Reasoning in complex theories and applications
Advanced Lecture, ESSLLI 2009

Viorica Sofronie-Stokkermans
Max-Planck-Institut für Informatik, Campus E 1.4, D-66125 Saarbrücken, Germany

1 Introduction

One of the most important research objectives in mathematics and computer science is reasoning in and about complex systems. Complex systems arise in a natural way in many areas such as mathematics, logic, computation, design of digital circuits, communication, language processing, databases, or artificial intelligence. They can be, for instance:

- complex mathematical theories;
- programs, or more general reactive or hybrid systