\Gamma, u' = u \Rightarrow \Delta}{(\Gamma \Rightarrow \Delta)\sigma}$	(v)
Eq. Factoring	$\dfrac{\Gamma \Rightarrow \Delta, u = t, u' = t'}{(\Gamma, t = t' \Rightarrow \Delta, u = t')\sigma}$	$(i), (vi)$

Legenda: a clause $\neg A_1 \vee \cdots \vee \neg A_n \vee B_1 \vee \cdots \vee B_n$ is written in sequent style as $\{A_1, \ldots, A_n\} \Rightarrow \{B_1, \ldots, B_m\}$ (where the A_i's and B_j's are literals), equality is the only predicate symbol, σ is the most general unifier of u and u', u' is not a variable in *Superposition* and *Paramodulation*, L is a literal, and the following hold:
(i) $u\sigma \not\preceq t\sigma$, *(ii)* $\forall L \in \Pi \cup \Sigma : (u = t)\sigma \not\preceq L\sigma$, *(iii)* $l[u']\sigma \not\preceq r\sigma$, *(iv)* $\forall L \in \Gamma \cup \Delta : (l[u'] = r)\sigma \not\preceq L\sigma$, *(v)* for all $L \in \Gamma \cup \Delta : (u' = u)\sigma \not\prec L\sigma$, and *(vi)* for all $L \in \Gamma : u\sigma \not\preceq L\sigma$, and for all $L \in \{u' = t'\} \cup \Delta : (u = t)\sigma \not\prec L\sigma$.

Figure 2.1: Expansion Inference Rules of $\mathcal{SP}$.

Subsumption	$\dfrac{S \cup \{C, C'\}}{S \cup \{C\}}$	if $C\vartheta \subseteq C'$ for some substitution ϑ
Simplification	$\dfrac{S \cup \{C[l'], l = r\}}{S \cup \{C[r\vartheta], l = r\}}$	if $l' \equiv l\vartheta$, $r\vartheta \prec l\vartheta$, and $\forall L \in C[l\vartheta] : (l\vartheta = r\vartheta) \prec L$
Deletion	$\dfrac{S \cup \{\Gamma \Rightarrow \Delta, t = t\}}{S}$	

where C and C' are clauses and S is a set of clauses.

Figure 2.2: Contraction Inference Rules of $\mathcal{SP}$.

returned by the prover contains the empty clause, then the T-satisfiability procedure returns *"unsatisfiable"*; otherwise, it returns *"satisfiable"*. This behavior respects the fact that $\mathcal{SP}$ is refutationally complete (see [71]).

2.4.2 Superposition Modules

Since we have to deal with constraints involving finitely (but arbitrarily) many new constants, we consider a countable set $\mathcal{K}$ of constants disjoint from Σ to form the expanded signature $\Sigma^{\mathcal{K}}$. Usual results on orderings can be extended to infinite signatures (see [67]); notice however that one can keep the signature $\Sigma^{\mathcal{K}}$ finite, by coding c_i as $s^i(0)$ (for new symbols $s, 0$), like e.g., in [28]. We collect all needed data in the following

Definition 2.4.2 (Suitable Ordering Triple)**.** A *suitable ordering triple* is a triple $(\Sigma, \mathcal{K}, \prec)$, where: (a) Σ is a finite signature; (b) $\mathcal{K} := \{c_1, c_2, c_3, \ldots\}$ is a countably infinite set of constant symbols such that Σ and $\mathcal{K}$ are disjoint; (c) $\prec$ is a reduction ordering over

$\Sigma^{\mathcal{K}}$-terms satisfying the following conditions:

(i) $\prec$ is total on ground $\Sigma^{\mathcal{K}}$-terms;

(ii) for every ground $\Sigma^{\mathcal{K}}$-term t with root symbol $f \in \Sigma$ and for every $c_i \in \mathcal{K}$, we have $c_i \prec t$;

(iii) for $c_i, c_j \in \mathcal{K}$, we have $c_i \prec c_j$ iff $i < j$.

The above conditions on the reduction ordering are similar to those adopted in [5, 4] to build rewrite-based decision procedures for the constraint satisfiability problem in theories of data structures, fragments of integer arithmetic, and their combinations. It is indeed very easy and natural to produce suitable ordering triples: for instance, if an LPO is adopted, it is sufficient to take a total precedence $>_p$ satisfying the condition $f >_p c_i >_p c_j$, for $f \in \Sigma$, $c_i \in \mathcal{K}$, $c_j \in \mathcal{K}$ and $i > j$.

Another key characteristic of a rewrite-based inference system is the possibility of associating a model to the set of derived clauses, defined by building incrementally a convergent term rewriting system.

Let $(\Sigma, \mathcal{K}, \prec)$ be a suitable ordering triple and let S be a set of $\Sigma^{\mathcal{K}}$-clauses not containing the empty clause. The set $gr(S)$ contains all ground $\Sigma^{\mathcal{K}}$-clauses that are instances of clauses in S. By transfinite induction on $C \in gr(S)$, we simultaneously define $Gen(C)$ and the ground rewrite system R_C as follows (see [71]):

(a) $R_C := \bigcup_{D \in gr(S), D \prec C} Gen(D)$;

(b) $Gen(C) := \{l \to r\}$ in case C is of the kind $\Delta_1 \Rightarrow l = r, \Delta_2$ and the following conditions are satisfied:

1. $R_C \not\models \Delta_1 \Rightarrow \Delta_2$, i.e. (i) for each $l = r \in \Delta_1$, l and r have the same normal form with respect to R_C (in symbols, $l \downarrow_{R_C} r$) and (ii) for each $s = t \in \Delta_2$, $s \not\downarrow_{R_C} t$;
2. $r \prec l$, $v \prec l$ (for all v occurring in Δ_1), $\{u, v\} \prec^{ms} \{r, l\}$, for every equation $u = v$ occurring in Δ_2, where $\prec^{ms}$ is the multi-set extension [11] of $\prec$;
3. l is not reducible by R_C, and
4. $R_C \not\models r = t'$, for every equation of the kind $l = t'$ occurring in Δ_2;

(c) $Gen(C) := \emptyset$, otherwise.

We say that C is productive if $Gen(C) \neq \emptyset$. Finally, let $R_S := \bigcup_{C \in gr(S)} Gen(C)$. Note that R_S is a convergent rewrite system, by conditions 2 and 3 above.

We are interested in a semantic notion of saturation based on model generation.

Definition 2.4.3. A set S of $\Sigma^{\mathcal{K}}$-clauses is *model-saturated* iff (i) S does not contain the empty clause and (ii) the rewrite system R_S is a model of S, i.e. the quotient of the Herbrand universe of $\Sigma^{\mathcal{K}}$ modulo R_S-convergence is a model of the universal closures of the clauses in S.

The following definition of reasoning module is precisely what we need to prove the main technical Lemma 2.4.10 below.

Definition 2.4.4 (Superposition Module)**.** Let $(\Sigma, \mathcal{K}, \prec)$ be a suitable ordering triple. A *superposition module* $\mathcal{SP}(\Sigma, \mathcal{K}, \prec)$ is a computable function which takes a finite set S_0 of $\Sigma^{\mathcal{K}}$-clauses as input and returns a (possibly infinite) sequence

$$S_0, S_1, \ldots, S_n, \ldots \tag{2.17}$$

of finite sets of $\Sigma^{\mathcal{K}}$-clauses, called an S_0-*derivation*, such that (i) if S_0 is unsatisfiable, then there exists $k \geq 0$ such that the empty clause is in S_k; (ii) if S_0 is satisfiable, then the set

$$S_\infty := \bigcup_{j \geq 0} \bigcap_{i \geq j} S_i$$

of *persistent clauses* is model-saturated, and (iii) the sets S_i and S_j are logically equivalent for $(0 \leq i, j \leq \infty)$. We say that $\mathcal{SP}(\Sigma, \mathcal{K}, \prec)$ *terminates* on the set of $\Sigma^{\mathcal{K}}$-clauses S_0 iff the S_0-derivation (2.17) is finite.

Superposition modules are *deterministic*, i.e. there exists just one S_0-derivation starting with a given finite set S_0 of clauses. Any implementation of the superposition calculus (see [71]) together with a fair strategy satisfies Definition 2.4.4.

2.4.3 Superposition Modules and Rewrite-based Decision Procedures

For the proofs below, we need a class of superposition modules which are invariant (in a sense to be made precise) under certain renamings of finitely many constants. Formally, an *n-shifting* (where n is an integer such that $n > 0$) is the operation that applied to a $\Sigma^{\mathcal{K}}$-expression E returns the $\Sigma^{\mathcal{K}}$-expression E^{+n} obtained from E by simultaneously replacing each occurrence of the free constant $c_i \in \mathcal{K}$ by the free constant c_{i+n}, for $i > 0$ (where the word "expression" may denote a term, a literal, a clause, or a set of clauses). In practice, an n-shifting enlarges the set of free constants occurring in the set of clauses by adding the extra constants $c_1, \ldots, c_n$ that are not in the range of the function $(\cdot)^{+n}$.

Example 2.4.5. Let us consider the set $S := \{f(c_1, c_4) = c_1, f(f(c_1, c_4), c_4) = c_2\}$ of ground $\Sigma^{\mathcal{K}}$-literals where $\Sigma := \{f\}$ and $\mathcal{K} := \{c_1, c_2, \ldots\}$. Then, we have that $S^{+5} := \{f(c_6, c_9) = c_6, f(f(c_6, c_9), c_9) = c_7\}$.

Definition 2.4.6 (Invariant Superposition Module)**.** Let $(\Sigma, \mathcal{K}, \prec)$ be a suitable ordering triple. A superposition module $\mathcal{SP}(\Sigma, \mathcal{K}, \prec)$ is *invariant* iff for every S_0-derivation $S_0, S_1, \ldots, S_j, \ldots$ (with S_0 being a set of $\Sigma^{\mathcal{K}}$-clauses), we have that $(S_0)^{+n}, (S_1)^{+n}, \ldots, (S_j)^{+n}, \ldots$ is an $(S_0)^{+n}$-derivation, for all $n \geq 0$.

Most of the actual implementations of the superposition calculus are *stable under signature extensions* (this is so because they need to handle Skolem symbols) and, hence, the behavior of a superposition prover is not affected by any proper extension of the signature and the ordering. The property of producing derivations that are invariant under shifting is weaker than stability under signature extensions. As a consequence, any superposition prover can be turned into an invariant superposition module. However, not all possible implementations of the superposition calculus are invariant superposition modules, as we shall discuss in Section 2.4.4.

Example 2.4.7. Suppose that in the suitable ordering triple $(\Sigma, \mathcal{K}, \prec)$, the term ordering $\prec$ is an LPO whose precedence satisfies $f >_p c_i >_p c_j$ (for $f \in \Sigma, c_i \in \mathcal{K}, c_j \in \mathcal{K}$, $i > j$). Let us consider the superposition module given by the standard superposition calculus (see Section 2.4.1) and let us take again the situation in Example 2.4.5. The (model-)saturated set output by $\mathcal{SP}(\Sigma, \mathcal{K}, \prec)$ when taking S as input is $S_s := \{f(c_1, c_4) = c_1, c_2 = c_1\}$. It is not difficult to see that the set $(S_s)^{+5} := \{f(c_6, c_9) = c_6, c_7 = c_6\}$ is exactly the set that we would obtain as output by the superposition module $\mathcal{SP}(\Sigma, \mathcal{K}, \prec)$ when taking as input the set $(S)^{+5}$ (see Example 2.4.5).

Definition 2.4.8. Let $(\Sigma, \mathcal{K}, \prec)$ be a suitable ordering triple. A universal and finitely axiomatized Σ-theory T is $\exists$-*superposition-decidable* iff there exists an invariant superposition module $\mathcal{SP}(\Sigma, \mathcal{K}, \prec)$ that is guaranteed to terminate when taking as input $T \cup \Gamma$, where Γ is a $\Sigma^{\mathcal{K}}$-constraint.

From the termination results for superposition calculus given in [5, 4], it follows that theories such as equality, (possibly cyclic) lists, arrays, and so on are $\exists$-decidable by superposition. According to Definition 2.4.8, any theory T which is $\exists$-superposition-decidable is $\exists$-decidable. In the following, we show that T is also $\exists_\infty$-decidable.

2.4.4 Invariant Superposition Modules and Cardinality Constraints

A *variable clause* is a clause containing only equations between variables or their negations. The *antecedent-mgu* (a-mgu, for short) of a variable clause $\Delta_1 \Rightarrow \Delta_2$ is the most general unifier of the unification problem $\{x \stackrel{?}{=} y \mid x = y \in \Delta_1\}$. A *cardinality constraint clause* is a variable clause $\Delta_1 \Rightarrow \Delta_2$ such that $\Rightarrow \Delta_2\mu$ does not contain any trivial equation like $x = x$, where μ is the a-mgu of $\Delta_1 \Rightarrow \Delta_2$; the number of free variables of $\Delta_2\mu$ is called the *cardinal* of the cardinality constraint clause $\Delta_1 \Rightarrow \Delta_2$. For example, the clause

$x = y \Rightarrow y = z_1, x = z_2$ is a cardinality constraint clause whose cardinal is 3 (notice that this clause is true only in the one-element model).

Lemma 2.4.9. *If a satisfiable set S of clauses contains a cardinality constraint clause $\Delta_1 \Rightarrow \Delta_2$, then S cannot have a model whose domain is larger than the cardinal of $\Delta_1 \Rightarrow \Delta_2$.*

Proof. Let μ be the a-mgu of $\Delta_1 \Rightarrow \Delta_2$. By definition of a cardinality constraint clause, the clause $\Rightarrow \Delta_2\mu$ does not contain trivial equations; if n is the number of distinct variables in $\Rightarrow \Delta_2\mu$, then there cannot be more than $n-1$ distinct elements in any model of S. □

The next crucial lemma expresses the property that an invariant superposition module will discover a cardinality constraint clause whenever the input set of clauses does not admit infinite models. In Section 2.4.4, we illustrate this behavior by showing how the superposition calculus can derive a cardinality constraint clause from $\Rightarrow x = a, x = b$.

Lemma 2.4.10. *Let $(\Sigma, \mathcal{K}, \prec)$ be a suitable ordering triple. Let $\mathcal{SP}(\Sigma, \mathcal{K}, \prec)$ be an invariant superposition module. If S_0 is a satisfiable finite set of clauses, then the following conditions are equivalent:*

(i) *the set S_∞ of persistent clauses in an S_0-derivation of $\mathcal{SP}(\Sigma, \mathcal{K}, \prec)$ contains a cardinality constraint clause;*

(ii) *S_0 does not admit infinite models.*

Proof. The implication (i) ⇒ (ii) is proved by Lemma 2.4.9. To show (ii) ⇒ (i), assume that the set S_0 does not have a model whose domain is infinite. By Lemma 2.3.1, there must exist a natural number N such that every model $\mathcal{M}$ of S_0 has a domain with at most N elements. Since a cardinality constraint clause does not contain constants, it is in S_∞ iff it is in $(S_\infty)^{+N}$. Hence, by Definition 2.4.6 of an invariant superposition module (considering $(S_0)^{+N}$ rather than S_0, if needed) we are free to assume that the constants $\{c_1, \ldots, c_N\}$ do not occur in S_∞. Recall also that, according to the definition of a suitable ordering triple, the constants $\{c_1, \ldots, c_N\}$ are the smallest ground $\Sigma^{\mathcal{K}}$-terms.

According to the definition of superposition module (see Definition 2.4.4), since S_0 is assumed to be satisfiable, S_∞ is model-saturated, which means that the convergent rewrite system R_{S_∞} is a model of S_∞ (hence also of S_0, which is logically equivalent to S_∞). Now, since S_0 does not have a model whose domain is of cardinality N or greater, there is at least one constant among $c_1, \ldots, c_N$ which is not in normal form (with respect to R_{S_∞}). Assume that c_i is not in normal form (with respect to R_{S_∞}) and that each c_j (for $j < i$) is. By model generation (see Section 2.4.2), to reduce c_i we need a rule $l \to r$ from a productive clause C of the kind $\Delta_1 \Rightarrow l = r, \Delta_2 \in gr(S_\infty)$; furthermore, c_i can be

reduced only to c_j for $j < i$. The maximality condition 2 of model generation in Section 2.4.2 on l implies that l is c_i and that the remaining terms in C are of the kind c_j for $j \leq i$.[2] By condition 1 of model generation in Section 2.4.2, the fact that all terms c_j ($j < i$) are in R_{S_∞}-normal form, and the fact that R_{S_∞} is a convergent rewrite system extending R_C, it follows that each equation in Δ_1 is of the form $c_j = c_j$. Furthermore, again by condition 1 of model generation in Section 2.4.2, there is no (trivial) equality of the form $c_j = c_j$ in Δ_2. Since the constants $\{c_1, \ldots, c_N\}$ do not occur in S_∞, we are entitled to conclude that the productive clause $\Delta_1 \Rightarrow l = r, \Delta_2$ is the ground instance of a variable clause, i.e. there must exist a variable clause $\tilde{C}$ of the form $\tilde{\Delta}_1 \Rightarrow \tilde{l} = \tilde{r}, \tilde{\Delta}_2$ in S_∞ such that $\tilde{C}\vartheta \equiv C$ for some ground substitution ϑ. Since the antecedent of C consists of trivial equalities, ϑ is less general than μ, where μ is the a-mgu of $\tilde{C}$, i.e. we have that $\vartheta = \mu\vartheta'$ for some substitution ϑ'. Furthermore, since there are no positive trivial equalities in $C \equiv \tilde{C}\mu\vartheta'$, there are no positive trivial equalities in $\tilde{C}\mu$ either, which implies that $\tilde{C}$ is a cardinality constraint clause belonging to S_∞. □

The following result immediately follows from Lemma 2.4.10 above, because unsatisfiability in infinite models can be detected by looking for a cardinality constraint clause among the finitely many final clauses of a terminating derivation:

Theorem 2.4.11. *Let T be a finitely axiomatized universal Σ-theory where Σ is finite. If T is $\exists$-superposition-decidable, then T is strongly $\exists_\infty$-decidable.*

Deriving a Cardinality Constraint Clause in Practice

As already said (see Section 2.4.1), superposition calculus is refutationally complete: model generation technique is the main tool to show this result. However, the completeness proof in [71] makes clear that the calculus is complete as well if the ordering constraints are interpreted as *symbolic constraint solving problems* (see, e.g., [27, 70, 55]): this means that, e.g., the condition *(i)* in Figure 2.1 can be rephrased as "there exists a ground substitution ϑ such that $t\sigma\vartheta \prec u\sigma\vartheta$" (where $\not\preceq$ can be replaced to $\prec$, because the ordering is total on ground terms). We can further restrict the ground substitution ϑ to take values in the *actual* signature (and not in a signature extending the actual one). These choices are not very convenient from a practical point of view, because the benefit of blocking some inference does not compensate the increase in complexity due to the intractability of symbolic constraint solving problems (which usually are NP-complete problems).

What we want to point out here is that this interpretation of ordering constraints as symbolic constraint solving problems in the actual signature destroys invariance in

[2] More precisely (this is important for the proof): terms occurring positively can only be c_j for $j \leq i$ and terms occurring negatively can only be c_j for $j < i$.

the sense of Definition 2.4.6 and also invalidates the statement of Lemma 2.4.10. To see why this is the case, let $c_1 \in \mathcal{K}$ be the smallest constant in the given suitable ordering triple. A clause like $x = c_1$ can be superposed with itself if the maximality constraint is interpreted as $x \not\succeq c_1$ (and the result of the superposition is $x = y$). On the other hand, if the maximality constraint is interpreted as a symbolic constraint solving problem in the signature $\Sigma^{\mathcal{K}}$, then no superposition applies because there is no ground term smaller than c_1 in $\Sigma^{\mathcal{K}}$. Unfortunately, if we apply a +2-shifting, then the symbolic constraint $x \prec c_3$? has, e.g., the solution $x \mapsto c_1$ and superposition is not blocked anymore. Notice also that the singleton set of clauses $\{x = c_1\}$ is model-saturated,[3] has no infinite models, but does not contain a cardinality constraint clause.

To illustrate the content of Lemma 2.4.10 in a simple but not entirely trivial case, let us consider the clause $\Rightarrow x = a, x = b$, which tells us that there are at most two elements in the domain of a model (these are the interpretations of the constants a and b). It is instructive to apply to this clause the superposition calculus (in the plain Figure 2.1 formulation, where ordering constraints are just $\not\preceq$-conditions). The following is a derivation of a cardinality constraint clause:

1.		$\Rightarrow$	$u = a, u = b$	
2.		$\Rightarrow$	$u = a, v = a, v = u$	[*Sup* 1.1, 1.1]
3.		$\Rightarrow$	$u = a, u = v, w = v, x = a, x = w$	[*Sup* 2.0, 2.0]
4.	$a = a$	$\Rightarrow$	$u = v, w = v, u = a, u = w$	[*Fac* 3.0, 3.3]
5.		$\Rightarrow$	$u = v, w = v, u = a, u = w$	[*Ref* 4.0]
6.		$\Rightarrow$	$u = v, w = v, u = w, x = y, z = y, x = u, x = z$	[*Sup* 5.2, 5.2]

where u, v, w, x, y, and z are variables, *Sup* abbreviates *Superposition*, *Fac* abbreviates *Factoring*, *Ref* abbreviates *Reflection*, and the sequences of non negative integers separated by "." denote positions. With a little bit of effort, it is possible to derive (by continuing the application of the rules of the calculus) a cardinality constraint clause

[3] Recall that we defined model-saturation of a set of clauses in terms of the rewrite system associated to the model generation construction (and not in terms of closure - up to redundancy - with respect to the rules of the calculus).

whose cardinal is 3:

7.	$v = y$	$\Rightarrow$	$z = y, w = v, z = w, x = y, x = z, x = z$	[*Fac* 6.0, 6.4]
8.		$\Rightarrow$	$z = y, w = y, z = w, x = y, x = z, x = z$	[*Ref* 7.0]
9.	$y = y$	$\Rightarrow$	$z = y, x = y, z = x, x = z, x = z$	[*Fac* 8.1, 8.3]
10.		$\Rightarrow$	$z = y, x = y, z = x, x = z, x = z$	[*Ref* 9.0]
11.	$z = z$	$\Rightarrow$	$z = y, x = y, z = x, x = z$	[*Fac* 10.3, 10.4]
12.		$\Rightarrow$	$z = y, x = y, z = x, x = z$	[*Ref* 11.0]
13.	$z = z$	$\Rightarrow$	$z = y, x = y, z = x$	[*Fac* 12.2, 12.3]
14.		$\Rightarrow$	$z = y, x = y, z = x$	[*Ref* 13.0]

Cardinality constraint clauses are always derived by common superposition provers, according to Lemma 2.4.10, when saturating sets of clauses not admitting infinite models. Such derivations, however, even in simple cases like the one above, seems to take considerable amount of time in state-of-the-art provers.

2.4.5 Combining Superposition Modules and SMT Procedures

Invariant superposition modules provide us with means to check whether a theory is strongly decidable or not. However, the situation is not really clear in practice. By using available state-of-the-art implementations of the superposition calculus, such as SPASS (see [88]) or E (see [79]), with suitable ordering, we have run concrete invariant superposition modules for a theory $T^{\leq k}$, admitting only finite models with at most $k-1$ elements, axiomatized by an appropriate "at-most" cardinality constraint, see (2.3). Indeed, according to Definition 2.4.6, the hard part is to prove termination for arbitrary input clauses of the form $T^{\leq k} \cup \Gamma$, where Γ is a set of ground literals. Our preliminary experiments were quite discouraging. In fact, both SPASS and E were able to handle only the trivial theory $T^{\leq 1}$ (axiomatized by $\Rightarrow x = y$). Already for $T^{\leq 2}$ (axiomatized by $\Rightarrow x = y, x = z, y = z$), the provers do not terminate in a reasonable amount of time although we experimented with various settings. For example, while SPASS is capable of finding a saturation for $T^{\leq 2} \cup \Gamma$ when $\Gamma := \emptyset$, it seems to diverge when $\Gamma := \{a \neq b\}$. This seems to dramatically reduce the scope of applicability of Theorem 2.4.11 and hence of Theorem 2.3.4.

Fortunately, this problem can be solved by the following two observations. First, although a superposition module may not terminate on instances of the constraint satisfiability problem of the form $T \cup \Gamma$, where Γ is a constraint and T does not admit infinite models (such as $T^{\leq k}$, above), Lemma 2.4.10 ensures that a cardinality constraint clause will eventually be derived in a finite amount of time: if a clause C is in the set S_∞ of persistent clauses of a derivation $S_0, S_1, \ldots$, then there must exists an integer $k \geq 0$ such

function *Grounding* (N : *integer*, T: *axioms*, Γ: *Ground literals*)
1 introduce fresh constants $c_1, \ldots, c_N$;
2 for every k-ary function symbol f in $\Gamma \cup T$ (with $k \geq 0$), generate the positive clauses

$$\bigvee_{i=1}^{N} f(a_1, \ldots, a_k) = c_i$$

for every $a_1, \ldots, a_k \in \{c_1, \ldots, c_N\}$ and let E be the resulting set of clauses;
3 for every clause $C \in T$, instantiate C in all possible ways by ground substitutions whose range is the set $\{c_1, \ldots, c_N\}$ and let T_g be the resulting set of clauses;
4 return the set $T_g \cup E \cup \Gamma$.
end

Figure 2.3: The *Grounding* function

that $C \in S_k$ (recall Definition 2.4.4). Second, when a cardinality constraint clause C is derived from $T \cup \Gamma$, a bound on the cardinality of the domains of any model can be immediately obtained by the cardinal associated to C. It is possible to use such a bound to build an equisatisfiable set of clauses (see Figure 2.3) and pass it to an efficient decision procedure for the pure theory of equality, based on congruence closure, such as those provided by many SMT tools (see, e.g., [32, 6, 36, 43]). The observations above motivate the following relaxation of the notion of an $\exists$-superposition-decidable theory.

Definition 2.4.12. Let $(\Sigma, \mathcal{K}, \prec)$ be a suitable ordering triple. A universal and finitely axiomatized Σ-theory T is *weakly $\exists$-superposition-decidable* iff there exists an invariant superposition module $\mathcal{SP}(\Sigma, \mathcal{K}, \prec)$ such that for every $\Sigma^{\mathcal{K}}$-constraint Γ, any $T \cup \Gamma$-derivation either (i) terminates or (ii) generates a cardinality constraint clause.

We can easily adapt Theorem 2.4.11 to this new definition.

Theorem 2.4.13. *Let T be a universal and finitely axiomatized Σ-theory, where Σ is finite. If T is weakly $\exists$-superposition-decidable, then T is strongly $\exists_\infty$-decidable.*

Proof. Decidability of Σ-constraints in models of T can be obtained by halting the invariant superposition module and then using any SMT procedure for the theory of equality with the set of clauses obtained by the function *Grounding* of Figure 2.3. Decidability in infinite models is answered negatively if a cardinality constraint clause is generated; otherwise, we have termination of the invariant superposition module and if the empty clause is not produced, satisfiability is reported by Lemma 2.4.10. □

Chapter 3

Combining Theories over Non-Disjoint Signatures

As already pointed out, the Nelson-Oppen combination schema yields to a decision procedure for the constraint satisfiability problem for the union of theories which are stably infinite and over disjoint signatures. In the previous chapter we were concerned with the weakening of the hypothesis of stable infiniteness; in this chapter we deal with the hypothesis of disjoint signatures. A first attempt to drop it can be found in [44], where it is replaced by two fundamental requirements: the first regards a sort of "compatibility" with respect a common subtheory and the second limits the number of terms different up to logical consequence to be finite. In the following we re-propose these results and we weaken that second requirement by introducing a "noetherianity" notion. Let us start fixing the context.

3.1 Some Notions from Model Theory

Here and in the following we will refer to the definitions presented in Section 1.1. Whenever confusion does not arise, we may improperly write clauses/positive clauses as disjunctions of the corresponding literals/atoms (in particular, the empty clause is always confused with the empty disjunction $\bot$ expressing syntactic falsity). Letters $\varphi, \psi, \ldots$ are used for formulae, whereas letters $A, B, \ldots$ are used for literals and letters $C, D, \ldots$ are used for clauses.

If T and T' are two Σ-theories, writing $T \subseteq T'$ means that all the axioms for T are logical consequences of the axioms for T', i.e. that the set of sentences in T are true in all the models of T'. The *diagram* $\Delta(\mathcal{A})$ of a Σ-structure $\mathcal{A}$ is the set of ground $\Sigma^{\mathcal{A}}$-literals which are true in $\mathcal{A}$, thus enlarging the signature Σ with the names of all the elements in the domain of $\mathcal{A}$; the *elementary diagram* $\Delta^e(\mathcal{A})$ of a Σ-structure $\mathcal{A}$ is the

set of $\Sigma^{\mathcal{A}}$-sentences which are true in $\mathcal{A}$. We recall a very useful theorem

Theorem 3.1.1 (Robinson's Diagram Lemma [25])**.** *There is an (elementary) embedding between the Σ-structures $\mathcal{A}$ and $\mathcal{B}$ iff it is possible to expand $\mathcal{B}$ to a $\Sigma^{\mathcal{A}}$-structure in such a way that it becomes a model of the (elementary) diagram of $\mathcal{A}$.*

Now we introduce a notion that will be deeply exploited in what follows.

Definition 3.1.2. Let T be a Σ-theory and let T^* be a further Σ-theory extending T. T^* is a *model completion* of T iff:

(i) every model of T embeds into a model of T^*;

(ii) for every Σ-structure $\mathcal{A}$ which is a model of T, we have that $T^* \cup \Delta(\mathcal{A})$ is a complete $\Sigma^{\mathcal{A}}$-theory.

A Σ theory T is said to be *sub-model complete* if and only if, given $\mathcal{M}$, model of T, and $\mathcal{A} \subseteq \mathcal{M}$ any substructure of $\mathcal{M}$, $T \cup \Delta(\mathcal{A})$ is a complete $\Sigma^{\mathcal{A}}$-theory. By the Diagram Lemma, an equivalent formulation is the following: T is sub-model complete iff given two models $\mathcal{M}_1, \mathcal{M}_2$ of T and given a common Σ-substructure $\mathcal{A}$ of them, we have that $\mathcal{M}_1$ is elementarily equivalent to $\mathcal{M}_2$ as a $\Sigma^{\mathcal{A}}$-structure. The notion of sub-model completeness is the semantic counterpart of quantifier elimination.

Proposition 3.1.3. *T admits elimination of quantifiers iff it is sub-model complete.*

Proof. Suppose that T admits elimination of quantifier; consider two models $\mathcal{N}_1, \mathcal{N}_2$ of its, a common substructure $\mathcal{A}$ of $\mathcal{N}_1$ and $\mathcal{N}_2$ and a $\Sigma_0^{\mathcal{A}}$-sentence $\varphi(\underline{a})$. $\varphi(\underline{x})$ is T-equivalent to a quantifier-free formula $\varphi'(\underline{x})$, hence if $\varphi(\underline{a})$ is true in $\mathcal{N}_1$, $\varphi'(\underline{a})$ is true in it, $\varphi'(\underline{a})$ is true in $\mathcal{A}$ and in $\mathcal{N}_2$, as well, because ground open formulae are preserved by both sub- and super-structures, thus establishing that $\varphi(\underline{a})$ is true in $\mathcal{N}_2$. Since $\varphi(\underline{a})$, $\mathcal{N}_1, \mathcal{N}_2$ are arbitrary, $T \cup \Delta(\mathcal{A})$ is complete.

Suppose now, for the other side, that T is sub-model complete and let $\varphi(\underline{x})$ be an arbitrary formula. For new constants $\underline{a}$ consider the set of sentences

$$\Theta \ := \ T \cup \{\varphi(\underline{a})\} \cup \{\neg\psi(\underline{a}) \mid \psi \text{ is quantifier-free and } T \models \psi(\underline{a}) \rightarrow \varphi(\underline{a})\}.$$

If Θ is inconsistent, then we have $T \models \varphi(\underline{a}) \rightarrow \psi_1(\underline{a}) \vee \ \cdots \ \vee \psi_n(\underline{a})$ for quantifier-free ψ_i implying φ, so that we have

$$T \models \varphi(\underline{x}) \leftrightarrow \psi_1(\underline{x}) \vee \ \cdots \ \vee \psi_n(\underline{x})$$

yielding the fact that T admits quantifier elimination. Consequently, it suffices to show that Θ cannot be consistent. Suppose it is and let $\mathcal{M}$ be a model of it. Let $\mathcal{A}$ be the

substructure of $\mathcal{M}$ generated by the $\underline{a}$; we must have

$$T \cup \Delta(\mathcal{A}) \models \varphi(\underline{a}),$$

because otherwise we would be able to build a model of T containing $\mathcal{A}$ as a substructure and falsifying $\varphi(\underline{a})$ (which cannot be because T is sub-model complete and $\varphi(\underline{a})$ is true in $\mathcal{M}$ being an element of Θ). This means that for some quantifier-free sentence $\psi(\underline{a})$ true in $\mathcal{A}$ we have that $T \models \psi(\underline{a}) \rightarrow \varphi(\underline{a})$. According to the definition of Θ, $\neg\psi(\underline{a})$ is true in $\mathcal{M}$ and also in $\mathcal{A}$ (because it is quantifier-free), contradiction. □

The following lemma shows an interesting relationship between model completion and sub-model completeness.

Lemma 3.1.4. *Let T be a universal Σ-theory and let $T^* \supseteq T$ a further Σ-theory; under the hypothesis that* (i) *every model of T embeds into a model of T^*, the following requirements are equivalent:*

(ii) *for every Σ-structure $\mathcal{A}$ which is a model of T, we have that $T^* \cup \Delta(\mathcal{A})$ is a complete $\Sigma^{\mathcal{A}}$-theory;*

(ii') *T^* is a sub-model complete theory.*

Proof. Let us assume that (i) and (ii) hold. We want to show that, if $\mathcal{M}$ is a model of T^* and $\mathcal{A}$ is a substructure of $\mathcal{M}$, then $T^* \cup \Delta(\mathcal{A})$ is a complete $\Sigma^{\mathcal{A}}$-theory. $T \subseteq T^*$, so $\mathcal{M}$ is a model of T; moreover, since T is universal, also $\mathcal{A}$ is a model of T. It is possible to apply (ii), obtaining that $T^* \cup \Delta(\mathcal{A})$ is a is a complete $\Sigma^{\mathcal{A}}$-theory.

Let us assume now that (i) and (ii') hold, and let $\mathcal{M}$ be a model of T. By hypothesis (i), $\mathcal{M}$ is a substructure of a model $\mathcal{N}$ of T^* for some $\mathcal{N}$; thus (ii') applies, yielding the fact that $T^* \cup \Delta(\mathcal{M})$ is a complete $\Sigma^{\mathcal{M}}$-theory. □

Notice that, under the hypothesis (i), (ii') implies (ii) even if we drop the assumption that T is a universal theory; moreover, as the sub-model completeness is equivalent to quantifier elimination, the following proposition holds.

Proposition 3.1.5. *Let T be a Σ-theory and let $T^* \supseteq T$ a further Σ-theory. T^* is a model-completion of T if*

1. *every model of T embeds into a model of T^*;*
2. *T^* admits quantifier elimination.*

It can be shown that a model completion T^* of a theory T is unique in case it exists, and if T has universal axioms, then T^* has a set of $\forall\exists$-axioms, i.e. every sentence in this

axiomatization of T^* is obtained from an open formula by prefixing it some existential quantifiers and finally some universal quantifiers (in this order).

We can give a brief list of examples of theories that are model-completion: the theory of an infinite set is the model completion of the pure theory of equality; the theory of dense total orders without endpoints is the model completion of the theory of total orders; the theory of algebraically closed fields is the model completion of the theory of integral domains; the theory of divisible torsion free abelian groups is the model completion of the theory of torsion free abelian groups. An old result in [92] says, in particular, that universal Horn theories T in finite signatures always have a model completion, provided the following two conditions are satisfied: (a) finitely generated models of T are all finite; (b) amalgamation property holds for models of T. This fact can be used in order to prove the existence of a model completion for theories axiomatizing many interesting discrete structures (like graphs, posets, etc.).

From Proposition 3.1.5 it follows that every theory T^* admitting elimination of quantifiers is the model completion of itself; moreover, T^* is also the model completion of the theory T axiomatized by the set of universal sentences which are logical consequences of T^* (see [25] for a proof).

3.2 Compatibility

A key ingredient for the combination of procedures is the following notion:

Definition 3.2.1. Let T be a theory in the signature Σ and let T_0 be a universal theory in a subsignature $\Sigma_0 \subseteq \Sigma$. We say that T is T_0-*compatible* iff

(i) $T_0 \subseteq T$;

(ii) T_0 has a model-completion T_0^*;

(iii) every model of T embeds into a model of $T \cup T_0^*$.

Remark 3.2.2. Let us make few observations.

- According to the above Definition, it is evident that T_0-compatibility reduces to the standard notion of stable infiniteness (used in the disjoint Nelson-Oppen combination procedure) in case T_0 is the pure theory of equality.

- Moreover, every theory including a universal theory T_0 that is the model completion of itself is T_0-compatible. Examples of that kind do exist: for example, let us consider the theory L of acyclic lists, whose signature consists on two unary function

symbols *car*, *cdr* and one binary function symbol *cons*, and whose axioms are:

$$cons(car(x), cdr(x)) = x$$
$$car(cons(x, y)) = x$$
$$cdr(cons(x, y)) = y$$
$$t(x) \neq x$$

where t is a term different from a variable and a constant, built up by using finitely many applications of the unary function symbols *car*, *cdr* and involving the variable x. L is universal and admits elimination of quantifiers (see [44]), so every theory including L is L-compatible.

– If T_0 has a model completion T_0^* and if $T \supseteq T_0^*$, then T is certainly T_0-compatible: this trivial case is often interesting (we may take e.g. T_0 to be the theory of linear orders and T to be real arithmetic or rational linear arithmetic).

– Let T_0 be a universal theory having a model completion T_0^* and let T be any extension of T_0 with free function symbols only. In this case, T is T_0-compatible: to see it, take any model $\mathcal{M}$ of T, embeds its Σ_0-reduct into a model $\mathcal{M}'$ of T_0^* and expand in any arbitrary way the interpretation of the free function symbols to the tuples of $\mathcal{M}'$ not entirely belonging to $\mathcal{M}$.

More examples of theories which are T_0-compatible for some T_0 will be supplied in section 3.4. Before going on to state the decidability result for the combined constraint satisfiability, we recall the Robinson's Joint Consistency Theorem (see [25]):

Theorem 3.2.3 (Robinson's Joint Consistency Theorem)**.** *Let H_0 be a complete theory in a signature $\Theta_0 = \Theta_1 \cap \Theta_2$ and H_1, H_2 be consistent extensions of its in the signatures Θ_1, Θ_2, respectively. Then $H_1 \cup H_2$ is consistent too in the signature $\Theta_1 \cup \Theta_2$.*

3.3 Combining Compatible Theories

Let us fix the main data for the whole chapter trough the following:

Assumption 3.3.1.

1. *T_1 is a theory in the signature Σ_1 and T_2 is a theory in the signature Σ_2; Σ_0 is the signature $\Sigma_1 \cap \Sigma_2$;*

2. *For finitely many new free constants $\underline{a}$, Γ_1 is a finite set of ground literals in the signature $\Sigma_1^{\underline{a}}$ and Γ_2 is a finite set of ground literals in the signature $\Sigma_2^{\underline{a}}$;*

3. There is a universal Σ_0-theory T_0 such that both T_1 and T_2 are T_0-compatible.

The main aim is that of (semi)deciding the universal fragment of $T_1 \cup T_2$, given that the corresponding universal fragments of T_1 and T_2 are (semi)decidable. By skolemization, this amounts to (semi)decide the consistency of

$$T_1 \cup T_2 \cup \Gamma, \tag{3.1}$$

where Γ is a finite set of ground literals in the signature $\Sigma_1 \cup \Sigma_2$, expanded with a finite set of new Skolem constants.

Recall, from Section 2.1, that Γ can be *purified* into the equisatisfiable (w.r.t. $T_1 \cup T_2$) set $\Gamma_1 \cup \Gamma_2$, where Γ_i is a set of ground literals over a simple expansion of Σ_i (for $i = 1, 2$). Thus the problem reduces to that of establishing the consistency of a set of literals like

$$(T_1 \cup \Gamma_1) \cup (T_2 \cup \Gamma_2), \tag{3.2}$$

where Γ_1, Γ_2 satisfy the requirements described in the Assumption 3.3.1-(2). Notice that at the end of the purification process, Σ_0-literals could be inserted either in Γ_1 or in Γ_2 or in both of them, indifferently.

Clearly the consistency of (3.2) cannot follow from the mere separate consistency of $T_1 \cup \Gamma_1$ and of $T_2 \cup \Gamma_2$ (for trivial reasons, take e.g. $T_1 = T_2 = \emptyset$, $\Gamma_1 = \{a_1 = a_2\}$ and $\Gamma_2 = \{a_2 = a_3,\ a_1 \neq a_3\}$). We need some *information exchange* between a reasoner dealing with $T_1 \cup \Gamma_1$ and a reasoner dealing with $T_2 \cup \Gamma_2$. Craig's interpolation theorem for first-order logic ensures that the inconsistency of (3.2) can be detected by the information exchange of a single Σ_0^a-sentence φ such that $T_1 \cup \Gamma_1 \models \varphi$ and $T_2 \cup \Gamma_2 \cup \{\varphi\} \models \bot$. However, as pointed out in [85], this observation is not very useful, as φ might be any first-order formula, whereas we would like - at least - φ to be quantifier-free: remind that most existing provers detect inconsistency just by skolemization and saturation, so they surely would not be able to find such a φ (if it is not quantifier-free), even in case they can efficiently handle with both $T_1 \cup \Gamma_1$ and $T_2 \cup \Gamma_2$.

Unfortunately, information exchange of quantifier-free Σ_0^a-formulae alone is not sufficient, even for syntactically simple T_1 and T_2, to establish the inconsistency of (3.2). Thus we need the further assumption (3) in order to get limited information exchange without affecting refutational completeness ((3) is the only relevant assumption we make, being (1) and (2) mere notational conventions).

Definition 3.3.2. A *positive residue chain* is a finite list

$$C_1, \ldots, C_n$$

of positive ground Σ_0^a-clauses such that for every $k = 1, \ldots, n$, there is $i = 1, 2$ such that

$$T_i \cup \Gamma_i \cup \{C_1, \ldots, C_{k-1}\} \models C_k.$$

We can now formulate the main combination results, whose proof can be found in the next section:

Theorem 3.3.3. *Under the Assumption 3.3.1, $(T_1 \cup \Gamma_1) \cup (T_2 \cup \Gamma_2)$ is inconsistent iff there is a positive residue chain $C_1, \ldots, C_n$ such that C_n is the empty clause.*

Thus inconsistency can be detected by repeated exchanges of positive ground clauses only; if we allow information exchange consisting on ground quantifier-free formulae, a single exchange step is sufficient:

Theorem 3.3.4. *Under the Assumption 3.3.1, $(T_1 \cup \Gamma_1) \cup (T_2 \cup \Gamma_2)$ is inconsistent iff there is a ground quantifier-free Σ_0^a-sentence φ such that*

$$T_1 \cup \Gamma_1 \models \varphi \qquad \textit{and} \qquad T_2 \cup \Gamma_2 \cup \{\varphi\} \models \bot.$$

We recall that the T_i's are said to be Σ_0-convex (see Section 2.1) iff whenever it happens that $T_i \cup \Gamma_i \models A_1 \vee \cdots \vee A_n$ (for $n \geq 1$ and for ground Σ_0^a-atoms $A_1, \ldots, A_n$), then there is $k = 1, \ldots, n$ such that $T_i \cup \Gamma_i \models A_k$. Among Σ_0-convex theories we have the important class of universal Horn theories, see [85]. For Σ_0-convex theories, an immediate subsumption argument refines Theorem 3.3.3 in the following way:

Corollary 3.3.5. *In addition to the Assumption 3.3.1, suppose also that T_1, T_2 are both Σ_0-convex. Then $(T_1 \cup \Gamma_1) \cup (T_2 \cup \Gamma_2)$ is inconsistent iff there is a positive residue chain $C_1, \ldots, C_n$ in which $C_1, \ldots, C_{n-1}$ are all ground Σ_0^a-atoms and C_n is $\bot$.*

It is clear that Theorems 3.3.3 and 3.3.4 yield *combined semidecision procedures for the constraint satisfiability problem* in $T_1 \cup T_2$ in case the corresponding constraint satisfiability problems for T_1 and T_2 are separately decidable. The procedure suggested by Theorem 3.3.3 is just a fair information exchange of positive ground Σ_0^a-clauses that eventually stops if $(T_1 \cup \Gamma_1) \cup (T_2 \cup \Gamma_2)$ is inconsistent. On the contrary, the procedure suggested by Theorem 3.3.4 (which is nothing but an interpolation theorem) identifies all ground Σ_0^a-clauses which are logical consequences of $T_1 \cup \Gamma_1$ and check whether their conjunction is consistent with $T_2 \cup \Gamma_2$.

3.3.1 Proofs of the Combination Result

Definition 3.3.6. A set Δ of Σ_0^a-clauses is *exhaustive* whenever it contains, for every ground Σ_0^a-literal A, either A itself or its negation.

Definition 3.3.7. Say that a set Γ_0 of *positive* ground Σ_0^a-clauses is *saturated* iff it is closed under the two rules

$$T_1 \cup \Gamma_1 \cup \Gamma_0 \models C \quad \Rightarrow \quad C \in \Gamma_0$$
$$T_2 \cup \Gamma_2 \cup \Gamma_0 \models C \quad \Rightarrow \quad C \in \Gamma_0.$$

Lemma 3.3.8. *Suppose that Γ_0 is saturated and does not contain the empty clause. Then there are Σ_i^a-models $\mathcal{M}_i$ $(i = 1, 2)$ such that*

$$\mathcal{M}_1 \models T_1 \cup \Gamma_1 \cup \Gamma_0 \quad \textit{and} \quad \mathcal{M}_2 \models T_2 \cup \Gamma_2 \cup \Gamma_0;$$

moreover $\mathcal{M}_1$ and $\mathcal{M}_2$ share the same Σ_0-substructure generated by the elements (denoted by) $\underline{a}$.

Proof. The statement of the lemma is proved if we are able to find an exhaustive set Δ of ground Σ_0^a-literals which is consistent with both $T_1 \cup \Gamma_1 \cup \Gamma_0$ and $T_2 \cup \Gamma_2 \cup \Gamma_0$. In this case, in fact, given any two models $\mathcal{M}_1 \models T_1 \cup \Gamma_1 \cup \Gamma_0 \cup \Delta$ and $\mathcal{M}_2 \models T_2 \cup \Gamma_2 \cup \Gamma_0 \cup \Delta$, we have that their Σ_0-substructures generated by $\underline{a}$ both have diagram Δ, consequently they are Σ_0-isomorphic. We may assume that they are just the same, by renaming some elements in one of the supports, if needed.

We shall adapt the notion of productive clause used in nowadays refutational completeness proofs for e.g. resolution or paramodulation based calculi. Consider any strict total terminating order on ground Σ_0^a-atoms and extend it to a strict total terminating order $>$ for positive ground Σ_0^a-clauses by taking standard multiset extension. We shall define increasing sets Δ_C^+ (varying $C \in \Gamma_0$) of ground Σ_0^a-atoms as follows. Recall that, as the empty clause is not in Γ_0, all positive clauses in Γ_0 are of the kind $A \vee A_1 \vee \cdots \vee A_n$ $(n \geq 0)$.

The definition is by transfinite induction on $>$. Say that the clause $C \equiv A \vee A_1 \vee \cdots \vee A_n$ from Γ_0 is *productive* iff

(i) $\{A\} > \{A_1, \ldots, A_n\}$;

(ii) $A_1, \ldots, A_n \notin \Delta_{<C}^+$ (where $\Delta_{<C}^+$ is $\bigcup_{D<C} \Delta_D^+$).

Now, if C is productive, we let Δ_C^+ to be $\Delta_{<C}^+ \cup \{A\}$, otherwise Δ_C^+ is simply $\Delta_{<C}^+$.

Let Δ^+ be $\bigcup_{C \in \Gamma_0} \Delta_C^+$ and Δ be $\Delta^+ \cup \{\neg A \mid A$ is a ground Σ_0^a-atom not belonging to $\Delta^+\}$. By construction, $\Delta \models \Gamma_0$, so we simply need to show that $T_1 \cup \Gamma_1 \cup \Delta$ and $T_2 \cup \Gamma_2 \cup \Delta$ are consistent.

It is straightforward to verify that if the clause $A \vee A_1 \vee \cdots \vee A_n$ is productive and A is the maximum atom in it, then $A_1, \ldots, A_n \notin \Delta^+$: actually, the A_i's could only be produced by clauses smaller than $A \vee A_1 \vee \cdots \vee A_n$.

Suppose now that $T_1 \cup \Gamma_1 \cup \Delta$ is not consistent (the case $i = 2$ is analogous). Then there are ground atoms $B_1, \ldots, B_m \notin \Delta^+$ and productive clauses

$$\begin{aligned} C_1 &\equiv A_1 \vee A_{11} \vee \cdots \vee A_{1k_1} \\ &\cdots \\ C_n &\equiv A_n \vee A_{n1} \vee \cdots \vee A_{nk_n} \end{aligned}$$

(with maximum atoms $A_1, \ldots, A_n$, respectively), such that

$$T_1 \cup \Gamma_1 \cup \{A_1, \ldots, A_n\} \models B_1 \vee \cdots \vee B_m.$$

By trivial logical manipulations, it follows that

$$T_1 \cup \Gamma_1 \cup \{C_1, \ldots, C_n\} \models \bigvee_{i,j} A_{ij} \vee B_1 \vee \cdots \vee B_m.$$

As $C_1, \ldots, C_n$ are clauses in Γ_0 and as Γ_0 is saturated, the clause

$$D \equiv \bigvee_{i,j} A_{ij} \vee B_1 \vee \cdots \vee B_m$$

is also in Γ_0. By construction (anyway, either D is productive or not) some of the atoms $\{A_{11}, \ldots, A_{nk_n}, B_1, \ldots, B_m\}$ is in Δ^+. By the remark above, $A_{11}, \ldots, A_{nk_n}$ cannot be there, so one of the B_j's is in Δ^+, contradiction. □

If $\mathcal{M}$ is a Σ-model and $X \subseteq \mathcal{M}$, we can consider $\mathcal{M}$ as a $\Sigma \cup X$-structure by interpreting the name $\bar{b}$ of each $b \in X$ into b. Next lemma uses the assumption 3.3.1:

Lemma 3.3.9. *Let $\mathcal{M}_1$ be a Σ_1-model of T_1 and let $\mathcal{M}_2$ be a Σ_2-model of T_2; suppose also that $\mathcal{M}_1$ and $\mathcal{M}_2$ share a common Σ_0-substructure $\mathcal{A}$. Then there are a $\Sigma_1 \cup \Sigma_2 \cup \mathcal{A}$-model $\mathcal{M}$ of $T_1 \cup T_2$ and two $\Sigma_i^{\mathcal{A}}$-embeddings $\mathcal{M}_i \longrightarrow \mathcal{M}$ ($i = 1, 2$).*

Proof. By T_0-compatibility, we can suppose that the $\mathcal{M}_i$ are models of $T_i \cup T_0^*$. By renaming some elements in the supports if needed, we can also freely suppose that the sets $\mathcal{M}_1 \setminus \mathcal{A}$ and $\mathcal{M}_2 \setminus \mathcal{A}$ are disjoint. Consider the elementary diagrams $\Delta^e(\mathcal{M}_1)$, $\Delta^e(\mathcal{M}_2)$ of $\mathcal{M}_1$, $\mathcal{M}_2$; we show that $\Delta^e(\mathcal{M}_1) \cup \Delta^e(\mathcal{M}_2)$ is consistent as a $\Sigma_1 \cup \Sigma_2 \cup \mathcal{M}_1 \cup \mathcal{M}_2$-theory.

First notice that $T_0^* \cup \Delta(\mathcal{A})$ is a complete $\Sigma_0^{\mathcal{A}}$-theory: this is just condition (ii) of the Definition 3.1.2 of model completion, recalling that every substructure of a model of T_i is a model of T_0, being T_0 a universal theory.

Having established that $T_0^* \cup \Delta(\mathcal{A})$ is a complete theory, we see that $\Delta^e(\mathcal{M}_1)$ and $\Delta^e(\mathcal{M}_2)$ are both extensions of its. Now we can simply invoke Robinson's Joint Consistency Theorem 3.2.3: in our case, extended signatures are $\Sigma_1 \cup \mathcal{M}_1$ and $\Sigma_2 \cup \mathcal{M}_2$, while

the common subsignature is $\Sigma_0 \cup \mathcal{A}$, because the sets $\mathcal{M}_1 \setminus \mathcal{A}$ and $\mathcal{M}_2 \setminus \mathcal{A}$ are disjoint. Having established that $\Delta^e(\mathcal{M}_1) \cup \Delta^e(\mathcal{M}_2)$ is consistent, clearly any model of its fits our $\mathcal{M}$ (notice that such an $\mathcal{M}$ is a model of T_1 and T_2, because it is a model of the elementary diagrams of $\mathcal{M}_1$ and $\mathcal{M}_2$). □

We are now ready to prove Theorems 3.3.3 and 3.3.4.

Proof of Theorem 3.3.3. Suppose that there is no positive residue chain ending up with the empty clause. We build a saturated set Γ_0 of of positive ground $\Sigma_0^{\underline{a}}$-clauses in ω steps. Let Θ_0 be the empty set; if Θ_k has already been defined, let Θ_{k+1} be the set of positive ground $\Sigma_0^{\underline{a}}$-clauses C such that $T_i \cup \Gamma_i \cup \Theta_k \models C$ holds for $i = 1$ or $i = 2$. Clearly $\Theta_k \subseteq \Theta_{k+1}$; moreover, by the compactness theorem for first-order logic, it is clear that $\Gamma_0 = \bigcup_k \Theta_k$ is saturated. Notice also that a clause C belongs to Θ_{k+1} ($k \geq 0$) iff there is a positive residue chain $C_1, \ldots, C_n, C$ such that $C_1, \ldots, C_n$ all belong to Θ_k: this is easily proved by induction on k and by compactness again. The induction step is as follows: if $C \in \Theta_{k+1}$, then there are $C_1, \ldots, C_n \in \Theta_k$ such that $T_i \cup \Gamma_i \cup \{C_1, \ldots, C_n\} \models C$ holds for $i = 1$ or $i = 2$. Now it is sufficient to append C to any juxtaposition of positive residue chains ending up in $C_1, \ldots, C_n$. Consequently, Γ_0 does not contain the empty clause and Lemma 3.3.8 applies. This means that there are models $\mathcal{M}_1 \models T_1 \cup \Gamma_1$ and $\mathcal{M}_2 \models T_2 \cup \Gamma_2$ whose Σ_0-substructures generated by $\underline{a}$ are the same. We can now apply Lemma 3.3.9 to $\mathcal{M}_1, \mathcal{M}_2, \mathcal{A}$ (where $\mathcal{A}$ is this Σ_0-substructure generated by $\underline{a}$). By that Lemma, there are a model $\mathcal{M} \models T_1 \cup T_2$ and $\Sigma_i^{\mathcal{A}}$-embeddings $\mathcal{M}_i \longrightarrow \mathcal{M}$. As $\mathcal{M}_i \models \Gamma_i$, we have also $\mathcal{M} \models \Gamma_i$ ($i = 1, 2$) (recall that the Γ_i's are sets of ground $\Sigma_0^{\underline{a}}$-literals, so their truth is preserved by $\Sigma_i^{\mathcal{A}}$-embeddings). Thus $\mathcal{M} \models T_1 \cup \Gamma_1 \cup T_2 \cup \Gamma_2$, so the latter set is indeed consistent. □

Proof of Theorem 3.3.4. We reduce this Theorem to the previous one. If $T_1 \cup \Gamma_1 \cup T_2 \cup \Gamma_2$ is inconsistent, there is a positive residue chain $C_1, \ldots, C_n$ ending up with the empty clause. Say that C_k is an i-residue ($i = 1, 2$) iff $T_i \cup \Gamma_i \cup \{C_1, \ldots, C_{k-1}\} \models C_k$. Let ψ_k (for $k = 1, \ldots, n$) be the quantifier-free ground $\Sigma_0^{\underline{a}}$-formula $\neg C_1 \vee \cdots \vee \neg C_{k-1} \vee C_k$ and let φ be the conjunction of the ψ_k such that C_k is a 1-residue. Clearly, $T_1 \cup \Gamma_1 \models \varphi$. Moreover, by induction, it is easy to see that $T_2 \cup \Gamma_2 \cup \{\varphi\} \models C_j$ for all $j = 1, \ldots, n$: in fact, if C_j is a 2-residue, then $T_2 \cup \Gamma_2 \cup \{C_1, \ldots, C_{j-1}\} \models C_j$ (hence $T_2 \cup \Gamma_2 \cup \{\varphi\} \models C_j$ by induction hypothesis) and if C_j is a 1-residue, then $\{\varphi\} \models \neg C_1 \vee \cdots \vee \neg C_{j-1} \vee C_j$ (hence $T_2 \cup \Gamma_2 \cup \{\varphi\} \models C_j$ again by induction hypothesis). As $T_2 \cup \Gamma_2 \cup \{\varphi\} \models C_j$ holds for all j, we have in particular that $T_2 \cup \Gamma_2 \cup \{\varphi\} \models \bot$ for $j = n$. □

3.4 Locally Finite Theories and their Combinations

An Σ_0-universal theory T_0 is said to be *locally finite* iff

(i) Σ_0 is finite;

(ii) for every finite set $\underline{a}$ of new free constants, there are finitely many $\Sigma_0^{\underline{a}}$-ground terms $t_1, \ldots, t_{k_{\underline{a}}}$ such that for every further $\Sigma_0^{\underline{a}}$-ground term u, we have $T_0 \models u = t_i$ (for some $i = 1, \ldots, k_{\underline{a}}$).

Condition (ii) simply means that in the class of the structures in which T_0 is interpreted, every $\Sigma_0^{\underline{a}}$-ground term is equal, as an interpreted function, to one of the t_i. The terms $t_1, \ldots, t_n$ are called the $\underline{x}$-representative terms of Φ_0.

As we are mainly dealing with computational aspects, we consider part of the definition the further request that such $t_1, \ldots, t_{k_{\underline{a}}}$ are effectively computable from $\underline{a}$. In this case T_0 is said to be *effectively locally finite*. Examples of locally finite theories are the theory of graphs, of partial orders (more generally, any theory whose signature does not contain function symbols), of commutative idempotent monoids, of Boolean algebras, etc.

In a locally finite theory T_0, there are restricted *finite* classes which are representatives, up to T_0-equivalence, of the whole classes of $\Sigma_0^{\underline{a}}$-ground literals, clauses, quantifier-free sentences, etc. (they are just the ground literals, clauses, quantifier-free sentences, etc. containing only the above mentioned representative terms $t_1, \ldots, t_{k_{\underline{a}}}$). As it is evident that we can limit information exchange to ground positive clauses and quantifier-free sentences in that restricted class, both Theorems 3.3.3 and 3.3.4 yield *combined decision procedures for the constraint satisfiability problem* in $T_1 \cup T_2$, in case T_0 is effectively locally finite and in case the corresponding constraint satisfiability problems for T_1 and T_2 are separately decidable.

Recalling the observations made after the Corollary 3.3.5, the procedure suggested by Theorem 3.3.3 is just a fair information exchange of positive ground $\Sigma_0^{\underline{a}}$-clauses, to be continued until the situation gets stable or until an inconsistency is detected. Notice that in case T_1 and T_2 are Σ_0-convex theories, information exchange can be further limited to ground $\Sigma_0^{\underline{a}}$-atoms by Corollary 3.3.5. This observation, as shown in [73] for the disjoint signatures case, may improve complexity bounds in certain significant situations.

Besides the already mentioned procedure suggested by Theorem 3.3.4, there is a third possible (non-deterministic) procedure, which is justified directly by Lemma 3.3.9: as there are only finitely many Σ_0-structures generated by $\underline{a}$ which are models of T_0 (recall that such structures cannot have more than $k_{\underline{a}}$-elements), one simply guesses one of them and check whether its diagram is consistent with both $T_1 \cup \Gamma_1$ and $T_2 \cup \Gamma_2$.

We give a first example to which the above outlined combined procedures apply.

Example 3.4.1. Let T_1 be rational linear arithmetic and let T_2 be the theory of total orders endowed with a strict monotonic function f. This means that f is subject to the axiom $\forall x \forall y\,(x < y \rightarrow f(x) < f(y))$. We take as T_0 the theory of total orders, whose model completion T_0^* is the theory of dense total orders without endpoints. T_1 is known to be decidable (see [19]). The universal fragment of T_2 is decidable too: in fact, it is easy to prove that any finite total order endowed with a *partial* strict monotonic function embeds into a model of T_2 (this is shown by successively inserting new points and by taking union in the limit). Henceforth the satisfiability of a set Γ_2 of Σ_2^a-ground literals can be decided by a non-deterministic guessing of such a finite total order endowed with a partial strict monotonic function. As $T_1 \supseteq T_0^*$, T_1 is certainly T_0-compatible. We only need to show that T_2 is T_0-compatible, by embedding each model $\mathcal{M}$ of T_2 into a model $\mathcal{M}'$ of $T_0^* \cup T_2$. It is sufficient to take as $\mathcal{M}'$ the lexicographic product of $\mathcal{M}$ with e.g. the poset of rational numbers. For density, observe that $(b, m) <_{lex} (b', m')$ implies $(b, m) <_{lex} (b, n) <_{lex} (b', m')$, where $n = \frac{m+m'}{2}$ in case $m < m'$ and $n = m + 1$ otherwise (notice that in this last case, we must have $b < b'$). The symbol f is interpreted by putting $f^{\mathcal{M}'}(b, m) = (f^{\mathcal{M}}(b), m)$, and the embedding $\mathcal{M} \longrightarrow \mathcal{M}'$ is defined by associating with $b \in \mathcal{M}$ the pair $(b, 0)$. Thus the combination results apply and we obtain the decidability of the universal fragment of the rational linear arithmetic endowed with a strict monotonic function. It is not difficult to see that the complexity of this combined decision algorithm, if applied to satisfiability of the existential closure of a quantifier-free formula, lies in the NP-class (just adapt the arguments in [73]).

Application to Fusion Decidability in Modal Logic In order to explain the applications to fusion decidability in modal logic, we need to fix some terminology. We shall not directly introduce modal logic, rather we insist on its algebraic counterpart (this might look not very orthodox, but makes things more simple for the purposes of this section). A *modal algebra* is just a Boolean algebra $\mathcal{B} = \langle B, \cap, 1, \cup, 0, (-)' \rangle$ endowed with an hemimorphism $\Box$ (a hemimorphism is a function preserving only binary meets and the top element). Hemimorphisms are also called modal (necessity) operators. The modal operator $\Box$ is said to be *transitive* iff the identity $\Box a \leq \Box\Box a$ holds for every $a \in B$.[1] Let now Σ_1 be the signature of Boolean algebras augmented with a unary function symbol $\Box_1$ and let Σ_2 be the signature of Boolean algebras augmented with a unary function symbol $\Box_2$. T_1 is the equational theory of a variety V_1 of modal algebras and T_2 is the equational theory of another variety V_2 of modal algebras. For $i = 1, 2$, T_i is a universal Horn theory, hence it is Σ_i-convex.

Proposition 3.4.2. *Let T_1, T_2 be as above; then the decidability of the universal Horn fragments of T_1 and T_2 implies the decidability of the universal Horn fragment of $T_1 \cup T_2$*

[1] Recall that, in a Boolean algebra, $y \leq z$ is a shorthand for $y \cap z = y$.

Proof. As already mentioned, we have for free the decidability of the universal fragment of T_1 and T_2; we take as T_0 the theory of Boolean algebras (which is locally finite and admits as a model completion the theory of atomless Boolean algebras): in order to apply the combination results, we simply need to show that T_1, T_2 are T_0-compatible. We do it for T_1. Let $\mathcal{M}$ be a model of T_1, we show how to embed it into a model $\mathcal{M}'$ of T_1 which is based on an atomless Boolean algebra: this is a well known and rather trivial fact, which is used also in [93] as a side preliminary lemma. Instead of reporting the argument used in [93], we give a more direct one. Define a sequence of models of T_1 by: $\mathcal{M}_0 := \mathcal{M}$, $\mathcal{M}_{k+1} := \mathcal{M}_k \times \mathcal{M}_k$; define also embeddings $\delta_k : \mathcal{M}_k \longrightarrow \mathcal{M}_{k+1}$ by $\delta_k(a) := \langle a, a\rangle$. Now take as $\mathcal{M}'$ the union (more precisely, the inductive limit) of this chain: clearly $\mathcal{M}'$ is atomless as a Boolean algebra (no non-zero element is minimal in it as any $a \in \mathcal{M}_k$ gets identified with $\langle a, a\rangle = \langle a, 0\rangle \cup \langle 0, a\rangle$ in $\mathcal{M}_{k+1}$). □

3.5 Positive Basis Enumerators

Now we want to weaken the assumption of local finiteness. Let T be a theory over the signature Σ, and $\underline{a}$ be a finite set of new constants. Given a set Γ of ground $\Sigma^{\underline{a}}$-clauses and a subtheory $T_0 \subseteq T$ over the signature $\Sigma_0 \subseteq \Sigma$, we call T_0-*basis for* Γ a set Δ of ground $\Sigma_0^{\underline{a}}$-clauses such that:

(i) all the clauses $D \in \Delta$ are positive and are such that $T \cup \Gamma \models D$;

(ii) for every positive ground $\Sigma_0^{\underline{a}}$-clause C, $T \cup \Gamma \models C$ if and only if $T_0 \cup \Delta \models C$.

For the sake of simplicity, we conventionally included $\bot$ among the atoms over any signature Σ, so $\bot$ is considered as a positive clause.

We are interested in exchange information concerning consequences over shared signatures, thus we need a notion that reminds the notion of residue, like in partial theory reasoning (see, e.g., [18] for comprehensive information on the subject and the relevant pointers to the literature). In Chapter 4 we will develop an abstract approach and we will treat residues as clauses which are recursively enumerated by a suitable device; at the moment a more superficial notion of basis enumerator will be sufficient to our aim.

Definition 3.5.1. Suppose we are given a subtheory T_0 of a theory T over the signature $\Sigma_0 \subseteq \Sigma$. A *positive basis enumerator for* T_0 (often shortened as T-p.b.e.) is a recursive function mapping a finite set $\underline{a}$ of constants and a finite set Γ of ground $\Sigma^{\underline{a}}$-clauses to a finite set of positive ground $\Sigma_0^{\underline{a}}$-clauses $\mathsf{B}_T^{\underline{a}}(\Gamma)$ (to be written simply as $\mathsf{B}_T(\Gamma)$) in such a way that:

– $T \cup \Gamma \models \mathsf{B}_T(\Gamma)$ (soundness);

– if $T \cup \Gamma \models \bot$, then $\mathsf{B}_T(\Gamma)$ is equal to $\bot$; otherwise $\mathsf{B}_T(\Gamma)$ is T_0-basis for Γ.

A positive basis enumerator is simply a machinery able to return a finite set of positive ground clauses that are "representative" of the logical consequences of a set of ground clauses, modulo a theory T. Actually, our positive basis T-enumerator is a little bit better, because it is able to point inconsistencies out. Notice however that, since we cannot always guarantee the existence of finite and recursive bases, a positive basis T-enumerator may not exist.

3.5.1 Noetherian Theories and their Combinations

The above mentioned notion of p.b.e.'s usually applies to the cases in which the subtheory T_0 is noetherian: this important notion is borrowed from algebra. Noetherianity conditions known from algebra (see, e.g., [64]) say that there are no infinite ascending chains of congruences. In finitely presented algebras, congruences are represented as sets of equations among terms, hence noetherianity can be expressed there by saying that there are no infinite ascending chains of sets of atoms, modulo logical consequence. If we translate this into our setting, we get the following definition.

A Σ_0-theory T_0 is called *noetherian* if and only if for every finite set of constants $\underline{a}$, every infinite ascending chain

$$\Theta_1 \subseteq \Theta_2 \subseteq \cdots \subseteq \Theta_n \subseteq \cdots$$

of sets of ground $\Sigma_0^{\underline{a}}$-atoms is eventually constant for T_0-consequence (meaning that there is an n such that for all m and $A \in \Theta_m$, we have $T_0 \cup \Theta_n \models A$).

In a noetherian theory the lack of strict ascending chains of positive ground atoms forbids the existence of strict ascending chains of positive ground clauses.

Proposition 3.5.2. *In a noetherian theory T_0 every infinite ascending chain of sets of positive $\Sigma_0^{\underline{a}}$-clauses is eventually constant for T_0-consequence.*

Proof. Suppose not; in this case there are infinitely many positive ground $\Sigma_0^{\underline{a}}$-clauses $C_1, C_2, \ldots$ such that for all i, the clause C_i is not a T_0-consequence of $\{C_k \mid k < i\}$.[2]

Let us build a chain of trees $\mathcal{T}_0 \subseteq \mathcal{T}_1 \subseteq \mathcal{T}_2 \subseteq \cdots$, whose nodes are labeled by ground $\Sigma_0^{\underline{a}}$-atoms as follows. $\mathcal{T}_0$ consists of the root only, which is labeled $\top$. Suppose $\mathcal{T}_{i-1}$ is already built and consider the clause $C_i \equiv B_1 \vee \cdots \vee B_m$. To build $\mathcal{T}_i$, do the following for every leaf K of $\mathcal{T}_{i-1}$ (let the branch leading to K be labeled by $A_1, \ldots, A_k$): append new sons to K labeled $B_1, \ldots, B_m$, respectively, if C_i is not a T_0-consequence of $\{A_1, \ldots, A_k\}$ (if it is, do nothing for the leaf K).

[2]This is an equivalent formulation of the negation of the statement of the proposition.

Consider now the union tree $\mathcal{T} := \bigcup \mathcal{T}_i$: since, whenever a node labeled A_{k+1} is added, A_{k+1} is not a T_0-consequence of the formulae labeling the predecessor nodes, by the noetherianity of Φ_0, all branches are then finite and by König lemma the whole tree is itself finite. This means that for some index j, the examination of clauses C_i (for $i > j$) did not yield any modification of the already built tree. Now, C_{j+1} is not a T_0-consequence of $\{C_1, \ldots, C_j\}$: this means that there is a Σ_0^a structure which is a model of T_0, in which all atoms of C_{j+1} are false and one atom in each of the $C_1, \ldots, C_j$ is true. By induction on $j = 0, \ldots, i$, it is easily seen that there is a branch in $\mathcal{T}_j$ whose labeling atoms are true in $\mathcal{A}$: this contradicts the fact that the tree $\mathcal{T}_i$ has not been modified in step $i + 1$. □

Suppose that T_0 is noetherian and that T is a theory extending T_0: by the above proposition, it is immediate to see that every finite set of ground $\Sigma^{\underline{a}}$-clauses Γ has a finite T_0-basis. This is the reason why noetherian theories are so suitable to deal with our notion of p.b.e.. The role of effective local finiteness is to make Nelson-Oppen procedures terminating (see also [44, 9, 8]): however noetherianity is a weaker condition that will turn out to be already sufficient for that, once it is accompanied by a suitable effectiveness condition.

Proposition 3.5.3. *If T_0 is an effectively locally finite theory, T is a theory extending T_0, and T has decidable constraint satisfiability problem, then there always exists a T-p.b.e. B_T for T_0.*

Proof. Once a finite number of $\Sigma^{\underline{a}}$-clauses Γ is given, first test Γ for consistency: if it is inconsistent, $\mathsf{B}_T(\Gamma)$ just returns $\bot$. If it is consistent, test the finitely many positive clauses containing the representative atoms $P(t_{i_1}, \ldots, t_{i_n})$ for being a T-consequence of Γ (here the t_i's are the finitely many representative terms involving the constants among the $\underline{a}$'s). $\mathsf{B}_T(\Gamma)$ will coincide with the set of all the clauses whose test is positive. Notice that this test can be effectively done, because the constraint satisfiability problem for T is decidable. □

Now we are ready to give a formal description of an algorithm for the constraint satisfiability problem for a combined theory $T_1 \cup T_2$.

The result of Theorem 3.3.3 can be restated in this new context:

Theorem 3.5.4. *Suppose that T_1 and T_2 are two theories respectively over the signature Σ_1 and Σ_2. Let Σ_0 be the subsignature common to Σ_1 and Σ_2 (i.e: $\Sigma_0 := \Sigma_1 \cap \Sigma_2$), and T_0 a universal theory such that:*

(i) *T_1 and T_2 are both T_0-compatible;*

(ii) *T_0 is noetherian.*

Algorithm 4 Extending Nelson-Oppen

Step 1. Purify the finite **input** set of ground literals Γ, thus producing, for some finite set $\underline{a}$ of free constants, a finite set Γ_1 of ground $\Sigma_1^{\underline{a}}$-literals and finite set Γ_2 of ground $\Sigma_2^{\underline{a}}$-literals (then $\Gamma_1 \cup \Gamma_2$ is $T_1 \cup T_2$-equisatisfiable with Γ). In the next loop, positive ground $\Sigma_0^{\underline{a}}$-clauses are added to Γ_1 and Γ_2;

Step 2. Using the T_1-p.b.e. B_{T_1} and T_2-p.b.e. B_{T_2}, check the output of $\mathsf{B}_{T_i}(\Gamma_i)$:

Step 2.1. if for at least an index $i \in \{1, 2\}$, $\mathsf{B}_{T_i}(\Gamma_i)$ is $\bot$, **return** *"unsatisfiable"*;

Step 2.2. if $\mathsf{B}_{T_i}(\Gamma_i) = \Delta_i$ and $\Delta_i \neq \bot$, for each $D \in \Delta_i$ such that $T_j \cup \Gamma_j \not\models D$, $(i \neq j)$, add D to Γ_j and rerun **Step 2**;

Step 3. If this step is reached, **return** *"satisfiable"*.

Suppose also that there exist a T_1-p.b.e. B_{T_1} for T_0 and a T_2-p.b.e. B_{T_2} for T_0.

Under these hypotheses the above Algorithm 4 is a decision procedure for the constraint satisfiability problem for $T_1 \cup T_2$.

Proof. First, we show that the algorithm always halts. In fact, the noetherianity of T_0 and Proposition 3.5.2 imply that only a finite number of positive ground $\Sigma_0^{\underline{a}}$-clauses can be added to Γ_1 and Γ_2; henceforth, either $\bot$ is discovered at a certain point, or eventually the loop will end.

Since the positive basis enumerator's are sound and the message *"unsatisfiable"* is returned if and only if at least a positive basis enumerator has discovered an inconsistency, we can conclude that, if the output of an execution of the Algorithm 4 on the constraint $\Gamma_1 \cup \Gamma_2$ is *"unsatisfiable"*, then $\Gamma_1 \cup \Gamma_2$ is unsatisfiable modulo $T_1 \cup T_2$.

On the other hand, if the final message is *"satisfiable"*, then (i) there exists a set of positive ground $\Sigma_0^{\underline{a}}$-clauses $\{C'_1, \ldots, C'_l\}$ such that $\mathsf{B}_{T_1}(\Gamma_1 \cup \{C'_1, \ldots, C'_l\}) = \Delta_1$, (ii) there exists a set of positive ground $\Sigma_0^{\underline{a}}$-clauses $\{C''_1, \ldots, C''_m\}$ such that $\mathsf{B}_{T_2}(\Gamma_2 \cup \{C''_1, \ldots, C''_m\}) = \Delta_2$, (iii) Δ_1 and Δ_2 are sets different from $\{\bot\}$; moreover (iv) for each clause $D_1 \in \Delta_1$, $T_2 \cup \Gamma_2 \cup \{C''_1, \ldots, C''_m\}) \models D_1$, and, (v) for each clause $D_2 \in \Delta_2$, $T_1 \cup \Gamma_1 \cup \{C'_1, \ldots, C'_l\}) \models D_2$.

As B_{T_1} and B_{T_2} are positive basis enumerator's, Δ_1 is a T_0-basis for $\Gamma_1 \cup \{C'_1, \ldots, C'_l\}$ and Δ_2 is a T_0-basis for $\Gamma_2 \cup \{C''_1, \ldots, C''_m\}$; recalling conditions (iv) and (v), this is sufficient to conclude that Δ_1 and Δ_2 are logically equivalent modulo T_0.

We want to show that it is possible to obtain a saturated set of clauses not containing $\bot$. Let us consider the set $\Delta^* := \{C \mid T_0 \cup \Delta_1 \models C\}$. Δ^* is a saturated set. Recalling the definition, (see Definition 3.3.7), we have to prove that, for every positive ground $\Sigma_0^{\underline{a}}$-clause C such $T_1 \cup \Gamma_1 \cup \Delta^* \models C$, then $C \in \Delta^*$. But $T_1 \cup \Gamma_1 \cup \Delta^* \models C$ iff $T_1 \cup \Gamma_1 \cup \Delta_1 \models C$. Being Δ_1 a T_0-basis for $(\Gamma_1) \cup \{C'_1, \ldots, C'_l\}$, it follows that $T_0 \cup \Delta_1 \models C$, henceforth

$C \in \Delta^*$. Analogously, being Δ_2 logically equivalent modulo T_0 to Δ_1, every positive ground $\Sigma_0^{\underline{a}}$-clause D such $T_2 \cup \Gamma_2 \cup \Delta^* \models D$, is already a member of Δ^*.

Δ^* does not contain the empty clause. In fact, suppose it does. Thus $T_1 \cup \Gamma_1 \cup \Delta^* \models \bot$. This implies that $T_1 \cup \Gamma_1 \cup \Delta_1 \models \bot$; this happens if and only if $T_1 \cup \Gamma_1 \cup \{C'_1, \ldots, C'_l\} \models \bot$; by the definition of B_{T_1}, $\mathsf{B}_{T_1}(\Gamma_1 \cup \{C'_1, \ldots, C'_l\})$ should be equal to $\bot$: contradiction with the hypothesis that $\mathsf{B}_{T_1}(\Gamma_1 \cup \{C'_1, \ldots, C'_l\}) = \Delta_1$, $\Delta_1 \neq \{\bot\}$.

Thus we have proved that, if the final message is *"satisfiable"*, it is possible to obtain a saturated set of clauses not containing $\bot$. From now on we follow the proof of Theorem 3.3.3: at this point Lemma 3.3.8 applies, implying that there are models $\mathcal{M}_1 \models T_1 \cup \Gamma_1$ and $\mathcal{M}_2 \models T_2 \cup \Gamma_2$ whose Σ_0-substructures generated by $\underline{a}$ are the same. By Lemma 3.3.9, choosing as a common substructure $\mathcal{A}$ the Σ_0-substructure generated by $\underline{a}$, there are a model $\mathcal{M} \models T_1 \cup T_2$ and two $\Sigma_i^{\mathcal{A}}$-embeddings $\mathcal{M}_i \longrightarrow \mathcal{M}$. As $\mathcal{M}_i \models \Gamma_i$, we have also $\mathcal{M} \models \Gamma_i$ $(i = 1, 2)$. □

Noetherianity is the essential ingredient for the termination of Nelson-Oppen combination procedures. Let us make some observations concerning an efficiency issue. In order to decide the satisfiability in a Σ theory T of ground clauses (or, more in general, of arbitrary ground formulae over a simple expansion $\Sigma^{\underline{a}}$) it is sufficient to decide the satisfiability in T of conjunction of ground literals over $\Sigma^{\underline{a}}$. In fact we can convert the ground formula we are checking for satisfiability modulo T into disjunctive normal form, and then we can test for satisfiability each disjunct. The original formula is satisfiable if and only if at least one of the disjuncts is. This approach is extremely inefficient because of the exponential blow up due to the conversion in disjunctive normal form.

In concrete cases, the most common solution is an integration of a decision procedure for the constraint satisfiability problem with a SAT-solver implementing refinements of the DPLL algorithm (see [30, 29]). A general framework that enables to combine decision procedures with SAT-solvers in such a way that can incorporate various types of optimization developed for the DPLL method can be found in [84]. Even in this refined case, deciding the satisfiability of ground clauses requires case splitting. This is the reason why, for efficiency issues, convexity is the crucial property.

Following the definition introduced in Section 2.1, a theory T is Σ_0-convex iff every finite set Γ of literals over Σ having as a T-consequence the disjunction of $n > 1$ Σ_0-atoms, actually has as a T-consequence one of them. Similarly, a T-p.b.e. for T_0 is Σ_0-*convex* iff for every set Γ of literals, $\mathsf{B}_T(\Gamma)$ is an atom (recall that by our conventions, this includes the case in which it is $\top$ or $\bot$). Any T-p.b.e. for T_0 can be turned into a Σ_0-convex T-p.b.e. for T_0, in case T is Σ_0-convex. As already hinted, the advantage lies on the efficiency of the procedure. In fact, **Step 2.2** of Algorithm 4 requires the calculus of a basis, which usually underlies a satisfiability test for a set of clauses, and an exchange

of clauses between the two p.b.e.'s. It is clear that, if the sets returned by the p.b.e.'s contain only atoms, i.e. the component theories are both convex with respect to the shared subtheory, the combination procedure checks the satisfiability only of sets of atoms, thus considerably improving in efficiency.

3.6 Examples

In this subsection we collect some examples which concretely illustrate the notions of a noetherian theory and of a noetherian T-p.b.e..

Example 3.6.1 (K-algebras)**.** Given a field K, let us consider the one-sorted signature $\Sigma_{K\text{-}alg}$ containing the constants $0, 1$, the two binary function symbols $+$, $\circ$, the unary function symbol $-$ and a K-indexed family of unary function symbols g_k (or simply k). We shall use infix notation for $+$ and write kv, v_1v_2 for $g_k(v)$, $\circ(v_1, v_2)$, respectively. As we want to describe the theory $K\text{-}alg$ of (commutative, for simplicity) K-algebras, the axioms will be the following:

(1) Abelian additive group axioms:

$$\begin{gathered}\forall x\, y\, z\ (x + y) + z = x + (y + z)\\ \forall x\ x + 0 = x\\ \forall x\ x - x = 0\\ \forall x\, y\ x + y = y + x\end{gathered}$$

(2) Abelian multiplicative monoid:

$$\begin{gathered}\forall x\, y\, z\ (xy)z = x(yz)\\ \forall x\ x1 = x\\ \forall x\, y\ xy = yx\end{gathered}$$

(3)

$$\forall x\, y\, z\ (x + y)z = xz + yz \tag{3.3}$$
$$\forall x\, y\ k(x + y) = kx + ky \qquad \text{for all } k \in K \tag{3.4}$$
$$\forall x\ (k_1 \oplus k_2)x = k_1x + k_2x \qquad \text{for all } k_1, k_2 \in K \tag{3.5}$$
$$\forall x\ (k_1 \cdot k_2)x = k_1(k_2x) \qquad \text{for all } k_1, k_2 \in K \tag{3.6}$$
$$\forall x\ 1_Kx = x \tag{3.7}$$
$$\forall x\, y\ k(xy) = (kx)y = x(ky) \qquad \text{for all } k \in K \tag{3.8}$$

where $\oplus$ and $\cdot$ are respectively the sum and multiplication operation in K, whereas 1_K is the multiplicative unit of K.

The ground atoms over a simple expansion $\Sigma^{\underline{a}}_{K\text{-}alg}$ of $\Sigma_{K\text{-}alg}$ have a normalized representation as $p(\underline{a}) = 0$, where p is a polynomial; the theory of K-algebras is equational, hence convex, so that the constraint satisfiability problem is just the problem of deciding whether an equation $p(\underline{a}) = 0$ is a logical consequence of a finite number of equations

$\{p_1(\underline{a}) = 0, \ldots, p_n(\underline{a}) = 0\}$. Since the polynomial ring $K[a_1, \ldots, a_n]$ is the free K-algebra on n-generators, this problem is equivalent to the membership of the polynomial p to the ideal $\langle p_1, \ldots, p_n \rangle$ generated by the polynomials $p_1, \ldots, p_n$. The Buchberger's algorithm solves the problem by computing the Gröbner basis associated to the ideal $\langle p_1, \ldots, p_n \rangle$ (Gröbner basis can morally be considered as confluent and terminating rewriting systems for deciding the universal fragment of theory of K-algebras). Since, by Hilbert's basis theorem, there exists no infinite properly ascending chain of ideals in $K[a_1, \ldots, a_n]$, *K-alg* is a noetherian theory.

Example 3.6.2 (K-vector spaces)**.** As a subtheory of the previous theory, let us consider the theory *K-vect* of K-vector spaces, whose signature $\Sigma_{K\text{-}vect}$ is obtained from $\Sigma_{K\text{-}alg}$ forgetting the ring multiplication $\circ$ and the ring unit 1. The axioms for *K-vect* are now axioms (1), (3.4), (3.5), (3.6) and (3.7). The ground terms over a simple expansion of $\Sigma_{K\text{-}vect}$ can consequently be represented as linear homogeneous polynomials with K-coefficients. *K-vect* is noetherian because there does not exist an infinite properly ascending chain of subspaces of a finitely generated K-vector space, but it is not locally finite because linear polynomials in n unknowns are infinite (if coefficients are infinite). In order to obtain a *K-alg*-p.b.e. for *K-vect* through Buchberger algorithm, we need membership of a linear polynomial to a finitely generated ideal to be decided only by linear reduction rules (in this case, our *K-alg*-p.b.e. for *K-vect may simply extract the linear polynomials from a Gröbner basis*). This can be obtained through a further requirement on the admissibility of the term ordering (in fact, since we need that linear terms can be rewritten to linear ones only, not every admissible order is suitable). As an example of an order matching such a requirement, one can take for instance the following one (see [11]): say that $a_1^{m_1} \cdots a_n^{m_n} \prec a_1^{l_1} \cdots a_n^{l_n}$ holds iff (1) $\sum_{i=1}^n m_i < \sum_{i=1}^n l_i$ or (2) $\sum_{i=1}^n m_i = \sum_{i=1}^n l_i$ and $(m_1, \cdots, m_n) <_{lex}^n (l_1, \cdots, l_n)$ (where $<_{lex}^n$ is the lexicographic extension of the natural numbers ordering). We can now briefly outline how to extract from the Buchberger's algorithm a *K-alg*-p.b.e. for *K-vect*. First notice that, according to the above requirement on the ordering for polynomials, only linear rewrite rules can be used to reduce a linear term. Thus in order to compute a *K-vect*-basis for the *K-alg*-constraint $\Gamma = \{p_1 = 0, \ldots, p_n = 0, r_1 \neq 0, \ldots, r_m \neq 0\}$ it is sufficient to: (a) run the Buchberger's algorithm procedure on $\{p_1, \ldots, p_n\}$ (let Q be the corresponding Gröbner basis); (b) normalize the r_j w.r.t. $\rightarrow_Q$ to check consistency; (c) if consistency holds, return the equations $q = 0$, where q in the basis is a linear polynomial (if consistency does not hold, simply return $\perp$).

Example 3.6.3. The observations of the previous example concerning noetherianity and convexity of the theory *K-vect* apply also in the analogous case of the theory of modules over a noetherian ring R. In fact, ground terms over a simple expansion of the signature

of the theory of R-modules represent elements in finitely generated free R-modules and, as a generalization of Hilbert's basis theorem, the following theorem holds (see, e.g., [64]).

Theorem 3.6.4. *Given a noetherian ring R, any finitely generated R-module is noetherian.*

This implies that the theory of abelian groups is a noetherian theory and, since integer or rational arithmetic (namely the theory of the integers or of the rationals under addition) is an extension of the latter, it is noetherian too (however noetherianity is lost if we add to the language a predicate symbol for the ordering).

We give a further example (to be combined with the fragment of Example 3.6.1):

Example 3.6.5 (K-vector spaces endowed with an endomorphism)**.** This is an extension of the theory *K-vect* in Example 3.6.2. We augment the signature $\Sigma_{K\text{-}vect}$ with a unary function symbol f and we add the following two axioms to the axioms of *K-vect*:

$$\forall x\, y\ f(x+y) = f(x) + f(y) \tag{3.9}$$

$$\forall x\ f(kx) = kf(x) \qquad \text{for all } k \in K \tag{3.10}$$

This theory - call it *f-K-vect* - is the theory of the K-vector spaces endowed with an endomorphism. Ground terms over a simple expansion $\Sigma^{a}_{f\text{-}K\text{-}vect}$ of $\Sigma_{f\text{-}K\text{-}vect}$ formally represent vectors in finitely generated free K-algebras endowed with an endomorphism, and hence normalize to the form $k_1 f^{m_1}(a_{i_1}) + \cdots + k_n f^{m_n}(a_{i_n})$ ($k_j \in K$, $j \in \{1, \ldots, n\}$). Given the convexity of the fragment, constraint satisfiability amounts to decide if an equation $p = 0$ is a logical consequence of given equations $p_1 = 0, \ldots, p_n = 0$. This is equivalent to check (in free f-K-vector spaces) if the vector p belongs to the subspace $\langle p_1, \ldots, p_n \rangle_f$ obtained from the closure under the endomorphism f of the subspace $\langle p_1, \ldots, p_n \rangle$ generated by the vectors $p_1, \ldots, p_n$. In order to show the decidability of the problem, we can use standard completion methods (reproducing Buchberger algorithm): we show here what to do, leaving the details to the reader.

(i) Let us call 'vector components' the terms of the form $f^m(a_i)$; vector components must be given a *terminating total order* (satisfying suitable admissibility requirements). An example of such an ordering is the following: define $f^{n_1}(a_i) \prec f^{n_2}(a_j)$ iff $\langle n_1, i \rangle <_{lex} \langle n_2, j \rangle$ (where $<_{lex}$ is the lexicographic order on $\mathbb{N} \times \mathbb{N}$). We will call head component the greatest vector component occurring in a term p (denoted by $H(p)$), head coefficient the coefficient of the head component (denoted by $HC(p)$) and remainder the term $R(p) = p - HC(p) \cdot H(p)$.

(ii) Given a finite set P of terms with head coefficient 1, the *reduction relation* $\rightarrow_P$ is introduced as follows: $p_1 \rightarrow_P p_2$ holds iff for some $p \in P$

(a) p_1 contains a component m whose coefficient is $k \neq 0$;

(b) there exists n such that $m = f^n(H(p))$;

(c) $p_2 = p_1 - k \cdot f^n(p)$.

(iii) Critical pairs are identified with *S-vectors*, in the sense that $\rightarrow_P$ is confluent iff all S-vectors $S(p_l, p_r)$ (for $p_l, p_r \in P$) normalize to 0. Here if $p_l = H(p_l) + R(p_l)$ and $p_r = H(p_r) + R(p_r)$, the term $S(p_l, p_r)$ is defined as follows:

- if there exists n such that $H(p_i) = f^n(H(p_j))$ $(i, j \in \{l, r\},\ i \neq j)$, then $S(p_l, p_r) = p_i - f^n(p_j)$;
- $S(p_l, p_r) = 0$ otherwise.

(iv) A terminating *completion procedure* turns an arbitrary $\rightarrow_P$ into some confluent and equivalent $\rightarrow_Q$: the procedure simply adds a normal form of the S-vectors that do not normalize to 0 to the current set of terms. For efficiency reasons, the procedure may also perform backward simplification steps.

We finally mention how to extract from the above completion procedure a *f-K-vect*-p.b.e. for *K-vect*: to this aim, notice that, according to the above ordering for vector components, only f-free rewrite rule can be used to reduce an f-free term. So it is possible to compute a *K-vect*-basis for the *f-K-vect*-constraint $\Gamma = \{p_1 = 0, \ldots, p_n = 0, r_1 \neq 0, \ldots, r_m \neq 0\}$ by running the completion procedure on $\{p_1, \ldots, p_n\}$ and by following the steps (b), (c) of Example 3.6.2.

We consider now the combined theory $T_1 \cup T_2$, where T_1 is the theory *K-alg* given in the Example 3.6.1 and T_2 is the fragment *f-K-vect* of Example 3.6.5 (here, however, we require K-algebras to be non degenerate, i.e. to satisfy the condition $0 \neq 1$). From the axioms of the two theories, it follows that the class of models for $T_1 \cup T_2$ consists of the models of the theory of the non degenerate K-algebras endowed with a linear endomorphism (i.e. endowed with a function f preserving sum and scalar multiplication). The common subtheory T_0 is the theory *K-vect* of K-vector spaces given in Example 3.6.2. Such a theory is universal and admits as a model completion the theory $T_0^\star = T_0 \cup \{\exists x\,(x \neq 0)\}$, if K is an infinite field, and the theory $T_0 \cup \{\exists x_1 \cdots \exists x_n \bigwedge_{i \neq j} x_i \neq x_j\}_{n \in \mathbb{N}}$, otherwise: in both cases, the models of $T_0^\star$ are just infinite K-vector spaces. Since every non degenerate K-algebra (resp. every *f-K*-vector space) can be embedded into an infinite K-algebra (resp. into an infinite *f-K*-vector space), the requirements about the T_0-compatibility of both *K-alg* and *f-K-vect* are completely fulfilled. The condition concerning the existence of a *K-alg*-p.b.e. and of a *f-K-vect*-p.b.e. for $T_0 = K$*-vect* is also satisfied, as pointed out in Examples 3.6.1, 3.6.2 and 3.6.5 above. Hence the combination procedure given by

Algorithm 4 decides the constraint satisfiability problem for the theory of (non degenerate) K-algebras endowed with a linear endomorphism.

Example 3.6.6 (Linear arithmetic over reals)**.** First of all, let us fix some notation conventions. Given a set of variables $\{x_1, \ldots, x_n\}$, each inequality C has the form $\sum_{i=1}^{n} a_i x_i \leq b$; moreover, if e is the linear equation $\sum_{i=1}^{n} a_i x_i = b$, the symbol $\overline{e}$ denotes the corresponding inequation $\sum_{i=1}^{n} a_i x_i \neq b$; finally, if C is the inequality $\sum_{i=1}^{n} a_i x_i \leq b$, $C^{=}$ denotes the equation $\sum_{i=1}^{n} a_i x_i = b$ obtained from C replacing the simbol $\leq$ by the symbol $=$ (similarly is defined $C^{<}$).

Let us now recall two important properties of systems of linear equations and inequalities with coefficients in the real field. The following two theorems hold:

Theorem 1. *If P is a finite set of inequalities and $\overline{e}_1, \overline{e}_2, \ldots, \overline{e}_m$ are inequations, $P \cup \{\overline{e}_1, \ldots, \overline{e}_m\}$ has a solution if and only if $P \cup \{\overline{e}_i\}$ has a solution for every i.*

Theorem 2. *Given a finite set of inequalities $P = \{C_1, \ldots C_m\}$, let S_P be the set of solutions of P, let $S_{C_k^{=}}$ be the set of solutions of the equation $C_k^{=}$, and let I_A be the set of indexes such that $k \in I_A$ if and only if $S_P \subseteq S_{C_k^{=}}$; moreover, given an equation e, let S_e denote the set of solution of e. If e is an equation such that $S_P \subseteq S_e$, then $\bigcap_{k \in I_A} S_{C_k^{=}} \subseteq S_e$.*

The proofs of Theorems 1 and 2 can be found, for example, in [60]. For the sake of completeness, we present them here and, to this aim, we recall some preliminary notions. Every set of solutions of a system P of inequalities can be considered as a convex set in an Euclidean space of finite dimension; moreover, adopting a more geometrical point of view, it is easy to associate to every convex set X its affine closure $\mathsf{Aff}(X)$, that is the intersection of all the affine subspaces of the Euclidean space containing X. These affine spaces are naturally topological spaces endowed with the topology induced by the Euclidean distance. Let us consider an Euclidean affine space of a given finite dimension.

Lemma 3. *Any convex set has non-empty interior in its affine closure, endowed with the induced topology.*

Proof. Let C be a non-empty convex set. The lemma is straightforward if C consists of a single point. Let us suppose that C has more than one element and, without loss of generality, let us suppose that the origin O of the affine space is in C. Consider the set of vectors p_C defined by $\{P - O \mid P \text{ is a point in } C\}$; let $\{p_1, p_2, \ldots, p_s\}$ be the maximal set of linearly independent vectors in p_C. Then the affine closure of C coincides with the vector space spanned by $p_1, p_2, \ldots, p_s$. All the points of the form $\sum((\frac{1}{(s+1)}) + \varepsilon_i)p_i$, where $|\varepsilon_i| < \frac{1}{s+1}$, are in C and constitute an open neighborhood of the point $\sum(\frac{1}{s+1})p_i$ (which is in C because C is convex) in the affine closure of C. □

Lemma 4. *Let C be a convex set whose affine closure has dimension d and suppose that $Q_1, Q_2, \ldots, Q_n$ are convex sets whose affine closure has dimension $d_i < d$ ($i \in \{1, \ldots, n\}$). Then $C \not\subseteq \bigcup_i Q_i$.*

Proof. Each intersection $X_i = \mathsf{Aff}(Q_i) \cap \mathsf{Aff}(C)$ is an affine subspace of $\mathsf{Aff}(C)$ of dimension strictly lower than d, and thus it is a closed nowhere dense subset of $\mathsf{Aff}(C)$. By Lemma 3, C has non-empty relative interior in $\mathsf{Aff}(C)$; as a consequence of Baire theorem (see, e.g., [75]), C cannot be contained in a finite (even denumerable) union of closed nowhere dense sets. □

Now it is possible to derive the proof of Theorem 1.

Proof of Theorem 1. Let $\mathsf{Aff}(S_P)$ be the affine closure of the set S_P of solutions of the system P. Suppose that, for each j, the system $P \cup \{\overline{e}_j\}$ admits at least a solution; then, for each j, we have $S_P \not\subseteq S_{e_j}$, where S_{e_j} is the set of solutions of the equation e_j. It follows that $\mathsf{Aff}(S_P) \not\subseteq S_{e_j}$. Therefore $S_{e_j} \cap \mathsf{Aff}(S_P)$ is an affine space of dimension strictly lower then $\mathsf{Aff}(S_P)$. So, by Lemma 4, S_P, which is clearly a convex set, is not included in $\bigcup_j S_{e_j}$. □

Theorem 1 allows also to prove Theorem 2.

Proof of Theorem 2. The theorem is proved if we show that $\mathsf{Aff}(S_P) = \bigcap_{k \in I_A} S_{C_k^=}$. Clearly $\mathsf{Aff}(S_P) \subseteq \bigcap_{k \in I_A} S_{C_k^=}$. For the converse we show first that the system of inequalities and equations $\{C_k^<\}_{k \notin I_A} \cup \{C_k^=\}_{k \in I_A}$ admits at least a solution. This system is equivalent to the system $P \cup \{\overline{C_k^=}\}_{k \notin I_A}$. By Theorem 1, this latter system admits at least a solution if and only if, for each $k \notin I_A$, the system $P \cup \{\overline{C_k^=}\}$ admits at least a solution, but this follows from the definition of I_A. Hence, the set X of solution of the system $\{C_k^<\}_{k \notin I_A} \cup \{C_k^=\}_{k \in I_A}$ is not empty. Moreover, X is an open subset of the affine space $\bigcap_{k \in I_A} S_{C_k^=}$, since the affine space inherits the induced topology from the Euclidean one. Therefore $\mathsf{Aff}(X)$ and $\bigcap_{k \in I_A} S_{C_k^=}$ have the same dimension, so $\bigcap_{k \in I_A} S_{C_k^=} = \mathsf{Aff}(X) \subseteq \mathsf{Aff}(S_P)$. □

If we consider a system P of inequalities with real coefficients, the Fourier-Motzkin algorithm decides if P admits at least a solution. The algorithm works as follows. Let P be a set of inequalities and let x be a variable. By obvious algebraic manipulations it is always possible to arrange the inequalities as follows:

$$\begin{array}{lll} N_i & l_i \leq x & i \in \{1, \ldots, p\} \\ P_j & x \leq r_j & j \in \{1, \ldots, q\} \\ V_l & d_l \leq 0 & l \in \{1, \ldots, s\} \end{array}$$

where $p, q, s \geq 0$, each polynomial l_i, r_j and d_l do not contain the variable x and N_i, P_j and V_l are the sets of original inequalities from which the corresponding equality was derived. The first p inequalities correspond to the ones in which x has a negative coefficient, the next q inequalities correspond to the ones in which x has a positive coefficient, and the remaining s inequalities correspond to the ones in which x does not occur.

A Fourier step eliminating x is the transformation which produces the inequalities

$$\begin{aligned} l_i &\leq r_j \qquad i \in \{1, \ldots, p\} \text{ and } j \in \{1, \ldots, q\} \\ d_l &\leq 0 \qquad l \in \{1, \ldots, s\}, \end{aligned}$$

which associates $N_i \cup P_j$ with each inequality $l_i \leq r_j$, and which associates V_l with each $d_l \leq 0$. A Fourier step is trivial if $p = 0$ or $q = 0$; such a step simply deletes all the inequalities containing x in case x does not have a positive (respectively negative) coefficient in any inequality of P. Fourier's algorithm repeats the above procedure until all variables are eliminated or a contradictory inequality is found. The labels associated to every inequality have the following meaning: the set S_k associated with each original inequality C_k is initialized to $\{C_k\}$, and then Fourier's algorithm is applied as described above. If C' is contained in a set S associated with an inequality C then we say that C *was produced using* C'.

Let us consider the signature $\Sigma = \{0, 1, +, -, \{f_r\}_{r \in \mathbb{R}}, <\}$ where $0, 1$ are constants, $-$ and f_r are unary function symbols whereas $+$ is binary, and $<$ is a binary predicate symbol. Let $T_{\mathbb{R}} = Th_{\Sigma}(\mathbb{R})$, that is the set of all Σ-sentences true in $\mathbb{R}$ considered as an $\mathbb{R}$-vector space (thus f_r's represent the external product). In this perspective, given a system of inequalities $P = \{C_1, \ldots, C_m\}$, we will call *implicit equality* any $C_k^=$ such that $T_{\mathbb{R}} \models P \rightarrow C_k^=$. Thus, since $T_{\mathbb{R}}$ is a complete theory, Theorem 1 establishes the Σ_0-convexity of $T_{\mathbb{R}}$ ($\Sigma_0 = \Sigma \setminus \{<\}$). On the other hand, Theorem 2 establishes that, given a system of inequalities P, if B is the collection of all the implicit equalities of P and e is an equality such that $T_{\mathbb{R}} \models P \rightarrow e$, then $T_{\mathbb{R}} \models B \rightarrow e$. So, in order to compute a basis for all the equalities implied by a system P, it is sufficient to recognize the implicit equalities in P.

The Fourier-Motzkin algorithm is actually an algorithm for the quantifier elimination for $T_{\mathbb{R}}$, and it solves the constraint satisfiability problem for $T_{\mathbb{R}}$. Moreover, in [59] it is shown how the Fourier-Motzkin algorithm can be used in order to establish which of the original inequalities are actually implicit equalities. Indeed, the following theorem holds.

Theorem 5. *An inequality C_k in a set of inequalities P is an implicit equality iff application of the Fourier algorithm will produce an inequality $0 \leq 0$ using C_k.*

To prove Theorem 5 we need to introduce some preliminaries lemmas. In the following,

if α is a real and C is the inequality $\sum_{j=1}^{n} a_j x_j \leq b$, αC is the inequality $\sum_{j=1}^{n} \alpha a_j x_j \leq \alpha b$, whereas if C' is the inequality $\sum_{j=1}^{n} a'_j x_j \leq b'$, $C + C'$ means the inequality $\sum_{j=1}^{n} (a_j + a'_j) x_j \leq b + b'$.

Lemma 6. *If P is a finite set of inequalities $\{C_1, C_2, \ldots C_m\}$ and there exists a set of positive reals $\{\alpha_1, \ldots, \alpha_m\}$ such that $\sum_{k=1}^{m} \alpha_k C_k$ is an inequality which can be reduced to the form $0 \leq 0$, then C_j is an implicit equality for every $j \in \{1, \ldots, m\}$.*

Proof. Suppose that there exists a set of positive reals $\{\alpha_1, \ldots, \alpha_m\}$ such that $\sum_{k=1}^{m} \alpha_k C_k$ has the form $0 \leq 0$. Hence $(-1)C_j = \sum_{k=1, k \neq j}^{m} \frac{\alpha_k}{\alpha_j} C_k$ for $j \in \{1, \ldots, m\}$. Since the set of consequences of P is closed under non negative linear combinations, $T_{\mathbb{R}} \models P \to (-1)C_j$. Clearly $T_{\mathbb{R}} \models P \to C_j$, and so $T_{\mathbb{R}} \models P \to C_j^{=}$ for $j \in \{1, \ldots, m\}$. □

Here and in the following we shall adopt the following notation: if P and P' are systems of inequalities and P' is the result of an application of a single step of the Fourier's algorithm, then we write $P \to_F P'$. The following lemmas will be necessary to show the completeness of Fourier's algorithm whenever is used to extract all the implicit equalities.

Lemma 7. *Let x be a variable and let P be a system of inequalities*

$$\begin{aligned} l_i &\leq x \qquad i \in \{1, \ldots, p\} \\ x &\leq r_j \qquad j \in \{1, \ldots, q\} \\ d_l &\leq 0 \qquad l \in \{1, \ldots, s\} \end{aligned}$$

If $P \to_F P'$ by the elimination of the variable x, then $T_{\mathbb{R}} \models \exists x P \leftrightarrow P'$.

Hence, if $P \to_F P'$ by the elimination of the variable x and $T_{\mathbb{R}} \models P \to e$, where e is an equation in which x does not occur, then $T_{\mathbb{R}} \models P' \to e$.

Proof. $T_{\mathbb{R}} \models P \to P'$, thus $T_{\mathbb{R}} \models \forall x (P \to P')$. Since x does not occur in P', $T_{\mathbb{R}} \models \exists x P \to P'$. Suppose s is an assignment of variables to real numbers such that $\mathbb{R} \models_s P'$. Then $s(l_i) \leq s(r_j)$ for each i and j and so $\max_i s(l_i) \leq \min_j s(r_j)$. Thus s' defined by $s'(z) = s(z)$ if $z \not\equiv x$ and $s'(x) = \max_i s(l_i)$ is an assignment such that $\mathbb{R} \models_{s'} \exists x P$. Hence $T_{\mathbb{R}} \models P' \to \exists x P$. □

Lemma 8. *Let P be the set of inequalities*

$$\begin{aligned} l_i &\leq x \qquad i \in \{1, \ldots, p\} \\ x &\leq r_j \qquad j \in \{1, \ldots, q\} \\ d_l &\leq 0 \qquad l \in \{1, \ldots, s\} \end{aligned}$$

over the variable x and other variables $\tilde{y}$, and suppose $T_{\mathbb{R}} \models P \rightarrow x = L(\tilde{y})$ where $L(\tilde{y})$ is a linear expression. If $P \rightarrow_F P'$ by the elimination of the variable x then, for some i and j, $T_{\mathbb{R}} \models P' \rightarrow l_i = r_j$.

Proof. Let S denote the set of solutions of P. For every $s \in S$ there is an i such that $s(l_i) = s(x)$. If not, then $\max_i s(l_i) < s(x)$ and so s' defined by $s'(z) = s(z)$ if $z \not\equiv x$ and $s'(x) = \max_i s(l_i)$ would be a solution of P. But $T_{\mathbb{R}} \models P \rightarrow x = L(\tilde{y})$ and so $s'(x) = s'(L(\tilde{y})) = s(L(\tilde{y})) = s(x)$ which is a contradiction.

Thus we have just shown that $T_{\mathbb{R}} \models P \rightarrow \bigvee_{i=1}^{p} l_i = x$. By Theorem 1, $T_{\mathbb{R}} \models P \rightarrow l_i = x$ for some i. Similarly $T_{\mathbb{R}} \models P \rightarrow x = r_j$ for some j. Hence $T_{\mathbb{R}} \models P \rightarrow l_i = r_j$. Lemma 7 implies that $T_{\mathbb{R}} \models P' \rightarrow l_i = r_j$ for some i and j. □

As a consequence of the proof of the previous lemma we have that the existence of implicit equalities is preserved under Fourier's algorithm steps.

Lemma 9. *If P contains an implicit equality $C_k^=$ produced using an original inequality C and $P \rightarrow_F P'$, then P' contains an implicit equality produced using C.*

Proof. Let x be the eliminated variable. There are three cases. If C_k has the form $l_i \leq x$ then, from the proof of Lemma 8, P has an implicit equality $x \leq r_j$ and P' contains $l_i \leq r_j$ produced using C. Similarly if C_k has the form $x \leq r_j$. If x does not occur in $C_k^=$, then P' contains C_k (produced using C). In each case we can apply Lemma 7 to show that the appropriate inequality is an implicit equality of P'. □

We can now prove Theorem 5 which establishes the correctness of Fourier's algorithm for computing implicit equalities.

Proof of Theorem 5. (Soundness) We can observe that every inequality C obtained during an execution of Fourier's algorithm is the sum of positive multiples of the inequalities in the set S associated with C. By Lemma 6, C_k is an implicit equality.

(Completeness) Suppose $T_{\mathbb{R}} \models P \rightarrow C_k^=$. Let $P \rightarrow_F P'$. By Lemma 9, there is an inequality C in P' which is an implicit equality, and which is produced using C_k. If $0 \leq 0$ is such an equality then we are done. Otherwise we repeat the argument for P' and C.

This process must terminate since eventually all variables in P will have been eliminated. At that stage, the set of inequalities must contain an implicit equality produced using C_k. Since all variables have been eliminated, the inequalities must have the form $0 \leq c$ for some constant c. The only possible implicit equality must come from $0 \leq 0$. □

As already observed, the constraint satisfiability problem for $T_{\mathbb{R}}$ is decidable: apart from the Fourier-Motzkin algorithm for the elimination of quantifiers, possible decision procedures can be built up relying on the Simplex algorithm or also on the SUP-INF

method. As a subtheory of $T_{\mathbb{R}}$ let us consider $\mathbb{R}$-*vect*, i.e. the theory of the vector spaces over $\mathbb{R}$. $\mathbb{R}$-*vect* admits as model completion the theory $\mathbb{R}$-*vect*$^\star = \mathbb{R}$-*vect* $\cup \{\exists x\,(x \neq 0)\}$, whose models are just infinite $\mathbb{R}$-vector spaces. Since for each n the "at least" constraint $\exists x_1, x_2, \ldots, x_n \bigwedge_{i \neq j} x_i \neq x_j$ is in $T_{\mathbb{R}}$, the models of $T_{\mathbb{R}}$ are necessarily infinite; this implies that every model of $T_{\mathbb{R}}$ is a model also of $\mathbb{R}$-*vect**, henceforth the requirements about the $\mathbb{R}$-*vect*-compatibility of $T_{\mathbb{R}}$ are completely fulfilled.

Two observations are in order. First, Theorem 1 shows that $T_{\mathbb{R}}$ is $\Sigma_{\mathbb{R}\text{-}vect}$-convex. Second, Theorem 5 shows that, given a set P of inequalities, it is always possible to extract all the entailed equalities. This is sufficient to state that there exists a $\mathbb{R}$-*vect*-p.b.e. for $T_{\mathbb{R}}$.

Example 3.6.7 (The theory of equality)**.** Let us consider the theory Eq of equality. Σ_{Eq} consists only of the binary predicate symbol $=$. Let $\mathcal{P} := \{P_1, P_2, \ldots, P_n, \ldots\}$ be a countable set of propositional letters, and consider the theory Eq^*, over the signature $\Sigma_{Eq^*} = \Sigma_{Eq} \cup \mathcal{P}$, which extends the theory of equality Eq by adding the following (countably many) axioms:

$$P_n \leftrightarrow \forall x_1 \ldots x_{n+1} \bigvee_{1 \leq i < j \leq n+1} x_i = x_j \qquad (n \in \mathbb{N})$$

We notice that, if P_n is true in some models $\mathcal{A}$ of Eq, then the domain of $\mathcal{A}$ has cardinality at most n; on the other hand, $\neg P_n$ is true in a structure $\mathcal{A}$ if and only if the domain of $\mathcal{A}$ has cardinality at least $n+1$. In other words, the role played by the propositional letter P_n is the same of an "at least" constraint, whereas the role played by $\neg P_n$ is the same of an "at most" constraint. Thus, every infinite structure $\mathcal{A}$ will satisfy $\neg P_n$ for each n.

Eq^* has essentially the same models of Eq, because the interpretation of the propositional letters in $\mathcal{P}$ is uniquely determined; moreover it is sub-model complete. If fact, given any substructure $\mathcal{A}$ of a model $\mathcal{N}$ of Eq^*, if n is the cardinality of the domain of $\mathcal{N}$, $\Delta(\mathcal{A})$ contains all the atoms of the kind P_m for each $m \geq n$, and literals of the kind $\neg P_l$ for each $l < n$; in case $\mathcal{N}$ is an infinite structure, $\Delta(\mathcal{A})$ contains $\neg P_n$ for each n. This is sufficient in order to uniquely determine the cardinality of any model of $Eq^* \cup \Delta(\mathcal{A})$, as the assignment over the propositional letters in $\mathcal{P}$ is an equivalent way of giving an axiomatization of the theory Eq^n of set with a fixed cardinality n, varying n in $\{1, 2, \ldots, \infty\}$. Now it is easy to see that $Eq^* \cup \Delta(\mathcal{A})$ is a complete $\Sigma^{\mathcal{A}}$-theory. Let us consider two models $\mathcal{B}_1$, $\mathcal{B}_2$ of $Eq^* \cup \Delta(\mathcal{A})$: if they have the same finite cardinality, they are isomorphic; if they are structures with infinite domain, then, by the upward Löwenheim-Skolem theorem, there exist two structures $\mathcal{B}'_1$ and $\mathcal{B}'_2$ with the same cardinality such that $\mathcal{B}_1$ embeds (elementarily) into $\mathcal{B}'_1$, and such that $\mathcal{B}_2$ embeds (elementarily) into $\mathcal{B}'_2$. Since $\mathcal{B}'_1$ and $\mathcal{B}'_2$

have the same cardinality, they are isomorphic; since $\mathcal{B}_1$ and $\mathcal{B}_2$ embed elementarily into structures that are isomorphic, they are elementary equivalent.

We have shown that Eq^* is sub-model complete; by Lemma 3.1.4, Eq^* admits elimination of quantifiers, so it can be considered the model-completion of itself. Moreover, as the models of Eq are the same of Eq^*, every theory T extending Eq can be automatically considered a theory extending Eq^*; in this way T always fulfills the condition of Eq^*-compatibility. Finally, Eq^* is a noetherian theory. Given a finite set of constants $\underline{a}$, there exists only a finite number of atoms over $\Sigma^{\underline{a}}_{Eq^*}$; moreover, as $P_n \models P_m$ for each $m \geq n$, every ascending chains of $\Sigma^{\underline{a}}_{Eq^*}$-atoms becomes eventually constant for logical consequences.

Thus, if we want to combine two theories T_1 and T_2 whose constraint satisfiability problem is decidable and which are over disjoint signatures, we only need to check if there exist a T_1-p.b.e. and a T_2-p.b.e. for Eq^*.

Example 3.6.8 (Strongly $\exists_\infty$-decidable theories)**.** Let us consider a theory T which is strongly-$\exists_\infty$-decidable (see Section 2.3). We recall that, given a constraint Γ, if T is a strongly-$\exists_\infty$-decidable theory then (i) it is decidable if $T \cup \Gamma$ is satisfiable, (ii) it is decidable if $T \cup \Gamma$ is satisfiable in an infinite structure, (iii) for every finite structure $\mathcal{A}$, it is decidable if $\mathcal{A}$ is a model for $T \cup \Gamma$; moreover Lemma 2.3.2 implies that it is possible to effectively compute the least integer N such that all models of $T \cup \Gamma$ have cardinality at most N.

As already said, the theory T can be considered equivalent to the $(\Sigma \cup \mathcal{P})$-theory $T_{Eq^*} := T \cup Eq^*$, since they have the same models, being the interpretation of the propositional letters P_n fixed; moreover, the following lemma holds.

Lemma 10. *If T is a strongly-$\exists_\infty$-decidable theory extending Eq, then (i) the constraint satisfiability problem for T_{Eq^*} is decidable, and (ii) given a constraint Γ, it is decidable if Γ is satisfiable in an infinite model of T_{Eq^*}.*

Proof. For (i), let us consider a $(\Sigma \cup \mathcal{P})^{\underline{a}}$-constraint Γ; notice that, being Σ and $\mathcal{P}$ disjoint, we can always write Γ as $\overline{\Gamma} \cup C$, where $\overline{\Gamma}$ is a $\Sigma^{\underline{a}}$-constraint not containing propositional letters belonging to $\mathcal{P}$ and C is a constraint containing only propositional letters belonging to $\mathcal{P}$. Moreover, we can freely suppose that C contains a positive occurrence of the atom P_N whenever the least integer N such that all models of $T \cup \overline{\Gamma}$ has cardinality at most N exists (if not, since T is strongly-$\exists_\infty$-decidable and because of Lemma 2.3.2, we can simply add it to C preserving the equisatisfiability of Γ with respect to T_{Eq^*}). To check the satisfiability of $\overline{\Gamma} \cup C$ with respect to T_{Eq^*}, first of all we test C for consistency (this can be easily done by a straightforward inspection). Suppose that C is consistent. If an atom P_m occurs positively in C, then the satisfiability of $\overline{\Gamma} \cup C$ can be detected by

inspecting all the (finitely many) $\Sigma^{\underline{a}}$-structures whose cardinality is less or equal to m. Otherwise the satisfiability of $\overline{\Gamma} \cup C$ with respect to T_{Eq^*} is reduced to the satisfiability of $\overline{\Gamma}$ with respect to T.

As far as (ii) is concerned, notice that every satisfiable constraint $\overline{\Gamma} \cup C$ admits an infinite model if and only if $\overline{\Gamma}$ is satisfiable in an infinite structure which is a model of T and C does not contain any positive occurrence of atoms belonging to $\mathcal{P}$. Being both requirements effective, it follows that condition (ii) is satisfied. □

As a consequence of the above lemma, for every constraint Γ which is satisfiable only in finite models of T_{Eq^*}, it is always possible to compute the least integer N such that all models of $T_{Eq^*} \cup \Gamma$ have cardinality at most N.

Now, given a set of ground clauses Δ over the simple expansion $(\Sigma \cup \mathcal{P})^{\underline{a}}$ of $\Sigma \cup \mathcal{P}$, we want to obtain a T_{Eq^*}-p.b.e. for Eq^*. We will deal with positive $(\Sigma \cup \mathcal{P})^{\underline{a}}$-clauses of the following kinds:

(1) clauses over the signature $\mathcal{P}$;

(2) clauses over the signature $\Sigma^{\underline{a}}$;

(3) clauses over the mixed signature $(\Sigma \cup \mathcal{P})^{\underline{a}}$.

As far as (1) and (3) are concerned, notice that we can assume that just one atom belonging to $\mathcal{P}$ occurs in them (indeed, $P_i \vee P_j$ is logically equivalent to P_k where $k = \max\{i, j\}$). Let us proceed in the following way:

(i) for clauses of the kind (1), we check if $T_{Eq^*} \cup \Delta$ is satisfiable and, if so, we decide whether $T_{Eq^*} \cup \Delta$ admits an infinite model, and, if not, we compute the least N such that all models of $T_{Eq^*} \cup \Delta$ have cardinality at most N.

This can be effectively done: the most naive method is converting the conjunction of the clauses in Δ into disjunctive normal form $\Gamma_1 \vee \Gamma_2 \vee \cdots \vee \Gamma_n$. By Lemma 10, we can check if $T_{Eq^*} \cup \Gamma_i$ is satisfiable; if so, we can decide if Γ_i is satisfiable in an infinite model of T_{Eq^*} and, if not, we can compute the least N_i such that all models of $T_{Eq^*} \cup \Gamma_i$ have cardinality at most N_i. If at least one N_i has been found and no Γ_i is satisfiable in infinite models, then the clause we are looking for is P_N, where N is the maximum of the N_i's (notice that, if $T_{Eq^*} \cup \Delta \models P_k$, then $Eq^* \cup \{P_N\} \models P_k$).

(ii) for clauses of the kind (2), it is sufficient to check if a positive clause in the form $a_{i_1} = a_{i_2} \vee \cdots \vee a_{i_{n-1}} = a_{i_n}$, where a_{i_j} is a constant among the $\underline{a}$'s, is a logical consequence of $T_{Eq^*} \cup \Delta$ (this can be effectively done by reducing it to a satisfiability problem in the usual way).

(iii) as far as clauses of the kind (3) are concerned, for each clause C of the form $a_{i_1} = a_{i_2} \vee \cdots \vee a_{i_{n-1}} = a_{i_n}$, where a_{i_j} is a constant among the $\underline{a}$'s, we check if $T_{Eq^*} \cup \Delta \cup \{\neg C\}$ is satisfiable in an infinite model, and, if not, we compute the least N such that all models of $T_{Eq^*} \cup \Delta \cup \{\neg C\}$ have cardinality at most N (to this aim, we can proceed as in (i)). In such a case, the required clause is $C \vee P_N$ (indeed, $T_{Eq^*} \cup \Delta \models C \vee P_N$ and, if $T_{Eq^*} \cup \Delta \models C \vee P_k$, then $Eq^* \cup \{C \vee P_N\} \models C \vee P_k$)

$\mathsf{B}_{T_i}(\Delta)$ is equal to $\bot$ if the satisfiability test in step (i) fails. Otherwise, it consists of all the ground $\Sigma^{\underline{a}}_{Eq}$-clauses computed in steps (i), (ii) and (iii).

Example 3.6.9 (Weakly $\exists$-superposition-decidable theories)**.** The previous example shows that, in order to extract a T_{Eq^*}-p.b.e. for Eq^* (where $T_{Eq^*} = T \cup Eq^*$ and T is any theory extending the pure theory of equality) it is sufficient to have that T is a strongly-$\exists_\infty$-decidable. As weakly $\exists$-superposition-decidable theories are strongly-$\exists_\infty$-decidable theories (see Theorem 2.4.13), it is possible to obtain a T_{Eq^*}-p.b.e. for Eq^* whenever T is weakly $\exists$-superposition-decidable.

Chapter 4

A Higher-Order Framework for Combination

In this chapter we try to push the Nelson-Oppen methodology further in order to solve in a uniform way as many problems as possible. We have already seen in the previous chapter that, when joined to model-theoretic results, the Nelson-Oppen schema succeeds in dealing with various classes of combination problems. Now we want to generalize even more this schema: in this perspective, we will plug it into an higher-order framework, adopting type-theoretic signatures in Church's style. The choice of a higher-order framework is justified by the fact that quite often the semantic specification language for decision problem is intrinsically higher-order, even if in practice problems themselves are not really such. For example, in the case of modal logics, decision problems are specified through the so-called standard translations: clearly, the problem of finding a structure satisfying the standard translation of a modal formula is (at least in principle) higher-order because the predicates symbols occurring in the problem are genuine second-order variables.

The interest of this approach relies on the existence of tractable fragments of general type theory whose 'combination' often turns out to be tractable. To develop the plan of plugging Nelson-Oppen procedure into a higher-order context, we will need to introduce a suitable notion of fragments and their combination (Sections 4.2 and 4.3), to adapt the notions of local finiteness and noetherianity to these new definitions (Section 4.2.4), to re-design the combination schema (Section 4.3.2). At this point we will be able to obtain a semidecision procedure for the combined constraint satisfiability; to guarantee the completeness of the procedure, we will need powerful semantically-driven tools (Section 4.4).

4.1 Type-Theoretic Languages

We fix our notation for higher-order syntax; we adopt a type theory in Church's style (see [3, 2, 58] for introductions to the subject).

4.1.1 Signatures

We use letters $S_1, S_2, \dots$ to indicate *sorts* (also called *primitive types*) of a signature. Formally, sorts are a set $\mathcal{S}$ and *types over* $\mathcal{S}$ are built inductively as follows:

- every sort $S \in \mathcal{S}$ is also a type;
- Ω is a type (this is called the *truth-values* type):
- if τ_1, τ_2 are types, so is $(\tau_1 \to \tau_2)$.

As usual external brackets are omitted; moreover, we shorten the expression $\tau_1 \to (\tau_2 \to \dots (\tau_n \to \tau))$ into $\tau_1 \dots \tau_n \to \tau$ (in this way, every type τ has the form $\tau_1 \dots \tau_n \to \tau$, where $n \geq 0$ and τ is a sort or it is Ω). In the following, we use the notation $\mathcal{T}(\mathcal{S})$ or simply $\mathcal{T}$ to indicate a *types set*, i.e. the totality of types that can be built up from the set of sorts $\mathcal{S}$. In this way, $\mathcal{S}$ is sometimes left implicit in the notation, however we always reserve to sorts the letters $S_1, S_2, \dots$ (as opposed to the letters τ, υ, etc. which are used for arbitrary types).

A *signature* (or a *language*) is a triple $\mathcal{L} = \langle \mathcal{T}, \Sigma, a \rangle$, where $\mathcal{T}$ is a types set, Σ is a set of constants and a is an *arity map*, namely a map $a : \Sigma \longrightarrow \mathcal{T}$; we write $f : \tau_1 \dots \tau_n \to \tau$ to express that f is a *constant of type* $\tau_1 \dots \tau_n \to \tau$, i.e. that $a(f) = \tau_1 \dots \tau_n \to \tau$. According to the above observation, we can assume that τ is a sort or that $\tau = \Omega$; in the latter case, we say that f is a *predicate* or a *relational* symbol (predicate symbols are preferably indicated with the letters $P, Q, \dots$).

We require the following *special constants* to be always present in a signature:

- $\top$ and $\bot$ of type Ω;
- $\neg$ of type $\Omega \to \Omega$;
- $\vee$ and $\wedge$ of type $\Omega\,\Omega \to \Omega$;
- $=_\tau$ of type $\tau\,\tau \to \Omega$ for each type $\tau \in \mathcal{T}$ (we usually write it as '=' without specifying the subscript τ).

The *proper symbols* of a signature are its sorts and its non special constants.

A signature is *one-sorted* iff its set of sorts is a singleton. A signature $\mathcal{L}$ is *first-order* if for any proper $f \in \Sigma$, we have that $a(f) = S_1 \dots S_n \to \tau$, where τ is a sort or it is Ω.

A first-order signature is called *relational* iff any proper $f \in \Sigma$ is a relational constant, that is $a(f) = S_1 \dots S_n \to \Omega$. By contrast, a first-order signature is called *functional* iff any proper $f \in \Sigma$ has arity $S_1 \dots S_n \to S$.

Let $\mathcal{L}_1 = \langle \mathcal{T}_1, \Sigma_1, a_1 \rangle$ and $\mathcal{L}_2 = \langle \mathcal{T}_2, \Sigma_2, a_2 \rangle$ be two signatures; we say that $\mathcal{L}_1$ is a subsignature of $\mathcal{L}_2$ (written $\mathcal{L}_1 \subseteq \mathcal{L}_2$) if $\mathcal{T}_1 \subseteq \mathcal{T}_2$, $\Sigma_1 \subseteq \Sigma_2$ and $a_1 \subseteq a_2$. Furthermore, given $\mathcal{L}_1 = \langle \mathcal{T}_1, \Sigma_1, a_1 \rangle$ and $\mathcal{L}_2 = \langle \mathcal{T}_2, \Sigma_2, a_2 \rangle$, in case a_1 and a_2 coincide[1] on $\Sigma_1 \cap \Sigma_2$, we define the *union signature* $\mathcal{L}_1 \cup \mathcal{L}_2$ to be (let $\mathcal{T}_1$ be $\mathcal{T}(\mathcal{S}_1)$ and $\mathcal{T}_2$ be $\mathcal{T}(\mathcal{S}_2)$) $\langle \mathcal{T}(\mathcal{S}_1 \cup \mathcal{S}_2), \Sigma_1 \cup \Sigma_2, a_1 \cup a_2 \rangle$ and the *intersection signature* $\mathcal{L}_1 \cap \mathcal{L}_2$ to be $\langle \mathcal{T}_1 \cap \mathcal{T}_2, \Sigma_1 \cap \Sigma_2, a_1 \cap a_2 \rangle$.

4.1.2 Terms

Given a signature $\mathcal{L} = \langle \mathcal{T}, \Sigma, a \rangle$ and a type $\tau \in \mathcal{T}$, we define the notion of an $\mathcal{L}$-*term* (or just *term*) of type τ, written $t : \tau$, as follows (for the definition we need, for every type $\tau \in \mathcal{T}$, a countable supply V_τ of variables of type τ):

- $x : \tau$ (for $x \in V_\tau$) is an $\mathcal{L}$-term of type τ;
- $c : \tau$ (for $c \in \Sigma$ and $a(c) = \tau$) is an $\mathcal{L}$-term of type τ;
- if $t : \upsilon \to \tau$ and $u : \upsilon$ are $\mathcal{L}$-terms of types $\upsilon \to \tau$ and υ, respectively, then $val_\upsilon(t, u) : \tau$ (also written as $t(u) : \tau$) is an $\mathcal{L}$-term of type τ ;
- if $t : \tau$ is an $\mathcal{L}$-term of type τ and $x \in V_\upsilon$ is a variable of type υ, $\lambda x^\upsilon\, t : \upsilon \to \tau$ is an $\mathcal{L}$-term of type $\upsilon \to \tau$.

In the following, we consider the notation x^τ (c^τ) equivalent to $x : \tau$ ($c : \tau$), where x (c) is a variable (a constant); if it can be deduced from the context, *the specification of the type of a term may be omitted*. Moreover, a term of type τ is also called a τ-*term* and terms of type Ω are also called *formulae*. Given a formula φ, we write $\{x \mid \varphi\}$ for $\lambda x\, \varphi$.

We shorten $val_{\upsilon_n}(\cdots(val_{\upsilon_1}(t, u_1), \cdots), u_n)$ to $t(u_1, \dots, u_n)$ where u_i is a term of type υ_i ($i \in \{1, \dots, n\}$) and t is a term of type $\upsilon_1 \dots \upsilon_n \to \tau$. For each term φ of type Ω, we define the Ω-terms $\forall x^\upsilon\, \varphi$ and $\exists x^\upsilon\, \varphi$ as $\{x^\upsilon \mid \varphi\} = \{x^\upsilon \mid \top\}$ and as $\neg\, \forall x^\upsilon\, \neg\varphi$, respectively (the latter can also be defined differently, in an intuitionistically acceptable way, see [58]). For terms φ_1, φ_2 of type Ω, the terms $\varphi_1 \to \varphi_2$ and $\varphi_1 \leftrightarrow \varphi_2$ of type Ω are classically defined by $\neg\varphi_1 \vee \varphi_2$ and by $(\varphi_1 \to \varphi_2) \wedge (\varphi_2 \to \varphi_1)$, respectively (but notice that $\varphi_1 \leftrightarrow \varphi_2$ can be defined in a semantically equivalent way also as $\varphi_1 = \varphi_2$).

By the above definitions, first-order formulae *can be considered as a subset* of the higher-order formulae defined in this section. More specifically, when we speak of *first-order terms*, we mean variables $x : S$, constants $c : S$ and terms of the kind $f(t_1, \dots, t_n) :$

[1] Modulo renaming some elements of Σ_1, we can assume that this condition is always satisfied, so that union and intersection signatures are always defined.

S, where $t_1, \ldots, t_n$ are (inductively given) first-order terms and $a(f) = S_1 \cdots S_n \to S$. Now *first-order formulae* are obtained from formulae of the kind $\top : \Omega, \bot : \Omega, P(t_1, \ldots, t_n) : \Omega$ (where $t_1, \ldots, t_n$ are first-order terms and $a(P) = S_1 \cdots S_n \to \Omega$) by applying $\exists x^S, \forall x^S, \wedge, \vee, \neg, \to, \leftrightarrow$.

4.1.3 Substitutions and Conversions

An occurrence of a variable x in a term t is *bound* if it appears in a subterm of t of the kind $\lambda x\, u$, otherwise it is said to be *free*. A variable x occurs free in a term t if and only if at least one occurrence of x in t is free; by $fvar(t)$ we mean the set of the variables that occur free in t (whereas $fvar_\tau(t)$ is the set of the variables of type τ that occur free in t). If E is a set of terms, $fvar(E)$ means $\bigcup_{t \in E} fvar(t)$. We often use notations like $\underline{x}, \underline{y}$ to mean tuples of distinct free variables.

A term without free variables is called a *closed* term and a formula without free variables is called a *sentence*. The notation $t[x_1, \ldots, x_n]$ (resp. $E[x_1, \ldots, x_n]$) means that $fvar(t) \subseteq \{x_1, \ldots, x_n\}$ (resp. $fvar(E) \subseteq \{x_1, \ldots, x_n\}$).

Two terms are said to be equivalent modulo *α-conversion* iff they differ only by a bound variables renaming; in the following, we shall identify α-equivalent terms, i.e. we consider terms as representatives of their equivalence class modulo α-conversion.

Let $\mathcal{V}$ be the disjoint union of the sets of variables V_τ ($\tau \in \mathcal{T}$). We define the notion of substitution as usual: a *substitution* is a map $\sigma : \mathcal{V} \to T$ (from the set $\mathcal{V}$ of the variables into the set T of the terms) that respects types (i.e. if $x \in V_\tau$ then $x\sigma$ is a term of type τ) and such that the set $\{x \mid x \not\equiv x\sigma\}$ is finite.[2] The set $dom(\sigma) := \{x \mid \not\equiv x\sigma\}$ is called the *domain* of the substitution σ. A substitution σ will be written as $x_1 \mapsto x\sigma_1, \ldots, x_n \mapsto x_n\sigma$, or equivalently as $x_1\sigma/x_1, \ldots, x_n\sigma/x_n$, where $dom(\sigma) \subseteq \{x_1, \ldots, x_n\}$. A substitution is a *renaming* iff it is a variable permutation.

Substitutions can be extended in the domain from variables to all terms in the usual way; notice however that, when defining inductively the term $t\sigma$, it might happen that α-conversions must be applied before actual replacements, in order to avoid clashes. If $\sigma = \{x_1 \mapsto u_1, \ldots, x_n \mapsto u_n\}$ and $fvar(t) \subseteq \{x_1, \ldots, x_n\}$, the term $t\sigma$ can also be written as $t[u_1, \ldots, u_n]$. Given two substitutions σ_1 and σ_2, the *composite substitution* $\sigma_1\sigma_2$ is the substitution that maps the variable x to $(x\sigma_1)\sigma_2$. The notion of $\beta\eta$-equivalence between terms is introduced through the following previous inductive definition of the relation $\rhd^1_{\beta\eta}$ (we follow [33]):

- (β) $\quad val(\lambda x\, t, u) \rhd^1_{\beta\eta} t\sigma$, where $\sigma : \{x \mapsto u\}$;
- (η) $\quad \lambda x\, val(t, x) \rhd^1_{\beta\eta} t$, if x is not free in t;

[2] Since the equality symbol '=' is present in the object language, we prefer to use '$\equiv$' in the metalanguage for coincidence of syntactic expressions.

– (μ) $val(t,u) \rhd^1_{\beta\eta} val(t,u')$, if $u \rhd^1_{\beta\eta} u'$;

– (ν) $val(t,u) \rhd^1_{\beta\eta} val(t',u)$, if $t \rhd^1_{\beta\eta} t'$;

– (ξ) $\lambda x\, t \rhd^1_{\beta\eta} \lambda x\, t'$, if $t \rhd^1_{\beta\eta} t'$.

The *$\beta\eta$-equivalence relation* $\sim_{\beta\eta}$ is now the reflexive, symmetric and transitive closure of the relation $\rhd^1_{\beta\eta}$. By definition, $\sim_{\beta\eta}$ is an equivalence relation compatible with the term constructors. It is known that the $\beta\eta$-reduction relation $\rhd_{\beta\eta}$ obtained from the transitive closure of $\rhd^1_{\beta\eta}$, gives a rewrite system that is strongly normalizable and confluent (see [16, 47]), i.e. each term has *a unique $\beta\eta$-normal form* modulo α-conversion. Sometimes, however, it is preferable to use the so-called *long-$\beta\eta$-normal form* of a term $t : \tau$ (instead of the $\beta\eta$-normal form of t). This is defined as follows: suppose $\tau = \tau_1 \cdots \tau_n \to \upsilon$, consider the $\beta\eta$-normal form

$$\lambda x_1 \cdots \lambda x_m\; y(u_1, \ldots, u_p)$$

of t (here $m \leq n$ and y is a variable or a constant) and then take

$$\lambda x_1 \cdots \lambda x_m \lambda x_{m+1} \cdots \lambda x_n\; y(u'_1, \ldots, u'_p, x'_{m+1}, \ldots, x'_n)$$

to be the long-$\beta\eta$-normal forms of t (where $u'_1, \ldots, u'_p, x'_{m+1}, \ldots, x'_n$ are the long-$\beta\eta$-normal forms of $u_1, \ldots, u_p, x_{m+1}, \ldots, x_n$, respectively). Thus, for instance, the long $\beta\eta$-normal form of a predicate constant $P : \tau \to \Omega$ is $\{x \mid P(x)\}$.

4.1.4 Models

In order to introduce our computational problems, we need to recall the notion of an interpretation for a type-theoretic language. Formulae of higher-order type theory which are valid in ordinary set-theoretic models do not form an axiomatizable class, as it is well-known from classical limitative results. Hence, in order to re-gain axiomatizability, one has to use Henkin models (see [3]) or to take interpretations into elementary toposes (see [58]). However, we shall confine ourselves to standard set-theoretic models, because we are not interested in the whole type theoretic language (nor in any calculus for it). On the other hand, the generalization to more powerful semantics of the definitions given in this subsections is well-known and can be found, for instance, in the aforementioned textbooks.

If we are given a map that assigns to every sort $S \in \mathcal{S}$ a set $[\![S]\!]$, we can inductively extend it to all types over $\mathcal{S}$, by taking $[\![\tau \to \upsilon]\!]$ to be the set of functions from $[\![\tau]\!]$ to $[\![\upsilon]\!]$. Given a language $\mathcal{L} = \langle \mathcal{T}, \Sigma, a \rangle$, a *$\mathcal{L}$-structure* (or just a *structure*) $\mathcal{A}$ is a pair $\langle [\![-]\!]_{\mathcal{A}}, \mathcal{I}_{\mathcal{A}} \rangle$, where:

(i) $[\![-]\!]_{\mathcal{A}}$ is a function assigning to a sort $S \in \mathcal{T}$, a set $[\![S]\!]_{\mathcal{A}}$;

(ii) $\mathcal{I}_{\mathcal{A}}$ is a function assigning to a constant $c \in \Sigma$ of type τ, an element $\mathcal{I}_{\mathcal{A}}(c^{\tau}) \in [\![\tau]\!]_{\mathcal{A}}$ (here $[\![-]\!]_{\mathcal{A}}$ has been extended from sorts to types as explained above).

In every structure $\mathcal{A}$, we finally require also that $[\![\Omega]\!]_{\mathcal{A}} = \{0, 1\}$, that $\mathcal{I}_{\mathcal{A}}(\bot) = 0$, that $\mathcal{I}_{\mathcal{A}}(\top) = 1$, that $\mathcal{I}_{\mathcal{A}}(=_{\tau})$ is the characteristic function of the identity relation on $[\![\tau]\!]_{\mathcal{A}}$, and that $\mathcal{I}_{\mathcal{A}}(\neg)$, $\mathcal{I}_{\mathcal{A}}(\vee)$, $\mathcal{I}_{\mathcal{A}}(\wedge)$ are the usual truth tables functions (notice that, in these and similar passages, we implicitly use the isomorphisms $(X^Y)^Z \simeq X^{Y \times Z}$ in order to treat in the natural way curryfied binary function symbols).

We do not exclude, in principle, that in a structure $\mathcal{A}$ we can have $[\![\tau]\!]_{\mathcal{A}} = \emptyset$ for some type τ (in fact, the use of *finitary* assignments below is compatible with empty domains);[3] however, when dealing with one-sorted signatures $\mathcal{L}$, we shall implicitly assume, for simplicity, that in $\mathcal{L}$-structures *the unique sort is always interpreted into a non-empty domain.*

Given a $\mathcal{L}$-structure $\mathcal{A} = \langle [\![-]\!]_{\mathcal{A}}, \mathcal{I}_{\mathcal{A}} \rangle$, let $\mathcal{L}^{\mathcal{A}}$ be the language enriched by a constant $\bar{a}$ of type τ for every $a \in [\![\tau]\!]_{\mathcal{A}}$; $\mathcal{A}$ can be canonically considered as a $\mathcal{L}^{\mathcal{A}}$-structure once $\mathcal{I}_{\mathcal{A}}$ is extended to the new constants by stipulating that $\mathcal{I}_{\mathcal{A}}(\bar{a}) := a$. By induction, it is now possible to extend $\mathcal{I}_{\mathcal{A}}$ to all closed $\mathcal{L}^{\mathcal{A}}$-terms t as follows:

– $\mathcal{I}_{\mathcal{A}}(val(t, u)) = \mathcal{I}_{\mathcal{A}}(t)(\mathcal{I}_{\mathcal{A}}(u))$ (this is set-theoretic functional application);

– $\mathcal{I}_{\mathcal{A}}(\lambda x^{\tau}\, t)$ is the function that maps each element $a \in [\![\tau]\!]_{\mathcal{A}}$ into $\mathcal{I}_{\mathcal{A}}(t[\bar{a}/x])$.

From now on, we shall not distinguish for simplicity between a and its name $\bar{a}$.

A $\mathcal{L}^{\mathcal{A}}$-sentence φ is *true* in $\mathcal{A}$ (in symbols $\mathcal{A} \models \varphi$) iff $\mathcal{I}_{\mathcal{A}}(\varphi) = 1$. Notice that, according to the above definition of universal quantification, we have that $\mathcal{A} \models \forall x^{\tau}\, \varphi$ if and only if for each $a \in [\![\tau]\!]_{\mathcal{A}}$, we have $\mathcal{I}_{\mathcal{A}}(\varphi[a/x]) = 1$.

To introduce the notion of satisfiability we use finite assignments. Let $\mathcal{A} = \langle [\![-]\!]_{\mathcal{A}}, \mathcal{I}_{\mathcal{A}} \rangle$ be a $\mathcal{L}$-structure and let $\underline{x}$ be a finite set of variables; an *$\underline{x}$-assignment* (or simply an assignment if $\underline{x}$ is clear from the context) α is a map associating with every variable $x^{\tau} \in \underline{x}$ an element $\alpha(x) \in [\![\tau]\!]_{\mathcal{A}}$. An $\mathcal{L}$-formula φ is *satisfied* in $\mathcal{A}$ under the $\underline{x}$-assignment α (where $\underline{x} \supseteq fvar(\varphi)$) iff $\mathcal{I}^{\alpha}_{\mathcal{A}}(\varphi) = 1$, where $\mathcal{I}^{\alpha}_{\mathcal{A}}(\varphi)$ is the $\mathcal{L}^{\mathcal{A}}$-sentence obtained by replacing in φ the variables $x \in \underline{x}$ by (the names of) $\alpha(x)$. We usually write $\mathcal{A} \models_{\alpha} \varphi$ for $\mathcal{I}^{\alpha}_{\mathcal{A}}(\varphi) = 1$.

A formula is *satisfiable* iff it is satisfied under some assignment and a set of formulae Θ (containing altogether only finitely many variables) is satisfiable iff for some assignment α we have that $\mathcal{A} \models_{\alpha} \varphi$ holds for each $\varphi \in \Theta$ (of course, for this to make sense, α must be

[3]Usual (total) assignments are inadequate if one wants tautological sentences to be satisfiable in a structure in which some sort is interpreted into the empty set. Finite assignments eliminate this inconvenient, however empty domains cause further problems on the syntactic side, if one wants to formulate suitable calculi. These questions do not concern the present work, however we recall that there is a simple well-known solution to them, namely the explicit indication of the variables involved in a proof (see [58]).

an $\underline{x}$-assignment for some $\underline{x} \supseteq fvar(\Theta)$ - and one can even assume $\underline{x} = fvar(\Theta)$ without loss of generality).

For signature inclusions $\mathcal{L}_0 \subseteq \mathcal{L}$, there is an obvious *taking reduct* operation mapping an $\mathcal{L}$-structure $\mathcal{A}$ to an $\mathcal{L}_0$-structure $\mathcal{A}_{|\mathcal{L}_0}$; we can similarly take the $\mathcal{L}_0$-reduct of an assignment, by ignoring the values assigned to variables whose types are not in $\mathcal{L}_0$ (we leave the reader to define these notions properly).

Two $\mathcal{L}$-structures $\mathcal{A}_1 = \langle [\![-]\!]_{\mathcal{A}_1}, \mathcal{I}_{\mathcal{A}_1} \rangle$ and $\mathcal{A}_2 = \langle [\![-]\!]_{\mathcal{A}_2}, \mathcal{I}_{\mathcal{A}_2} \rangle$ are said to be *isomorphic* iff there are bijections $\iota_\tau : [\![\tau]\!]_{\mathcal{A}_1} \longrightarrow [\![\tau]\!]_{\mathcal{A}_2}$ (varying $\tau \in \mathcal{T}$) such that $\iota_\tau(\mathcal{I}_{\mathcal{A}_1}(c)) = \mathcal{I}_{\mathcal{A}_2}(c)$ holds for all $c : \tau \in \Sigma$ and such that $\iota_{\tau \to \upsilon}(h) = \iota_\upsilon \circ h \circ \iota_\tau^{-1}$ holds for all $h \in [\![\tau \to \upsilon]\!]_{\mathcal{A}_1}$.[4] Isomorphic structures are in fact indistinguishable (in particular, the same sentences are true in them).

4.2 Fragments

General type theory is very hard to attack from a computational point of view, this is why we are basically interested only in more tractable fragments and in combinations of them. Fragments are defined as follows:

Definition 4.2.1. A *fragment* is a pair $\langle \mathcal{L}, T \rangle$ where $\mathcal{L} = \langle \mathcal{T}, \Sigma, a \rangle$ is a signature and T is a recursive set of $\mathcal{L}$-terms.

4.2.1 Algebraic Fragments

We want to use fragments as ingredients of larger and larger combined fragments: a crucial notion in this sense is that of an algebraic fragment.

Definition 4.2.2. A fragment $\langle \mathcal{L}, T \rangle$ is said to be an *algebraic fragment* iff T satisfies the following conditions:

(i) T *is closed under composition*, i.e. if $u[x_1, \ldots, x_n] \in T$, then $u\,\sigma \in T$, where $\sigma : \{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ is a substitution such that $t_i \in T$ for all $i = 1, \ldots, n$;

(ii) T *contains domain variables*, i.e. if τ is a type such that some variable of type τ occurs free in a term $t \in T$, then every variable of type τ belongs to T;

(iii) T *contains codomain variables*, i.e. if $t : \tau$ belongs to T, then every variable of type τ belongs to T.

Observe that from the above definition it follows that T is closed under renamings, i.e. that if $t \in T$ and σ is a renaming, then $t\sigma \in T$. The role of Definition 4.2.2(i) is that of

[4] Thus, once again, to give an isomorphism it is sufficient to specify the bijections ι_S for all sorts S.

making fragment combinations non trivial, whereas the other conditions of Definition 4.2.2 will be needed in order to apply preprocessing purification steps to combined constraints.

Quite often, one is interested in interpreting the terms of a fragment not in the class of all possible structures for the language of the fragment, but only in some selected ones (e.g. when checking satisfiability of some temporal formulae, one might be interested only in checking satisfiability in particular flows of time, those which are for instance discrete or continuous). This is the reason for 'interpreting' fragments:

Definition 4.2.3. An *interpreted algebraic fragment* (to be shortened as i.a.f.) is a triple $\Phi{=}\langle \mathcal{L}, T, \mathcal{S}\rangle$, where $\langle \mathcal{L}, T\rangle$ is an algebraic fragment and $\mathcal{S}$ is a class of $\mathcal{L}$-structures closed under isomorphisms.

The set of terms T in an i.a.f. $\Phi{=}\langle \mathcal{L}, T, \mathcal{S}\rangle$ is called the set of Φ-*terms* and the set of types τ such that $t : \tau$ is a Φ-term for some t is called the set of Φ-*types*. A Φ-*variable* is a variable x^τ such that τ is a Φ-type (or equivalently, a variable which is a Φ-term). It is also useful to *identify a (non-interpreted) algebraic fragment* $\langle \mathcal{L}, T\rangle$ with the interpreted algebraic fragment $\Phi{=}\langle \mathcal{L}, T, \mathcal{S}\rangle$, where $\mathcal{S}$ is taken to be the class of all $\mathcal{L}$-structures.

Definition 4.2.4. Given an i.a.f. fragment Φ, a Φ-*atom* is an equation $t_1 = t_2$ between Φ-terms t_1, t_2 of the same type; a Φ-*literal* is a Φ-atom or a negation of a Φ-atom, a Φ-*constraint* is a finite conjunction of Φ-literals, a Φ-*clause* is a finite disjunction of Φ-literals. Infinite sets of Φ-literals (representing an infinite conjunction) are called *generalized* Φ-*constraints* (provided they contain altogether only finitely many free variables).

Some Conventions. Without loss of generality, we may assume that $\top$ *is a* Φ-*atom* in every i.a.f. Φ (in fact, to be of any interest, a fragment should at least contain one term t and we can let $\top$ to be $t = t$). As a consequence, $\bot$ will always be a Φ-literal; by convention, however, we shall *include* $\bot$ *among* Φ-*atoms* (hence a Φ-atom is either an equation among Φ-terms - $\top$ included - or it is $\bot$). Since we have $\bot$ as an atom, there is no need to consider the empty clause as a clause, so clauses will be disjunctions of *at least one* literal. The reader should keep in mind these slightly non standard conventions for the rest of the chapter.

A Φ-clause is said *positive* if only Φ-atoms occur in. A Φ-atom $t_1 = t_2$ is closed if and only if t_i is closed ($i \in \{1, 2\}$); the definition of closed Φ-literals, -constraints and -clauses is analogous. For a finite set $\underline{x}$ of variables and an i.a.f. Φ, a $\Phi(\underline{x})$-atom (-term, -literal, -clause, -constraint) is a Φ-atom (-term, -literal, -clause, -constraint) A such that $fvar(A) \subseteq \underline{x}$.

In this chapter we mainly deal with the *constraint satisfiability problem* for an interpreted algebraic fragment $\Phi{=}\langle \mathcal{L}, T, \mathcal{S}\rangle$: this is *the problem of deciding whether a* Φ-*constraint is satisfiable in some structure* $\mathcal{A} \in \mathcal{S}$. On the other hand, the *word problem*

for Φ is the problem of deciding if the universal closure of a given Φ-atom is true in every structure $\mathcal{A} \in \mathcal{S}$.

The literature on fragments and on decision procedures for fragments is extremely large (in a sense, one may argue that mathematical logic itself consists of studying the various fragments and their syntactic and semantic properties). Notice however that our definition of a fragment refers to an embedding into a higher-order typed language: the consequence of this approach is that a given well-known fragment (in the naive sense) can formally be turned into a fragment in our sense *in many ways* and the differences among such ways are crucial when concretely applying the definition of a combined fragment to be given in Section 4.3.

The reason is the following: although we have not yet given the relevant definition, the reader may imagine that combining algebraic fragments means, roughly speaking, taking the smallest algebraic fragment containing some given ones. Now, when *defining* the set of Φ-terms of a fragment, it does not actually matter whether certain symbols are treated as free constants or as free variables: since every variable is existentially quantified in the definition of a satisfiable constraint, then one may indifferently use free constants or variables in Φ-terms. However, constants are not good to be used as placeholders when defining the composition (=substitution) of terms, so *the (ab)use of free constants reduces the expressive richness of the combined fragments that can be build over the given one.*

Another opportunity in defining the set of Φ-terms of a fragment, is that of taking a final λ-abstraction in order to get rid of free variables.[5] Clearly the choice of closing by λ-abstraction the Φ-terms of a fragment changes the nature of the satisfiability of the resulting constraints (e.g. it makes the difference between local and global satisfiability, in the case of the standard translation of modal propositional formulae). However such a choice has another relevant and more hidden effect: having taken λ-abstraction, we produced higher-order terms which are now ready to be *substituted for* higher-order variables when taking combined fragments.

Sometimes the above options cannot be used together: for instance, the set of prenex first-order formulae having a certain given prefix shape are *not* an algebraic fragment if the predicate letters in them are treated as second-order variables and if first-order variables are λ-abstracted (closure under substitutions of Φ-terms for variables fails).

The moral of this discussion is that our framework is quite general and flexible, but just for this reason, *it needs to be handled with some care.* In next subsection we shall give examples of algebraic fragments (the reader now knows why we will apparently make 'many different copies' of seemingly the same fragment).

[5] For Φ-terms $\varphi[\underline{x}]$ of type Ω, this usually has the effect of taking universal closure in Φ-atoms: the term $\{\underline{x} \mid \top\}$ is usually in Φ, so that taking the Φ-atom $\{\underline{x} \mid \top\} = \{\underline{x} \mid \varphi\}$ amounts to consider the universal closure of φ.

We would like to draw the reader's attention to the fact that in Definition 4.2.2, when formulating the closure under composition requirement for the set of the terms T of an algebraic fragment, we asked that if $t[x_1, \ldots, x_n] \in T$ and $u_1, \ldots, u_n \in T$, then *precisely* the term $t[u_1/x_1, \ldots, u_n/x_n]$ belongs to T (and not just some other term which is $\beta\eta$-equivalent to it, like for instance its $\beta\eta$-normal form). The reason for this strict requirement is that we want a term belonging to a combined i.a.f. to be effectively decomposable into some iterated composition of pure terms (see Subsection 4.3.1). With the present version of Definition 4.2.2, there is an evident algorithm for computing such a decomposition. Of course, we did not eliminate the $\beta\eta$-conversion problem in this way, but we simply left to the user of our combination procedure the responsibility of effectively certifying that the terms forming the constraints he is interested to decide for satisfiability really belong (maybe up to $\beta\eta$-equivalence) to the combined fragment to which he is going to apply the procedure. For instance, before claiming that our procedure decides relativized satisfiability in fusions of modal logics, we shall have to produce such a certificate (this is the content of Lemma 4.4.13 below).

Before closing this subsection, we make a little *digression* (not relevant for the comprehension of the remaining part of the chapter) about the choice of the word 'algebraic' in order to name our fragments (this digression may also help future development in a more conceptual setting).

It is well-known (see [58]) that higher-order (intuitionistic) type theories correspond to elementary toposes; thus our signatures must in particular be related to free toposes. Algebraic fragments in this context are cartesian subcategories of such toposes (we use the name 'cartesian category' for 'category with finite products'). In fact, algebraic fragments are closed under compositions and contains projections, namely variables. Now it is well-known from Lawvere functorial semantics (see [61]) that cartesian categories correspond to equational theories in a similar way as toposes correspond to higher-order type theories. Thus our algebraic fragments, if considered *from ouside*, are just equational (i.e. algebraic) theories. In fact, if we take for instance the algebraic fragment given by the standard translation of modal propositional formulae, if we consider it as a cartesian category by itself and if we then come back to a presentation of it as a first-order equational theory, we get the theory of modal algebras (i.e., Boolean algebras plus meet-preserving operators), namely the theory of the algebras which are used as the standard algebraic semantics for modal logic.

However, the embedding of an algebraic fragment into a higher-order language (i.e., the consideration of a specific topos in which a cartesian category is embedded - we recall that one such always exists) gives new information on the *internal* structure of the fragment itself and we want this information *to be part of our data*. In fact, when interpreting an algebraic fragment, we consider not just set-valued cartesian functors having as a domain

the cartesian category corresponding to the fragment itself, but just those such functors which are restrictions of set-valued logical functors defined on the bigger recipient topos (in the case of the above presentation of the theory of modal algebras, for instance, this means that we are considering Kripke models, not just arbitrary algebraic models). Secondly, the specification of the recipient topos seems to influence the construction of our combined larger fragments. Thirdly, the internal information on the fragment is useful to identify certain ad hoc operations on the models of the recipient topos and to exploit specific preservation properties (with respect to the formulae in the fragment) of such operations: these preservation properties will be essential ingredients for justifying completeness of combined decision procedures.

4.2.2 Examples

We give here a list of examples of i.a.f.'s; we shall mainly concentrate on those examples which will play a central role in the positive results of this thesis. In all cases, the proof that the properties of Definition 4.2.2 are satisfied is just sketched or entirely left to the reader (such proofs are all immediate or they reduce to easy inductive arguments based on standard information from Subsection 4.1.3).

Example 4.2.5 (Simply typed λ-calculus)**.** This is the i.a.f. Φ that one gets by keeping only the terms that can be built by 'omitting any reference to the type Ω'. According to Friedman theorem (see [39]), this i.a.f. has decidable word problem,[6] because $\beta\eta$-normalization can decide equality of Φ-terms in all interpretations. However, constraint satisfiability problem is no longer decidable.

Example 4.2.6 (First-order equational fragments)**.** Let us consider a first-order language $\mathcal{L} = \langle \mathcal{T}, \Sigma, a \rangle$ (for simplicity, we also assume that $\mathcal{L}$ is one-sorted). Let T be the set of the first-order $\mathcal{L}$-terms and let $\mathcal{S}$ consists of the $\mathcal{L}$-structures which happen to be models of a certain first-order theory in the signature $\mathcal{L}$. Obviously, the triple $\Phi = \langle \mathcal{L}, T, \mathcal{S} \rangle$ is an i.a.f.. The Φ-atoms will be equalities between Φ-terms, i.e. first-order atomic formulae of the kind $t_1 = t_2$. Deciding the word problem in $\Phi = \langle \mathcal{L}, T, \mathcal{S} \rangle$ is equivalent to the decidability of the standard equational fragment (as defined for the case of equational theories for instance in [11]), whereas constraint satisfiability problem is the problem of deciding satisfiability of a finite set of equations and inequations.

Example 4.2.7 (Universal first-order fragments)**.** The previous example disregards the relational symbols of the first-order signature $\mathcal{L}$. To take also them into consideration, it is sufficient to make some slight adjustment: besides first-order terms, also atomic

[6] Remember that, when no semantic class $\mathcal{S}$ is mentioned in the definition of an i.a.f., it is intended that $\mathcal{S}$ consists of all possible interpretations for the language.

formulae ($\top, \bot$ included), as well as propositional variables (namely variables having type Ω) will be terms of the fragment.[7] The semantic class $\mathcal{S}$ where the fragment is to be interpreted can be taken to be again the class of the models of some first-order theory. Then, for $\Phi{=}\langle \mathcal{L}, T, \mathcal{S}\rangle$ so defined, the constraint satisfiability problem becomes the problem of deciding the satisfiability of an arbitrary finite set of $\mathcal{L}$-literals in the models belonging to $\mathcal{S}$[8] (the complementary problem is equivalent to the problem of deciding validity of a universal first-order formula in $\mathcal{S}$).

We now define different kinds of i.a.f.'s starting from the set F of first-order formulae of a first-order signature $\mathcal{L}$; for simplicity, let us suppose also that $\mathcal{L}$ is relational and one-sorted (call W its unique sort).

Example 4.2.8 (Full first-order language, plain version)**.** We take T to be the union of F with the sets of the individual variables and of the propositional variables. Of course, $\Phi{=}\langle \mathcal{L}, T\rangle$ so defined is an algebraic fragment, whose types are W and Ω. By Church theorem, both word and constraint satisfiability problem are undecidable here (the two problems reduce to satisfiability of a first-order formula with equality); they may be decidable in case the fragment is interpreted into some specific semantic class $\mathcal{S}$. If $\mathcal{S}$ is an *elementary class* (i.e. it is the class of the models of a first-order theory), then the i.a.f. $\Phi = \langle \mathcal{L}, T, \mathcal{S}\rangle$ is called a *first-order* fragment..

In the next example, we build formulae (out of the symbols of our fixed first-order relational one-sorted signature $\mathcal{L}$) by using at most N (free or bound) individual variables; however we are allowed to use also second-order variables of arity at most K:

Example 4.2.9 (Full first-order language, NK-version)**.** Fix cardinals $K \leq N \leq \omega$ and consider, instead of F, the set F_{NK} of formulae φ that contains at most N (free or bound) individual variables and that are built up by applying boolean connectives and individual quantifiers to atomic formulae of the following two kinds:

- $P(x_{i_1}, \ldots, x_{i_n})$, where P is a relational constant and $x_{i_1}, \ldots, x_{i_n}$ are individual variables (since at most $x_1, \ldots, x_N$ can be used, we require that $i_1, \ldots, i_n \leq N$);
- $X(x_{i_1}, \ldots, x_{i_n})$, where $i_1, \ldots, i_n \leq N$, and X is a variable of type $W^n \to \Omega$ with $n \leq K$ (here W^n abbreviates $W \cdots W$, n-times).

[7] Propositional variables are added to the set of terms in order for closure under codomain variables to be satisfied, see Definition 4.2.2.

[8] $\mathcal{L}$-atomic formulae A (resp. negated $\mathcal{L}$-atomic formulae $\neg A$) can be seen as the Φ-atoms $A = \top$ (resp. $A = \bot$). One should include also equations $A = B$ and inequations $A \neq B$ among $\mathcal{L}$-atomic formulae and/or propositional variables. However, for instance, $A = B$ is satisfiable iff $A \wedge B$ is satisfiable or $\neg A \wedge \neg B$ is satisfiable: this means that, by case splitting, we can anyway reduce satisfiability of Φ-constraints to satisfiability of conjunctions of $\mathcal{L}$-atomic and negated $\mathcal{L}$-atomic formulae.

The terms in the algebraic fragment $\Phi^{\mathcal{L}}_{NK} = \langle \mathcal{L}_{NK}, T^{\mathcal{L}}_{NK} \rangle$ are now the terms t such that $t \sim_{\beta\eta} \{x_1, \ldots, x_n \mid \varphi\}$, for some $n \leq K$ and for some $\varphi \in F_{NK}$, with $fvar_W(\varphi) \subseteq \{x_1, \ldots, x_n\}$.[9] Types in such $\Phi^{\mathcal{L}}_{NK}$ are now $W^n \to \Omega$ ($n \leq K$) and this fact makes a big difference with the previous example (the difference will be sensible when combined fragments enter into the picture). Constraint satisfiability problems still reduce to satisfiability problems for sentences: in fact, once second-order variables are replaced by the names of the subsets assigned to them by some assignment α in a $\mathcal{L}$-structure, $\Phi^{\mathcal{L}}_{NK}$-atoms like $\{\underline{x} \mid \varphi\} = \{\underline{x} \mid \psi\}$ are equivalent to first-order sentences $\forall \underline{x}(\varphi \leftrightarrow \psi)$ and conversely any first-order sentence ϑ (with at most N bound individual variables) is equivalent to the $\Phi^{\mathcal{L}}_{NK}$-atom $\vartheta = \top$.

The cases $N = 1, 2$ are particularly important, because in these cases the satisfiability problem for sentences (and hence also constraint satisfiability problems in our fragments) becomes decidable (see [63, 80, 68, 81, 31]).

We mention that the previous two examples admit very important weaker versions in which some of the first-order operators are omitted. For instance, if universal quantifiers and negations are omitted, constraint satisfiability in the $\lambda_{\omega\omega}$-version becomes the problem of deciding whether a geometric sequent is entailed by a finitely axiomatized geometric theory (for this terminology, see [8] or some book in categorical logic, like [65]).

Further examples can be obtained by using the large information contained in the textbook [21] (see also [35]). We shall continue here by investigating fragments that arise from research in knowledge representation area, especially in connection to modal and description logics.

Example 4.2.10 (Modal/Description Logic fragments, global version)**.** A *modal signature* is a set O_M, whose elements are called unary 'Diamond' modal operators (the case of n-ary modal operators does not create special difficulties and it is left to the reader). O_M-modal formulae are built up from a countable set of propositional variables $\mathrm{x}, \mathrm{y}, \mathrm{z}, \ldots$ by applying $\top, \bot, \neg, \wedge, \vee$ as well as the operators $\Diamond_k \in O_M$.

With every modal signature O_M we associate the first-order signature $\mathcal{L}_M$, containing a unique sort W and, for every $\Diamond_k \in O_M$, a relational constant R_k of type $WW \to \Omega$. Suppose we are given a bijective correspondence $\mathrm{x} \longmapsto X$ between propositional variables and second-order variables of type $W \to \Omega$. Given an O_M-modal formula φ and a variable w of type W, the standard translation $ST(\varphi, w)$ is the $\mathcal{L}_M$-term of type Ω inductively

[9] We need to use $\beta\eta$-equivalence here to show that the properties of Definition 4.2.2 (namely closure under composition and under domain/codomain variables) are satisfied.

defined as follows:

$$\begin{aligned} ST(\top, w) &= \top \\ ST(\bot, w) &= \bot \\ ST(\mathrm{x}, w) &= X(w) \\ ST(\neg\psi, w) &= \neg ST(\psi, w) \\ ST(\psi_1 \vee \psi_2, w) &= ST(\psi_1, w) \vee ST(\psi_2, w) \\ ST(\psi_1 \wedge \psi_2, w) &= ST(\psi_1, w) \wedge ST(\psi_2, w) \\ ST(\Diamond\psi, w) &= \exists v(R(w, v) \wedge ST(\psi, v)) \end{aligned}$$

where v is a variable of type W (different from w). Let T_M be the set of those $\mathcal{L}_M$-terms t for which there exists a modal formula φ s.t. $t \sim_{\beta\eta} \{w \mid ST(\varphi, w)\}$. The pair $\langle \mathcal{L}_M, T_M \rangle$ is an algebraic fragment and it becomes an i.a.f. $\Phi_M = \langle \mathcal{L}_M, T_M, \mathcal{S}_M \rangle$ if we specify also a class $\mathcal{S}_M$ of $\mathcal{L}_M$-structures closed under isomorphisms (notice that $\mathcal{L}_M$-structures, usually called *Kripke frames* in modal logic, are just sets endowed with a binary relation R_k for every $\Diamond_k \in O_M$).

Φ_M-constraints can be equivalently represented in the form

$$\{w \mid ST(\psi, w)\} = \{w \mid \top\} \wedge \{w \mid ST(\varphi_1, w)\} \neq \{w \mid \bot\} \wedge \cdots \wedge \{w \mid ST(\varphi_n, w)\} \neq \{w \mid \bot\};$$

they are satisfied iff there exists a Kripke model (i.e. a Kripke frame endowed with an assignment of subsets for second-order variables of type $W \to \Omega$) based on a frame in $\mathcal{S}_M$ in which ψ holds globally (namely in every state), whereas $\varphi_1, \ldots, \varphi_n$ hold in some states $s_1, \ldots, s_n$, respectively. If $\mathcal{S}_M$ is closed under disjoint unions, we can limit ourselves to the case $n = 1$: thus constraint satisfiability problem becomes, in the description logics terminology, just the relativized satisfiability problem for a given concept description w.r.t. to a given T-Box (we call *T-Box* a Φ_M-atom like $\{w \mid ST(\psi, w)\} = \{w \mid \top\}$).[10]

Example 4.2.11 (Modal/Description Logic fragments, local version)**.** If we want to capture A-Box reasoning too, we need to build a slightly different fragment. The type-theoretic signature $\mathcal{L}_{ML}$ of our fragment $\langle \mathcal{L}_{ML}, T_{ML} \rangle$ is again $\mathcal{L}_M$, but T_{ML} now contains: a) the set of terms which are $\beta\eta$-equivalent to terms of the kind $ST(\varphi, w)$ (these terms are called 'concept assertions'); b) the terms of the kind $R_k(v, w)$ (these terms are called 'role assertions'); c) the variables of type W, Ω and $W \to \Omega$.

The pair $\langle \mathcal{L}_{ML}, T_{ML} \rangle$ is an algebraic fragment and it becomes an interpreted algebraic fragment $\Phi_{ML} = \langle \mathcal{L}_{ML}, T_{ML}, \mathcal{S}_{ML} \rangle$ if we specify also a class $\mathcal{S}_{ML}$ of $\mathcal{L}_{ML}$-structures closed

[10] Usually, a T-Box is defined as a conjunction of 'generalized concept inclusions' that are required to hold globally: this can be reduced to the requirement for a single formula to hold globally, because all boolean connectives are at our disposal.

under isomorphisms. Let us now analyze constraints in this fragment: as in Example 4.2.7, we can eliminate (by Boolean case splitting) atoms of the kind $ST(\varphi, w) = ST(\psi, v)$, $ST(\varphi, w) = R(v_1, v_2)$, etc. (and their negations), in favor of plain concept assertions and role assertions. In addition we have: a) identities among individual names (i.e. among variables of type W); b) identities among atomic concepts (i.e. among second-order variables of type $W \to \Omega$); c) propositional variables (i.e. variables of type Ω); d) negations of identities among atomic concepts; e) negations of propositional variables; f) negations of role assertions; g) negations of identities among individual names.

Now, a)-b)-c)-d)-e) can be eliminated without loss of generality: in fact, (i) all variable identities can be eliminated by replacements; (ii) negations of identities among atomic concepts can be replaced by concept assertions involving fresh variables; (iii) propositional variables and their negations do not interact with the remaining part of the constraint and can be ignored. In conclusion, Φ_{LM}-constraints are just standard A-Boxes with, in addition, negations of role assertions and of identities among individual names (notice that traditional A-Boxes automatically include all negations of identities among distinct individual variables by the so-called 'unique name assumption', see [7]. Let us call *A-Boxes* these slightly more general constraints and let us reserve the name of *positive A-Boxes* to conjunctions of concept assertions and role assertions.

Example 4.2.12 (Modal/Description Logic fragments, full version)**.** If we want to deal with satisfiability of an A-Box w.r.t. a T-Box, it is sufficient to join the two previous fragments. More precisely, we can build the fragments $\Phi_{MF} = \langle \mathcal{L}_{MF}, T_{MF}, \mathcal{S}_{MF} \rangle$, where $\mathcal{L}_{MF} = \mathcal{L}_M$ and $T_{MF} = T_M \cup T_{ML}$. Types in this fragment are W, Ω and $W \to \Omega$ constraints are conjunctions of a T-Box and an A-Box.

Example 4.2.13 (Modal/Description Logic fragments, non-normal case)**.** If we want to consider the case in which some of the operators in O_M are non-normal, we can use higher-order constants $f_k : (W \to \Omega) \to (W \to \Omega)$ (instead of binary relations $R_k : WW \to \Omega$) and define a different translation. Such a translation $NT(\varphi, w)$ differs from $ST(\varphi, w)$ for the inductive step relative to modal operators which now reads as follows:

$$NT(\Diamond_k \psi, w) = f_k(\{w \mid NT(\psi, w)\})(w).$$

Now global, local and full algebraic fragments can be defined as in the normal case. If the easy extension to n-ary non-normal cases is included and if we also interpret the resulting fragments, we get precisely the abstract description systems of [10].

Example 4.2.14 (μ-calculus)**.** We show how to build a truly second-order fragment out of a modal signature O_M (in the sense of Example 4.2.10). In the syntax of μ-calculus (see [56]), we are allowed to apply the minimum fixed point operator $\mu x\, D$ to a concept

D provided x occurs only positively or only negatively in D. According to well-known fixed point characterization, we can extend the translation ST from Example 4.2.10, by using the second-order formulae

$$ST(\mu \mathrm{x}\, D, w) := \forall Y((\{w \mid ST(D, w)\}[Y \mapsto X] \subseteq Y) \rightarrow Y(w))$$

Armed by this translation, we can easily design suitable μ-fragments.

Guarded and packed guarded fragments were introduced as generalizations of modal fragments (see [1, 49, 66]): in fact, they form classes of formulae which are remarkably large but still inherit relevant syntactic and semantic features of the more restricted modal formulae. In particular, guarded and packed guarded formulae are decidable for satisfiability (with the appropriate settings, decision procedures can be obtained also by running standard superposition provers - see [42]).

For simplicity, we give here the instructions on how to build only one version of the packed guarded fragment with equality (other versions can be built by following the methods we used above for the first-order and the modal cases). We notice that packed guarded fragments without equality are also important: to built them it is sufficient to erase any reference to the equality predicate in the relevant definitions.

Example 4.2.15 (Packed guarded fragments)**.** Let us consider a first-order one-sorted relational signature $\mathcal{L}_G$. A *guard* π is a $\mathcal{L}_G$-formula like $\bigwedge_{i=1}^{k} \pi_i$, where:

- π_i is obtained by applying existential quantifiers to atomic formulae $P_i(x_{i1}, \ldots, x_{in_i})$ where the P_i are constants of type $W^{n_i} \rightarrow \Omega$ and $x_{i1}, \ldots, x_{in_i}$ are variables of type W;
- for all $x_1, x_2 \in fvar(\pi)$, there exists an $i \in \{1, \ldots, k\}$ such that $\{x_1, x_2\} \subseteq fvar(\pi_i)$.

We define the *packed guarded formulae* as follows:

- if $X : W \rightarrow \Omega$ and $x : W$ are variables, $X(x)$ is a packed guarded formula;
- if $P : W^n \rightarrow \Omega$ is a constant and $y_1 : W, \ldots, y_n : W$ are variables, $P(y_1, \ldots, y_n)$ is a packed guarded formula;
- if φ is a packed guarded formula, $\neg\varphi$ is a packed guarded formula;
- if φ_1 and φ_2 are packed guarded formulae, $\varphi_1 \wedge \varphi_2$ and $\varphi_1 \vee \varphi_2$ are packed guarded formulae;
- if φ is a packed guarded formula and π is a guard such that $fvar_W(\varphi) \subseteq fvar(\pi)$, then $\forall \underline{y}(\pi[\underline{x}, \underline{y}] \rightarrow \varphi[\underline{x}, \underline{y}])$ and $\exists \underline{y}(\pi[\underline{x}, \underline{y}] \wedge \varphi[\underline{x}, \underline{y}])$ are packed guarded formulae.[11]

[11] If $\underline{y} = \{y_1, \ldots, y_n\}$, then $\forall \underline{y}$ means $\forall y_1 \cdots \forall y_n$ and $\exists \underline{y}$ means $\exists y_1 \cdots \exists y_n$.

Notice that we used second-order variables of type $W \to \Omega$ only (and not of type $W^n \to \Omega$ for $n > 1$): the reason, besides the applications to combined decision problems we have in mind, is that we want constraint problems to be equivalent to sentences which are still packed guarded, see below. Packed guarded formulae not containing variables of type $W \to \Omega$ are called *elementary* (or first-order) packed guarded formulae.

If we let T_G be the set of $\mathcal{L}_G$-terms t such that t is $\beta\eta$-equivalent to a term of the kind $\{w \mid \varphi\}$ (where φ is a packed guarded formula such that $fvar_W(\varphi) \subseteq \{w\}$), then the pair $\langle \mathcal{L}_G, T_G \rangle$ is an algebraic fragment. The only type in this fragment is $W \to \Omega$ and constraint satisfiability problem in this fragment is equivalent to satisfiability of guarded sentences: this is because, in case φ_1, φ_2 are packed guarded formulae with $fvar_W(\varphi_i) \subseteq \{w\}$ (for $i = 1, 2$), then $\{w \mid \varphi_1\} = \{w \mid \varphi_2\}$ is equivalent to $\forall w(\varphi_1 \leftrightarrow \varphi_2)$ which is packed guarded (just use $w = w$ as a guard).

4.2.3 Reduced Fragments and Residues

If $\Phi = \langle \mathcal{L}, T, \mathcal{S} \rangle$ is an i.a.f., we shall use Greek letters $\Gamma, \Delta, \Lambda \ldots$ for Φ-constraints (i.e. for finite sets of Φ-literals) and letters $\Theta, \Xi, \ldots$ for finite sets of Φ-clauses. We recall that a $\Phi(\underline{x})$-constraint is a Φ-constraint in which at most the variables $\underline{x}$ occur free; analogously, a $\Phi(\underline{x})$-clause is a Φ-clause in which at most the variables $\underline{x}$ occur free. If Θ is a set of such $\Phi(\underline{x})$-clauses and $C \equiv L_1 \vee \cdots \vee L_k$ is a $\Phi(\underline{x})$-clause, we say that C is a Φ-*consequence of* Θ (written $\Theta \models_\Phi C$) iff the set of formulae $\Theta \cup \{\neg L_1, \ldots, \neg L_k\}$ is not Φ-satisfiable (i.e. iff such a set is not satisfiable in any $\mathcal{A} \in \mathcal{S}$).

The notion of consequence is too strong for certain applications; for instance, when we simply need to delete certain deductively useless data, a weaker notion of redundancy (based e.g. on subsumption) is preferable. We shall give abstract notions of redundancy and of consequences enumeration in this subsection, however these notions are less sophisticated than similar notions introduced within saturation-based theorem proving (see, e.g., [15]). The reason why we need them here is that they will make our combined decision procedure more flexible, as explained below.

Our abstract axiomatization of a notion of redundancy is the following (recall that we conventionally included $\top$ and $\bot$ among Φ-atoms in any i.a.f. Φ):

Definition 4.2.16. A *redundancy notion* for a fragment Φ is a *recursive* binary relation Red_Φ between a finite set of Φ-clauses Θ and a Φ-clause C satisfying the following properties:

(i) $Red_\Phi(\Theta, C)$ implies $\Theta \models_\Phi C$ (soundness);

(ii) $Red_\Phi(\emptyset, \top)$ and $Red_\Phi(\{\bot\}, C)$ both hold;

(iii) $Red_\Phi(\Theta, C)$ and $\Theta \subseteq \Theta'$ imply $Red_\Phi(\Theta', C)$ (monotonicity);

(iv) $Red_\Phi(\Theta, C)$ and $Red_\Phi(\Theta \cup \{C\}, D)$ imply $Red_\Phi(\Theta, D)$ (transitivity);

(v) if C is subsumed by some $C' \in \Theta$,[12] then $Red_\Phi(\Theta, C)$ holds.

Whenever a redundancy notion Red_Φ is fixed, we say that C is Φ-redundant w.r.t. Θ when $Red_\Phi(\Theta, C)$ holds.

For example, the *minimum* redundancy notion is obtained by stipulating that $Red_\Phi(\Theta, C)$ holds precisely when ($\bot \in \Theta$ or $C \equiv \top$ or $C \equiv \top \vee D$ or C is subsumed by some $C' \in \Theta$). On the contrary, if the constraint solving problem for Φ is decidable, there is a maximum redundancy notion (called the *full* redundancy notion) given by the Φ-consequence relation. In fact, it is evident that a recursive procedure for Φ-constraint solving is a recursive procedure deciding $\Theta \models_\Phi C$, for finite Θ.

Let $\Phi = \langle \mathcal{L}, T, \mathcal{S} \rangle$ be an i.a.f. on the signature $\mathcal{L} = \langle \mathcal{T}, \Sigma, a \rangle$ and let $\mathcal{L}_0 = \langle \mathcal{T}_0, \Sigma_0, a_0 \rangle$ be a subsignature of $\mathcal{L}$. The i.a.f. *restricted to* $\mathcal{L}_0$ is the i.a.f. $\Phi_{|\mathcal{L}_0} = \langle \mathcal{L}_0, T_{|\mathcal{L}_0}, \mathcal{S}_{|\mathcal{L}_0} \rangle$ so defined:

- $T_{|\mathcal{L}_0}$ is the set of terms obtained by intersecting T with the set of $\mathcal{L}_0$-terms;
- $\mathcal{S}_{|\mathcal{L}_0}$ consists of the structures of the kind $\mathcal{A}_{|\mathcal{L}_0}$, varying $\mathcal{A} \in \mathcal{S}$.

An i.a.f. $\Phi_0 = \langle \mathcal{L}_0, T_0, \mathcal{S}_0 \rangle$ is said to be a $\mathcal{L}_0$-*subfragment* (or simply a subfragment, leaving the subsignature $\mathcal{L}_0 \subseteq \mathcal{L}$ as understood) of $\Phi = \langle \mathcal{L}, T, \mathcal{S} \rangle$ iff $T_0 \subseteq T_{|\mathcal{L}_0}$ and $\mathcal{S}_0 \supseteq \mathcal{S}_{|\mathcal{L}_0}$. In this case, we may also say that Φ is an *expansion* (or an *extension*) of Φ_0.

Given the notions of redundancy and of subfragment, we are now ready to restate in these general settings the notions of basis and of enumerators already introduced in Section 3.5. Given a set Θ of $\Phi(\underline{x})$-clauses and a redundancy notion Red_{Φ_0} on a subfragment Φ_0 of Φ, we call Φ_0-*basis for* Θ a set Ξ of $\Phi_0(\underline{x}_0)$-clauses such that (here $\underline{x}_0$ collects those variables among the $\underline{x}$ which happen to be Φ_0-variables):

(i) all clauses $D \in \Xi$ are positive and are such that $\Theta \models_\Phi D$;[13]

(ii) every positive $\Phi_0(\underline{x}_0)$-clause C such that $\Theta \models_\Phi C$ is Φ_0-redundant with respect to Ξ.

As in the first-order case, the intuitive meaning of residues is that of clauses which are recursively enumerated by a suitable device (the device may for instance be an enumerator of certain proofs of a calculus, but there is no need to think of it in this way):

Definition 4.2.17. Suppose we are given a subfragment Φ_0 of a fragment Φ. A *positive residue* Φ-*enumerator for* Φ_0 (often shortened as Φ-p.r.e.) is a recursive function mapping

[12] As usual, this means that every literal of C' is also in C.

[13] Recall that we conventionally included $\bot$ among Φ-atoms, so $\bot$ is considered as a positive clause.

a finite set $\underline{x}$ of Φ-variables, a finite set Θ of $\Phi(\underline{x})$-clauses and a natural number i to a Φ_0-clause $Res^{\underline{x}}_{\Phi}(\Theta, i)$ (to be written simply as $Res_\Phi(\Theta, i)$) in such a way that:

- $Res_\Phi(\Theta, i)$ is a positive clause;
- $fvar(Res_\Phi(\Theta, i)) \subseteq \underline{x}$;
- $\Theta \models_\Phi Res_\Phi(\Theta, i)$ (soundness).

Any Φ_0-clause of the kind $Res_\Phi(\Theta, i)$ (for some $i \geq 0$) will be called a Φ_0-*residue* of Θ.

Having also a redundancy notion for Φ_0 at our disposal, we can axiomatize the notion of an 'optimized' (i.e. of a non-redundant) Φ-p.r.e. for Φ_0. The version of the Nelson-Oppen combination procedure we give in Subsection 4.3.2 has non-redundant p.r.e.'s as main ingredients and it is designed to be 'self-adaptive' for termination in the relevant cases when termination follows from our results. These are basically the noetherian and the locally finite cases mentioned in Subsection 4.2.4, where p.r.e.'s which are non-redundant with respect to the full redundancy notion usually exist and enjoy the termination property below.

Definition 4.2.18. A Φ-p.r.e. Res_Φ for Φ_0 is said to be *non-redundant* (w.r.t. a redundancy notion Red_{Φ_0}) iff it satisfies also the following properties for every $\underline{x}$, for every finite set Θ of $\Phi(\underline{x})$-clauses and for every $i \geq 0$ (we write $\Theta_{|\Phi_0}$ for the set of clauses in Θ which are Φ_0-clauses):

(i) if $Res_\Phi(\Theta, i)$ is Φ_0-redundant with respect to $\Theta_{|\Phi_0} \cup \{Res_\Phi(\Theta, j) \mid j < i\}$, then $Res_\Phi(\Theta, i)$ is either $\bot$ or $\top$;

(ii) if $\bot$ is Φ_0-redundant with respect to $\Theta_{|\Phi_0} \cup \{Res_\Phi(\Theta, j) \mid j < i\}$, then $Res_\Phi(\Theta, i)$ is equal to $\bot$;

(iii) if $Res_\Phi(\Theta, i)$ is equal to $\top$, then $\Theta_{|\Phi_0} \cup \{Res_\Phi(\Theta, j) \mid j < i\}$ is a Φ_0-basis for Θ.

Definition 4.2.19. A non-redundant Φ-p.r.e. for Φ_0 is said to be *complete* iff for every $\underline{x}$, for every finite set Θ of $\Phi(\underline{x})$-clauses and for every positive $\Phi_0(\underline{x})$-clause C, we have that $\Theta \models_\Phi C$ implies that C is Φ_0-redundant w.r.t. $\Theta_{|\Phi_0} \cup \{Res_\Phi(\Theta, j) \mid j \leq i\}$ for some i.

A non-redundant Φ-p.r.e. Res_Φ is said to be *terminating* iff for for every $\underline{x}$, for every finite set Θ of $\Phi(\underline{x})$-clauses there is an i such that $Res_\Phi(\Theta, i)$ is equal to $\bot$ or to $\top$.

Let us make a few comments on Definition 4.2.18: first, only non-redundant residues can be produced at each step (condition (i)), if possible. If it is not possible, this means that all the relevant information has been accumulated (a Φ_0-basis has been reached).

In this case, if the inconsistency $\bot$ is discovered (in the sense that it is perceived as redundant), then the residue enumeration in practice stops, because it becomes constantly equal to $\bot$ (condition (ii)). The tautology $\top$ has the special role of marking the opposite outcome: it is the residue that is returned precisely when Θ is consistent and a Φ_0-basis has been produced, meaning that all relevant semantic consequences of Θ have been discovered (conditions (ii)-(iii)).

If the redundancy notion we use is trivial (i.e. it is the minimum one), then only very mild corrections are needed for a Φ-p.r.e. for Φ_0 to become non-redundant: apart from minor ad hoc modifications,[14] we only need to make it constantly equal to $\bot$, as soon as $\bot$ becomes redundant in the enumeration. This observation shows that, in practice, any Φ-p.r.e. for Φ_0 can be made non-redundant and can consequently be used as input of our combined decision procedure.

The role of $\top$ as a residue is precious in case for some special reasons (typically exemplified in computational algebra, see Subsection 4.2.5 below), we have an effective procedure which is able to recognize whether a given set of positive Φ_0-clauses forms a Φ_0-basis for Θ with respect to the full redundancy notion: if these *full* Φ_0-*bases* for Θ can be effectively recognized and if also Φ_0-consequence is decidable, we can always turn a complete Φ-p.r.e. for Φ_0 into a non-redundant one with respect to the full redundancy notion. The advantage of this optimization is that the combined decision procedure of Subsection 4.3.2, after getting $\top$ or $\bot$ as residues, automatically recognizes that the residue exchange is over and halts.

4.2.4 Noetherian, Locally Finite and Convex Fragments

The above mentioned optimization for p.r.e.'s usually apply to the cases in which the 'small' fragment Φ_0 is noetherian. In Section 3.5.1 we have already defined this peculiar kind of theories; if we translate it into our general setting, we get the following definition.

An i.a.f. Φ_0 is called *noetherian* if and only if for every finite set of variables $\underline{x}$, every infinite ascending chain

$$\Delta_1 \subseteq \Delta_2 \subseteq \cdots \subseteq \Delta_n \subseteq \cdots$$

of sets of $\Phi_0(\underline{x})$-atoms is definitively constant for Φ_0-consequence (meaning that there is an n such that for all m and $A \in \Delta_m$, we have $\Delta_n \models_{\Phi_0} A$).

[14] These modifications are possible provided that there are countably many closed Φ_0-atoms equivalent to $\top$ but syntactically different from it: if there are such infinitely many closed Φ_0-atoms which are 'copies' of $\top$, then we can replace $Res_\Phi(\Theta, i)$ by one of them in case $Res_\Phi(\Theta, i)$ is redundant with respect to $\Theta_{|\Phi_0} \cup \{Res_\Phi(\Theta, j) \mid j < i\}$. By using this trick, conditions (i) and (iii) of Definition 4.2.18 can be forced, if the underlying redundancy notion for Φ_0 is the *minimum* one. The hypothesis that Φ_0 is endowed with such infinitely many 'copies' of $\top$ is not really restrictive and can be always obtained by slight modifications of Φ_0.

We want to recapture also the class of locally finite theories: to this aim, it is sufficient to adapt the definition of Section 3.4 as follows.

An i.a.f. Φ_0 is said to be *effectively locally finite* iff

(i) the set of Φ_0-types is recursive and constraint satisfiability problem for Φ_0 is decidable;

(ii) for every finite set of Φ_0-variables $\underline{x}$, there are finitely many computable $\Phi_0(\underline{x})$-terms $t_1, \ldots, t_n$ such that for every further $\Phi_0(\underline{x})$-term u one of the literals $t_1 \neq u, \ldots, t_n \neq u$ is not Φ_0-satisfiable (that is, in the class of the structures in which Φ_0 is interpreted, every $\Phi_0(\underline{x})$-term is equal, as an interpreted function, to one of the t_i).

The terms $t_1, \ldots, t_n$ in (ii) are called the $\underline{x}$-representative terms of Φ_0.

The adaptation of Proposition 3.5.2 to these new definitions is immediate.

Proposition 4.2.20. *In a noetherian fragment Φ_0 every infinite ascending chain of sets of positive $\Phi_0(\underline{x})$-clauses is definitively constant for Φ_0-consequence.*

Suppose that Φ_0 is noetherian and that Φ is an expansion of it: by the above proposition, it is immediate to see that every finite set of $\Phi(\underline{x})$-clauses Θ has a finite full Φ_0-basis (i.e. there is a finite Φ_0-basis for Θ with respect to the full redundancy notion). The following noetherianity requirement for a p.r.e. is intended to be nothing but an effectiveness requirement for the computation of finite full Φ_0-bases.

A Φ-p.r.e. Res_Φ for a noetherian fragment Φ_0 is said to be *noetherian* iff it is non-redundant with respect to the full redundancy notion for Φ_0. An immediate consequence of Proposition 4.2.20 is that:

Proposition 4.2.21. *A noetherian Φ-p.r.e. Res_Φ for Φ_0 is terminating and also complete.*

It is straightforward, given Proposition 3.5.3, to verify that the following proposition holds:

Proposition 4.2.22. *If Φ_0 is effectively locally finite and Φ is any extension of it having decidable constraint satisfiability problem, then there always exists a noetherian Φ-p.r.e. for Φ_0.*

Proof. Do as in the proof of Proposition 3.5.3; at the end, instead of returning a basis, list (up to Φ_0-redundancy, which can be effectively checked) the clauses whose test for Φ-consequence is positive and to give $\top$ as a final message. □

We shall see that, when dealing with noetherian p.r.e.'s over a noetherian shared fragment, the combination procedure of Subsection 4.3.2 becomes automatically terminating. However, the informal remarks we made at the end of Section 3.5.1 about complexity issue are still valid: in fact noetherianity is the key ingredient for termination, but convexity is the key property for efficiency.

We paraphrase the definition of convexity with respect a subfragment as follows: an i.a.f. Φ is Φ_0-*convex* (here Φ_0 is a subfragment of Φ) iff every finite set Γ of Φ-literals having as a Φ-consequence the disjunction of $n > 1$ Φ_0-atoms, actually has as a Φ-consequence one of them. When we say that a fragment Φ is *convex* tout court, we mean that it is Φ-convex. The fragments $\Phi = \langle \mathcal{L}, T, \mathcal{S} \rangle$ analyzed in Example 4.2.6 are convex in case $\mathcal{S}$ is the class of the models of a first-order Horn theory. Similarly, a Φ-p.r.e. for Φ_0 is Φ_0-*convex* iff $Res_\Phi(\Gamma, i)$ is always an atom (recall that by our conventions, this includes the case in which it is $\top$ or $\bot$). Any complete non-redundant Φ-p.r.e. for Φ_0 can be turned into a Φ_0-convex complete non-redundant Φ-p.r.e. for Φ_0, in case Φ is Φ_0-convex. Thus the combination procedure of Subsection 4.3.2 is designed in such a way that it becomes automatically *deterministic* if the component fragments are both convex with respect to the shared fragment.

4.2.5 Further Examples

We can easily translate the Examples 3.6.1, 3.6.2, 3.6.5 of Section 3.6 of noetherian first-order theories into our framework.

Example 4.2.23 (K-algebras)**.** Given a field K, let us consider the one-sorted language $\mathcal{L}_{Kalg}$, whose signature contains the constants $0, 1$ of type V (V is the unique sort of $\mathcal{L}_{Kalg}$), the two binary function symbols $+$, $\circ$ of type $VV \to V$, the unary function symbol $-$ of type $V \to V$ and a K-indexed family of unary function symbols g_k of type $V \to V$. We consider the i.a.f. $\Phi_{Kalg} = \langle \mathcal{L}_{Kalg}, T_{Kalg}, \mathcal{S}_{Kalg} \rangle$ where T_{Kalg} is the set of first-order terms in the above signature (we shall use infix notation for $+$ and write kv, v_1v_2 for $g_k(v)$, $\circ(v_1, v_2)$, respectively). Furthermore, the class $\mathcal{S}_{Kalg}$ consists of the structures which happen to be models for the theory of (commutative, for simplicity) K-algebras. In this way the conditions of the definition of interpreted algebraic fragment are completely fulfilled, and also in this formalism, the atoms of the fragment are simply the equation of the kind $p = 0$, where p is a polynomial.

Example 4.2.24 (K-vector spaces)**.** As a subfragment of the previous fragment, let us consider the fragment $\Phi_K = \langle \mathcal{L}_K, T_K, \mathcal{S}_K \rangle$, where we forget in the signature the ring multiplication $\circ$ and the ring unit 1; the structures in $\mathcal{S}_K$ are now the K-vector spaces and the terms in T_K (namely the first-order terms in $\mathcal{L}_K$) can consequently be represented as linear homogeneous polynomials with K-coefficients.

Finally, we translate Example 3.6.5 as follows.

Example 4.2.25 (K-vector spaces with an endomorphism)**.** We add to the signature $\mathcal{L}_K$ a unary function symbol f and, in order to interpret the fragment, we take K-vector spaces endowed with an endomorphism (call this fragment $\Phi_{Kend} = \langle \mathcal{L}_{Kend}, T_{Kend}, \mathcal{S}_{Kend} \rangle$ and the structures in $\mathcal{S}_{Kend}$ f-K-vector spaces). Terms in this fragment formally represent vectors in finitely generated free $\mathcal{S}_{Kend}$-algebras and hence normalize to the form $k_1 f^{m_1}(x_{i_1}) + \cdots + k_n f^{m_n}(x_{i_n})$ ($k_j \in K$, $j \in \{1, \ldots, n\}$). The algorithm in order to extract a *noetherian* Φ_{Kend}-p.r.e. for the subfragment Φ_K is exactly the same described in Example 3.6.5.

4.3 Combined Fragments

We give now the formal definition for the operation of combining fragments.

Definition 4.3.1. Let $\Phi_1 = \langle \mathcal{L}_1, T_1, \mathcal{S}_1 \rangle$ and $\Phi_2 = \langle \mathcal{L}_2, T_2, \mathcal{S}_2 \rangle$ be i.a.f.'s on the languages $\mathcal{L}_1$ and $\mathcal{L}_2$ respectively; we define the *shared fragment* of Φ_1, Φ_2 as the i.a.f. $\Phi_0 = \langle \mathcal{L}_0, T_0, \mathcal{S}_0 \rangle$, where

- $\mathcal{L}_0 := \mathcal{L}_1 \cap \mathcal{L}_2$;
- $T_0 := T_{1|\mathcal{L}_0} \cap T_{2|\mathcal{L}_0}$;
- $\mathcal{S}_0 := \mathcal{S}_{1|\mathcal{L}_0} \cup \mathcal{S}_{2|\mathcal{L}_0}$.

Thus the Φ_0-terms are the $\mathcal{L}_0$-terms that are both Φ_1-terms and Φ_2-terms, whereas the Φ_0-structures are the $\mathcal{L}_0$-structures which are reducts either of a Φ_1- or of a Φ_2-structure. According to the above definition, Φ_0 is a subfragment of both Φ_1 and Φ_2.

Definition 4.3.2. The *combined fragment* of the i.a.f.'s Φ_1 and Φ_2 is the i.a.f.

$$\Phi_1 \oplus \Phi_2 = \langle \mathcal{L}_1 \cup \mathcal{L}_2, T_1 \oplus T_2, \mathcal{S}_1 \oplus \mathcal{S}_2 \rangle$$

in the language $\mathcal{L}_1 \cup \mathcal{L}_2$ such that:

- $T_1 \oplus T_2$ is the smallest set of $\mathcal{L}_1 \cup \mathcal{L}_2$-terms which includes $T_1 \cup T_2$, is closed under composition and contains domain and codomain variables;
- $\mathcal{S}_1 \oplus \mathcal{S}_2 = \{\mathcal{A} \mid \mathcal{A}$ is a $\mathcal{L}_1 \cup \mathcal{L}_2$-structure s.t. $\mathcal{A}_{|\mathcal{L}_1} \in \mathcal{S}_1$ and $\mathcal{A}_{|\mathcal{L}_2} \in \mathcal{S}_2\}$.

$T_1 \oplus T_2$ is defined in such a way that conditions (i)-(ii)-(iii) from Definition 4.2.2 are matched;[15] of course, since $\Phi_1 \oplus \Phi_2$-types turn out to be just the types which are either Φ_1- or Φ_2-types, closure under domain and codomain variables comes for free.

[15] In Subsection 4.3.1 we shall prove that $T_1 \oplus T_2$ is recursive (given that T_1 and T_2 are recursive).

4.3.1 The Purification Steps

We say that a $\Phi_1 \oplus \Phi_2$-term is *pure* iff it is a Φ_i-term ($i = 1$ or $i = 2$) and that a $\Phi_1 \oplus \Phi_2$-constraint Γ is *pure* iff it for each literal $L \in \Gamma$ there is $i = 1$ or $i = 2$ such that L is a Φ_i-literal. Constraints in combined fragments can be purified, as we shall see. Before giving the related procedure, we first have a better look to terms in a combined fragment $\Phi_1 \oplus \Phi_2 = \langle \mathcal{L}_1 \cup \mathcal{L}_2, T_1 \oplus T_2, \mathcal{S}_1 \oplus \mathcal{S}_2 \rangle$. For a $\mathcal{L}_1 \cup \mathcal{L}_2$-term t and for a natural number n, the relation $\delta(t, n)$ (written as $\delta(t) \leq n$) holds whenever one of the following non mutually exclusive conditions apply:

- $n \geq 0$ and t is a shared variable (i.e. a Φ_0-variable);
- $n \geq 1$ and $t \in T_1 \cup T_2$;
- $n \geq 2$ and there are $r, s > 0$, there are terms $u[x_1, \ldots, x_k], t_1, \ldots, t_k$ such that $n = r + s$, $\delta(u) \leq r$, $\delta(t_1) \leq s, \ldots, \delta(t_k) \leq s$ and t is equal to $u[t_1/x_1, \ldots, t_k/x_k]$.

Notice that if $\delta(t) \leq n$ holds and if $n \leq m$, then $\delta(t) \leq m$ holds too. The *degree* $\delta(t)$ of a $\mathcal{L}_1 \cup \mathcal{L}_2$-term t is the minimum d such that $\delta(t) \leq d$ holds (provided such a d exists, otherwise the degree of t is said to be infinite). It turns out that terms having degree 0 are just shared variables and terms having degree 1 are pure Φ_i-terms which are not shared variables. The following lemma is easily proved by induction:

Lemma 4.3.3. *$\mathcal{L}_1 \oplus \mathcal{L}_2$-terms t satisfying $\delta(t) \leq n$ are closed under substitutions mapping variables into variables.*

Lemma 4.3.4. *A term $t \in \mathcal{L}_1 \cup \mathcal{L}_2$ belongs to $T_1 \oplus T_2$ iff it has a finite degree.*

Proof. For $n \geq 1$, the set of terms t satisfying $\delta(t) \leq n$ contains domain and codomain variables (essentially because T_1 and T_2 contain their domain and codomain variables).

Let us show that terms having finite degree are closed under composition: take terms $u[x_1, \ldots, x_k]$ and $t_1, \ldots, t_k$ (all having finite degree) and suppose that types are compatible for substitution. We must show that $u[t_1/x_1, \ldots, t_k/x_k]$ has finite degree: this is obvious if u is a variable and, if all t_i are variables, we can use the above mentioned fact that terms having finite degree are closed under substitutions mapping variables into variables. Otherwise $\delta(u[x_1, \ldots, x_k]) = s_1 > 0$ and $\delta(t_1) = m_1, \ldots, \delta(t_k) = m_k$. For $s_2 := \max(m_1, \ldots, m_k) > 0$, we have that $\delta(u[t_1/x_1, \ldots, t_k/x_k]) \leq s_1 + s_2$ has finite degree too.

We proved that terms of finite degree satisfy conditions (i)-(ii)-(iii) of Definition 4.2.2. Vice versa, if $\delta(t) \leq n$, then it is immediate to see that t belongs to any set of $\mathcal{L}_1 \cup \mathcal{L}_2$-terms containing $T_1 \cup T_2$ and satisfying such conditions. □

Corollary 4.3.5. *$T_1 \oplus T_2$ is recursive.*

Proof. This is an effective procedure (based on Lemma 4.3.4) that determines whether a given term $t \in \mathcal{L}_1 \oplus \mathcal{L}_2$ belongs to the combined fragment. We associate with t the complexity measure $\rho(t)$ given by the sum of the size of t and of the number of occurrences of constants in t. We first check whether t is a pure Φ_i-term; if not, in order for t to belong to $T_1 \oplus T_2$, the degree of t (namely the *smallest* n such that $\delta(t) \leq n$ holds) must be some $n > 1$, which means that we can split it as $u[t_1, \ldots, t_k]$, where $\delta(u) \leq r$, $\delta(t_1), \ldots, \delta(t_k) \leq s$, $r + s = n$, and $r, s > 0$. Since $n > r$, by Lemma 4.3.3 it follows that at least one of the t_i is not a variable and u cannot be a variable too because $n > s$; but this means that $\rho(u) < \rho(t), \rho(t_1) < \rho(t), \ldots, \rho(t_k) < \rho(t)$, i.e. that we can recursively check whether $u, t_1, \ldots, t_k$ have finite degree. □

The membership problem $t \in T_1 \oplus T_2$ however might be computationally hard: since we basically have to guess a subtree of the position tree of the term t, *the procedure we sketched is in NP*. For instance, if we combine $T_1 = \{f_1(f_0^{2n+1}(x)) \mid n \geq 0\}$ with $T_2 = \{f_0^{2n+1}(f_2(x)) \mid n \geq 0\}$,[16] then it is evident that in order to get a good splitting of $f_1(f_0^4(f_2(x)))$ one might need to backtrack a first inappropriate attempt like

$$f_1(f_0^2(y)) \quad \text{and} \quad y \mapsto f_0^2(f_2(x)).$$

Notice however that these complications in complexity (with respect to the plain Nelson-Oppen case) are due to our level of generality and that they disappear in customary situations where don't know non-determinism can be avoided by looking for 'alien' subterms, see [12] for a thorough discussion of the problem in standard first-order cases.

Let Γ be now any $\Phi_1 \oplus \Phi_2$-constraint: we shall provide finite sets Γ_1, Γ_2 of Φ_1- and Φ_2-literals, respectively, such that Γ is $\Phi_1 \oplus \Phi_2$-satisfiable iff $\Gamma_1 \cup \Gamma_2$ is $\Phi_1 \oplus \Phi_2$-satisfiable.

The purification process is obtained by iterated applications of the following:

Purification Rule

$$\frac{\Gamma', A[t, \underline{x}]}{\Gamma',\ A[y, \underline{x}],\ y = t} \tag{4.1}$$

where (we use notations like $\Gamma', A[t, \underline{x}]$ for the constraint $\Gamma' \cup \{A[t, \underline{x}]\}$)

- t is a non-variable term (let τ be its type);
- y is a variable of type τ occurring in $A[y, \underline{x}]$ but not occurring in $\Gamma', A[t, \underline{x}]$;[17]

[16] To complete the settings for this example, we may assume that $a(f_1) = S_0 \to S_1$, $a(f_2) = S_2 \to S_0$, $a(f_0) = S_0 \to S_0$ (f_0 is the unique shared symbol). Suitable variables should also be added to T_1, T_2 to formally fulfill the conditions of Definition 4.2.2.

[17] Recall that, from our conventions in Subsection 4.1.3, the notation $A[y, \underline{x}]$ means that $fvar(A) \subseteq \{y, \underline{x}\}$ and $A[t, \underline{x}]$ means the formula obtained by applying to A the substitution $y \mapsto t$.

– the literal $A[y, \underline{x}]$ is not an equation between variables;

– Γ', $A[y, \underline{x}]$, $y = t$ is a $\Phi_1 \oplus \Phi_2$-constraint (this means that it still consists of equations and inequations among $\Phi_1 \oplus \Phi_2$-terms).

The meaning of the Purification Rule is that *we are allowed to simultaneously abstract out in a constraint one or more occurrences of a non-variable subterm* t, provided we still produce a $\Phi_1 \oplus \Phi_2$-constraint (for termination, we must also take care of not introducing variable equations).

Proposition 4.3.6. *An application of Purification Rule produces an equisatisfiable constraint.*

Proof. The constraint $\Gamma', A[t, \underline{x}]$ is satisfied in a $\mathcal{L}_1 \cup \mathcal{L}_2$-structure $\mathcal{A} \in \mathcal{S}_1 \oplus \mathcal{S}_2$ under the assignment α iff the constraint Γ', $A[y, \underline{x}]$, $y = t$ produced by the rule is satisfied in $\mathcal{A}$ under the assignment obtained by incrementing α with $y \mapsto \mathcal{I}^{\alpha}_{\mathcal{A}}(t)$. □

The *purification process* takes as input an arbitrary $\Phi_1 \oplus \Phi_2$-constraint Γ and applies it the Purification Rule as far as possible. The Purification Rule can be applied in a don't care non-deterministic way (however recall that in order to apply the rule one must before take care of the fact that the constraint produced by it still consists of $\Phi_1 \oplus \Phi_2$-literals, hence don't know non-determinism may arise inside a single application of the rule).

Proposition 4.3.7. *The purification process terminates and returns a set* $\Gamma_1 \cup \Gamma_2$*, where* Γ_i *is a set of* Φ_i*-literals.*

Proof. The terminating property is proved as follows. First notice that, after an application of the Purification Rule, the number N of the non-variable subterm positions of the current constraint cannot increase. New equations are added by the rule, but these are only equations between a variable and a non-variable term occurring in the constraint, so that the overall number of equations that can be added during the purification process does not exceed N (notice that, after the rule has produced Γ', $A[y, \underline{x}], y = t$, the new position in which the subterm t is now is not available for another purification step, since purification steps cannot produce variables equations).

Let us now show that if the Purification Rule does not apply to Γ, then Γ splits into two pure Φ_i-constraints. We first claim that, since Purification Rule does not apply to Γ, any term t in a literal $t = v$ or $t \neq v$ of Γ has degree at most 1 (i.e. it is either in T_1 or in T_2): otherwise we have $t \equiv u[t_1, \ldots, t_k]$, with $u[x_1, \ldots, x_k], t_1, \ldots, t_k$ all having lower degree than t. Since the degree of u and of the t_i's is lower than the degree of t, both u and at least one of the t_i are not a variable (see Lemma 4.3.3); suppose for instance

that t_1 is not a variable and that the constraint Γ is $\Gamma', u[t_1, \ldots, t_k] = v$. Contrary to the assumption, the Purification Rule applies to Γ and produces the constraint

$$\Gamma',\ u[x_1, t_2, \ldots, t_k] = v,\ x_1 = t_1 \tag{4.2}$$

(x_1 can be renamed, if needed):[18] in fact fragments are closed under domain/codomain variables, hence the variable we need is at our disposal, so that (4.2) is a $\Phi_1 \oplus \Phi_2$-constraint (notice that $u[x_1, t_2, \ldots, t_k]$ has a degree, hence it is a $\Phi_1 \oplus \Phi_2$-term).

Having established that terms in Γ are all pure, we wonder whether there are impure (in)equations. It is also impossible, because the Purification Rule can replace e.g. $t_1 = t_2$ by $t_1 = x \wedge x = t_2$ in case $t_1 \in T_1, t_2 \in T_2$ are non-variable terms (since fragments are closed under codomain variables, if $t_1 : \tau_1 \in T_1$, $t_2 : \tau_2 \in T_2$ and $\tau := \tau_1 = \tau_2$, then the type τ is shared and $t_i = x$ is a Φ_i-atom for every variable $x : \tau$). □

Actually, one can prove that the Purification Rule (if exhaustively applied) can bring the current constraint into a pure and *flat* form (i.e. in a form in which negative literals just contain variables and positive literals do not contain equations among two non-variable terms).

4.3.2 The Combination Procedure

In this subsection, we develop a procedure which is designed to solve constraint satisfiability problems in combined fragments: the procedure is sound and we shall investigate afterwards sufficient conditions for it to be terminating and complete. Let us fix (once and for all) relevant notation for the involved data.

Assumptions/Notational Conventions.

We suppose that we are given two i.a.f.'s $\Phi_1 = \langle \mathcal{L}_1, T_1, \mathcal{S}_1 \rangle$ *and* $\Phi_2 = \langle \mathcal{L}_2, T_2, \mathcal{S}_2 \rangle$*, with shared fragment* $\Phi_0 = \langle \mathcal{L}_0, T_0, \mathcal{S}_0 \rangle$*. We suppose also that a redundancy notion* Red_{Φ_0} *for* Φ_0 *and two non-redundant* Φ_i*-p.r.e.'s for* Φ_0 *(call them* Res_{Φ_1}*,* Res_{Φ_2}*) are available.*[19] *We also fix a purified* $\Phi_1 \oplus \Phi_2$*-constraint* $\Gamma_1 \cup \Gamma_2$ *to be tested for* $\Phi_1 \oplus \Phi_2$*-consistency; we can freely suppose that* Γ_1 *and* Γ_2 *contain the same subset* Γ_0 *of* Φ_0*-literals (i.e. that* $\Gamma_0 := \Gamma_{1|\Phi_0} = \Gamma_{2|\Phi_0}$*).*[20] *We indicate by* $\underline{x}_i$ *the free variables occurring in* Γ_i *(*$i = 1, 2$*);* $\underline{x}_0$ *are those variables among* $\underline{x}_1 \cup \underline{x}_2$ *which happen to be* Φ_0*-variables (again we can freely suppose that* $\underline{x}_0 = \underline{x}_1 \cap \underline{x}_2$*).*[21]

[18] This might be needed in order to fulfill the Purification Rule requirement that x_1 does not occur in Γ.

[19] Of course, Res_{Φ_1} and Res_{Φ_2} are assumed to be both non-redundant with respect to Red_{Φ_0}.

[20] Otherwise, Φ_0-literals can be added to Γ_1 and Γ_2 till this holds.

[21] Otherwise, equations like $x = x$ can be added to the Γ_i.

In order to describe the procedure we also need a selection function in the sense of the following definition:

Definition 4.3.8. A *selection function* CHOOSE(Λ) is a recursive function accepting as input a set Λ of $\Phi_0(\underline{x}_0)$-atoms and returning a positive $\Phi_0(\underline{x}_0)$-clause C such that:

(i) C is a Φ_i-consequence of $\Gamma_i \cup \Lambda$, for $i = 1$ or $i = 2$;

(ii) if $\bot$ is Φ_0-redundant w.r.t. $\Gamma_0 \cup \Lambda$, then C is $\bot$;

(iii) if C is Φ_0-redundant w.r.t. $\Gamma_0 \cup \Lambda$, then C is $\top$ or $\bot$.

The recursive function CHOOSE(Λ) will be subject also to a fairness requirement that will be explained below.

The Function FComb

Our combined procedure generates a tree whose internal nodes are labeled by sets of $\Phi_0(\underline{x}_0)$*-atoms; leaves are labeled by "unsatisfiable" or by "saturated". The root of the tree is labeled by the empty set and if a node is labeled by the set* Λ*, then the successors are:*

- *a single leaf labeled "unsatisfiable", if* CHOOSE(Λ) *is equal to* $\bot$*;*
- *or a single leaf labeled "saturated", if* CHOOSE(Λ) *is equal to* $\top$*;*
- *or nodes labeled by* $\Lambda \cup \{A_1\}, \ldots, \Lambda \cup \{A_k\}$*, if* CHOOSE($\Lambda$) *is* $A_1 \vee \cdots \vee A_k$*.*

The branches which are infinite or end with the *"saturated"* message are called *open*, whereas the branches ending with the *"unsatisfiable"* message are called *closed*. The procedures stops (and the generation of the above tree is interrupted) iff all branches are closed or if there is an open finite branch (of course termination is not guaranteed in the general case).[22]

Fair Selection Functions.

The function CHOOSE(Λ) is *fair* iff the following happens for every open branch $\Lambda_0 \subseteq \Lambda_1 \subseteq \cdots$: if $C \equiv Res_{\Phi_i}(\Gamma_i \cup \Lambda_k, l)$ for some $i = 1, 2$ and for some $k, l \geq 0$, then C is

[22] During our combination procedure Φ_0-residues are exchanged, till a saturation state is reached or till an inconsistency is detected. One may worry about the fact that information concerning $\mathcal{L}_0$-terms which are not Φ_0-terms is not exchanged (there might *in principle* be such terms, according to Definition 4.3.1). However, we just pointed out that, without additional conditions, the procedure is not complete and, when sufficient conditions for completeness are introduced (see Proposition 4.4.5), these will be strong enough to guarantee that the exchanged information is sufficient: more specifically, the set of Φ_0-terms will be big enough for Φ_0-*equivalence* of structures to be converted into $\mathcal{L}_0$-*isomorphism*, through Φ_i-equivalence preserving semantic operations.

Algorithm 5 The combination procedure

```
1: function FComb(Λ)
2:     C ← Choose(Λ)
3:     if C = ⊥ then
4:         return "unsatisfiable"
5:     else if C = ⊤ then
6:         return "saturated"
7:     end if
8:     for all A ∈ C do
9:         if FComb(Λ ∪ {A}) = "saturated" then
10:            return "saturated"
11:        end if
12:    end for
13:    return "unsatisfiable"
14: end function
```

Φ_0-redundant with respect to $\Gamma_0 \cup \Lambda_n$ for some n (roughly, *residues w.r.t. Φ_i of an open branch are redundant with respect to the atoms in the branch*).

We first show how to build fair selection functions (under the current assumptions/notational conventions):

Proposition 4.3.9. *There always exists a fair selection function.*

Proof. For a finite set Λ and for a list Θ of $\Phi_0(\underline{x}_0)$-clauses, let us define the auxiliary procedure $\mathrm{LMin}(\Lambda, \Theta)$. We have that

- $\mathrm{LMin}(\Lambda, [C]) = C$;
- $\mathrm{LMin}(\Lambda, [C|\Theta]) = \mathrm{LMin}(\Lambda, \Theta)$, if $Red_{\Phi_0}(\Lambda \cup \{D\}, C)$ holds for some $D \in \Theta$;
- $\mathrm{LMin}(\Lambda, [C|\Theta]) = C$, otherwise

(roughly, the procedure takes the leftmost Red_{Φ_0}-maximal element of the list Θ, using the set Λ as a parameter).

Fix now a surjective recursive function

$$\delta = (\delta_1, \delta_2, \delta_3) : \mathbb{N} \longrightarrow \{1, 2\} \times \mathbb{N} \times \mathbb{N}$$

such that $n \geq \delta_2(n)$ holds for every n (this function can be easily built by using a recursive encoding of pairs, see, e.g., [72]).

For $\Lambda = \{A_1, \ldots, A_n\}$, define now $\mathrm{Choose}(\Lambda)$ to be

$$\mathrm{LMin}(\Gamma_0 \cup \Lambda, [Res_{\Phi_{\delta_1(n)}}(\Gamma_{\delta_1(n)} \cup \{A_1, \ldots, A_{\delta_2(n)}\}, \delta_3(n)), Res_{\Phi_1}(\Gamma_1 \cup \Lambda, 0), Res_{\Phi_2}(\Gamma_2 \cup \Lambda, 0)]). \tag{4.3}$$

The intuitive explanation of this definition is as follows: for $i = 1, 2$ and $j \leq n$, residues $Res_{\Phi_i}(\Gamma_i \cup \{A_1, \ldots, A_j\}, k)$ can be disposed in two matrices having n infinite rows ($Res_{\Phi_i}(\Gamma_i \cup \{A_1, \ldots, A_j\}, k)$ is the k-th entry of the j-th row in the i-th matrix). Now our selection function explores the rows of these two matrices by a diagonal path, but before making the final choice it checks whether in the first entries of the two last rows there is anything more informative.

Clearly CHOOSE(Λ) is a Φ_i-consequence of $\Gamma_i \cup \Lambda$, for $i = 1$ or $i = 2$, because of the soundness condition of Definition 4.2.17; moreover if CHOOSE(Λ) is redundant w.r.t. $\Gamma_0 \cup \Lambda$, then (by Definition 4.2.16 (iii)) it must be equal to $Res_{\Phi_2}(\Gamma_2 \cup \Lambda, 0)$, hence it is $\top$ or $\bot$, according to Definition 4.2.18 (i) (and it is $\bot$, if $\bot$ is redundant w.r.t. $\Gamma_0 \cup \Lambda$, by Definition 4.2.18 (ii)).

To show fairness, pick a consistent branch labeled by the increasing sets of Φ_0-atoms $\Lambda_0 \subseteq \Lambda_1 \subseteq \cdots$ and suppose that $C \equiv Res_{\Phi_i}(\Gamma_i \cup \Lambda_k, l)$ for some $i = 1, 2$ and for some $k, l \geq 0$. Let us distinguish the case in which the consistent branch is finite and the case in which it is infinite.

If it is finite, it ends with a saturation message, which means that for some n, we have CHOOSE(Λ_n) $\equiv \top$. From (4.3) and Definition 4.2.16 (ii)-(iii)-(iv), we must have that $Res_{\Phi_1}(\Gamma_1 \cup \Lambda_n, 0)$ and $Res_{\Phi_2}(\Gamma_2 \cup \Lambda_n, 0)$ are both equal to $\top$. To show this, notice that: (a) a residue equal to $\top$ selected by the function CHOOSE(Λ_n) according to (4.3) cannot be either $Res_{\Phi_{\delta_1(n)}}(\Gamma_{\delta_1(n)} \cup \{A_1, \ldots, A_{\delta_2(n)}\}, \delta_3(n))$ or $Res_{\Phi_1}(\Gamma_1 \cup \Lambda_n, 0)$, because $\top$ is always redundant; (b) hence it must be $Res_{\Phi_2}(\Gamma_2 \cup \Lambda_n, 0)$, which implies however (by the way the procedure LMIN is defined) that $Res_{\Phi_1}(\Gamma_1 \cup \Lambda_n, 0)$ is redundant w.r.t. $\Gamma_0 \cup \Lambda_n \cup \{Res_{\Phi_2}(\Gamma_2 \cup \Lambda_n, 0)\}$ (which is equal to $\Gamma_0 \cup \Lambda_n \cup \{\top\}$) and hence w.r.t. $\Gamma_0 \cup \Lambda_n$ by transitivity. The latter implies that $Res_{\Phi_1}(\Gamma_1 \cup \Lambda_n, 0)$ is also equal to $\top$ by Definition 4.2.18 (i).[23]

Since $Res_{\Phi_1}(\Gamma_1 \cup \Lambda_n, 0)$ and $Res_{\Phi_2}(\Gamma_2 \cup \Lambda_n, 0)$ are both equal to $\top$, by Definition 4.2.18 (iii), we conclude that $\Gamma_0 \cup \Lambda_n$ is a Φ_0-basis for both $\Gamma_1 \cup \Lambda_n$ and $\Gamma_2 \cup \Lambda_n$, which means that the Φ_i-residue C is redundant with respect to $\Gamma_0 \cup \Lambda_n$: in fact, since $C \equiv Res_{\Phi_i}(\Gamma_i \cup \Lambda_k, l)$, by Definition 4.2.17, C is a Φ_i-consequence of $\Gamma_i \cup \Lambda_k$ (for $\Lambda_k \subseteq \Lambda_n$) and hence also of $\Gamma_i \cup \Lambda_n$, thus the definition of a Φ_0-basis applies.

If the branch is infinite, for some n, we have $\delta_1(n) = i, \delta_2(n) = k, \delta_3(n) = l$. Hence, either C has been selected, or some better choice (from the redundancy point of view) has been made according to (4.3). Since this better choice D cannot be $\top$ or $\bot$ because the branch is infinite, some atom of D (or of C, if C has been directly selected) is in Λ_{n+1}: this means that C is redundant with respect to $\Gamma_0 \cup \Lambda_{n+1}$ because of Definition 4.2.16(iii)-(iv)-(v). □

[23] It cannot be equal to $\bot$, because it is redundant w.r.t. $\Gamma_0 \cup \Lambda_n$: in that case, $Res_{\Phi_2}(\Gamma_2 \cup \Lambda_n, 0)$ would be $\bot$ too by Definition 4.2.18(ii).

We underline that the fair selection function given in (4.3) above can be optimized in specific situations, where extra information on the input residue enumerators is available; however, the existence of a uniform schema for defining a fair selection function is an interesting property of our combination procedure.

4.3.3 Soundness

One possible exit of our procedure is when it generates a finite tree whose leaves are all labeled *"unsatisfiable"*: this is precisely the case in which the whole procedure returns *"unsatisfiable"*.

Proposition 4.3.10 (Soundness)**.** *If the procedure* FComb *returns "unsatisfiable", then the purified constraint* $\Gamma_1 \cup \Gamma_2$ *is* $\Phi_1 \oplus \Phi_2$*-unsatisfiable.*

Proof. We consider the tree generated by the execution of the procedure described in section 4.3.2. The thesis consists of proving that, if such a tree is closed, then the purified constraint $\Gamma_1 \cup \Gamma_2$ is $\Phi_1 \oplus \Phi_2$-unsatisfiable. The proof applies an inductive argument on the tree.

Consider a node labeled with Λ which is the parent of a leaf labeled with *"unsatisfiable"*: by construction Choose(Λ) is equal to $\bot$. By Definition 4.3.8, there exists an i such that $\Gamma_i \cup \Lambda$ is Φ_i-unsatisfiable. Recalling the definition of combined fragment, it follows that $\Gamma_i \cup \Lambda$ is $\Phi_1 \oplus \Phi_2$-unsatisfiable, thus $\Gamma_1 \cup \Gamma_2 \cup \Lambda$ is unsatisfiable too.

Consider now a tree whose leaves are labeled with *"unsatisfiable"* and whose root is labeled by Λ. Suppose now, by inductive hypothesis, that each child of the root (labeled by $\Lambda \cup \{A_j\}$) is such that $\Gamma_1 \cup \Gamma_2 \cup \Lambda \cup \{A_j\}$ is $\Phi_1 \oplus \Phi_2$-unsatisfiable ($j \in \{1, \ldots, k\}$). $\Gamma_1 \cup \Gamma_2 \cup \Lambda \cup \{A_j\}$ is $\Phi_1 \oplus \Phi_2$-unsatisfiable for each j iff $\Gamma_1 \cup \Gamma_2 \cup \Lambda \cup \{A_1 \vee \cdots \vee A_k\}$ is $\Phi_1 \oplus \Phi_2$-unsatisfiable: this means that our inductive hypothesis entails the $\Phi_1 \oplus \Phi_2$-unsatisfiability of $\Gamma_1 \cup \Gamma_2 \cup \Lambda \cup \{A_1 \vee \cdots \vee A_k\}$. By construction, our internal nodes are labeled by $\Lambda \cup \{A_1\}, \ldots, \Lambda \cup \{A_k\}$ iff Choose(Λ) is $A_1 \vee \cdots \vee A_k$; hence, by Definition 4.3.8, there exists an $i \in \{1, 2\}$ such that $\Gamma_i \cup \Lambda \models_{\Phi_i} A_1 \vee \cdots \vee A_k$, thus being $A_1 \vee \cdots \vee A_k$ a $\Phi_1 \oplus \Phi_2$-consequence of $\Gamma_1 \cup \Gamma_2 \cup \Lambda$.

This means that for each $\Phi_1 \oplus \Phi_2$-structure in which $\Gamma_1 \cup \Gamma_2 \cup \Lambda$ is true w.r.t. an assignment, $A_1 \vee \cdots \vee A_k$ is true w.r.t. to the same assignment too. Moreover, the $\Phi_1 \oplus \Phi_2$-unsatisfiability of $\Gamma_1 \cup \Gamma_2 \cup \Lambda \cup \{A_1 \vee \cdots \vee A_k\}$ means that there does not exist a $\Phi_1 \oplus \Phi_2$-structure in which $\Gamma_1 \cup \Gamma_2 \cup \Lambda$ and $\{A_1 \vee \cdots \vee A_k\}$ are both true w.r.t. at least an assignment. It follows that $\Gamma_1 \cup \Gamma_2 \cup \Lambda$ is $\Phi_1 \oplus \Phi_2$-unsatisfiable itself.

The thesis follows by the consideration that, when we run the procedure for the pure constraint $\Gamma_1 \cup \Gamma_2$, the root of the tree is labeled by the empty set by construction. □

4.3.4 Termination

Next, we identify a relevant termination case:

Proposition 4.3.11 (Termination)**.** *If Φ_0 is noetherian and Red_{Φ_0} is the full redundancy notion, then the procedure* FComb *terminates on the purified constraint $\Gamma_1 \cup \Gamma_2$.*

Proof. Let us consider the tree T generated by the execution of the procedure FComb as described in Section 4.3.2. Recalling that T is finite iff FComb$(\emptyset)$ terminates, we now suppose that FComb$(\emptyset)$ does not terminate. In this way T, which is a finitely branching tree by construction, is not finite and it has an infinite branch by König lemma.

This means that there is a infinite chain of sets of $\Phi_0(\underline{x}_0)$-atoms $\Lambda_1 \subset \Lambda_2 \subset \cdots \subset \Lambda_n \subset \cdots$, where Λ_i is the label of a node that belongs to that infinite path, $\Lambda_{i+1} = \Lambda_i \cup \{A_i\}$ and $Red_{\Phi_0}(\Gamma_0 \cup \Lambda_i, A_i)$ does not hold by Definitions 4.2.16(v) and 4.3.8. Since Red_{Φ_0} is the full redundancy notion, we obtained an infinite sequence $A_1, A_2, A_3, \ldots$ such that $\Gamma_0 \cup \{A_j \mid j < i\} \not\models_{\Phi_0} A_i$ for every i. This contradicts our hypothesis on the noetherianity of Φ_0. □

4.3.5 Towards Completeness

Completeness of the procedure FComb cannot be achieved easily, heavy conditions are needed. In this section, we nevertheless identify what is the 'semantic meaning' of a run of the procedure that either does not terminate or terminates with a saturation message.

Since our investigations are taking a completeness-oriented route, it is quite obvious that we must consider from now on only the case in which the input Φ_i-p.r.e.'s are *complete* (see Definition 4.2.19). In addition we need a compactness-like assumption. We say that an i.a.f. Φ is Φ_0-*compact* (where Φ_0 is a subfragment of Φ) iff, given a Φ-constraint Γ and a generalized Φ_0-constraint Γ_0, we have that $\Gamma \cup \Gamma_0$ is Φ-satisfiable if and only if for all finite $\Delta_0 \subseteq \Gamma_0$, we have that $\Gamma \cup \Delta_0$ is Φ-satisfiable.

Proposition 4.3.12. *Any extension Φ of a locally finite fragment Φ_0 is Φ_0-compact.*

Proof. Recall that, according to the definition of constraints, a generalized Φ_0-constraint Γ_0 is an infinite set of Φ_0-literals in which only finitely many Φ_0-variables (call them $\underline{x}$) occur free. Since Φ_0 is locally finite, there exist finitely many $\Phi_0(\underline{x})$-terms representing all $\Phi_0(\underline{x})$-terms up to Φ_0-equivalence: for this reason, a generalized $\Phi_0(\underline{x})$-constraint Γ_0 is equivalent to the constraint in which all terms have been replaced by their representatives. □

The above Proposition means that, if we assume effective local finiteness in order to guarantee termination, Φ_0-compactness is guaranteed too. Notice that only special kinds of generalized Φ-constraints are involved in the definition of Φ_0-compactness, namely

those that contain finitely many proper Φ-literals; thus, Φ_0-compactness is a rather weak condition (that's why it may hold for any extension whatsoever of a given fragment, as shown by Proposition 4.3.12). Finally, it goes without saying that, by the compactness theorem for first-order logic, Φ_0-compactness is guaranteed whenever Φ is a first-order fragment.

Proposition 4.3.13. *Suppose that Φ_1, Φ_2 are both Φ_0-compact, that the function* CHOOSE*(Λ) is fair w.r.t. two complete Φ_i-p.r.e.'s and that the procedure* FCOMB *does not return "unsatisfiable" on the purified constraint $\Gamma_1 \cup \Gamma_2$. Then there are $\mathcal{L}_i$-structures $\mathcal{M}_i \in \mathcal{S}_i$ and $\mathcal{L}_i$-assignments α_i ($i = 1, 2$) such that:*

(i) *$\mathcal{M}_1 \models_{\alpha_1} \Gamma_1$ and $\mathcal{M}_2 \models_{\alpha_2} \Gamma_2$;*

(ii) *for every $\Phi_0(\underline{x}_0)$-atom A, we have that $\mathcal{M}_1 \models_{\alpha_1} A$ iff $\mathcal{M}_2 \models_{\alpha_2} A$.*

Proof. A set of positive $\Phi_0(\underline{x}_0)$-clauses $\Theta_0^\star$ is *saturated* if and only if it is closed under the two rules:

$$\Gamma_1 \cup \Theta_0^\star \models_{\Phi_1} C \quad \Rightarrow \quad C \in \Theta_0^\star$$
$$\Gamma_2 \cup \Theta_0^\star \models_{\Phi_2} C \quad \Rightarrow \quad C \in \Theta_0^\star,$$

for every positive $\Phi_0(\underline{x}_0)$-clause C.

Let us suppose that Res_{Φ_1} and Res_{Φ_2} are complete p.r.e.'s. Consider now the tree T generated by an execution as described in Section 4.3.2, in case FCOMB does not return *"unsatisfiable"* on the purified constraint $\Gamma_1 \cup \Gamma_2$. If T is finite, then it has a branch whose leaf is labeled by *"saturated"*, otherwise it has an infinite branch (we recall that, by construction, T is a finitely branching tree and, by König lemma, a finitely branching tree which is infinite has an infinite branch).

Let us consider that (finite or infinite) open branch labeled $\Lambda_0 \subseteq \Lambda_1 \subseteq \cdots$ and let us take $\Lambda := \bigcup_j \Lambda_j$; we define $\Theta_0^\star := \{C \mid C \text{ is a positive } \Phi_0(\underline{x}_0)\text{-clause s.t. } \Gamma_1 \cup \Lambda \models_{\Phi_1} C\}$ (we remark that $\Lambda \subseteq \Theta_0^\star$). $\Theta_0^\star$ is saturated and, for $i = 1, 2$, the generalized constraints $\Gamma_i \cup \Theta_0^\star$ are Φ_i-satisfiable, as shown by Lemma 4.3.14. Thus, Lemma 4.3.15 applies and there are two $\mathcal{L}_i$-structures $\mathcal{M}_i \in \mathcal{S}_i$ satisfying $\Gamma_i \cup \Theta_0^\star$ under assignments α_i, such that $\mathcal{M}_1, \alpha_1$ and $\mathcal{M}_2, \alpha_2$ satisfy the same $\Phi_0(\underline{x}_0)$-atoms. □

Lemma 4.3.14. *The set $\Theta_0^\star$ defined above is saturated, $\Gamma_1 \cup \Theta_0^\star$ is Φ_1-satisfiable and $\Gamma_2 \cup \Theta_0^\star$ is Φ_2-satisfiable.*

Proof. To prove that $\Theta_0^\star$ is saturated, we need to show that

$$\Gamma_2 \cup \Theta_0^\star \models_{\Phi_2} C \quad \Rightarrow \quad C \in \Theta_0^\star$$

where C is a positive $\Phi_0(\underline{x}_0)$-clause. We prove that $\Gamma_1 \cup \Lambda \models_{\Phi_1} C$ implies $\Gamma_2 \cup \Lambda \models_{\Phi_2} C$ (and conversely, but the proof of the converse is the same).

$\Gamma_1 \cup \Lambda \models_{\Phi_1} C$ iff there exists n such that $\Gamma_1 \cup \Lambda_n \models_{\Phi_1} C$ (by Φ_0-compactness of Φ_1) iff there exists k such that[24] $Red_{\Phi_0}(\Gamma_0 \cup \Lambda_n \cup \{C_0, \ldots, C_k\}, C)$ holds, where $C_j \equiv Res_{\Phi_1}(\Gamma_1 \cup \Lambda_n, j)$ (by the completeness of the Φ_1-p.r.e.). By the fairness requirement on the CHOOSE function, there exist m_j's such that $Red_{\Phi_0}(\Gamma_0 \cup \Lambda_{m_j}, C_j)$ holds ($j \in \{1, \ldots, k\}$), hence by monotonicity of redundancy there exists $m \geq n, m \geq m_j$ such that $Red_{\Phi_0}(\Gamma_0 \cup \Lambda_m, C_j)$ holds for each $j \in \{1, \ldots, k\}$; by transitivity of redundancy we have $Red_{\Phi_0}(\Gamma_0 \cup \Lambda_m, C)$ and consequently also $\Gamma_0 \cup \Lambda_m \models_{\Phi_0} C$. Thus $\Gamma_2 \cup \Lambda_m \models_{\Phi_2} C$ and finally $\Gamma_2 \cup \Lambda \models_{\Phi_2} C$.

We showed that $\Gamma_1 \cup \Lambda \models_{\Phi_1} C$ holds iff $\Gamma_2 \cup \Lambda \models_{\Phi_2} C$ holds; it follows that:

$$\begin{aligned}\Theta_0^\star &= \{C \text{ is a positive } \Phi_0(\underline{x}_0)\text{-clause} \mid \Gamma_1 \cup \Lambda \models_{\Phi_1} C\} = \\ &= \{C \text{ is a positive } \Phi_0(\underline{x}_0)\text{-clause} \mid \Gamma_2 \cup \Lambda \models_{\Phi_2} C\},\end{aligned}$$

that is $\Gamma_2 \cup \Theta_0^\star \models_{\Phi_2} C \Rightarrow C \in \Theta_0^\star$.

We finally prove that $\Gamma_i \cup \Theta_0^\star$ is Φ_i-satisfiable for $i = 1, 2$. To this aim notice that $\Gamma_i \cup \Lambda$ is Φ_i-satisfiable iff $\Gamma_i \cup \Delta$ is Φ_i-satisfiable for each $\Delta \subseteq \Lambda$, Δ finite (by Φ_0-compactness of Φ_i). For each such Δ, there exists an index n such that $\Delta \subseteq \Lambda_n$. $\Gamma_i \cup \Lambda_n$ is Φ_i-satisfiable for each n (we have $\Gamma_i \cup \Lambda_n \not\models_{\Phi_i} \bot$ by completeness of Res_{Φ_i}, by definition 4.2.18 (ii) and by the fairness requirement of the selection function CHOOSE),[25] thus every $\Gamma_i \cup \Delta$ is Φ_i-satisfiable and consequently so is $\Gamma_i \cup \Lambda$. Since we noticed above that $\Theta_0^\star$ consists of the clauses C such that $\Gamma_i \cup \Lambda \models_{\Phi_1} C$, it follows that $\Gamma_i \cup \Theta_0^\star$ is Φ_i-satisfiable. □

We now state the following lemma whose proof is exactly the translation into this high-order framework of Lemma 3.3.8.

Lemma 4.3.15. *Suppose that we are given a saturated set of positive $\Phi_0(\underline{x}_0)$-clauses $\Theta_0^\star$, such that $\Gamma_1 \cup \Theta_0^\star$ is Φ_1-satisfiable and $\Gamma_2 \cup \Theta_0^\star$ is Φ_2-satisfiable. Then there are structures $\mathcal{M}_1 \in \mathcal{S}_1$, $\mathcal{M}_2 \in \mathcal{S}_2$ and two assignments α_1, α_2 such that $\mathcal{M}_1 \models_{\alpha_1} \Gamma_1 \cup \Theta_0^\star$ and $\mathcal{M}_2 \models_{\alpha_2} \Gamma_2 \cup \Theta_0^\star$. Moreover, for every $\Phi_0(\underline{x}_0)$-atom A, $\mathcal{M}_1 \models_{\alpha_1} A$ holds if and only if $\mathcal{M}_2 \models_{\alpha_2} A$ holds.*

Proof. A set Δ of $\Phi_0(\underline{x}_0)$-literals is *exhaustive* iff for each $\Phi_0(\underline{x}_0)$-atom A, $A \in \Delta$ or $\neg A \in \Delta$.

[24] Recall that $\Gamma_0 = \Gamma_{1|\Phi_0} = \Gamma_{2|\Phi_0}$ according to the 'Notational Conventions' at the beginning of Subsection 4.3.2.

[25] In more detail, if $\Gamma_i \cup \Lambda_n \models_{\Phi_i} \bot$ for some n, then $\bot$ appears as a residue of $\Gamma_i \cup \Lambda_n$ (by completeness of Res_{Φ_i} and by Definition 4.2.18 (ii)). By the fairness of the selection function it is then redundant with respect to some $\Gamma_0 \cup \Lambda_m$, which implies that CHOOSE(Λ_m) is $\bot$, by Definition 4.3.8, contrary to fact that the branch is not closed.

Let us consider any terminating strict total order on $\Phi_0(\underline{x}_0)$-atoms (it exists by the well ordering principle) and let us extend it to a terminating strict total order on multisets.[26] We use such an ordering to define increasing subsets Δ_C, varying C among positive $\Phi_0(\underline{x}_0)$-clauses in $\Theta_0^\star$ (positive clauses are identified here with multisets of atoms).

We say that the $\Phi_0(\underline{x}_0)$-clause $C \equiv A \vee A_1 \vee \cdots \vee A_n$ from $\Theta_0^\star$ is *productive* (and produces the $\Phi_0(\underline{x}_0)$-atom A) iff $\{A\} > \{A_1, \ldots, A_n\}$ and $A_1, \ldots, A_n \notin \Delta^+_{<C}$ where $\Delta^+_{<C} := \bigcup_{D<C, D\in\Theta_0^\star} \Delta^+_D$. If C is productive and produces A, then $\Delta^+_C := \Delta^+_{<C} \cup \{A\}$, otherwise $\Delta^+_C := \Delta^+_{<C}$.

Let us define $\Delta^+ := \bigcup_{C\in\Theta_0^\star} \Delta^+_C$ and $\Delta := \Delta^+ \cup \{\neg A \mid A \text{ is a } \Phi_0(\underline{x}_0)\text{-atom and } A \notin \Delta^+\}$. By construction, $\Delta \models_{\Phi_0} \Theta_0^\star$ (because Δ contains a $\Phi_0(\underline{x}_0)$-atom for every $\Phi_0(\underline{x}_0)$-clause in $\Theta_0^\star$).

We need to show that $\Gamma_1 \cup \Delta$ is Φ_1-satisfiable and $\Gamma_2 \cup \Delta$ is Φ_2-satisfiable. First of all, we *claim* that if a $\Phi_0(\underline{x}_0)$-clause $C \equiv A \vee A_1 \vee \cdots \vee A_n$ is productive and $\{A\} > \{A_1, \ldots, A_n\}$, then $A_1, \ldots, A_n \notin \Delta^+$. To show this, recall that, by definition, $A_i \in \Delta^+$ ($i \in \{1, \ldots, n\}$) iff A_i belongs to a productive $\Phi_0(\underline{x}_0)$-clause C_i and A_i is the maximum atom in it, thus $C_i < C$ (by multisets ordering): however none of the A_i can be in $\Delta^+_{<C}$, because C is productive, thus justifying our claim.

We suppose now that $\Gamma_1 \cup \Delta$ is Φ_1-unsatisfiable. By Φ_0-compactness of the i.a.f. Φ_1, there are $\Phi_0(\underline{x}_0)$-atom $B_1, \ldots, B_m \notin \Delta^+$ and productive $\Phi_0(\underline{x}_0)$-clauses

$$\begin{aligned} C_1 &\equiv A_1 \vee A_{11} \vee \cdots \vee A_{1k_1} \\ &\cdots \\ C_n &\equiv A_n \vee A_{n1} \vee \cdots \vee A_{nk_n} \end{aligned}$$

(with maximum $\Phi_0(\underline{x}_0)$-atoms $A_1, \ldots, A_n$ respectively) s.t. $\Gamma_1 \cup \{A_1, \ldots, A_n, \neg B_1, \ldots, \neg B_m\}$ is Φ_1-unsatisfiable. It follows that

$$\Gamma_1 \cup \{C_1, \ldots, C_n\} \cup \{\neg A_{11}, \cdots, \neg A_{nk_n}, \neg B_1, \ldots, \neg B_m\}$$

is also unsatisfiable. As $C_1, \ldots, C_n$ are positive $\Phi_0(\underline{x}_0)$-clauses in $\Theta_0^\star$ and $\Theta_0^\star$ is saturated, the positive $\Phi_0(\underline{x}_0)$-clause

$$D \equiv \bigvee_{i,j} A_{ij} \vee B_1 \vee \cdots \vee B_m$$

is also in $\Theta_0^\star$. By construction, some of its $\Phi_0(\underline{x}_0)$-atoms belongs to Δ^+. $A_{11}, \ldots, A_{nk_n}$ cannot be there because the $C_1, \ldots, C_n$ are productive (see the above claim), thus at least one of the B_j's is in Δ^+: contradiction.[27] The case of Γ_2 is analogous.

[26] We are using basic information on multiset orderings that can be found in textbooks like [11].

[27] Notice that we cannot have $k_1 = \cdots = k_n = m = 0$, because $\Gamma_1 \cup \{C_1, \ldots, C_n\} \subseteq \Gamma_1 \cup \Theta_0^\star$ and the latter is consistent by hypothesis.

We finally show that, given two structures $\mathcal{M}_1 \in \mathcal{S}_1$, $\mathcal{M}_2 \in \mathcal{S}_2$ and two assignments α_1, α_2 such that $\mathcal{M}_1 \models_{\alpha_1} \Gamma_1 \cup \Delta$ and $\mathcal{M}_2 \models_{\alpha_2} \Gamma_2 \cup \Delta$, we have that $\mathcal{M}_1, \alpha_1$ and $\mathcal{M}_2, \alpha_2$ satisfy the same $\Phi_0(\underline{x}_0)$-atoms. This is clear, because Δ an exhaustive set of $\Phi_0(\underline{x}_0)$-literals. □

4.4 Isomorphism Theorems and Completeness

Proposition 4.3.13 explains what is the main problem for completeness: we would like an open branch to produce Φ_i-structures ($i = 1, 2$) whose $\mathcal{L}_0$-reducts are isomorphic and we are only given Φ_i-structures whose $\mathcal{L}_0$-reducts are $\Phi_0(\underline{x}_0)$-equivalent (in the sense that they satisfy the same $\Phi_0(\underline{x}_0)$-atoms). Hence we need a powerful semantic device that is able to *transform* $\Phi_0(\underline{x}_0)$*-equivalence into* $\mathcal{L}_0$*-isomorphism*: this device will be called isomorphism theorem. The precise formulation of what we mean by isomorphism theorem needs some preparation. First of all, we introduce fragments extended with free constants; in fact, whereas the use of variables in constraints was precious so far as it emphasized the role of substitutions (and substitutions act on variables, not on free constants), now it is better to have at hand also the formalism of free constants, otherwise standard model-theoretic results would get an unnatural formulation.

Given an i.a.f. $\Phi = \langle \mathcal{L}, T, \mathcal{S} \rangle$, we denote by $\Phi(\underline{c}) = \langle \mathcal{L}(\underline{c}), T(\underline{c}), \mathcal{S}(\underline{c}) \rangle$ the following i.a.f.: (i) $\mathcal{L}(\underline{c}) := \mathcal{L} \cup \{\underline{c}\}$ is obtained by adding to $\mathcal{L}$ finitely many new constants $\underline{c}$ (the types of these new constants must be types of Φ); (ii) $T(\underline{c})$ contains the terms of the kind $t[\underline{c}/\underline{x}, \underline{y}]$ for $t[\underline{x}, \underline{y}] \in T$; (iii) $\mathcal{S}(\underline{c})$ contains precisely the $\mathcal{L}(\underline{c})$-structures whose $\mathcal{L}$-reduct is in $\mathcal{S}$. Fragments of the kind $\Phi(\underline{c})$ are called *finite expansions* of Φ.

Let $\Phi(\underline{c})$ be a finite expansion of $\Phi = \langle \mathcal{L}, T, \mathcal{S} \rangle$ and let $\mathcal{A}, \mathcal{B}$ be $\mathcal{L}(\underline{c})$-structures. We say that $\mathcal{A}$ is $\Phi(\underline{c})$*-equivalent* to $\mathcal{B}$ (written $\mathcal{A} \equiv_{\Phi(\underline{c})} \mathcal{B}$) iff for every closed $\Phi(\underline{c})$-atom A we have that $\mathcal{A} \models A$ iff $\mathcal{B} \models A$. By contrast, we say that $\mathcal{A}$ is $\Phi(\underline{c})$*-isomorphic* to $\mathcal{B}$ (written $\mathcal{A} \simeq_{\Phi(\underline{c})} \mathcal{B}$) iff there is an $\mathcal{L}(\underline{c})$-isomorphism from $\mathcal{A}$ onto $\mathcal{B}$.

We can now specify what we mean by a structural operation on an i.a.f. $\Phi_0 = \langle \mathcal{L}_0, T_0, \mathcal{S}_0 \rangle$. We will be very liberal here and define *structural operation on* Φ_0 any family of correspondences $O = \{O^{\underline{c}_0}\}$ associating with any finite set of free constants $\underline{c}_0$ and with any $\mathcal{A} \in \mathcal{S}_0(\underline{c}_0)$ some $O^{\underline{c}_0}(\mathcal{A}) \in \mathcal{S}_0(\underline{c}_0)$ such that $\mathcal{A} \equiv_{\Phi_0(\underline{c}_0)} O^{\underline{c}_0}(\mathcal{A})$. If no confusion arises, we omit the indication of $\underline{c}_0$ in the notation $O^{\underline{c}_0}(\mathcal{A})$ and write it simply as $O(\mathcal{A})$.[28]

A collection $\mathcal{O}$ of structural operations on Φ_0 *admits a* Φ_0*-isomorphism theorem* if and only if, for every $\underline{c}_0$, for every $\mathcal{A}, \mathcal{B} \in \mathcal{S}_0(\underline{c}_0)$, if $\mathcal{A} \equiv_{\Phi_0(\underline{c}_0)} \mathcal{B}$ then there exist $O_1, O_2 \in \mathcal{O}$ such that $O_1(\mathcal{A}) \simeq_{\Phi_0(\underline{c}_0)} O_2(\mathcal{B})$.

[28] The notion of a structural operation we propose here is sufficient to state and prove our results. However all the structural operations we shall use in the chapter enjoy additional properties: for instance, since they are induced by suitable endofunctors on the category of sets, they are functorial. For similar reasons, properties like $(O^{\underline{c}_0}(\mathcal{A}))_{|\mathcal{L}_0(\underline{c}'_0)} = O^{\underline{c}'_0}(\mathcal{A}_{|\mathcal{L}_0(\underline{c}'_0)})$ (for $\underline{c}'_0 \subseteq \underline{c}_0$) are always true in our examples.

Example 4.4.1 (Ultrapowers)**.** Ultrapowers (see [25]) are basic constructions in the model theory of first-order logic. An ultrapower $\prod_{\mathcal{U}}$ (technically, an ultrafilter $\mathcal{U}$ over a set of indexes is needed to describe the operation) transforms a first-order structure $\mathcal{A}$ into a first-order structure $\prod_{\mathcal{U}} \mathcal{A}$ which is elementarily equivalent to it (meaning that $\mathcal{A}$ and $\prod_{\mathcal{U}} \mathcal{A}$ satisfy the same first-order sentences). Hence if we take a fragment $\Phi_0 = \langle \mathcal{L}_0, T_0, \mathcal{S}_0 \rangle$, where $\mathcal{S}_0$ is an elementary class and $\langle \mathcal{L}_0, T_0 \rangle$ is an algebraic fragment of the kind analyzed in Example 4.2.8, then $\prod_{\mathcal{U}}$ is a structural operation on Φ_0. The following deep result in classical model theory (known as the *Keisler-Shelah isomorphism theorem* - see [25]) gives here a Φ_0-isomorphism theorem in our sense.[29]

Theorem 4.4.2 (Keisler-Shelah Isomorphism Theorem)**.** *Let $\mathcal{L}$ be a first-order signature and let $\mathcal{A}, \mathcal{B}$ be $\mathcal{L}$-structures. Then $\mathcal{A}$ is elementarily equivalent to $\mathcal{B}$ iff there is an ultrafilter $\mathcal{U}$ such that the ultrapowers $\prod_{\mathcal{U}} \mathcal{A}$ and $\prod_{\mathcal{U}} \mathcal{B}$ are $\mathcal{L}$-isomorphic.*

We shall mainly be interested into operations that can be extended to a preassigned expanded fragment. Here is the related definition. Let an i.a.f. $\Phi = \langle \mathcal{L}, T, \mathcal{S} \rangle$ extending $\Phi_0 = \langle \mathcal{L}_0, T_0, \mathcal{S}_0 \rangle$ be given; a structural operation O on Φ_0 is Φ-*extensible* if and only if for every $\underline{c}$ and every $\mathcal{A} \in \mathcal{S}(\underline{c})$ there exist $\mathcal{B} \in \mathcal{S}(\underline{c})$ such that

$$\mathcal{B}_{|\mathcal{L}_0(\underline{c}_0)} \simeq_{\Phi_0(\underline{c}_0)} O(\mathcal{A}_{|\mathcal{L}_0(\underline{c}_0)}) \quad \text{and} \quad \mathcal{B} \equiv_{\Phi(\underline{c})} \mathcal{A},$$

(where $\underline{c}_0$ denotes the set of those constants in $\underline{c}$ whose type is a Φ_0-type).

Example 4.4.3. Taking the reduct of a first-order structure to a smaller signature commutes with ultrapowers, hence if $\Phi = \langle \mathcal{L}, T, \mathcal{S} \rangle$ is an extension of $\Phi_0 = \langle \mathcal{L}_0, T_0, \mathcal{S}_0 \rangle$, both $\langle \mathcal{L}, T \rangle, \langle \mathcal{L}_0, T_0 \rangle$ are fragments of the kind analyzed in Example 4.2.8 and $\mathcal{S}, \mathcal{S}_0$ are elementary classes, then the Φ_0-structural operation $\prod_{\mathcal{U}}$ is Φ-extensible (the structure $\mathcal{B}$ required in the definition of Φ-extensibility is again $\prod_{\mathcal{U}} \mathcal{A}$, where the ultrapower is now taken at the level of $\mathcal{L}$-structures).

Sometimes an isomorphism theorem does not hold precisely for a fragment $\Phi_0 = \langle \mathcal{L}_0, T_0, \mathcal{S}_0 \rangle$, but for an inessential variation (called specialization) of it. A *specialization* of Φ_0 is an i.a.f. $\Phi_0^\star$ which has the same language and the same terms as Φ_0, but whose class of $\mathcal{L}_0$-structures is a smaller class $\mathcal{S}_0^\star \subseteq \mathcal{S}_0$ satisfying the following condition: for every $\underline{c}_0$ and for every $\mathcal{A} \in \mathcal{S}_0(\underline{c}_0)$, there exists $\mathcal{A}^\star \in \mathcal{S}_0^\star(\underline{c}_0)$ such that $\mathcal{A} \equiv_{\Phi_0(\underline{c}_0)} \mathcal{A}^\star$. Thus, the condition simply means that Φ_0 and its specialization $\Phi_0^\star$ satisfy the same generalized constraints.

[29] If $\Phi_0 = \langle \mathcal{L}_0, T_0, \mathcal{S}_0 \rangle$ is from Example 4.2.6 or 4.2.7 and quantifier elimination holds in $\mathcal{S}_0$, then the $\prod_{\mathcal{U}}$'s are also structural operations on Φ_0 admitting a Φ_0-isomorphism theorem (this observation will be implicitly used in the proof of Theorem 4.4.7 below.)

Given an i.a.f. $\Phi = \langle \mathcal{L}, T, \mathcal{S} \rangle$ extending Φ_0, we say that Φ is *compatible* with respect to a given specialization $\Phi_0^\star = \langle \mathcal{L}_0, T_0, \mathcal{S}_0^\star \rangle$ of Φ_0 iff $\Phi^\star = \langle \mathcal{L}, T, \mathcal{S}^\star \rangle$ is a specialization of Φ, where $\mathcal{S}^\star$ contains exactly those $\mathcal{L}$-structures from $\mathcal{S}$ whose $\mathcal{L}_0$-reduct belongs to $\mathcal{S}_0^\star$. This Φ_0-compatibility notion is intended to recapture, in our general setting, T_0-compatibility as introduced in [44]. The latter generalizes, in its turn, the standard stable infiniteness requirement of Nelson-Oppen procedure:

Example 4.4.4 (Stably infinite first-order theories)**.** Let $\Phi = \langle \mathcal{L}, T, \mathcal{S} \rangle$ be an i.a.f. of the kind considered in Example 4.2.6 or in Example 4.2.7: we say that Φ is *stably infinite* iff every satisfiable Φ-constraint is satisfiable in some infinite $\mathcal{L}$-structure $\mathcal{A} \in \mathcal{S}$. To see that this is a Φ_0-compatibility requirement, consider the i.a.f. $\Phi_0 = \langle \mathcal{L}_0, T_0, \mathcal{S}_0 \rangle$ so specified: (i) $\mathcal{L}_0$ is the empty one-sorted signature; (ii) T_0 contains only the individual variables; (iii) $\mathcal{S}_0$ is the totality of $\mathcal{L}_0$-structures (i.e. the totality of sets). A specialization $\Phi_0^\star$ of Φ_0 is obtained by considering the class $\mathcal{S}_0^\star$ formed by the infinite sets. By an easy compactness argument (compactness theorem holds here because Φ is a first-order fragment), it is easily seen that Φ is stably infinite iff it is compatible with respect to the specialization $\Phi_0^\star$ of Φ_0.

During the chapter, we shall see other examples of extensible structural operations and of isomorphisms theorems (we shall divide indeed our applications into three groups, depending on the particular isomorphism theorem which is involved).[30]

4.4.1 The Main Combination Result

We are now ready to formulate a sufficient condition for our combined procedure to be complete:

Proposition 4.4.5. *Suppose that Φ_1, Φ_2 are both Φ_0-compact and Φ_0-compatible with respect to a specialization $\Phi_0^\star$ of Φ_0; suppose also that there is a collection $\mathcal{O}$ of structural operations on $\Phi_0^\star$ which are all $\Phi_1^\star$ and $\Phi_2^\star$-extensible and admit a $\Phi_0^\star$-isomorphism theorem. In this case, if the function* Choose*(Λ) is fair w.r.t. two complete Φ_i-p.r.e.'s and the procedure* FComb *does not return "unsatisfiable" on the purified constraint $\Gamma_1 \cup \Gamma_2$, then such a constraint is $\Phi_1 \oplus \Phi_2$-satisfiable.*

Proof. By Proposition 4.3.13, there are two structures $\mathcal{M}_1$, $\mathcal{M}_2$ and two assignments α_1, α_2 such that: (i) $\mathcal{M}_1 \in \mathcal{S}_1$, $\mathcal{M}_2 \in \mathcal{S}_2$; (ii) $\mathcal{M}_1 \models_{\alpha_1} \Gamma_1$ and $\mathcal{M}_2 \models_{\alpha_2} \Gamma_2$; (iii) $\mathcal{M}_1, \alpha_1$ and $\mathcal{M}_2, \alpha_2$ satisfy the same $\Phi_0(\underline{x}_0)$-atoms. If we put variables into bijective correspondence with free constants, we may identify the pairs $(\mathcal{M}_i, \alpha_i)$ with structures in $\mathcal{S}_i(\underline{c}_i)$, for finite sets of free constants $\underline{c}_i$. Thus we can say that there are structures $\mathcal{N}_1 \in \mathcal{S}(\underline{c}_1), \mathcal{N}_2 \in$

[30] Still, there might be further examples which are not considered in this thesis and which deserve further investigation: model theory of modal logic (see [48]) seems to be interesting in this sense.

$\mathcal{S}(\underline{c}_2)$ satisfying $\Gamma_1[\underline{c}_1], \Gamma_2[\underline{c}_2]$, respectively, such that $\mathcal{N}_{1|\mathcal{L}_0(\underline{c}_0)} \equiv_{\Phi_0(\underline{c}_0)} \mathcal{N}_{2|\mathcal{L}_0(\underline{c}_0)}$ (where $\underline{c}_0 = \underline{c}_1 \cap \underline{c}_2$ are the free constants whose types are Φ_0-types).[31]

Now we will show that there is a $\mathcal{L}_1(\underline{c}_1) \cup \mathcal{L}_2(\underline{c}_2)$-structure $\mathcal{M}$ such that $\mathcal{M}_{|\mathcal{L}_i} \in \mathcal{S}_i$ and $\mathcal{M} \models \Gamma_i[\underline{c}_i]$ $(i = 1, 2)$. By Φ_0-compatibility with respect to $\Phi_0^\star$, we may assume that $\mathcal{N}_{1|\mathcal{L}_0(\underline{c}_0)}$ and $\mathcal{N}_{2|\mathcal{L}_0(\underline{c}_0)}$ are in a class $\mathcal{S}_0^\star$, over which the collection of structural operations $\mathcal{O}$ admits an isomorphism theorem.

Thus there are two structural operations $O_1, O_2 \in \mathcal{O}$ such that $O_1(\mathcal{N}_{1|\mathcal{L}_0(\underline{c}_0)}) \simeq_{\Phi_0(\underline{c}_0)} O_2(\mathcal{N}_{2|\mathcal{L}_0(\underline{c}_0)})$. Since O_1, O_2 are $\Phi_1^\star$- and $\Phi_2^\star$-extensible, there exist two structures $\mathcal{B}_1 \in \mathcal{S}_1^\star(\underline{c}_1)$ and $\mathcal{B}_2 \in \mathcal{S}_2^\star(\underline{c}_2)$ such that $\mathcal{B}_{1|\mathcal{L}_0(\underline{c}_0)} \simeq_{\Phi_0(\underline{c}_0)} O_1(\mathcal{N}_{1|\mathcal{L}_0(\underline{c}_0)})$ and $\mathcal{B}_{2|\mathcal{L}_0(\underline{c}_0)} \simeq_{\Phi_0(\underline{c}_0)} O_2(\mathcal{N}_{2|\mathcal{L}_0(\underline{c}_0)})$, $\mathcal{B}_1 \equiv_{\Phi_1(\underline{c}_1)} \mathcal{N}_1$ and $\mathcal{B}_2 \equiv_{\Phi_2(\underline{c}_2)} \mathcal{N}_2$. Thus, $\mathcal{B}_1$ satisfies $\Gamma_1[\underline{c}_1]$ and $\mathcal{B}_2$ satisfies $\Gamma_2[\underline{c}_2]$. Moreover, $\mathcal{B}_{1|\mathcal{L}_0(\underline{c}_0)} \simeq_{\Phi_0(\underline{c}_0)} \mathcal{B}_{2|\mathcal{L}_0(\underline{c}_0)}$; we can now easily build the desired $\mathcal{M}$ in two steps.

In the first step, we define $\mathcal{B}_2'$ such that $\mathcal{B}_{1|\mathcal{L}_0(\underline{c}_0)} = \mathcal{B}_{2|\mathcal{L}_0(\underline{c}_0)}'$ and $\mathcal{B}_2 \simeq_{\Phi_2(\underline{c}_2)} \mathcal{B}_2'$ (notice that $\mathcal{B}_2' \in \mathcal{S}_2(\underline{c}_2)$ by the closure under isomorphisms of $\mathcal{S}_2$, see Definition 4.2.3). Let ι be the isomorphism $\mathcal{B}_{1|\mathcal{L}_0(\underline{c}_0)} \longrightarrow \mathcal{B}_{2|\mathcal{L}_0(\underline{c}_0)}$; to define $\mathcal{B}_2'$, we interpret $\mathcal{L}_0$-sorts as in $\mathcal{B}_1$ and $\mathcal{L}_2 \setminus \mathcal{L}_0$-sorts as in $\mathcal{B}_2$. Put now $\iota'_S := \iota_S$ for $S \in \mathcal{L}_0$ and let ι'_S be the identity for $\mathcal{L}_2 \setminus \mathcal{L}_0$-sorts; by taking standard inductive extension to all $\mathcal{L}_2$-types, we get a family of bijections $\iota' = \{\iota'_\tau : [\![\, \tau \,]\!]_{\mathcal{B}_2'} \longrightarrow [\![\, \tau \,]\!]_{\mathcal{B}_2}\}$ (indexed by the $\mathcal{L}_2$-types) that can be used in order to complete the definition of $\mathcal{B}_2'$ (in the sense that we define the $\mathcal{B}_2'$-interpretation of every constant $d : \tau$ of $\mathcal{L}_2(\underline{c}_2)$ as $(\iota'_\tau)^{-1}(\mathcal{I}_{\mathcal{B}_2}(d))$). It is easily seen that the $\mathcal{L}_2(\underline{c}_2)$-structure $\mathcal{B}_2'$ matches the desired requirements.

Since the $\mathcal{L}_0(\underline{c}_0)$-reducts of $\mathcal{B}_1$ and $\mathcal{B}_2'$ are now just the same structure, it is easy to define (through a trivial join of both sorts and constants interpretations) a $\mathcal{L}_1(\underline{c}_1) \cup \mathcal{L}_2(\underline{c}_2)$-structure $\mathcal{M}$ such that $\mathcal{M}_{|\mathcal{L}_1(\underline{c}_1)} = \mathcal{B}_1$ and $\mathcal{M}_{|\mathcal{L}_2(\underline{c}_2)} = \mathcal{B}_2'$. Thus, the $\mathcal{L}_1 \cup \mathcal{L}_2$-reduct of $\mathcal{M}$ belongs to $\mathcal{S}_1 \oplus \mathcal{S}_2$ and satisfies $\Gamma_1[\underline{x}_1] \cup \Gamma_2[\underline{x}_2]$. □

The facts we established so far can be collected into our main decidability transfer theorem.

Theorem 4.4.6. *Suppose that:*

(1) *the interpreted algebraic fragments Φ_1, Φ_2 have decidable constraint satisfiability problems;*

(2) *the shared fragment Φ_0 is effectively locally finite (or more generally, Φ_1, Φ_2 are both Φ_0-compact, Φ_0 is noetherian and there exist noetherian positive residue Φ_1- and Φ_2-enumerators for Φ_0);*

[31] See the parallel convention for $\underline{x}_0$ at the beginning of Subsection 4.3.2. Notice that because of the Definition 4.3.1 concerning the shared fragment Φ_0, we have that both $\mathcal{N}_{1|\mathcal{L}_0(\underline{c}_0)}$ and $\mathcal{N}_{2|\mathcal{L}_0(\underline{c}_0)}$ belong to $\mathcal{S}_0(\underline{c}_0)$.

(3) Φ_1 *and* Φ_2 *are both* Φ_0*-compatible with respect to a specialization* $\Phi_0^\star$ *of* Φ_0*;*

(4) *there is a collection* $\mathcal{O}$ *of structural operations on* $\Phi_0^\star$ *which are all* $\Phi_1^\star$*- and* $\Phi_2^\star$*-extensible and admit a* $\Phi_0^\star$*-isomorphism theorem.*

Then the procedure FComb *(together with the preprocessing Purification Rule) decides constraint satisfiability in the combined fragment* $\Phi_1 \oplus \Phi_2$*.*

Proof. From Propositions 4.3.6, 4.3.7, 4.3.9, 4.3.10, 4.2.22, 4.3.11, 4.3.12, 4.2.21, and 4.4.5. □

Remark. In case the shared fragment Φ_0 is locally finite, a combination procedure can be obtained also simply by guessing a maximal set Θ_0 of $\Phi_0(\underline{x}_0)$-literals and by testing the Φ_i-satisfiability of $\Theta_0 \cup \Gamma_i$. This non-deterministic version of the procedure does not require the machinery developed in Section 4.2.3 (but it does not apply to noetherian cases and does not yield automatic optimizations in Φ_0-convexity cases).

Remark. Theorem 4.4.6 cannot be used to transfer decidability of word problems to our combined fragments: the reason is that, in case the procedure FComb is initialized with only a single negative literal, constraints containing positive literals are nevertheless generated during the execution (and also by the Purification Rule). However, since negative literals are never run-time generated, Theorem 4.4.6 can be used to transfer decidability of *conditional word problems*, namely of satisfiability problems for constraints containing just one negative literal.[32]

In the next three subsections we shall investigate families of concrete applications of Theorem 4.4.6, based on suitable isomorphism theorems.

4.4.2 Applications: Decidability Transfer through Ultrapowers

We shall use the isomorphism Theorem 4.4.2 to get the transfer decidability results of [44] as a special case of Theorem 4.4.6. For simplicity, we show how to do it for one-sorted functional first-order signatures (and leave to the reader the easy extension to first-order signatures containing also relational symbols like in Example 4.2.7).

Let $\Phi_1 = \langle \mathcal{L}_1, T_1, \mathcal{S}_1 \rangle$ and $\Phi_2 = \langle \mathcal{L}_2, T_2, \mathcal{S}_2 \rangle$ be equational first-order i.a.f.'s (i.e. i.a.f.'s of the kind considered in the Example 4.2.6) and let $\Phi_0 = \langle \mathcal{L}_0, T_0, \mathcal{S}_0 \rangle$ be their shared fragment. The hypotheses for the decidability transfer result of [44] are (equivalent to) the following:

[32] For instance, usual relativized satisfiability problems in modal logic are the same as conditional word problems in the fragments $\Phi_M = \langle \mathcal{L}_M, T_M, \mathcal{S}_M \rangle$ from Example 4.2.10; here constraint satisfiability problems can be reduced to conditional word problems in case of closure under disjoint unions of the Kripke frames in Φ_M (this is a special case of the straightforward general fact that, *in convex fragments, conditional word problems and constraint satisfiability problems are inter-reducible*).

(C1) there is a universal theory T_0 in the shared signature $\mathcal{L}_0$ such that every $\mathcal{A} \in \mathcal{S}_0$ is a model of T_0;

(C2) T_0 admits a model-completion $T_0^\star$;

(C3) for $i = 1, 2$, for every tuple $\underline{c}$ of free constants and for every $\mathcal{A} \in \mathcal{S}_i(\underline{c})$, there is some $\mathcal{A}' \in \mathcal{S}_i(\underline{c})$, s.t. $\mathcal{A}(\underline{c}) \equiv_{\Phi_i(\underline{c})} \mathcal{A}'(\underline{c})$ and $\mathcal{A}'_{|\mathcal{L}_0}$ is a model of $T_0^\star$;[33]

(C4) Φ_0 is effectively locally finite.

Condition (C2) means that: (a) $T_0 \subseteq T_0^\star$; (b) every model of T_0 embeds into a model of $T_0^\star$; (c) the following holds for every finite set of free $\mathcal{L}_0$-constants $\underline{c}_0$ and for every pair of $\mathcal{L}_0(\underline{c}_0)$-structures $\mathcal{A}, \mathcal{B}$ which are models of T_0^*: if $\mathcal{A} \equiv_{\Phi_0(\underline{c}_0)} \mathcal{B}$, then $\mathcal{A}$ and $\mathcal{B}$ are $\mathcal{L}_0(\underline{c}_0)$-elementarily equivalent (see any textbook on model theory like [25] for more information).[34]

Theorem 4.4.7 (Ghilardi [44])**.** *Suppose that Φ_1 and Φ_2 are first-order equational i.a.f.'s satisfying conditions* (C1)-(C4) *above; if constraint satisfiability problems are decidable in Φ_1 and Φ_2, then they are decidable in $\Phi_1 \oplus \Phi_2$ too.*

Proof. We check the conditions of Theorem 4.4.6. We take $\mathcal{S}_0^\star$ to be the class of $\mathcal{L}_0$-structures $\mathcal{B}$ such that $\mathcal{B} \in \mathcal{S}_0$ and $\mathcal{B}$ is a model of the model-completion $T_0^\star$ of T_0 (see (C2) above). To check that $\Phi_0^\star = \langle \mathcal{L}_0, T_0, \mathcal{S}_0^\star \rangle$ is a specialization of Φ_0 argue as follows. If $\mathcal{A} \in \mathcal{S}_0(\underline{c}_0)$, then by Definition 4.3.1, there is $\mathcal{M} \in \mathcal{S}_i(\underline{c}_0)$ $(i = 1, 2)$ such that $\mathcal{M}_{|\mathcal{L}_0(\underline{c}_0)} = \mathcal{A}$. By (C3) above there is some $\mathcal{M}' \in \mathcal{S}_i$ which is a model of $T_0^\star$ and such that $\mathcal{M}' \equiv_{\Phi_i(\underline{c}_0)} \mathcal{M}$ holds. It follows that the $\mathcal{L}(\underline{c}_0)$-reduct of $\mathcal{M}'$ is in $\mathcal{S}_0^\star$ and it is $\Phi_0(\underline{c}_0)$-equivalent to $\mathcal{A}$. The same argument proves also the $\Phi_0^\star$-compatibility of Φ_i.

If we have two decision procedures for the constraint satisfiability problem over Φ_1 and Φ_2, then 4.4.6 (1) holds; (C4) guarantees 4.4.6 (2) and 4.4.6 (3) has been just proved. To check the remaining condition 4.4.6 (4), we use ultrapowers and the isomorphism Theorem 4.4.2.

In fact, for every $\underline{c}_0$, if $\mathcal{A}, \mathcal{B} \in \mathcal{S}_0^\star(\underline{c}_0)$ are such that $\mathcal{A} \equiv_{\Phi_0(\underline{c}_0)} \mathcal{B}$, then $\mathcal{A}$ and $\mathcal{B}$ are $\mathcal{L}_0(\underline{c}_0)$-elementarily equivalent because they are models of $T_0^\star$. Hence we have $\prod_{\mathcal{U}} \mathcal{A} \simeq_{\Phi_0(c0)} \prod_{\mathcal{U}} \mathcal{B}$, for a suitable ultrafilter $\mathcal{U}$.

Ultrapowers as structural operations are $\Phi_i^\star$-extensible, meaning that for every $\mathcal{U}$, for every $\underline{c}$ and for every $\mathcal{A} \in \mathcal{S}_i^\star(\underline{c})$, there exists $\mathcal{B} \in \mathcal{S}_i^\star(\underline{c})$ such that $\mathcal{B}_{|\mathcal{L}_0(\underline{c})} \simeq_{\Phi_0(\underline{c})}$

[33] Instead of (C3), in [44] it is asked the (apparently stronger but in fact equivalent) condition: (C3') every $\mathcal{A} \in \mathcal{S}_i$ embeds into some $\mathcal{A}' \in \mathcal{S}_i$ which is a model of $T_0^\star$.

[34] Usually condition (c) is formulated by saying that: (c') the union of $T_0^\star$ and of the Robinson diagram of a model of T_0 is a complete first-order theory. It can be shown that (c) is equivalent to (c'), but for our purpose of deriving the results from [44], it is sufficient to observe that (c') implies (c): this is clear since $\mathcal{A} \equiv_{\Phi_0(\underline{c}_0)} \mathcal{B}$ means precisely that $\mathcal{A}$ and $\mathcal{B}$ are both models of the Robinson diagram of the substructure generated by the $\underline{c}_0$ (the latter is a model of T_0 because T_0 is universal).

$\prod_{\mathcal{U}}(\mathcal{A}_{|\mathcal{L}_0(\underline{c})})$ and $\mathcal{B} \equiv_{\Phi_i(\underline{c})} \mathcal{A}$.[35] To see this, take $\mathcal{B} := \prod_{\mathcal{U}} \mathcal{A}$; since the class of structures which are models of an elementary theory is closed under ultrapowers, $\mathcal{B} \in \mathcal{S}_i^\star(\underline{c})$. Furthermore we have

$$\mathcal{B}_{|\mathcal{L}_0(\underline{c})} = (\prod_{\mathcal{U}} \mathcal{A})_{|\mathcal{L}_0(\underline{c})} \simeq_{\Phi_0(\underline{c})} \prod_{\mathcal{U}} (\mathcal{A}_{|\mathcal{L}_0(\underline{c})}),$$

as desired. □

If we take as T_0 the empty theory (in the one-sorted first-order empty language with equality), then $T_0^\star$ is the theory of an infinite set and condition (C3) is equivalent to stable infiniteness (by a simple argument based on compactness); thus, Theorem 4.4.7 *reduces to the standard Nelson-Oppen result* (see [69, 73, 86]) *concerning stably infinite theories over disjoint signatures.*

We recall from [44] that among relevant examples of theories to which Theorem 4.4.7 is easily seen to apply, we have Boolean algebras with operators (namely the theories axiomatizing algebraic semantics of modal logic): thus, decidability of conditional word problem transfers from two theories axiomatizing varieties of Boolean algebras with operators to their union (provided only Boolean operators are shared). This result, proved in [93] by specific techniques, is the algebraic version of the *fusion transfer of decidability of global consequence relation in modal logic.*

We remark that condition (C4) can be weakened to

(C4′) Φ_0 is noetherian and there exist noetherian positive residue Φ_1- and Φ_2-enumerators for Φ_0

(as suggested by Theorem 4.4.6 (2)) and we give an example of an application of Theorem 4.4.7 under this weaker condition.

Example 4.4.8 (Combination of Noetherian fragments)**.** We consider the combined fragment $\Phi_1 \oplus \Phi_2$ where Φ_1 is the fragment Φ_{Kalg} of the Example 4.2.23 and Φ_2 is the fragment Φ_{Kend} of Example 4.2.25 (here, however, we require K-algebras to be non degenerate, i.e. to satisfy the condition $0 \neq 1$). From Definition 4.3.2, it follows that the class $\mathcal{S}_1 \oplus \mathcal{S}_2$ consists of the models of the theory of the non degenerate K-algebras endowed with a linear endomorphism (i.e. endowed with a function f preserving sum and scalar multiplication). The set $\mathcal{S}_0$ of structures of the shared fragment Φ_0 consists of the models of the theory T_0 of K-vector spaces. The theory T_0 is universal and admits as a model completion the theory $T_0^\star = T_0 \cup \{\exists x\,(x \neq 0)\}$, if K is an infinite field, and the theory

[35] Since, for the sake of simplicity, we limited our analysis to the one-sorted case, the set of types of Φ_0 is the same as the set of types of Φ_i, so the $\underline{c}_0$ in the definition of an extensible operation are equal to the $\underline{c}$ here.

$T_0 \cup \{\exists x_1 \cdots \exists x_n \bigwedge_{i \neq j} x_i \neq x_j\}_{n \in \mathbb{N}}$, otherwise: in both cases, the models of $T_0^\star$ are just infinite K-vector spaces. Thus conditions (C1) and (C2) are satisfied. Since every non degenerate K-algebra (resp. every f-K-vector space) can be embedded into an infinite K-algebra (resp. into an infinite f-K-vector space), condition (C3) holds too. Condition (C4′) is also satisfied, as pointed out in Subsection 4.2.5 when discussing Examples 4.2.23, 4.2.24 and 4.2.25. Hence the combination procedure FCOMB decides conditional word problems for the theory of (non degenerate) K-algebras endowed with a linear endomorphism.

As another application of Theorem 4.4.6 based on Keisler-Shelah isomorphism theorem, we show *how to include a first-order equational theory within description logic A-Boxes*. An *A-Box fragment* is an i.a.f. of the kind $\Phi_{ML} = \langle \mathcal{L}_{ML}, T_{ML}, \mathcal{S}_{ML} \rangle$, where $\langle \mathcal{L}_{ML}, T_{ML} \rangle$ is defined (out of a modal signature O_M) as in Example 4.2.11 and $\mathcal{S}_{ML}$ is a class of $\mathcal{L}_{ML}$-structures closed under isomorphisms and disjoint I-copies. The latter operation is defined as follows:

Definition 4.4.9 (Disjoint I-copy)**.** Consider a first-order one-sorted relational signature $\mathcal{L}$ and a (non-empty) index set I. The operation $\sum_I$, defined on $\mathcal{L}$-structures and called *disjoint I-copy*, associates with an $\mathcal{L}$-structure $\mathcal{M} = \langle [\![-]\!]_{\mathcal{M}}, \mathcal{I}_{\mathcal{M}} \rangle$ the $\mathcal{L}$-structure $\sum_I \mathcal{M}$ such that $[\![W]\!]_{\sum_I \mathcal{M}}$ is the disjoint union of I-copies of $[\![W]\!]_{\mathcal{M}}$ (here W is the unique sort of $\mathcal{L}$). The interpretation of relational predicates is defined as follows[36]

$$\sum_I \mathcal{M} \models P(\langle d_1, i_1 \rangle, \ldots, \langle d_n, i_n \rangle) \iff i_1 = i_2 = \cdots = i_n \text{ and } \mathcal{M} \models P(d_1, \ldots, d_n) \tag{4.4}$$

for every n-ary predicate P.

Disjoint I-copy is a special case of a more general disjoint union operation: the latter is defined again by (4.4) and applies to any I-indexed family of structures (which may not coincide with each other). Our specific interest for disjoint I-copies is motivated by the following Lemma, concerning satisfiability of packed guarded formulae:[37]

Lemma 4.4.10. *Consider a first-order one-sorted relational signature $\mathcal{L}$, the $\mathcal{L}$-structure $\mathcal{M}$ and its disjoint I-copy $\sum_I \mathcal{M}$. The following statements hold:*

(i) *for every elementary packed guarded formula $\varphi[x_1, \ldots, x_n]$ ($n \geq 0$), for every $d_1, \ldots, d_n$ in the support of $\mathcal{M}$ and for every index $i \in I$, we have that*

$$\sum_I \mathcal{M} \models \varphi[\langle d_1, i \rangle, \ldots, \langle d_n, i \rangle] \iff \mathcal{M} \models \varphi[d_1, \ldots, d_n];$$

[36] Elements of the disjoint union of I-copies of a set S are represented as pairs $\langle s, i \rangle$ (meaning that $\langle s, i \rangle$ is the i-th copy of $s \in S$).

[37] See Example 4.2.15 for the related definition.

(ii) *a packed guarded elementary sentence is satisfiable in $\mathcal{M}$ iff it is satisfiable in $\sum_I \mathcal{M}$.*

Proof. We check the first claim by induction on φ (the second claim follows immediately for the case $n = 0$). If φ is atomic, just apply (4.4), and the case of Boolean connectives is immediate. Suppose now that φ is the packed guarded existential quantification

$$\exists y_1 \cdots \exists y_m(\pi[x_{i_1}, \ldots, x_{i_k}, y_1, \ldots, y_m] \wedge \psi[x_{i_1}, \ldots, x_{i_k}, y_1, \ldots, y_m])$$

where $x_{i_1}, \ldots, x_{i_k}$ are the variables among $x_1, \ldots, x_n$ that really occur free in $\varphi[x_1, \ldots, x_n]$ (notice that they must all occur free in the guard π, as well as the $y_1, \ldots, y_m$). That $\mathcal{M} \models \varphi[d_1, \ldots, d_n]$ implies $\sum_I \mathcal{M} \models \varphi[\langle d_1, i\rangle, \ldots, \langle d_n, i\rangle]$ is trivial; for the converse suppose that

$$\sum_I \mathcal{M} \models \pi[\langle d_{i_1}, i\rangle, \ldots, \langle d_{i_k}, i\rangle, \langle e_1, j_1\rangle, \ldots, \langle e_m, j_m\rangle] \wedge \\ \wedge \psi[\langle d_{i_1}, i\rangle, \ldots, \langle d_{i_k}, i\rangle, \langle e_1, j_1\rangle, \ldots, \langle e_m, j_m\rangle]$$

for some $\langle e_1, j_1\rangle, \ldots, \langle e_m, j_m\rangle$. By (4.4) and the definition of a guard, all indexes $j_1, \ldots, j_m$ must be equal to some j (and, if $k \neq 0$, j must be i). Thus $\mathcal{M} \models \pi[d_{i_1}, \ldots, d_{i_k}, e_1, \ldots, e_m] \wedge \psi[d_{i_1}, \ldots, d_{i_k}, e_1, \ldots, e_m]$ holds by induction hypothesis and by (4.4). □

Let O_M be a modal signature, as defined in Example 4.2.10: notice that formulae like $ST(\varphi, w)$ and $\forall w ST(\varphi, w)$ are packed guarded, hence if we replace in them the second-order variables of type $W \to \Omega$ by free constants for subsets of W (which are first-order relational symbols), Lemma 4.4.10 (i)-(ii) applies to the formulae so obtained.

We now want to combine an equational first-order i.a.f. $\Phi = \langle \mathcal{L}, T, \mathcal{S}\rangle$ from Example 4.2.6 and an A-Box fragment $\Phi_{ML} = \langle \mathcal{L}_{ML}, T_{ML}, \mathcal{S}_{ML}\rangle$ (we suppose that the signatures $\mathcal{L}$ and $\mathcal{L}_{ML}$ are disjoint). Assume in addition that $\mathcal{S}_{ML}$ is an elementary class (i.e it is the class of the models of a first-order $\mathcal{L}_{ML}$-theory) and that Φ is stably infinite.

Since all our data are first-order, the argument of the proof of Theorem 4.4.7 works, provided conditions (C1)-(C4) hold. We take as T_0 the empty theory and as $T_0^\star$ the theory of an infinite set, so that we only have to check condition (C3) for both Φ and Φ_{ML}. For the former, the condition holds trivially (the situation is the same as in the standard disjoint Nelson-Oppen case mentioned above). For the latter, for every $\mathcal{L}_{ML}(\underline{c})$-structure $\mathcal{A}(\underline{c})$ and for an infinite I, by Lemma 4.4.10 (i) we can expand $\sum_I \mathcal{A}$ to a $\mathcal{L}_{ML}(\underline{c})$-structure in such a way that $\mathcal{A}(\underline{c}) \equiv_{\Phi_{ML}(\underline{c})} \sum_I \mathcal{A}(\underline{c})$ holds[38]: this proves condition (C3) (obviously, $\sum_I \mathcal{A}(\underline{c})$ is infinite, because I is infinite). We so proved the following result:

Theorem 4.4.11. *Suppose that we are given an equational first-order i.a.f. $\Phi = \langle \mathcal{L}, T, \mathcal{S}\rangle$ and an A-Box fragment $\Phi_{ML} = \langle \mathcal{L}_{ML}, T_{ML}, \mathcal{S}_{ML}\rangle$; suppose also that the signatures $\mathcal{L}$ and*

[38]To this aim, interpret a free individual constant $c \in \underline{c}$ in $\sum_I \mathcal{A}$ as $\langle \mathcal{I}_{\mathcal{A}(\underline{c})}(c), i\rangle$, where i is some arbitrarily chosen element of I (to be the same for all the free individual constants c that belong to $\underline{c}$).

$\mathcal{L}_{ML}$ are disjoint, that Φ is stably infinite and that $\mathcal{S}_{ML}$ is an elementary class. Then decidability of constraint satisfiability problems transfers from Φ and Φ_{ML} to $\Phi \oplus \Phi_{ML}$.

The fragment $\Phi \oplus \Phi_{ML}$ of Theorem 4.4.11 is quite peculiar, because its combined terms all arise from a single composition step (they all have degree 2, in the terminology of Lemma 4.3.4).

4.4.3 Applications: Decidability Transfer through Disjoint Copies

Disjoint copies are the key tool for transfer decidability results in modal fragments too. Let O_M be a modal signature, as defined in Example 4.2.10. A *modal i.a.f. over O_M* is a fragment of the kind $\Phi_M = \langle \mathcal{L}_M, T_M, \mathcal{S}_M \rangle$, where $\mathcal{L}_M$ and T_M are as defined in Example 4.2.10, whereas $\mathcal{S}_M$ is a class of $\mathcal{L}_M$-structures *closed under isomorphisms and disjoint I-copies*. The next proposition translates into our settings the main ingredient of the decidability transfer proof for relativized satisfiability in [10].

In the following, we indicate by O_{M_0} the empty modal signature.

Proposition 4.4.12. *Let Φ_M be a modal i.a.f. over the modal signature O_M and consider a modal subfragment Φ_{M_0} of it, based on the empty modal signature; the structural operations $\{\sum_I\}_I$ over Φ_{M_0} are Φ_M-extensible and form a collection admitting a Φ_{M_0}-isomorphism theorem.*

Proof. Recall from Example 4.2.10 that $W \to \Omega$ is the only type of the i.a.f. Φ_M, hence the relevant free constants $\underline{c}$ in expanded languages are second-order constants for subsets of W (this means, in particular, that their interpretation can be extended to disjoint I-copies like any other relational first-order predicate symbol as shown in Definition 4.4.9).

That taking disjoint I-copies $\sum_I$ is a structural Φ_M-extensible operation is clear: to define $\mathcal{N} \in \mathcal{S}_M(\underline{c})$ which is $\Phi_M(\underline{c})$-equivalent to some $\mathcal{M} \in \mathcal{S}_M(\underline{c})$ and whose $\Phi_{M_0}(\underline{c})$-reduct is $\mathcal{L}_{M_0}(\underline{c})$-isomorphic to $\sum_I(\mathcal{M}_{|\mathcal{L}_{M_0}(\underline{c})})$ it is sufficient to take the $\mathcal{L}_M(\underline{c})$-structure $\sum_I \mathcal{M}$ as $\mathcal{N}$ and apply Lemma 4.4.10 (ii) (recall that constraints in Φ_M are equivalent to conjunctions of formulae of the kind $\forall w ST(\varphi, w)$ and their negations). That taking disjoint I-copies is a structural operation (i.e. that a $\mathcal{L}_{M_0}(\underline{c})$-structure and its disjoint I-copy are $\Phi_{M_0}(\underline{c})$-equivalent) is clear by the same reasons.

To show that a Φ_{M_0}-isomorphism theorem holds, suppose that we are given free constants $\underline{c}_0 := \{P_1, \ldots, P_n\}$ and two structures $\mathcal{M}_1$ and $\mathcal{M}_2$ in $\mathcal{S}_{M_0}(\underline{c}_0)$ such that $\mathcal{M}_1 \equiv_{\Phi_{M_0}(\underline{c}_0)} \mathcal{M}_2$; we show that $\sum_I \mathcal{M}_1 \simeq_{\Phi_{M_0}(\underline{c}_0)} \sum_I \mathcal{M}_2$ holds for some I.

Consider every boolean combination of the form $\varepsilon(w) = Q_1(w) \wedge \cdots \wedge Q_n(w)$ where $Q_j \equiv P_j$ or $Q_j \equiv \neg P_j$ (thus the number of such formulae is 2^n). For a given $\mathcal{L}_{M_0}(\underline{c}_0)$-structure $\mathcal{N}$, let $\varepsilon(\mathcal{N}) := \{a \in [\![W]\!]_{\mathcal{N}} \mid \mathcal{N} \models \varepsilon(a)\}$ and let us associate with $\mathcal{N}$ the 2^n cardinal invariants $a_\varepsilon(\mathcal{N}) := \sharp\varepsilon(\mathcal{N})$. Now two $\mathcal{L}_{M_0}(\underline{c}_0)$-structures $\mathcal{N}_1$ and $\mathcal{N}_2$ having

the same invariants are isomorphic, because we can glue bijections $\varepsilon(\mathcal{N}_1) \longrightarrow \varepsilon(\mathcal{N}_2)$ to a $\mathcal{L}_{M_0}(\underline{c}_0)$-isomorphism $\mathcal{N}_1 \simeq \mathcal{N}_2$.

Finally, we note that $\mathcal{M}_1 \equiv_{\Phi_{M_0}(\underline{c}_0)} \mathcal{M}_2$ means that $\mathcal{M}_1 \models A$ holds iff $\mathcal{M}_2 \models A$ holds for every closed $\Phi_{M_0}(\underline{c}_0)$-atom A. In particular, $\mathcal{M}_1 \models \{w \mid \varepsilon(w)\} = \{w \mid \bot\}$ iff $\mathcal{M}_2 \models \{w \mid \varepsilon(w)\} = \{w \mid \bot\}$: thus $\varepsilon(\mathcal{M}_1) = \emptyset$ iff $\varepsilon(\mathcal{M}_2) = \emptyset$ holds for all ε. Let now consider a set I whose cardinality m is such that $m \geq \varepsilon(\mathcal{M}_i)$ for all ε and for $i \in \{1, 2\}$: we show that $\sum_I \mathcal{M}_1 \simeq_{\Phi_{M_0}(\underline{c}_0)} \sum_I \mathcal{M}_2$ proving that the two structures have the same invariants. In fact the cardinal identities $a_\varepsilon(\sum_I \mathcal{M}_1) = m \cdot a_\varepsilon(\mathcal{M}_1) = m = m \cdot a_\varepsilon(\mathcal{M}_2) = a_\varepsilon(\sum_I \mathcal{M}_2)$ hold for all ε. □

If O_{M_1} and O_{M_2} are modal signatures, we let $O_{M_1 \oplus M_2}$ indicate their disjoint union ($O_{M_1 \oplus M_2}$ is called the fusion of the modal signatures O_{M_1} and O_{M_2}). Given a modal i.a.f. Φ_{M_1} over O_{M_1} and a modal i.a.f. Φ_{M_2} over O_{M_2}, let us define their *fusion* as the modal i.a.f.

$$\Phi_{M_1 \oplus M_2} = \langle \mathcal{L}_{M_1 \oplus M_2}, T_{M_1 \oplus M_2}, \mathcal{S}_{M_1} \oplus \mathcal{S}_{M_2} \rangle.$$

Notice that for two modal i.a.f.'s $\Phi_{M_1} = \langle \mathcal{L}_{M_1}, T_{M_1}, \mathcal{S}_{M_1} \rangle$ and $\Phi_{M_2} = \langle \mathcal{L}_{M_1}, T_{M_1}, \mathcal{S}_{M_1} \rangle$ over disjoint modal signatures, the shared fragment $\Phi_{M_0} = \langle \mathcal{L}_{M_0}, T_{M_0}, \mathcal{S}_{M_0} \rangle$ is locally finite, because it is a modal i.a.f. over the empty modal signature (for any finite set of Φ_{M_0}-variables $\underline{x}_0$, the representative $\Phi_{M_0}(\underline{x}_0)$-terms are those of the kind $\{w \mid \psi(w)\}$, where ψ is a boolean combination of the second-order variables $\underline{x}_0$).

Now if Φ_{M_1} and Φ_{M_2} have decidable constraint satisfiability problems, then so does the combined i.a.f. $\Phi_{M_1} \oplus \Phi_{M_2}$: in fact, the hypotheses of Theorem 4.4.6 are satisfied by the previous proposition.[39] To infer the transfer decidability result to the fusion modal i.a.f., we need to clarify the relationship between $\Phi_{M_1 \oplus M_2}$ and $\Phi_{M_1} \oplus \Phi_{M_2}$.

Given two i.a.f.'s $\Phi_1 = \langle \mathcal{L}_1, T_1, \mathcal{S}_1 \rangle$ and $\Phi_2 = \langle \mathcal{L}_2, T_2, \mathcal{S}_2 \rangle$, we say that they are $\beta\eta$-*equivalent* (written $\Phi_1 \sim_{\beta\eta} \Phi_2$) iff $\mathcal{L}_1 = \mathcal{L}_2$, $\mathcal{S}_1 = \mathcal{S}_2$ and moreover for every $t_1 \in T_1$ one can effectively compute some $t_2 \in T_2$ such that $t_1 \sim_{\beta\eta} t_2$, and vice versa. Clearly, $\beta\eta$-equivalent i.a.f.'s can be considered to be just the same.

Lemma 4.4.13. *If $\Phi_{M_1 \oplus M_2}$ and $\Phi_{M_1} \oplus \Phi_{M_2}$ are as above, we have that $\Phi_{M_1 \oplus M_2} \sim_{\beta\eta} \Phi_{M_1} \oplus \Phi_{M_2}$.*

Proof. Since $T_{M_1} \oplus T_{M_2}$ is defined to be the minimum set of terms closed under substitutions and containing T_{M_1} and T_{M_2} and since $T_{M_1 \oplus M_2}$ enjoys these properties, clearly any $t \in T_{M_1} \oplus T_{M_2}$ belongs to $T_{M_1 \oplus M_2}$.

Conversely, let us take $t \in T_{M_1 \oplus M_2}$; then $t \sim_{\beta\eta} \{w \mid ST(\varphi, w)\}$ for some $O_{M_1 \oplus M_2}$-modal formula φ.[40] By induction on φ, we define $u \in T_{M_1} \oplus T_{M_2}$ such that $u \sim_{\beta\eta}$

[39] We obviously take $\Phi_0^\star$ to be Φ_{M_0} in 4.4.6 (3).

[40] Notice that $\{w \mid ST(\varphi, w)\}$ - hence also φ - can be effectively computed because it is in long-$\beta\eta$-normal form and so it is the long-$\beta\eta$-normal form of t.

$\{w \mid ST(\varphi, w)\}$ (then $t \sim_{\beta\eta} u$ follows by transitivity). If φ is a propositional variable we can take u to be $\{w \mid ST(\varphi, w)\}$. If φ is $\psi_1 \wedge \psi_2$, by induction there are $u_1, u_2 \in T_{M_1} \oplus T_{M_2}$ such that $u_i \sim_{\beta\eta} \{w \mid ST(\psi_i, w)\}$ for $i = 1, 2$. Then $\{w \mid ST(\varphi, w)\} = \{w \mid ST(\psi_1, w) \wedge ST(\psi_2, w)\} \sim_{\beta\eta} \{w \mid \{w \mid ST(\psi_1, w)\}(w) \wedge \{w \mid ST(\psi_2, w)\}(w)\} \sim_{\beta\eta} \{w \mid u_1(w) \wedge u_2(w)\}$. The latter is obtained by replacing in the term $\{w \mid ST(\mathrm{x}_1 \wedge \mathrm{x}_2, w)\} = \{w \mid X_1(w) \wedge X_2(w)\}$ the terms $u_1, u_2 \in T_{M_1} \oplus T_{M_2}$ for the second-order variables X_1, X_2, respectively, hence it is a term that belongs to $T_{M_1} \oplus T_{M_2}$ too, because the latter is closed under substitution. The cases of $\vee, \neg, \Diamond_k$ are analogous. □

We have so proved the following well-known decidability transfer result (see, e.g., [10] and the literature quoted therein).

Theorem 4.4.14 (Decidability Transfer for Modal i.a.f.'s)**.** *If two modal i.a.f.'s Φ_{M_1} and Φ_{M_2} have decidable constraint satisfiability problems, so does their fusion $\Phi_{M_1 \oplus M_2}$.*

Fragments of the kind examined in Example 4.2.11 are not interesting for being combined with each other, because the absence of the type $W \to \Omega$ makes such combinations trivial. On the contrary, full modal fragments from Example 4.2.12 are quite interesting in this respect (we recall that they reproduce both A-Box and T-Box reasoning from the point of view of description logics). In fact very slight modifications are sufficient to get a result analogous to Theorem 4.4.14: we just sketch how to do it.

Let O_M be a modal signature; a *full modal i.a.f. over O_M* is a fragment of the kind $\Phi_{FM} = \langle \mathcal{L}_{FM}, T_{FM}, \mathcal{S}_{FM} \rangle$, where $\mathcal{L}_{FM}$ and T_{FM} are as defined in Example 4.2.12, whereas $\mathcal{S}_{FM}$ is again a class of $\mathcal{L}_M$-structures closed under isomorphisms and disjoint I-copies.

There is a little complication arising now: since W is a type of an i.a.f. like Φ_{FM}, when we expand languages with free constants, we now get (besides constants of type $W \to \Omega$) also individual constants of type W. The interpretation of these constants is not defined in disjoint I-copies, because taking disjoint I-copies is an operation defined only for first-order relational signatures. We proceed as follows: we take index sets I which are pointed, namely some $i_0 \in I$ is specified. Then, we define the interpretation of an individual constant c of type W in $\sum_I \mathcal{M}$ as $\langle \mathcal{I}_{\mathcal{M}}(c), i_0 \rangle$.

The definition of fusion for full modal i.a.f.'s is the obvious one and it leads to the following:

Theorem 4.4.15 (Decidability Transfer for Full Modal i.a.f.'s [10])**.** *If two full modal i.a.f.'s have decidable constraint satisfiability problems, so does their fusion.*

Proof. We sketch the little modifications required to prove Proposition 4.4.12 in the present context (Lemma 4.4.13 does not need any essential change). Let Φ_{FM} be a full modal i.a.f. over O_M and let Φ_{FM_0} be a subfragment of it on the empty modal signature.

According to the considerations in Examples 4.2.10 and 4.2.11, when considering languages expanded with free constants $\underline{c}$, closed $\Phi_{FM}(\underline{c})$-atoms are now of the kind $c_1 = c_2$, $R_k(c_1, c_2)$, $ST(\psi, c/w)$, and $\forall w ST(\psi, w)$ (where second-order variables in ψ have been replaced by first-order unary predicate constants). From the pointed definition of disjoint I-copy given above and Lemma 4.4.10, it is then clear that the $\mathcal{L}_{FM}(\underline{c})$-structures $\mathcal{M}$ and $\sum_I \mathcal{M}$ still are $\Phi_{FM}(\underline{c})$-equivalent and this is all what matters in order to check that pointed disjoint I-copies are Φ_{FM}-extensible structural operations over Φ_{FM_0}.

For the Φ_{FM_0}-isomorphism theorem, we just need to add to the invariants of a $\mathcal{L}_{FM_0}(\underline{c})$-structure $\mathcal{N}$ considered in the proof of Proposition 4.4.12 also the indication about the truth/falsity in $\mathcal{N}$ of the ground atoms of the kind $\varepsilon(c)$ and $c_1 = c_2$, varying c, c_1, c_2 among the individual constants in $\underline{c}$. □

The statement of Theorem 4.4.15 seems not to allow the decidability transfer of only *positive* A-Box satisfiability with respect to T-Box axioms; however this further decidability transfer result follows immediately once one realizes that the combined algorithm TComb never adds negative information to current constraints, so if non positive A-Boxes are not present from the very beginning, there won't be any call for a decision procedure involving them (see also the Remark following Theorem 4.4.6 for the same observation).

The decidability transfer theorem for the non-normal case of Example 4.2.13 (i.e. for the full strength of abstract description systems in the sense of [10]) requires a simple adaptation of Definition 4.4.9 and of Lemma 4.4.10. We can also extend our transfer results to fragments involving the μ-calculus fixed-points constructors of Example 4.2.14: in fact, these constructors are invariant under bisimulation, hence Lemma 4.4.10 still holds (notice also that fixed points can be eliminated from empty modal signatures, hence local finiteness of the shared fragment is not compromised, even in case we wish to combine two 'μ-fragments' with each other).

We now try to extend our decidability transfer results to cover also combinations of packed guarded or of two-variable fragments. However, to get positive results, we need to keep shared signatures under control (otherwise undecidability phenomena arise). In addition, we still want to exploit the isomorphism theorem of Proposition 4.4.12 and, for that, we need the shared signature to be empty and second-order variables appearing as terms in the fragments to be monadic only. The kind of combination that arise in this way is a form of fusion that we shall call monadic fusion. We begin by identifying a class of fragments to which our techniques apply.

Let us call $\Phi_\emptyset = \langle \mathcal{L}_\emptyset, T_\emptyset, \mathcal{S}_\emptyset \rangle$ the following i.a.f.: (i) $\mathcal{L}_\emptyset$ is the empty one-sorted first-order signature (that is, $\mathcal{L}_\emptyset$ does not contain any proper symbol, except for its unique sort which is called D); (ii) $T_\emptyset$ consists of the terms which are $\beta\eta$-equivalent to terms

belonging to $T_{11}^{\mathcal{L}_\emptyset}$;[41] (iii) $\mathcal{S}_\emptyset$ contains all $\mathcal{L}_\emptyset$-structures.

Definition 4.4.16. A *monadically suitable* i.a.f. $\Phi = \langle \mathcal{L}, T, \mathcal{S} \rangle$ is an i.a.f. such that:

(i) $\mathcal{L}$ is a relational one-sorted first-order signature;

(ii) $T_{11}^{\mathcal{L}_\emptyset} \subseteq T \subseteq T_{\omega 1}^{\mathcal{L}}$;[42]

(iii) the $\Phi_\emptyset$-structural operation of taking disjoint I-copies is Φ-extensible.

We remark that, despite the fact that the definition of a monadically suitable fragment needs the present chapter settings to be formulated, there is some anticipation of it in the literature on monodic fragments (see for instance statements like that of Theorem 11.21 in [40]). We give a couple of interesting examples of monadically suitable decidable fragments:

Example 4.4.17. *Packed guarded fragments* are i.a.f.'s of the kind $\Phi_G = \langle \mathcal{L}_G, T_G, \mathcal{S}_G \rangle$, where T_G is as defined in Example 4.2.15, whereas $\mathcal{S}_G$ is a class of $\mathcal{L}_G$-structures closed under isomorphisms and disjoint I-copies. To see that these are monadically suitable fragments, recall Lemma 4.4.10: by this Lemma, it is easy to see that for every free constants $\underline{c}$ of type $D \to \Omega$, for every $\mathcal{A} \in \mathcal{S}_G(\underline{c})$ and for every non-empty set of indexes I, we have that $\mathcal{A} \equiv_{\Phi_G(\underline{c})} \sum_I \mathcal{A}$. Thus taking disjoint I-copies is trivially Φ_G-extensible.

Before giving the second family of examples of monadically suitable fragments, we introduce an alternative construction for proving extensibility of the operation of taking disjoint I-copies. This construction is nicely behaved only for fragments without identity and is called I-conglomeration:

Definition 4.4.18 (I-conglomeration)**.** Consider a first-order one-sorted relational signature $\mathcal{L}$ and a (non-empty) index set I. The operation $\sum^I$, defined on $\mathcal{L}$-structures and called *I-conglomeration*, associates with an $\mathcal{L}$-structure $\mathcal{M} = \langle [\![-]\!]_\mathcal{M}, \mathcal{I}_\mathcal{M} \rangle$ the $\mathcal{L}$-structure $\sum^I \mathcal{M}$ such that $[\![D]\!]_{\sum^I \mathcal{M}}$ is the disjoint union of I-copies of $[\![D]\!]_\mathcal{M}$ (here D is the unique sort of $\mathcal{L}$). The interpretation of relational constants is defined in such a way that we have

$$\sum^I \mathcal{M} \models P(\langle d_1, i_1 \rangle, \ldots, \langle d_n, i_n \rangle) \iff \mathcal{M} \models P(d_1, \ldots, d_n)$$

for every n-ary relational predicate P different from equality.

[41] See Example 4.2.9 for this notation and for other similar notation used below.

[42] The inclusion $T \subseteq T_{\omega 1}^{\mathcal{L}}$ should be intended up to $\beta\eta$-equivalence (namely, for every $t \in T$ there is $t' \in T_{\omega 1}^{\mathcal{L}}$ such that $t \sim_{\beta\eta} t'$).

Notice that, contrary to disjoint union, I-conglomeration cannot be defined for families of I-indexed structures different from each other; on the other hand, I-conglomerations and disjoint I-copies *coincide* for relational first-order signatures having only unary predicates. The preservation Lemma 4.4.10 can be reformulated as follows:

Lemma 4.4.19. *Consider a first-order one-sorted relational signature $\mathcal{L}$ and the $\mathcal{L}$-structures $\mathcal{M}$ and $\sum^I \mathcal{M}$. The following statements hold:*

(i) *for every first-order formula $\varphi[x_1, \ldots, x_n]$* not containing the equality predicate*, for every $d_1, \ldots, d_n$ in the support of $\mathcal{M}$ and for every indexes $i_1, \ldots, i_n \in I$, we have that*

$$\sum^I \mathcal{M} \models \varphi[\langle d_1, i_1\rangle, \ldots, \langle d_n, i_n\rangle] \iff \mathcal{M} \models \varphi[d_1, \ldots, d_n];$$

(ii) *a first-order formula not containing the equality predicate is satisfiable in $\mathcal{M}$ iff it is satisfiable in $\sum^I \mathcal{M}$.*

Example 4.4.20. For a first-order relational one-sorted signature $\mathcal{L}_{2V}$, a *two variables i.a.f.* over $\mathcal{L}_{2V}$ is a fragment of the kind $\Phi_{2V} = \langle \mathcal{L}_{2V}, T_{2V}, \mathcal{S}_{2V}\rangle$, where: (i) T_{2V} contains the terms *without identity* which are $\beta\eta$-equivalent to terms belonging to the set $T_{NK}^{\mathcal{L}_{2V}}$ of Example 4.2.9 for $K = 1$ and $N = 2$; (ii) $\mathcal{S}_{2V}$ is a class of $\mathcal{L}_{2V}$-structures closed under isomorphisms and I-conglomerations.[43]

For two monadically suitable i.a.f.'s Φ_1 and Φ_2 operating on disjoint signatures, let us call the combined fragment $\Phi_1 \oplus \Phi_2$ the *monadic fusion* of Φ_1 and Φ_2. For monadic fusions we have the following

Theorem 4.4.21 (Decidability Transfer for Monadically Suitable i.a.f.'s)**.** *If two monadically suitable i.a.f.'s Φ_1, Φ_2 operating on disjoint signatures have decidable constraint satisfiability problems, so does their monadic fusion.*

Proof. Using Definition 4.4.16, we can say the following about the shared fragment $\Phi_0 = \langle \mathcal{L}_0, T_0, \mathcal{S}_0\rangle$: (i) $\mathcal{L}_0$ is the empty signature $\mathcal{L}_\emptyset$; (ii) T_0 contains $T_{11}^{\mathcal{L}_\emptyset}$ and hence it includes the terms T_{M_0} of Example 4.2.10 relative to the empty modal signature O_{M_0}; (iii) for every tuple of free constants $\underline{c}_0$, the closed $\Phi_0(\underline{c}_0)$-terms $t[\underline{c}_0]$, modulo $\beta\eta$-equivalence, are a subset of the terms of the kind $\{x \mid \varphi[x]\}$, where $\varphi[x]$ is a monadic formula of first-order language, possibly with equality (that is, to build $\varphi[x]$, at most equality and the free constants $\underline{c}_0$ of type $D \to \Omega$ can be used); (iv) the structures in $\mathcal{S}_0(\underline{c}_0)$ are closed under disjoint I-copies and are $\Phi_0(\underline{c}_0)$-equivalent to their disjoint I-copies.

To justify (iv), argue as follows: if $\mathcal{A} \in \mathcal{S}_0(\underline{c}_0)$, then by Definition 4.3.1 it is the $\mathcal{L}_\emptyset(\underline{c}_0)$-reduct of a $\Phi_i(\underline{c}_0)$-structure $\mathcal{B}$ $(i = 1, 2)$; since taking disjoint I-copies of $\mathcal{L}_\emptyset(\underline{c}_0)$-structures

[43] These closure properties are guaranteed if $\mathcal{L}_{2V}$ is axiomatized by first-order formulae without identity (see [53] for recent interesting material, both in the decidable and in the undecidable case).

is $\Phi_i(\underline{c}_0)$-extensible by Definition 4.4.16(iii), we have that for every index set I, there is a $\Phi_i(\underline{c}_0)$-structure $\mathcal{B}'$ having $\sum_I \mathcal{A}$ as a $\mathcal{L}_\emptyset(\underline{c}_0)$-reduct and such that $\mathcal{B}$ is $\Phi_i(\underline{c}_0)$-equivalent to $\mathcal{B}'$. Taking $\mathcal{L}_\emptyset(\underline{c}_0)$-reducts, it follows that $\mathcal{A} \equiv_{\Phi_0(\underline{c}_0)} \sum_I \mathcal{A}$.

Using (ii) and (iv) above, we can repeat word-by-word the proof of Proposition 4.4.12 and in order to apply Theorem 4.4.6 we only have to show that Φ_0 is effectively locally finite. Despite the fact that there are infinitely many non equivalent monadic first-order sentences with equality, we shall show by using (iii)-(iv) that there are only finitely many closed $\Phi_0(\underline{c}_0)$-terms $t[\underline{c}_0]$ which are differently interpreted in at least one structure from $\mathcal{S}_0(\underline{c}_0)$ (here $\underline{c}_0 := \{P_1, \ldots, P_n\}$ are free constants, which must be of type $D \to \Omega$, because this is the only type of Φ_0). Recall that $t[\underline{c}_0] \sim_{\beta\eta} \{x \mid \varphi[x]\}$, where $\varphi[x]$ is as in (iii) above.

By closure under disjoint I-copies and $\Phi_0(\underline{c}_0)$-equivalence to disjoint I-copies (see (iv)), we can limit ourselves to the consideration of at most 2^{2^n}-structures from $\mathcal{S}_0(\underline{c}_0)$: each of these structures is uniquely determined by the fact that the cardinal invariants[44] a_ε are either 0 or m in it (here m is an infinite, big enough, cardinal).

Each of these at most 2^{2^n} structures $\mathcal{A}_S$ is identified by a set S of formulae of the kind $\varepsilon(x)$, in the sense that we have $\mathcal{A}_S \models \psi_S$, where ψ_S is the one-variable monadic sentence $\bigwedge_{\varepsilon \in S} \exists x\, \varepsilon(x) \wedge \bigwedge_{\varepsilon \notin S} \neg\exists x\, \varepsilon(x)$ (notice also that for $S \neq S'$, we have $\mathcal{A}_S \not\models \psi_{S'}$).

We *claim* that quantifier elimination holds in $\mathcal{A}_S$, i.e. that for every first-order formula $\varphi[x]$ (built up from equality and from the unary predicates P_j) one can effectively compute from S a formula $\vartheta_S[x]$ such that $\mathcal{A}_S \models \forall x(\varphi[x] \leftrightarrow \vartheta_S[x])$ holds and such that $\vartheta_S[x]$ does not contain quantifiers. To show the claim, we use the fact that for every $\varepsilon \in S$, the set $\varepsilon(\mathcal{A}_S)$ is infinite; recall also that in order to eliminate quantifiers, it is sufficient to eliminate them from primitive formulae, i.e. from formulae of the kind $\exists y\, \chi(y, \underline{x})$, where $\chi(y, \underline{x})$ is a conjunction of literals. In our case, these literals can only be $y = x_i, y \neq x_i, P_j(y), \neg P_j(y)$ (of course, literals in which y does not occur are not relevant). Since equations $y = x_i$ causes the quantifier $\exists y$ to be removed by replacement, we can assume that our χ is equivalent to a conjunction of negative literals $y \neq x_i$ and of a Boolean combination of atomic formulae of the kind $P_j(y)$. The set defined by this Boolean combination in $\mathcal{A}_S$ is either infinite or empty, so within $\mathcal{A}_S$, the formula $\exists y\, \chi(y, \underline{x})$ is equivalent either to $\bot$ or to $\top$. As a consequence of the above claim, in the case in which the tuple $\underline{x}$ reduces to a single variable x, $\vartheta_S[x]$ is a boolean combination of the atomic formulae $P_j(x)$. Thus, in all the structures that belongs to $\mathcal{S}_0(\underline{c}_0)$, the $\Phi_0(\underline{c}_0)$-atom

$$\{x \mid \varphi[x]\} \;=\; \{x \mid \bigvee_S (\psi_S \wedge \vartheta_S[x])\}$$

is true, yielding the claim (because there are only finitely many possibilities for $\vartheta_S[x]$). □

[44] Here and below, we freely use notation from Proposition 4.4.12.

Theorem 4.4.21 offers various combination possibilities, however notice that: (a) the conditions for a fragment to be monadically suitable are rather strong (for instance, the two variable fragment with identity is not monadically suitable); (b) the notion of monadic fusion is a restricted form of combination, because only unary second-order variables are available for replacement when forming formulae of the combined fragment (thus, for instance, the monadic fusion of two two-variables fragments does not contain sentences like $\exists x \exists y(R_1(x, y) \wedge R_2(x, y))$, if R_1, R_2 do not belong to the same component signature).

As already pointed out, the main ingredient of Theorem 4.4.21 (namely the notion of a monadically suitable fragment) is somewhat implicit in the literature on monodic temporal fragments (see for instance [40]). In the next chapter we will show that our main decidability transfer result (Theorem 4.4.6) can be used to prove a decidability theorem for monodic temporal/modal fragments whose extensional component is a monadically suitable fragment, thus extending relevant results from the literature (see again [40]).

Chapter 5

Combination for Monodic Fragments

Applications often require to handle properties that can be expressed as formulae belonging to first-order temporal or modal predicate logics. Unfortunately, fragments in first-order temporal and modal predicate logics become undecidable quite soon: for instance, classical decidability results for the monadic or the two-variables cases do not extend to modal languages (see [57, 41, 54] and also [24] for an essential account of these and related results). However, there are interesting modal predicate fragments which are decidable: one-variable fragments (corresponding to products with S5) are usually decidable (see [82, 40]), as well as many monodic fragments. We recall that a *monodic* formula is a modal first order formula in which modal operators are applied only to subformulae containing at most one free variable; monodic fragments are intended to be classes of monodic formulae (of course the entire class of monodic formulae is too big to be decidable, being an extension of classical first order language).

The interest in monodic fragments relies in the fact that properties which are relevant, for example, in the field of temporal databases can be expressed using formulae belonging to such fragments (see again [40]). Monodic fragments whose extensional (i.e. non modal) components is decidable seem to be decidable too (see [94, 40]): we shall give this fact a formulation in terms of a decidability transfer result for monodic fragments which are obtained as combinations of a suitable extensional fragment and of a one-variable first-order modal fragment. Since we prefer, for simplicity, not to introduce a specific formal notion of a modal fragment, we shall proceed through standard translations and rely on our usual notion of an i.a.f..

5.1 Constant Domains and Standard Translation

Modal predicate formulae are built up from first order atomic formulae of a given first-order one-sorted relational signature $\mathcal{L}$ and from formulae of the kind $X(x)$ (where X is a unary second-order variable), by using boolean connectives, individual quantifiers and a diamond operator $\Diamond$.[1]

There are actually different standard translations for first-order modal languages (see [24]), we shall concentrate here on the translation corresponding to *constant domain* semantics. The latter is defined as follows. The signature $\mathcal{L}^W$ has, in addition to the unique sort D of $\mathcal{L}$, a new sort W; relational constants of type $D^n \to \Omega$ have corresponding relational constants in $\mathcal{L}^W$ of type $D^n W \to \Omega$. We use equal names for corresponding constants: this means for instance that if P has type $D^2 \to \Omega$ in $\mathcal{L}$, the same P has type $D^2 W \to \Omega$ in $\mathcal{L}^W$. We shall make the same conventions for second-order variables: hence a second-order $\mathcal{L}$-variable X of type $D \to \Omega$ has a corresponding second-order variable X of type $DW \to \Omega$ in $\mathcal{L}^W$.

Notice that a $\mathcal{L}^W$-structure $\mathcal{A}$ is nothing but a $[\![W]\!]_{\mathcal{A}}$-indexed class of $\mathcal{L}$-structures, all having the same domain $[\![D]\!]_{\mathcal{A}}$: we indicate by $\mathcal{A}_w$ the structure corresponding to $w \in [\![W]\!]_{\mathcal{A}}$ and call it the *index structure over* w. To be more precise, $\mathcal{A}_w$ is the $\mathcal{L}$-structure having $[\![D]\!]_{\mathcal{A}}$ as a support and moreover, for $P : D^n \to \Omega$, we have that $\mathcal{I}_{\mathcal{A}_w}(P)$ contains the tuples $\underline{a} \in [\![D]\!]^n_{\mathcal{A}}$ such that $\mathcal{A} \models P(\underline{a}, w)$ holds.

The signature $\mathcal{L}^{WR}$ is obtained from $\mathcal{L}^W$ by adding it also a binary 'accessibility' relation R of type $WW \to \Omega$. This is the signature we need for defining the standard translation.

For a modal predicate $\mathcal{L}$-formula $\varphi[x_1^D, \ldots, x_n^D]$ and for a variable $w : W$, we define the (non modal) $\mathcal{L}^{WR}$-formula $ST(\varphi, w)$ as follows:

$$
\begin{aligned}
ST(\top, w) &= \top \\
ST(\bot, w) &= \bot \\
ST(X(x_i), w) &= X(x_i, w) \\
ST(P(x_{i_1}, \ldots, x_{i_m}), w) &= P(x_{i_1}, \ldots, x_{i_m}, w) \\
ST(\neg\psi, w) &= \neg ST(\psi, w) \\
ST(\psi_1 \vee \psi_2, w) &= ST(\psi_1, w) \vee ST(\psi_2, w) \\
ST(\psi_1 \wedge \psi_2, w) &= ST(\psi_1, w) \wedge ST(\psi_2, w) \\
ST(\Diamond\psi, w) &= \exists v^W (R(w, v) \wedge ST(\psi, v)) \\
ST(\exists x^D \psi, w) &= \exists x^D ST(\psi, w).
\end{aligned}
$$

[1] All the results in this section extend to the case of multimodal languages and to the case of n-ary modalities like SINCE, UNTIL, etc.

5.2 Monodic Fusions for Fragments

Next two definitions identify the ingredients of our combined problems.

Definition 5.2.1. *Let $\mathcal{F}_{1M}$ be a class of Kripke frames*[2] *closed under disjoint unions and isomorphisms. We call* one-variable modal fragment *induced by $\mathcal{F}_{1M}$ the interpreted algebraic fragment $\Phi_{1M} = \langle \mathcal{L}_{1M}, T_{1M}, \mathcal{S}_{1M} \rangle$, where:*

(i) *$\mathcal{L}_{1M} := \mathcal{L}_{\emptyset}^{WR}$, where $\mathcal{L}_{\emptyset}$ is the empty one-sorted first-order signature;*[3]

(ii) *T_{1M} contains the terms which are $\beta\eta$-equivalent to terms of the kind $\{w^W, x^D \mid ST(\varphi, w)\}$, where φ is a modal predicate formula having x as the only (free or bound) variable;*

(iii) *$\mathcal{S}_{1M}$ is the class of the $\mathcal{L}_{1M}$-structures $\mathcal{A}$ such that $[\![D]\!]_{\mathcal{A}}$ is not empty and such that the Kripke frame $([\![W]\!]_{\mathcal{A}}, \mathcal{I}_{\mathcal{A}}(R))$ belongs to $\mathcal{F}_{1M}$.*

Definition 5.2.2. *For a monadically suitable i.a.f. $\Phi_e = \langle \mathcal{L}_e, T_e, \mathcal{S}_e \rangle$ (recall Definition 4.4.16), we define the i.a.f. $\Phi_e^W = \langle \mathcal{L}_e^W, T_e^W, \mathcal{S}_e^W \rangle$, as follows:*

(i) *T_e^W contains the terms $\beta\eta$-equivalent to terms of the kind $\{w^W, x^D \mid ST(\varphi, w)\}$, for $\{x^D \mid \varphi\} \in T_e$;*

(ii) *$\mathcal{S}_e^W$ contains the $\mathcal{L}_e^W$-structures $\mathcal{A}$ whose index structures $\mathcal{A}_w$ are all in $\mathcal{S}_e$.*

We first make sure that decidability is not lost when passing from Φ_e to Φ_e^W:

Lemma 5.2.3. *If constraint satisfiability in a monadically suitable fragment Φ_e is decidable, so is constraint satisfiability in Φ_e^W.*

Proof. Consider a Φ_e^W-constraint:

$$\bigwedge_{i=1}^{n}(\{w^W, x^D \mid ST(\varphi_i, w)\} = \{w^W, x^D \mid ST(\varphi_i', w)\}) \wedge$$
$$\wedge \bigwedge_{j=1}^{m}(\{w^W, x^D \mid ST(\psi_j, w)\} \neq \{w^W, x^D \mid ST(\psi_j', w)\});$$

we claim that is is satisfiable iff the Φ_e-constraints

$$\bigwedge_{i=1}^{n}(\{x \mid \varphi_i\} = \{x \mid \varphi_i'\}) \wedge \{x \mid \psi_j\} \neq \{x \mid \psi_j'\} \tag{5.1}$$

are all satisfiable. One side is clear (just look at index structures); for the other side, consider Φ_e-structures $\mathcal{A}_j$ $(j = 1, \ldots, m)$ satisfying (5.1); applying to their $\mathcal{L}_{\emptyset}$-reducts

[2] Recall that a Kripke frame is a nonempty set W endowed with a binary relation R.
[3] This means that $\mathcal{L}_{\emptyset}$ contains just the sort D and no other proper symbol.

a large disjoint I-copy, the supports of these structures get the same cardinality, which means that such supports can be renamed to make them just the same. Since extensibility of taking disjoint I-copies is part of the definition of a monadically suitable fragment, this gives the desired model $\mathcal{A}$ for the original constraint: to this aim, it is sufficient to take $[\![W]\!]_{\mathcal{A}} := \{1, \ldots, m\}$ and to let $\mathcal{A}$ be the $\mathcal{L}^W$-structure having the $\mathcal{A}_j$ so modified as index structures. □

Fix a one variable modal fragment Φ_{1M} and a first-order monadically suitable fragment Φ_e; we call *monodic fusion* of Φ_e and Φ_{1M} the combined fragment $\Phi_e^W \oplus \Phi_{1M}$.

Thus one may for instance combine packed guarded or two-variables fragments with one-variables modal fragments to get monodic fusions corresponding to the relevant cases analyzed in [94, 40].[4] In fact (modulo taking standard translation), in combined fragments like $\Phi_e^W \oplus \Phi_{1M}$ we can begin with formulae $\varphi[x]$ of Φ_e, apply to them a modal operator, then use the formulae so obtained to replace second-order variables in other formulae from Φ_e, etc. Fragments of the kind $\Phi_e^W \oplus \Phi_{1M}$ formalize the intuitive notion of a monodic modal fragment whose extensional component is Φ_e. Since Φ_{1M} is also interpreted, constraint satisfiability in $\Phi_e^W \oplus \Phi_{1M}$ is restricted to a desired specific class of modal frames/flows of time. The class of Kripke frames on which Φ_{1M} is based is taken to be closed under disjoint unions, but this assumption is not really relevant for relativized satisfiability (i.e. for conditional word problems): notice that a constraint containing at most one negative literal is satisfied in a disjoint union iff it is satisfied in a component, hence one can always close under disjoint unions the class of Kripke frames under consideration, without loss of generality, as far as relativized satisfiability is concerned. We shall prove the following general transfer result for monodic fusions:

Theorem 5.2.4. *If the one variable modal i.a.f. Φ_{1M} and the monadically suitable i.a.f. Φ_e have decidable constraint satisfiability problems, then their monodic fusion $\Phi_e^W \oplus \Phi_{1M}$ also has decidable constraint satisfiability problems.*

If we try to use directly Theorem 4.4.6 to prove this result, we find problems: these problems are basically due to the fact that for the modal component the identity of two individuals living on different worlds is an important information which is completely out of the control of the extensional component. The idea is to include 'trans-world' identification into the semantics as an explicit data, following the classical suggestion of counterpart theory (see [62]). Since we want our alternative models to provide a semantics which is equivalent to the constant domains semantics, the most elegant solution seems to be that of representing individual domains as descent data.

[4] We recall that the two-variable fragment is monadically suitable only if we take out identity; consequently decidability of the monodic modal two-variable fragments with identity is not covered by our results (and as a matter of fact, it is not true - see [40] for the relevant pointers to the literature).

5.3 An Alternative Translation

Fix a set W; we call *descent data* for W[5] a triple (E, p, ϑ) where $p : E \to W$ and $\vartheta : E \times W \to E$ are functions satisfying the following three requirements for all $e \in E, w, w_1, w_2 \in W$:[6]

$$p(\vartheta(e, w)) = w \tag{5.2}$$

$$\vartheta(e, p(e)) = e \tag{5.3}$$

$$\vartheta(\vartheta(e, w_1), w_2) = \vartheta(e, w_2). \tag{5.4}$$

To understand this definition from a modal point of view, we may think of E as the domain of all possible individuals and of p as the function that associates with an individual e the world $p(e)$ where e lives: in this sense, $\vartheta(e, w)$ has to be thought as the counterpart of e in the world w. Since we want to mimic a constant domain semantics, we would like from conditions (5.2)-(5.4) to follow that counterparts behave in such a way that fibers over W are in fact 'constant'. This is true (provided W is not empty, which means that the map $W \to \{*\}$ is onto) by a 'descent theorem' that holds for instance in exact categories: we shall explain (and check) what we need just for the very easy special case we are interested in.

We call *canonical descent data* the descent data of the kind $(D \times W, p_W, \vartheta_W)$ where D is a set, p_W is the projection on the second component and $\vartheta_W(\langle d, w' \rangle, w) = \langle d, w \rangle$. Now the descent theorem says the following: *every triple (E, p, ϑ) forming descent data for a non-empty set W is isomorphic to a canonical descent data.* This is proved as follows: take the equivalence relation over E given by $e_1 \simeq e_2$ iff (there is $w \in W$ such that $\vartheta(e_1, w) = e_2$). We now let D to be the quotient set of E under $\simeq$. To check that the canonical descent data $(D \times W, p_W, \vartheta_W)$ are isomorphic to (E, p, ϑ), consider the bijective function $h : E \to D \times W$ associating with $e \in E$ the pair $h(e) := \langle [e], p(e) \rangle$ and observe that this bijection commutes with p and ϑ (in the sense that we have $p_W(h(e)) = p(e)$ and $\vartheta_W(h(e), w) = h(\vartheta(e, w))$).

The idea is now that of using descent data to define a new translation for modal first order formulae: this translation (corresponding to an alternative equivalent 'descent' semantics) has the first advantage that we do not need to modify the original first order signature $\mathcal{L}$ by altering the types of the relational symbols in it (as it happened for the definition of $\mathcal{L}^W$ in the case of the standard translation).

[5] One should better say 'descent data for the unique map $W \to \{*\}$'. Descent theory is a powerful and deep theory in pure mathematics (see, e.g., [52, 51]).

[6] From now on, we shall use the letters $e, w, \ldots$ both for (sorted) variables in a logical language and for concrete elements of given structures (and for the names of these concrete elements in expanded languages, like in Subsection 4.1.4): the context will carefully clarify, case by case, the intended use.

Let $\mathcal{L}$ be a first-order relational one-sorted signature; the signature $\mathcal{L}^d$ is obtained by changing the name of the unique sort of $\mathcal{L}$ from D to E and then by adding to $\mathcal{L}$ a new sort W, a binary relation $R : WW \to \Omega$ and function symbols $p : E \to W$, $\vartheta : EW \to E$.

For a modal $\mathcal{L}$-formula $\varphi[e_1, \ldots, e_n]$ (here $e_1, \ldots, e_n$ are just individual variables of the unique sort D of $\mathcal{L}$) and for a variable $w : W$, we define the (non modal) $\mathcal{L}^d$-formula $DT(\varphi, w)$ as follows:

$$\begin{aligned}
DT(\top, w) &= \top \\
DT(\bot, w) &= \bot \\
DT(X(e_i), w) &= X(e_i) \\
DT(P(e_{i_1}, \ldots, e_{i_m}), w) &= P(e_{i_1}, \ldots, e_{i_m}) \\
DT(\neg\psi, w) &= \neg DT(\psi, w) \\
DT(\psi_1 \vee \psi_2, w) &= DT(\psi_1, w) \vee DT(\psi_2, w) \\
DT(\psi_1 \wedge \psi_2, w) &= DT(\psi_1, w) \wedge DT(\psi_2, w) \\
DT(\Diamond\psi, w) &= \exists v^W (R(w, v) \wedge DT(\psi, w)[\vartheta(e_1, v), \ldots, \vartheta(e_n, v), v]) \\
DT(\exists e^D \psi, w) &= \exists e^E (p(e) = w \wedge DT(\psi, w)),
\end{aligned}$$

where, according to our usual conventions, the notation $DT(\psi, w)[\vartheta(e_1, v), \ldots, \vartheta(e_n, v), v]$ means the formula obtained from $DT(\psi, w)$ by replacing w by v and the free variables e_i by the $\mathcal{L}^d$-terms $\vartheta(e_i, v)$. In the following, we shall use notations like $DT(\psi[u_1, \ldots, u_n], t)$, where $u_1, \ldots, u_n$ are $\mathcal{L}^d$-term of type E and t is an $\mathcal{L}^d$-term of type W, to mean the formula obtained from $DT(\psi[x_1, \ldots, x_n], w)$ by applying it the substitution $x_1 \mapsto u_1, \ldots, x_n \mapsto u_n, w \mapsto t$.

We now *reformulate* the notions of a one variable modal i.a.f. and of a monadically suitable i.a.f. into this equivalent alternative descent semantics. We shall be interested in the terms of the kind $\{e \mid DT(\varphi[e], p(e))\}$, where φ is a first order modal formula. However, these terms are not precisely closed under substitutions for second-order variables: to get closure under substitution, some equational reasoning (partially based on the descent equations) would be needed, in addition to $\beta\eta$-equivalence. Since we are not precisely interested here in investigating the related technical details, we prefer to close under substitution the set of terms we need to build our i.a.f.'s. Such an operation of taking the smallest substitution closed set $cl(T)$ of $\mathcal{L}^d$-terms extending a given set T of $\mathcal{L}^d$-terms is rather harmless for our purposes: for instance, in case T contains the relevant variables (as it will always be the case in this Section), we can use a mechanism similar to the purification procedure explained in Subsection 4.3.1 and assume without loss of generality that the $cl(T)$-terms appearing in constraints are in fact terms from the original set T.

We first translate one-variable modal fragments into descent semantics:

Definition 5.3.1. *Let $\Phi_{1M} = \langle \mathcal{L}_{1M}, T_{1M}, \mathcal{S}_{1M} \rangle$ be the one-variable modal fragment induced by the frame class $\mathcal{F}_{1M}$. The i.a.f. $\Phi^d_{1M} = \langle \mathcal{L}^d_{1M}, T^d_{1M}, \mathcal{S}^d_{1M} \rangle$ is so defined:*

(i) *T^d_{1M} is the substitution closure of the set of the terms of the kind $\{e^E \mid DT(\varphi[e], p(e))\}$, where $\varphi[x]$ is a one-variable modal predicate formula;*

(ii) *$\mathcal{S}^d_{1M}$ contains the $\mathcal{L}^d_{1M}$-structures $\mathcal{A}$ such that $[\![E]\!]_{\mathcal{A}}$, $\mathcal{I}_{\mathcal{A}}(p)$, $\mathcal{I}_{\mathcal{A}}(\vartheta)$ satisfy the descent data equations* (5.2)-(5.4)*, $\mathcal{I}_{\mathcal{A}}(p)$ is surjective and the Kripke frame $([\![W]\!]_{\mathcal{A}}, \mathcal{I}_{\mathcal{A}}(R))$ belongs to $\mathcal{F}_{1M}$.*

The requirement of surjectivity of $\mathcal{I}_{\mathcal{A}}(p)$ corresponds to the fact that the domain of the constant domain semantics is not empty and, on the basis of the descent equations, it is equivalent to the fact that $[\![E]\!]_{\mathcal{A}}$ is not empty. Suppose now we are given a monadically suitable i.a.f. $\Phi_e = \langle \mathcal{L}_e, T_e, \mathcal{S}_e \rangle$; this is a purely extensional fragment, hence the DT-translations of the terms in it do not contain the descent multiplication ϑ: this is why we can translate Φ_e into the signature $\mathcal{L}^{-d}_e$ (the latter is like $\mathcal{L}^d_e$ except that ϑ and R are omitted). The related definition is the following one:

Definition 5.3.2. *Let $\Phi_e = \langle \mathcal{L}_e, T_e, \mathcal{S}_e \rangle$ be a monadically suitable i.a.f.. We let Φ^{-d}_e to be the i.a.f. $\langle \mathcal{L}^{-d}_e, T^{-d}_e, \mathcal{S}^{-d}_e \rangle$, where:*

(i) *T^{-d}_e is the substitution closure of the set of the terms $\{e^E \mid DT(\varphi[e], p(e))\}$ such that $\{x \mid \varphi[x]\} \in T_e$;*

(ii) *$\mathcal{S}^{-d}_e$ contains the $\mathcal{L}^{-d}_e$-structures $\mathcal{A}$ which are isomorphic to $[\![W]\!]_{\mathcal{A}}$-indexed disjoint unions of structures $\mathcal{A}_w$ from $\mathcal{S}_e$ (these $\mathcal{A}_w$, varying $w \in [\![W]\!]_{\mathcal{A}}$, are called the* fiber components *of $\mathcal{A}$).*

In full details, condition (ii) from Definition 5.3.2 means the following. Notice first that it makes sense to consider $\mathcal{L}_e$ as a subsignature of $\mathcal{L}^{-d}_e$ (modulo the change of name of the unique sort of $\mathcal{L}_e$ from D to E). Thus in (ii) we are asking that $\mathcal{A} \in \mathcal{S}^{-d}_e$ if and only if (up to isomorphism): (a) the $\mathcal{L}_e$-reduct of $\mathcal{A}$ is the disjoint union $\sum_{w \in [\![W]\!]_{\mathcal{A}}} \mathcal{A}_w$ of some $\mathcal{L}_e$-structures $\mathcal{A}_w$, varying $w \in [\![W]\!]_{\mathcal{A}}$; (b) the $\mathcal{L}_e$-structures $\mathcal{A}_w$ all belong to $\mathcal{S}_e$; (c) $\mathcal{I}_{\mathcal{A}}(p)$ is given by $\mathcal{I}_{\mathcal{A}}(p)(d, w) = w$, for every $(d, w) \in [\![E]\!]_{\mathcal{A}}$.

The definition of $\mathcal{I}_{\mathcal{A}}(p)$ is well set because, by (a), the elements of $[\![E]\!]_{\mathcal{A}}$ can be represented as pairs (d, w), where $w \in [\![W]\!]_{\mathcal{A}}$ and d is from the support of $\mathcal{A}_w$; moreover, since the supports of the $\mathcal{A}_w$'s are not empty, $\mathcal{I}_{\mathcal{A}}(p)$ is surjective. In other words, this means that *the following quite simple schema builds (up to isomorphism) precisely the structures in $\mathcal{S}^{-d}_e$*: take a family $\{\mathcal{A}_w \mid w \in I\}$ of structures from $\mathcal{S}_e$, interpret W as the index set I, E as the disjoint union of the supports of the $\mathcal{A}_w$, all predicates as in

standard disjoint union of relational structures and p as the functions associating with an element the index of its support.

We now make an important observation, to be fixed in Lemma 5.3.3 below. If the $\mathcal{L}_e^{-d}$-structure $\mathcal{A}$ is isomorphic to the $[\![W]\!]_{\mathcal{A}}$-indexed disjoint union of the $\mathcal{L}_e$-structures $\mathcal{A}_w$, *satisfiability in $\mathcal{A}$ of Φ_e^{-d}-constraints is fiberwise*, in the following sense. For every $w \in [\![W]\!]_{\mathcal{A}}$, for every $d_1, \ldots, d_n$ in the support of $\mathcal{A}_w$ and for every (non modal) $\mathcal{L}_e$-formula $\varphi[x_1, \ldots, x_n]$, we have

$$\mathcal{A} \models DT(\varphi[(d_1, w), \ldots, (d_n, w)], w) \quad \text{iff} \quad \mathcal{A}_w \models \varphi[d_1, \ldots, d_n] \tag{5.5}$$

(a trivial induction is sufficient to establish this fact). Notice that (5.5) holds also in case $\mathcal{L}_e$ is expanded by free constants $\underline{c}$ whose type is a Φ_e-type (there is only one Φ_e-type, namely the type of the subsets of the domain). Since $[\![E]\!]_{\mathcal{A}}$ is the disjoint union of the supports of the $\mathcal{A}_w$, the interpretation of the $\underline{c}$'s in $\mathcal{A}$ is obtained by gluing their restrictions to the supports of the $\mathcal{A}_w$. Thus, if we consider $\mathcal{A}$ as a $\mathcal{L}_e(\underline{c})$-structure, it is still isomorphic to the $[\![W]\!]_{\mathcal{A}}$-indexed disjoint unions of $\mathcal{L}_e(\underline{c})$-structures (that we still call $\mathcal{A}_w$). Now a $\Phi_e^{-d}(\underline{c})$-closed constraint is formed by positive literals

$$\{e^E \mid DT(\varphi_i[e], p(e))\} = \{e^E \mid DT(\varphi_i'[e], p(e))\} \quad (i = 1, \ldots, n)$$

and by negative literals

$$\{e^E \mid DT(\psi_j[e], p(e))\} \neq \{e^E \mid DT(\psi_j'[e], p(e))\} \quad (j = 1, \ldots, m);$$

according to (5.5), such a constraint is satisfied in $\mathcal{A}$ iff (i) for every $w \in [\![W]\!]_{\mathcal{A}}$, we have $\mathcal{A}_w \models \bigwedge_i (\{x \mid \varphi_i[x]\} = \{x \mid \varphi_i'[x]\})$[7] and (ii) for every j there is $w_j \in [\![W]\!]_{\mathcal{A}}$ such that $\mathcal{A}_{w_j} \models \{x \mid \psi_j[x]\} \neq \{x \mid \psi_j'[x]\}$. Of course, the same observation applies to generalized constraints too.

To sum up, we introduce the following notion: if a generalized $\Phi_e^{-d}(\underline{c})$-closed constraint Γ is given, a $\Phi_e(\underline{c})$-positive literal of Γ is a positive literal of the form $\{x \mid \varphi[x]\} = \{x \mid \varphi'[x]\}$ such that $\{e^E \mid DT(\varphi[e], p(e))\} = \{e^E \mid DT(\varphi'[e], p(e))\} \in \Gamma$ (the definition of a $\Phi_e(\underline{c})$-negative literal of Γ is analogous). The above observation now reads as:

Lemma 5.3.3. *Let $\Phi_e = \langle \mathcal{L}_e, T_e, \mathcal{S}_e \rangle$ be a monadically suitable i.a.f. and let Φ_e^{-d} be the i.a.f. of Definition 5.3.2. Given free constants $\underline{c}$, suppose that the $\mathcal{L}_e^{-d}(\underline{c})$-structure $\mathcal{A}$ is isomorphic to the $[\![W]\!]_{\mathcal{A}}$-indexed disjoint union of the $\mathcal{L}_e(\underline{c})$-structures $\mathcal{A}_w \in \mathcal{S}_e(\underline{c})$. Now $\mathcal{A}$ satisfies a generalized $\Phi_e^{-d}(\underline{c})$-closed constraint Γ iff the $\Phi_e(\underline{c})$-positive literals of Γ hold*

[7] In more detail: we have that $\mathcal{A} \models \{e^E \mid DT(\varphi_i[e], p(e))\} = \{e^E \mid DT(\varphi_i'[e], p(e))\}$ iff for every $w \in [\![W]\!]_{\mathcal{A}}$ and for every d in the support of $\mathcal{A}_w$, we have $\mathcal{A} \models DT(\varphi[(d, w)] \leftrightarrow \varphi'[(d, w)], w)$. By (5.5), this is the same as $\mathcal{A}_w \models \varphi[d] \leftrightarrow \varphi'[d]$ for all w and d in $\mathcal{A}_w$, which means that the $\Phi_e(\underline{c})$-atom $\{x \mid \varphi_i[x]\} = \{x \mid \varphi_i'[x]\}$ is true in all fiber structures $\mathcal{A}_w$.

in all fiber components $\mathcal{A}_w$ and every $\Phi_e(\underline{c})$-negative literal of Γ holds in at least one fiber component $\mathcal{A}_w$.

Remark 5.3.4. The following strong consequence of the above Lemma will be repeatedly used in the following: we know that a structure $\mathcal{A} \in \mathcal{S}_e^{-d}(\underline{c})$ is isomorphic to a $[\![W]\!]_{\mathcal{A}}$-indexed disjoint union of structures $\mathcal{A}_w$ from $\mathcal{S}_e(\underline{c})$. Suppose now that we replace, in such a disjoint union, the structures $\mathcal{A}_w$ by some structures $\mathcal{A}'_w \in \mathcal{S}_e(\underline{c})$ such that $\mathcal{A}_w \equiv_{\Phi_e(\underline{c})} \mathcal{A}'_w$: call $\mathcal{A}'$ the $[\![W]\!]_{\mathcal{A}'}$-indexed structure obtained in this way (here the interpretation of W has not changed, namely we have $[\![W]\!]_{\mathcal{A}'} := [\![W]\!]_{\mathcal{A}}$). Clearly $\mathcal{A}' \in \mathcal{S}_e^{-d}(\underline{c})$ and Lemma 5.3.3 implies that we have $\mathcal{A} \equiv_{\Phi_e^{-d}(\underline{c})} \mathcal{A}'$: thus *$\Phi_e^{-d}(\underline{c})$-equivalence is preserved, whenever we apply fiberwise a $\Phi_e(\underline{c})$-equivalence preserving construction.*

5.4 Proof of the Monodic Decidability Transfer Result

Let now Φ_{1M} be a one-variable modal i.a.f. and Φ_e be a monodically suitable i.a.f.. Our plan is the following: *we first check that $\Phi_e^W \oplus \Phi_{1M}$ can be equivalently replaced by $\Phi_e^{-d} \oplus \Phi_{1M}^d$ and then we apply Theorem 4.4.6 to the latter.*

The first part of the plan just consists of unwinding the definitions we gave. In fact Φ_e^W and Φ_e^{-d} (as well as Φ_{1M} versus Φ_{1M}^d, and $\Phi_e^W \oplus \Phi_{1M}$ versus $\Phi_e^{-d} \oplus \Phi_{1M}^d$), are basically the same i.a.f.; however for our purposes the statement of the following lemma is sufficient:

Lemma 5.4.1. *Satisfiability of pure constraints in $\Phi_e^W \oplus \Phi_{1M}$ can be reduced to satisfiability of pure constraints in $\Phi_e^{-d} \oplus \Phi_{1M}^d$, and vice versa. Constraint satisfiability for Φ_e^W (resp. for Φ_{1M}) can also be reduced to constraint satisfiability for Φ_e^{-d} (resp. for Φ_{1M}^d), and vice versa.*

Proof. A pure $\Phi_e^W \oplus \Phi_{1M}$-constraint contains equations and inequations among terms of the kind $\{w^W, x^D \mid ST(\varphi, w)\}$, where $\varphi[x]$ is either a one-variable modal predicate formula or it is such that $\{x \mid \varphi\}$ is a Φ_e-term. On the other hand, a pure $\Phi_e^{-d} \oplus \Phi_{1M}^d$-constraint contains equations and inequations among terms of the kind $\{e^E \mid DT(\varphi[e], p(e))\}$, where $\varphi[x]$ is either a one-variable modal predicate formula or it is such that $\{x \mid \varphi\}$ is a Φ_e-term. Hence it is clear how to convert a pure $\Phi_e^W \oplus \Phi_{1M}$-constraint Γ into a pure $\Phi_e^{-d} \oplus \Phi_{1M}^d$-constraint Γ^d, and vice versa: it remains to show the equisatisfiability.

To any $\Phi_e^W \oplus \Phi_{1M}$-structure $\mathcal{A}$ we can associate a $\Phi_e^{-d} \oplus \Phi_{1M}^d$-structure $\mathcal{A}^d$ as follows: we let $[\![W]\!]_{\mathcal{A}^d} := [\![W]\!]_{\mathcal{A}}$ and $\mathcal{I}_{\mathcal{A}^d}(R) := \mathcal{I}_{\mathcal{A}}(R)$; the symbols E, p, ϑ are interpreted as canonical descent data for $[\![W]\!]_{\mathcal{A}}$ relatively to $[\![D]\!]_{\mathcal{A}}$. Thus by definition $[\![E]\!]_{\mathcal{A}^d} = [\![D]\!]_{\mathcal{A}} \times [\![W]\!]_{\mathcal{A}}$, so it makes sense to put $\mathcal{I}_{\mathcal{A}^d}(P) := \{\langle (d_1, w), \ldots, (d_n, w) \rangle \mid \langle d_1, \ldots, d_n, w \rangle \in \mathcal{I}_{\mathcal{A}}(P)\}$ for every predicate symbol P having type $D^n \to \Omega$ in $\mathcal{L}$.[8] Actually every

[8] Recall that, in correspondence to a $P : D^n \to \Omega$, the signature $\mathcal{L}^{WR}$ contains $P : D^n W \to \Omega$, whereas $\mathcal{L}^d$ contains $P : E^n \to \Omega$.

$\Phi_e^{-d} \oplus \Phi_{1M}^d$-structure is isomorphic to one of the kind $\mathcal{A}^d$, for some $\Phi_e^W \oplus \Phi_{1M}$-structure $\mathcal{A}$: in fact, the $\mathcal{L}_e^{-d}$-reduct of a $\Phi_e^{-d} \oplus \Phi_{1M}^d$-structure $\mathcal{B}$ is a $[\![W]\!]_{\mathcal{B}}$-disjoint union of $\mathcal{L}_e$-structures $\mathcal{B}_w$ (see Definition 5.3.2(ii)), whereas *by the descent theorem* we can assume that, up to isomorphism, the descent data in $\mathcal{B}$ are canonical. The combination of these two facts means that $\mathcal{B} \simeq \mathcal{A}^d$ for some $\mathcal{A}$. To explain this fact in full details, we can reason as follows: since descent data in $\mathcal{B}$ are canonical, we can assume that $[\![E]\!]_{\mathcal{B}} = \tilde{D} \times [\![W]\!]_{\mathcal{B}}$, for some $\tilde{D}$ and that the descent symbols p, ϑ are interpreted in the canonical way. We define $\mathcal{A}$ by taking $[\![W]\!]_{\mathcal{A}} := [\![W]\!]_{\mathcal{B}}$, $\mathcal{I}_{\mathcal{A}}(R) := \mathcal{I}_{\mathcal{B}}(R)$, $[\![D]\!]_{\mathcal{A}} := \tilde{D}$ and $\mathcal{I}_{\mathcal{A}}(P) := \{\langle d_1, \ldots, d_n, w\rangle \mid \langle (d_1, w), \ldots, (d_n, w)\rangle \in \mathcal{I}_{\mathcal{B}}(P)\}$, for every n-ary predicate symbol P (the index structures of $\mathcal{A}$ are now isomorphic to the corresponding fiber components of $\mathcal{B}$, hence $\mathcal{A} \in \mathcal{S}_e^W$). Since the $\mathcal{L}_e$-reduct of $\mathcal{B}$ is the disjoint union of the $\mathcal{B}_w$, it turns out that $\mathcal{B} \simeq \mathcal{A}^d$.

That Γ is satisfied in $\mathcal{A}$ iff Γ^d is satisfied in $\mathcal{A}^d$ is straightforward: to see it, just check by induction that, for every modal formula $\varphi[x_1, \ldots, x_n]$,

$$\mathcal{A} \models ST(\varphi[d_1, \ldots, d_n], w) \quad \text{iff} \quad \mathcal{A}^d \models DT(\varphi[(d_1, w), \ldots, (d_n, w)], w) \tag{5.6}$$

holds for all $d_1 \ldots, d_n \in [\![D]\!]_{\mathcal{A}}$ and all $w \in [\![W]\!]_{\mathcal{A}}$ under the corresponding assignments to the second-order variables. By 'corresponding' we obviously mean here that (d, w) belongs to the subset assigned to X in $\mathcal{A}^d$ iff (d, w) belongs to the subset assigned to X in $\mathcal{A}$. To understand this and to check inductively the above condition, recall that descent data in $\mathcal{A}^d$ are canonical (thus, e.g. elements of $[\![E]\!]_{\mathcal{A}^d}$ are pairs (d, w), the symbol p is interpreted as the second projection, etc.).

The statement for the fragments Φ_{1M} and Φ_{1M}^d is shown in the same way. For fragments Φ_e^W and Φ_e^{-d}, we need a preliminary observation, because the supports of the fiber components in a Φ_e^{-d}-structure may not be isomorphic (there are no full descent data here). The observation is the following: Φ_e is monadically suitable, so by taking suitably large disjoint I-copies we can expand the cardinalities of the non-empty supports in the fiber components of a Φ_e^{-d}-structure and make such supports to coincide, up to renaming of their elements (see Remark 5.3.4). At this point, canonical descent data exists and the above argument based on (5.6) applies. □

In view of Lemma 5.4.1, to complete the proof of Theorem 5.2.4 it is sufficient now to show the following

Proposition 5.4.2. *If the one variable modal i.a.f. Φ_{1M} and the monadically suitable i.a.f. Φ_e have decidable constraint satisfiability problems, then the combined fragment $\Phi_e^{-d} \oplus \Phi_{1M}^d$ also has decidable constraint satisfiability problems.*

Proof. Since by Lemmas 5.2.3, 5.4.1 the component fragments Φ_e^{-d} and Φ_{1M}^d have decid-

able constraint satisfiability problems, we can try to apply Theorem 4.4.6, by checking the remaining conditions.

Notice that both i.a.f.'s have $E \to \Omega$ as the only type for their terms and that the shared signature $\mathcal{L}_0$ contains the two sorts E, W and the function symbol p of type $E \to W$. According to the definitions of a one variable modal and of a monadically suitable fragment, the set of terms T_0 in the shared fragment $\Phi_0 = \langle \mathcal{L}_0, T_0, \mathcal{S}_0 \rangle$ contains the terms of the kind $\{e^E \mid DT(\varphi[e], p(e))\}$, where $\varphi[x]$ is a one-variable (non modal) first-order formula in the empty one-sorted signature $\mathcal{L}_\emptyset$: this implies, in particular, that Φ_0 *is effectively locally finite* (only second-order variables for subsets, Boolean connectives and the quantifier $\forall x, \exists x$ can be used to build φ). We do not have complete information about the $\mathcal{L}_0$-structures $\mathcal{A} \in \mathcal{S}_0$, but we know that $\mathcal{I}_\mathcal{A}(p)$ is always surjective in them (because the interpretation of p is a surjective function in structures coming from both $\mathcal{S}^d_{1M}$ and $\mathcal{S}^{-d}_e$).

We need to identify suitable structural operations on Φ_0 to apply Theorem 4.4.6,[9] but first we study invariants for $\mathcal{L}_0(\underline{c}_0)$-structures. To this aim, suppose we are given free constants $\underline{c}_0 := \{P_1, \ldots, P_n\}$ (all having type $E \to \Omega$, which is the only Φ_0-type) and a $\mathcal{L}_0(\underline{c}_0)$-structure $\mathcal{A}$. For $w \in [\![W]\!]_\mathcal{A}$, we denote by $\mathcal{A}_w$ the fiber component over w: this is the $\mathcal{L}_\emptyset(\underline{c}_0)$-structure whose support is given by the $e \in [\![E]\!]_\mathcal{A}$ such that $\mathcal{I}_\mathcal{A}(p)(e) = w$ (in $\mathcal{A}_w$ the predicates P_j are interpreted by taking the restriction of $\mathcal{I}_\mathcal{A}(P_j)$ to the support of $\mathcal{A}_w$).

Consider, as in the proof of Proposition 4.4.12, the boolean combinations of the form $\varepsilon(e) = Q_1(e) \wedge \cdots \wedge Q_n(e)$ where $Q_j \equiv P_j$ or $Q_j \equiv \neg P_j$. With each $w \in [\![W]\!]_\mathcal{A}$ we associate the 2^n cardinal invariants of Proposition 4.4.12 for the fiber component $\mathcal{A}_w$. That is: for $\varepsilon = 1, \ldots, 2^n$, we let $\kappa_\varepsilon(w)$ to be the cardinality of the set of the $e \in [\![E]\!]_\mathcal{A}$ living in the support of the w-fiber component such that $\mathcal{A}_w \models \varepsilon(e)$; also, we let $\mu(w)$ to be equal to the set of the ε such that $\kappa_\varepsilon(w) > 1$. Finally, we let $\mu(\mathcal{A})$ to be the set of sets formed by the $\mu(w)$, varying $w \in [\![W]\!]_\mathcal{A}$. Notice that the fact that $\mu(\mathcal{A}) = J$ is equivalent to

$$\begin{aligned} \mathcal{A} \models \quad & \textstyle\bigwedge_{S \in J} \exists w \, [\bigwedge_{\varepsilon \in S} \exists e \, (p(e) = w \wedge \varepsilon(e)) \wedge \bigwedge_{\varepsilon \notin S} \neg \exists e \, (p(e) = w \wedge \varepsilon(e))] \wedge \\ & \wedge \textstyle\bigwedge_{S \notin J} \neg \exists w \, [\bigwedge_{\varepsilon \in S} \exists e \, (p(e) = w \wedge \varepsilon(e)) \wedge \bigwedge_{\varepsilon \notin S} \neg \exists e \, (p(e) = w \wedge \varepsilon(e))] \end{aligned}$$

i.e. to

$$\begin{aligned} \mathcal{A} \models \quad & \textstyle\bigwedge_{S \in J} \exists w \, DT[\bigwedge_{\varepsilon \in S} \exists x \, \varepsilon(x) \wedge \bigwedge_{\varepsilon \notin S} \neg \exists x \, \varepsilon(x), w] \wedge \\ & \wedge \textstyle\bigwedge_{S \notin J} \neg \exists w \, DT[\bigwedge_{\varepsilon \in S} \exists x \, \varepsilon(x) \wedge \bigwedge_{\varepsilon \notin S} \neg \exists x \, \varepsilon(x), w] \end{aligned}$$

[9] In the statement of Theorem 4.4.6, we take $\Phi_0^\star$ equal to Φ_0, so only condition 4.4.6 (4) has not yet been checked.

Since $\mathcal{I}_{\mathcal{A}}(p)$ is surjective, this is the same as

$$\begin{aligned}\mathcal{A} \models \quad & \textstyle\bigwedge_{S\in J} \exists e\, DT[\bigwedge_{\varepsilon\in S} \exists x\, \varepsilon(x) \wedge \bigwedge_{\varepsilon\notin S} \neg\exists x\, \varepsilon(x), p(e)] \wedge \\ & \wedge \textstyle\bigwedge_{S\notin J} \neg\exists e\, DT[\bigwedge_{\varepsilon\in S} \exists x\, \varepsilon(x) \wedge \bigwedge_{\varepsilon\notin S} \neg\exists x\, \varepsilon(x), p(e)];\end{aligned}$$

the latter simply says that the $\Phi_0(\underline{c}_0)$-closed constraint

$$\begin{aligned}&\textstyle\bigwedge_{S\in J}(\{e \mid DT[\bigwedge_{\varepsilon\in S} \exists x\, \varepsilon(x) \wedge \bigwedge_{\varepsilon\notin S} \neg\exists x\, \varepsilon(x), p(e)]\} \neq \{e \mid DT(\bot, p(e))\}) \wedge \\ &\wedge \textstyle\bigwedge_{S\notin J}(\{e \mid DT[\bigwedge_{\varepsilon\in S} \exists x\, \varepsilon(x) \wedge \bigwedge_{\varepsilon\notin S} \neg\exists x\, \varepsilon(x), p(e)]\} = \{e \mid DT(\bot, p(e))\})\end{aligned}$$

is satisfied in $\mathcal{A}(\underline{c}_0)$.

Suppose now that the $\mathcal{L}_0(\underline{c}_0)$-structures $\mathcal{A}_1$ and $\mathcal{A}_2$ are $\Phi_0(\underline{c}_0)$-equivalent: then we have $\mu(\mathcal{A}_1) = \mu(\mathcal{A}_2)$, as explained above. *We will make $\mathcal{A}_1$ and $\mathcal{A}_2$ $\Phi_0(\underline{c}_0)$-isomorphic in two steps: each step needs a Φ_0-structural operation that will be proved to be extensible both to Φ_e^{-d} and to Φ_{1M}^d* (notice that the composition of extensible structural operations is extensible).

The *first structural operation* is taking disjoint I-copies $\sum_I$: notice that taking disjoint I-copies operation applies to structures over a multi-sorted first-order relational language having also *unary* function symbols (like our p).[10] Consequently, the operation applies to the language $\mathcal{L}_e^{-d}$ (and, in particular, to $\mathcal{L}_0$); it is a Φ_e^{-d}-structural operation (hence its restriction to $\mathcal{L}_0$ is in particular Φ_e^{-d}-extensible), because the fiber components in $\sum_I \mathcal{A}$ are the same as the fiber components in $\mathcal{A}$ (we just made more copies of each fiber component) and Lemma 5.3.3 applies, guaranteeing that $\mathcal{A} \equiv_{\Phi_e^{-d}(\underline{c}_0)} \sum_I \mathcal{A}$.

The operation $\sum_I$ is Φ_{1M}^d-extensible: to see this, first observe that the frame class $\mathcal{F}_{1M}$ defining Φ_{1M} is closed under disjoint unions. If $\mathcal{A}$ is now a structure from $\mathcal{S}_{1M}(\underline{c}_0)$, in order to make $\sum_I \mathcal{A}$ a $\mathcal{S}_{1M}(\underline{c}_0)$-structure as well, we only need to introduce a descent multiplication on $\sum_I \mathcal{A}$. This is done as follows: put $\mathcal{I}_{\sum_I \mathcal{A}}(\vartheta)((e,i),(w,j)) := (\mathcal{I}_{\mathcal{A}}(\vartheta)(e,w), j)$. The descent equations (5.2)-(5.4) can be checked in a straightforward way. In addition, to show that truth of closed $\Phi_{1M}^d(\underline{c}_0)$-atoms is preserved, one simply check by induction that, for every modal one-variable $\mathcal{L}_\emptyset$-formula $\varphi[x]$ (in which second-order variables have been replaced by the free constants $\underline{c}_0$), for every index $i \in I$, for every $e \in [\![E]\!]_{\mathcal{A}}$, for every $w \in [\![W]\!]_{\mathcal{A}}$, if $\mathcal{I}_{\mathcal{A}}(p)(e) = w$, then we have

$$\sum_I \mathcal{A} \models DT(\varphi[(e,i)], (w,i)) \quad \text{iff} \quad \mathcal{A} \models DT(\varphi[e], w).$$

If we apply $\sum_I$ for sufficient large I to our $\Phi_0(\underline{c}_0)$-equivalent structures $\mathcal{A}_1$ and $\mathcal{A}_2$,

[10] This is because one can put, for unary p and $i \in I$, $\mathcal{I}_{\sum_I \mathcal{A}}(p)(e,i) := (\mathcal{I}_{\mathcal{A}}(p)(e), i)$.

then we get $\mathcal{L}_0(\underline{c}_0)$-structures $\mathcal{A}'_1$ and $\mathcal{A}'_2$[11] such that for every S the cardinality of the $w_1 \in [\![W]\!]_{\mathcal{A}'_1}$ with $\mu(w_1) = S$ is the same as the cardinality of the $w_2 \in [\![W]\!]_{\mathcal{A}'_2}$ with $\mu(w_2) = S$. Thus we have a bijection $\iota_W : [\![W]\!]_{\mathcal{A}'_1} \to [\![W]\!]_{\mathcal{A}'_2}$, preserving the invariant μ. To make $\mathcal{A}'_1$ and $\mathcal{A}'_2$ $\Phi_0(\underline{c}_0)$-isomorphic, we need the fiber components over w and $\iota(w)$ to be isomorphic (for all w): to that aim, since $\mu(w)$ is equal to $\mu(\iota(w))$, it is sufficient to apply 'fiberwise' the argument of Proposition 4.4.12, provided we are allowed to take suitably large disjoint I-copies *of the sets* $[\![E]\!]_{\mathcal{A}'_1}$ *and* $[\![E]\!]_{\mathcal{A}'_2}$ *only.* This will be achieved through the *second structural operation* we are going to introduce.

Let $\mathcal{A}$ be a $\mathcal{L}_0(\underline{c}_0)$-structure and let I be a non-empty set of indexes; we call $\sum_I^E(\mathcal{A})$ the $\mathcal{L}_0(\underline{c})$-structure so defined: (i) we interpret the sort W as in $\mathcal{A}$ and the sort E as the disjoint union $\sum_I [\![E]\!]_{\mathcal{A}}$; (ii) we interpret the symbol p as the function mapping (e, i) to $\mathcal{I}_{\mathcal{A}}(p)(e)$; (iii) we interpret the unary predicate $P \in \underline{c}_0$ as the set of all (e, i) such that $e \in \mathcal{I}_{\mathcal{A}}(P)$. That $\sum_I^E(\mathcal{A})$ is $\Phi_0(\underline{c}_0)$-equivalent to $\mathcal{A}$ will be checked below (directly for the stronger cases of $\Phi^d_{1M}(\underline{c}_0)$- and of $\Phi_e^{-d}(\underline{c}_0)$-equivalence).

The operation $\sum_I^E$ is Φ_e^{-d}-extensible: this comes from Remark 5.3.4 and from the fact that Φ_e is a monadically suitable fragment (so that taking disjoint copies is Φ_e-extensible at each fiber component separately).

Finally, the operation $\sum_I^E$ is Φ^d_{1M}-extensible: we can interpret the descent multiplication symbol ϑ in $\sum_I^E(\mathcal{A})$ into the function associating $(\mathcal{I}_{\mathcal{A}}(\vartheta)(e, w), i)$ with the pair $((e, i), w)$ (equations (5.2)-(5.4) are easily checked).[12] Of course, the accessibility relation R is interpreted in $\sum_I^E(\mathcal{A})$ as it was interpreted in $\mathcal{A}$. Thus it remains to prove that $\mathcal{A} \equiv_{\Phi^d_{1M}(\underline{c}_0)} \sum_I^E(\mathcal{A})$: to this aim, it is sufficient to check inductively that for every one-variable modal formula $\varphi[x]$, for $w \in [\![W]\!]_{\mathcal{A}}, e \in [\![E]\!]_{\mathcal{A}}$ (such that $\mathcal{I}_{\mathcal{A}}(p)(e) = w$) and for $i \in I$, we have that

$$\sum_I^E(\mathcal{A}) \models DT(\varphi[(e, i)], w) \quad \text{iff} \quad \mathcal{A} \models DT(\varphi[e], w).$$

This completes the proof because, as already pointed out, for sufficiently large I the structures $\sum_I^E(\mathcal{A}'_1)$ and $\sum_I^E(\mathcal{A}'_2)$ are now $\Phi_0(\underline{c}_0)$-isomorphic. □

[11] $\mathcal{A}'_1$ and $\mathcal{A}'_2$ are still $\Phi_0(\underline{c}_0)$-equivalent (in fact, we observed that, more generally, truth of closed $\Phi^d_{1M}(\underline{c}_0)$- and $\Phi_e^{-d}(\underline{c}_0)$-literals is preserved).

[12] One may also use the complete formulation of the descent theorem here, saying that the category of sets is equivalent to the category of descent data for the non-empty set W, and realize that the above definition is just the definition of an I-indexed coproduct.

Conclusions

In this thesis we considered the decidability of fragments of different logical languages and the problem of transferring the decidability to their combination. In particular, after presenting a result of ours about the decidability of the universal fragment of the theory of arrays with dimension, we mainly focused on one of the simplest methodologies for the combination of decision procedures, the *Nelson-Oppen procedure*, which was originally designed only for the disjoint signatures case and which is guaranteed to be terminating and complete under the following assumptions: (i) Σ_1 and Σ_2 are disjoint; (ii) the theories T_1 and T_2 are stably infinite.

We proved that, if we drop the hypothesis (ii), it is possible to incur in undecidability. On the other hand, if we consider *strongly* $\exists_\infty$*-decidable theories* over disjoint signatures, we proved that the decidability can be transferred to the union of the theories. Moreover, we considered the assumption (i) about the disjointness of the signatures. By introducing a suitable notions of *noetherian theory* and T_i*-basis enumerator*, we extended the results in [44] offering, for example, the opportunity of guaranteeing the decidability of the constraint satisfiability problem for the combination of theories coming from the field of computer algebra.

Finally, relying on a suitable notion of algebraic fragment, we showed that it is possible to recast the Nelson-Oppen schema into an higher-order framework by adopting type-theoretic signatures in Church's style and that, using it in conjunction with model-theoretic results, it succeeds in dealing with various classes of combination problems. Thus, we were able to prove a general decidability transfer result that covers as special cases, besides new applications, the extension of Nelson-Oppen procedure to non-disjoint signatures, the fusion transfer for decidability of global consequence relation in modal logic, the fusion transfer of decidability of A-Boxes with respect to T-Boxes axioms in local abstract description systems, and the transfer decidability result for the monodic fusion of the one variable modal fragment and the monadically suitable one.

Index

List Of Symbols

$(\cdot)^{+n}$	n-shifting function, page 37
$(S_i)^{+n}$	n-shifting of S_i, page 38
$\prec$	Term reduction ordering, page 33
$\{x \mid \varphi\}$	If φ is a formula, it means $\lambda x\, \varphi$, page 76
a-mgu	Antecedent most general unifier, page 38
$\triangleright^1_{\beta\eta}$	$\beta\eta$-rewriting relation (one-step), page 77
$\sim_{\beta\eta}$	$\beta\eta$-equivalence relation, page 78
$C(\overrightarrow{\mathrm{x}}, \overrightarrow{\forall R.\mathrm{X}})$	It indicates that the $\mathcal{ALC}$ atoms of C not occurring in under the scope of the connective $\forall R$ are $\overrightarrow{\mathrm{x}}$ and $\overrightarrow{\forall R.\mathrm{X}}$, where $\overrightarrow{\mathrm{x}}$ are the $\mathcal{ALC}$ atoms which are atomic concepts and $\overrightarrow{\forall R.\mathrm{X}}$ are the remaining ones, page 8
$\Delta(\mathcal{A})$	Diagram of the structure $\mathcal{A}$, page 44
$\Delta^e(\mathcal{A})$	Elementary diagram of the structure $\mathcal{A}$, page 44
$fvar(t)$	Set of variables that occur free in the term t, page 77
$fvar_\tau(t)$	Set of variables of type τ that occur free in the term t, page 77
$\langle \mathcal{L}, T \rangle$	(Algebraic) fragment, page 80
$\langle \mathcal{L}, T, \mathcal{S} \rangle$	Interpreted algebraic fragment, page 81
Φ	Interpreted algebraic fragment, page 81
$\Phi_{\mid \mathcal{L}_0}$	Restriction of the interpreted algebraic fragment to the language $\mathcal{L}_0$, page 91
$\Phi(\underline{\mathrm{c}})$	(Finite) expansion of the interpreted algebraic fragment Φ, page 109
$\Phi^\star$	Specialization of the interpreted algebraic fragment Φ, page 110
$\Phi_1 \oplus \Phi_2$	Combination of the interpreted algebraic fragments Φ_1 and Φ_2, page 96
$\Phi^{\mathcal{L}}_{NK}$	First-order algebraic fragment, NK version, page 85

$\mathcal{L}_{NK}$	First-order language, NK version, page 85
$k(e,n)$	(Non-computable) function associating to each pair (e,n) the number $k(e,n)$ of computation steps of the Turing Machine e on the input n, page 27
$\mathcal{I}$	Given a language $\mathcal{L} = \langle \mathcal{T}, \Sigma, a \rangle$, it is a function assigning to a constant $c \in \Sigma$ of type τ, an element $\mathcal{I}(c^\tau) \in [\![\tau]\!]$, page 79
$[\![-]\!]$	Given a language $\mathcal{L} = \langle \mathcal{T}, \Sigma, a \rangle$, it is an (inductively extensible to all types) function assigning to a sort $S \in \mathcal{T}$, a set $[\![S]\!]$, page 78
$\langle \mathcal{T}, \Sigma, a \rangle$	Higher-order language, page 75
$\prod_{\mathcal{U}}$	Ultrapower operation over the ultrafilter $\mathcal{U}$, page 110
$\sum^I$	I-conglomeration operation over the index set I, page 122
$\sum_I$	Disjoint I-copy operation over the index set I, page 116
Res_Φ	Positive residue enumerator, page 91
Red_Φ	Redundancy notion, i.e. recursive binary relation between a finite set of Φ-clauses and a Φ-clause, page 90
$\Sigma_T^{\underline{a}}$	Expansion of the signature Σ_T with the finite set of constant $\underline{a}$ (if a T is specified, usually Σ_T means the signature of the theory T), page 1
$\Sigma^{\mathcal{K}}$	Expansion of the signature Σ with a countable set of constants $\mathcal{K}$, page 35
$\langle [\![-]\!], \mathcal{I} \rangle$	$\mathcal{L}$-structure, page 78
$\mathcal{A}_{\mid \mathcal{L}_0}$	Reduct of the structure $\mathcal{A}$ to the language $\mathcal{L}_0$, page 80
$\equiv_{\Phi(\underline{c})}$	$\Phi(\underline{c})$-equivalence relation between $\mathcal{L}(\underline{c})$-structures, page 109
$\simeq_{\Phi(\underline{c})}$	$\Phi(\underline{c})$-isomorphism between $\mathcal{L}(\underline{c})$-structures, page 109
$\mathcal{SP}$	Superposition calculus, page 33
$ST(\cdot,\cdot)$	Standard translation function, page 86
$(\Sigma, \mathcal{K}, \prec)$	Suitable ordering triple, page 35
$t : \tau$, t^τ	Term t of type τ, page 76
$t[x_1, \ldots, x_n]$	If t is a term, it means that $fvar(t) \subseteq \{x_1, \ldots, x_n\}$, page 77
TM_∞	$\exists$-decidable theory which is not $\exists_\infty$-decidable, page 27
TM_ω	$\exists$-decidable theory over a finite signature which is not $\exists_\infty$-decidable, page 28
$TM_{\forall\omega}$	Universal $\exists$-decidable theory over a finite signature which is not $\exists_\infty$-decidable, page 30

Bibliography

[1] H. Andréka, J. van Benthem, and I. Nemeti. Modal languages and bounded fragments of predicate logics. *Journal of Philosophical Logic*, 27:217–274, 1998.

[2] P. B. Andrews. Classical type theory. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 2, chapter 15, pages 966–1007. Elsevier and MIT Press, 2001.

[3] P. B. Andrews. *An Introduction to Mathematical Logic and Type Theory: To Truth Through Proof*, volume 27 of *Applied Logic Series*. Kluwer Academic Publishers, Dordrecht (Holland), second edition, 2002.

[4] A. Armando, M. P. Bonacina, S. Ranise, and S. Schulz. On a rewriting approach to satisfiability procedures: extension, combination of theories and an experimental appraisal. In *Proceedings of 5th International Workshop on Frontiers of Combining Systems (FroCoS 2005)*, volume 3717 of *Lecture Notes in Computer Science*, pages 65–80, Wien (Austria), 2005. Springer.

[5] A. Armando, S. Ranise, and M. Rusinowitch. A rewriting approach to satisfiability procedures. *Information and Computation*, 183(2):140–164, 2003.

[6] G. Audemard, P. Bertoli, A. Cimatti, A. Korniłowicz, and R. Sebastiani. A SAT based approach for solving formulas over boolean and linear mathematical propositions. In *Proceedings of 18th International Conference on Automated Deduction (CADE 2002)*, volume 2392 of *Lecture Notes in Computer Science*, pages 195–210, Copenhagen (Denmark), 2002. Springer.

[7] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[8] F. Baader and S. Ghilardi. Connecting many-sorted theories. In *Proceedings of the 20th International Conference on Automated Deduction (CADE 2005)*, volume 3632 of *Lecture Notes in Computer Science*, pages 278–294, Tallinn (Estonia), 2005. Springer.

[9] F. Baader, S. Ghilardi, and C. Tinelli. A new combination procedure for the word problem that generalizes fusion decidability results in modal logics. In *Proceedings of the Second International Joint Conference on Automated Reasoning (IJCAR 2004)*, volume 3097 of *Lecture Notes in Computer Science*, pages 183–197, Cork (Ireland), 2004. Springer.

[10] F. Baader, C. Lutz, H. Sturm, and F. Wolter. Fusions of description logics and abstract description systems. *Journal of Artificial Intelligence Research*, 16:1–58, 2002.

[11] F. Baader and T. Nipkow. *Term Rewriting and All That*. Cambridge University Press, Cambridge (UK), 1998.

[12] F. Baader and C. Tinelli. Deciding the word problem in the union of equational theories. *Information and Computation*, 178(2):346–390, 2002.

[13] L. Bachmair and H. Ganzinger. On restrictions of ordered paramodulation with simplification. In *Proceedings of 10th International Conference on Automated Deduction (CADE 1990)*, volume 449 of *Lecture Notes in Computer Science*, pages 427–441, Kaiserslautern (Germany), 1990. Springer-Verlag.

[14] L. Bachmair and H. Ganzinger. Rewrite-based equational theorem proving with selection and simplification. *Journal of Logic and Computation*, 4(3):217–247, 1994.

[15] L. Bachmair, H. Ganzinger, and U. Waldmann. Theorem proving for hierarchic first-order theories. In *Algebraic and Logic Programming (ALP 1992)*, volume 632 of *Lecture Notes in Computer Science*, pages 420–434, Volterra (Italy), 1992. Springer-Verlag.

[16] H. P. Barendregt. *The Lambda Calculus. Its Syntax and Semantics*, volume 103 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam (Holland), revised edition, 1984.

[17] C. W. Barrett, D. L. Dill, and A. Stump. A generalization of Shostak's method for combining decision procedures. In *Proceedings of the 4th International Workshop on Frontiers of Combining Systems (FroCoS 2002)*, volume 2309 of *Lecture Notes in Computer Science*, pages 132–147, Santa Margherita Ligure (Italy), 2002. Springer.

[18] P. Baumgartner, U. Furbach, and U. Petermann. A unified approach to theory reasoning. Research Report 15–92, Universität Koblenz-Landau, Koblenz (Germany), 1992. Fachberichte Informatik.

[19] A. Bockmayr and V. Weispfenning. Solving numerical constraints. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, chapter 12, pages 751–842. Elsevier and MIT Press, 2001.

[20] M. P. Bonacina, S. Ghilardi, E. Nicolini, S. Ranise, and D. Zucchelli. Decidability and undecidability results for Nelson-Oppen and rewrite-based decision procedures. In U. Furbach and N. Shankar, editors, *Proceedings of the 3rd International Joint Conference on Automated Reasoning (IJCAR 2006)*, volume 4130 of *Lecture Notes in Computer Science*, pages 513–527, Seattle (USA), 2006. Springer.

[21] E. Börger, E. Grädel, and Y. Gurevich. *The Classical Decision Problem*. Universitext. Springer, Berlin (Germany), 2001. Reprint of the 1997 original.

[22] R. J. Brachman. What's in a concept: Structural foundations for semantic networks. *International Journal of Man-Machine Studies*, 9(2):127–152, 1977.

[23] R. J. Brachman. Structured inheritance networks. In *Research in Natural Language Understanding*, pages 13–46. Bolt, Beranek & Newman Inc., Cambridge (MA, USA), 1978. Quarterly Progress Report No. 1, BBN Report No. 3742.

[24] T. Bräuner and S. Ghilardi. First-order modal logic. In J. van Benthem, P. Blackburn, and F. Wolter, editors, *Handbook of Modal Logic*. 2005. (To appear).

[25] C.-C. Chang and H. J. Keisler. *Model Theory*. North-Holland, Amsterdam (Holland), third edition, 1990.

[26] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. MIT Press, Cambridge (MA, USA), 1999.

[27] H. Comon. Solving symbolic ordering constraints. *International Journal of Foundations of Computer Science*, 1(4):387–412, 1990.

[28] H. Comon, P. Narendran, R. Nieuwenhuis, and M. Rusinowitch. Decision problems in ordered rewriting. In *Proceedings of the 13th IEEE Symposium on Logic in Computer Science (LICS 1998)*, pages 276–286, Indianapolis (IN, USA), 1998. IEEE Computer Society Press.

[29] M. Davis, G. Longemann, and D. Loveland. A machine program for theorem proving. *Communication of the ACM*, 5(7):394–397, 1962.

[30] M. Davis and H. Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7(3):201–215, 1960.

[31] H. de Nivelle and I. Pratt-Hartmann. A resolution-based decision procedure for the two-variable fragment with equality. In *Proceedings of the 1st International Joint Conference on Automated Reasoning (IJCAR 2001)*, volume 2083 of *Lecture Notes in Computer Science*, pages 211–225, Siena (Italy), 2001. Springer.

[32] D. Déharbe and S. Ranise. Light-weight theorem proving for debugging and verifying units of code. In *Proceedings of the 1st International Conference on Software Engineering and Formal Methods (SEFM 2003)*, pages 220–228, Brisbane (Australia), 2003. IEEE Computer Society Press.

[33] G. Dowek. Higher order unification and matching. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 2, chapter 16, pages 1009–1062. Elsevier and MIT Press, 2001.

[34] H. B. Enderton. *A Mathematical Introduction to Logic*. Academic Press, New York-London, 1972.

[35] C. Fermüller, A. Leitsch, T. Tammet, and N. Zamov. *Resolution Methods for the Decision Problem*, volume 679 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin (Germany), 1993.

[36] J.-C. Filliâtre, S. Owre, H. Rueß, and N. Shankar. ICS: Integrated canonizer and solver. In *Proceedings of the 13th International Conference on Computer-Aided Verification (CAV 2001)*, volume 2101 of *Lecture Notes in Computer Science*, pages 246–249, Paris (France), 2001. Springer.

[37] M. J. Fischer and M. O. Rabin. Super-exponential complexity of Presburger arithmetic. In *Proceedings of the SIAM-AMS Symposium in Applied Mathematics*, volume 7, pages 27–41, New York (USA), 1974. American Mathematical Society.

[38] C. Flanagan, K. R. M. Leino, M. Lillibridge, G. Nelson, J. B. Saxe, and R. Stata. Extended static checking for Java. In *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2002)*, volume 37 of *ACM SIGPLAN Notices*, pages 234–245, Berlin (Germany), 2002. ACM Press.

[39] H. Friedman. Equality between functionals. In *Logic Colloquium*, volume 453 of *Lecture Notes in Mathemathics*, pages 22–37, Boston (MA, USA), 1975. Springer-Verlag.

[40] D. M. Gabbay, A. Kurucz, F. Wolter, and M. Zakharyaschev. *Many-Dimensional Modal Logics: Theory and Applications*, volume 148 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam (Holland), 2003.

[41] D. M. Gabbay and V. B. Shehtman. Undecidability of modal and intermediate first-order logics with two individual variables. *Journal of Symbolic Logic*, 58:800–823, 1993.

[42] H. Ganzinger and H. de Nivelle. A superposition decision procedure for the guarded fragment with equality. In *14th Symposium on Logic in Computer Science (LICS 1999)*, pages 295–303, Trento (Italy), 1999. IEEE Computer Society Press.

[43] H. Ganzinger, G. Hagen, R. Nieuwenhuis, A. Oliveras, and C. Tinelli. DPLL(T): Fast decision procedures. In *Proceedings of the 16th International Conference on Computer-Aided Verification (CAV 2004)*, volume 3114 of *Lecture Notes in Computer Science*, pages 175–188, Boston (MA, USA), 2004. Springer.

[44] S. Ghilardi. Model theoretic methods in combined constraint satisfiability. *Journal of Automated Reasoning*, 33(3-4):221–249, 2004.

[45] S. Ghilardi, E. Nicolini, S. Ranise, and D. Zucchelli. Deciding extensions of the theory of arrays by integrating decision procedures and instantiation strategies. In M. Fischer, W. van der Hoek, B. Konev, and A. Lisitsa, editors, *Proceedings of the 10th European Conference on Logic in Artificial Intelligence (JELIA 2006)*, volume 4160 of *Lecture Notes in Computer Science*, pages 177–189, Liverpool (UK), 2006. Springer.

[46] S. Ghilardi, E. Nicolini, and D. Zucchelli. A comprehensive combination framework. *ACM Transactions on Computational Logic*, 2006. (To appear).

[47] J. Y. Girard, P. Taylor, and Y. Lafont. *Proofs and Types*, volume 7 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, Cambridge (UK), 1989.

[48] V. Goranko and M. Otto. Model theory of modal logic. In J. van Benthem, P. Blackburn, and F. Wolter, editors, *Handbook of Modal Logic*. 2005. (To appear).

[49] E. Grädel. Decision procedures for guarded logics. In *Proceedings of 16th International Conference on Automated Deduction (CADE 1999)*, volume 1632 of *Lecture Notes in Computer Science*, pages 31–51, Trento (Italy), 1999. Springer-Verlag.

[50] D. Jackson and M. Vaziri. Finding bugs with a constraint solver. In *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA 2000)*, ACM SIGSOFT Software Engineering Notes, pages 14–25, Portland (OR, USA), 2000. ACM Press.

[51] G. Janelidze and W. Tholen. Facets of descent, I. *Applied Categorical Structures*, 2(3):245–281, 1994.

[52] A. Joyal and M. Tierney. An extension of the Galois theory of Grothendieck. *Memoirs of the American Mathematical Society*, 51(309):vii+71, 1984.

[53] E. Kieronski and M. Otto. Small substructures and decidability issues for first-order logic with two variables. In *Twentieth Annual IEEE Symposium on Logic in Computer Science (LICS 2005)*, pages 448–457, Chicago (IL, USA), 2005. IEEE Computer Society Press.

[54] R. Kontchakov, A. Kurucz, and M. Zakharyaschev. Undecidability of first-order intuituionistic and modal logics with two variables. Manuscript, 2004.

[55] K. Korovin and A. Voronkov. Knuth-Bendix constraint solving is NP-complete. *ACM Transactions on Computational Logic*, 6(2):361–388, 2005.

[56] D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27(3):333–354, 1983.

[57] S. Kripke. The undecidability of monadic modal quantificational theory. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 8:113–116, 1962.

[58] J. Lambek and P. J. Scott. *Introduction to higher order categorical logic*, volume 7 of *Cambridge Studies in Advanced Mathematics*. Cambridge University Press, Cambridge (UK), 1988. Reprint of the 1986 original.

[59] J.-L. Lassez and M. J. Maher. On Fourier's algorithm for linear arithmetic constraints. *Journal of Automated Reasoning*, 9(3):373–379, 1992.

[60] J.-L. Lassez and K. McAloon. A canonical form for generalized linear constraints. *Journal of Symbolic Computation*, 13(1):1–24, 1992.

[61] F. W. Lawvere. Functorial semantics of algebraic theories. *Proceedings of the National Academy of Science*, 50:869–872, 1963.

[62] D. Lewis. Counterpart theory and quantified modal logic. *Journal of Philosophy*, 65(5):113–126, 1968.

[63] L. Löwhenheim. Über möglichkeiten im relativkalkül. *Mathematische Annalen*, 76:228–251, 1915.

[64] S. MacLane and G. Birckhoff. *Algebra*. Chelsea Publishing Co., New York (USA), third edition, 1988.

[65] M. Makkai and G. E. Reyes. *First-Order Categorical Logic*, volume 611 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin (Germany), 1977.

[66] M. Marx. Tolerance logic. *Journal of Logic, Language and Information*, 10:353–374, 2001.

[67] A. Middeldorp and H. Zantema. Simple termination revisited. In *Proceedings of 12th International Conference on Automated Deduction (CADE 1994)*, volume 814 of *Lecture Notes in Computer Science*, pages 451–465, Nancy (France), 1994. Springer-Verlag.

[68] M. Mortimer. On languages with two variables. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 21:135–140, 1975.

[69] G. Nelson and D. C. Oppen. Simplification by cooperating decision procedures. *ACM Transaction on Programming Languages and Systems*, 1(2):245–257, 1979.

[70] R. Nieuwenhuis and J. M. Rivero. Practical algorithms for deciding path ordering constraint satisfaction. *Information and Computation*, 178(2):422–440, 2002.

[71] R. Nieuwenhuis and A. Rubio. Paramodulation-based theorem proving. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 1, chapter 7, pages 371–443. Elsevier and MIT Press, 2001.

[72] P. Odifreddi. *Classical Recursion Theory*, volume 125 of *Studies in Logic and the Foundations of Mathematics*. North-Holland Publishing Co., Amsterdam (Holland), 1989.

[73] D. C. Oppen. Complexity, convexity and combinations of theories. *Theoretical Computer Science*, 12:291–302, 1980.

[74] RTI - Health, Social and Economics Research. The economic impacts of inadequate infrastructure for software testing. Planning Report 02-3, National Institute of Standards & Technology (NIST) - U.S. Department of Commerce, 2002. Available at `http://www.nist.gov/director/prog-ofc/report02-3.pdf`.

[75] W. Rudin. *Functional Analysis*. International Series in Pure and Applied Mathematics. McGraw-Hill, Inc., Boston (MA, USA), second edition, 1991.

[76] K. Schild. A correspondence theory for terminological logics: Preliminary report. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI 1991)*, pages 466–471, Sidney (Australia), 1991. Morgan Kaufmann.

[77] M. Schmidt-Schauss and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.

[78] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley, New York (USA), 1986.

[79] S. Schulz. E - a brainiac theorem prover. *AI Communications*, 15(2/3):111–126, 2002.

[80] D. Scott. A decision method for for validity of sentences in two variables. *Journal of Symbolic Logic*, 27:477, 1962.

[81] K. Segerberg. Two-dimensional modal logic. *Journal of Philosophical Logic*, 2:77–96, 1973.

[82] V. B. Shehtman. On some two-dimensional modal logics. In *8th Congress on Logic Methodology and Philosophy of Science*, volume 1, pages 326–330. Nauka, Moskow (Russia), 1987.

[83] A. Stump, C. W. Barrett, D. L. Dill, and J. Levitt. A decision procedure for an extensional theory of arrays. In *Proceedings of the 16th IEEE Symposium on Logic in Computer Science (LICS 2001)*, pages 29–37, Boston (MA, USA), 2001. IEEE Computer Society.

[84] C. Tinelli. A DPLL-based calculus for ground satisfiability modulo theories. In *Proceedings of the 8th European Conference on Logics in Artificial Intelligence (JELIA 2002)*, volume 2424 of *Lecture Notes in Artificial Intelligence*, pages 308–319, Cosenza (Italy), 2002. Springer.

[85] C. Tinelli. Cooperation of background reasoners in theory reasoning by residue sharing. *Journal of Automated Reasoning*, 30(1):1–31, 2003.

[86] C. Tinelli and M. T. Harandi. A new correctness proof of the Nelson–Oppen combination procedure. In *Proceedings of the 1st International Workshop on Frontiers of Combining Systems (FroCoS 1996)*, Applied Logic, pages 103–120, Munich (Germany), 1996. Kluwer Academic Publishers.

[87] D. van Dalen. *Logic and Structure*. Springer-Verlag, Berlin (Germany), second edition, 1989.

[88] C. Weidenbach. Combining superposition, sorts and splitting. In A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*, volume 2, chapter 27, pages 1965–2013. Elsevier and MIT Press, 2001.

[89] V. Weispfenning. Existential equivalence of ordered abelian groups with parameters. *Archive for Mathematical Logic*, 29(4):237–248, 1990.

[90] V. Weispfenning. Complexity and uniformity of elimination in Presburger arithmetic. In *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation (ISSAC 1997)*, pages 48–53, Maui, Hawaii (USA), 1997. ACM Press.

[91] C. A. Welty and N. Guarino. Supporting ontological analysis of taxonomic relationships. *Data & Knowledge Engineering*, 39(1):51–74, 2001.

[92] W. H. Wheeler. Model-companions and definability in existentially complete structures. *Israel Journal of Mathematics*, 25:305–330, 1976.

[93] F. Wolter. Fusions of modal logics revisited. In M. Kracht, M. de Rijke, H. Wansing, and M. Zakharyaschev, editors, *Advances in Modal Logic*, volume 87 of *CSLI Lecture Notes*, pages 361–379. CSLI Publ., Stanford (CA, USA), 1998.

[94] F. Wolter and M. Zakharyaschev. Decidable fragments of first-order modal logics. *Journal of Symbolic Logic*, 66:1415–1438, 2001.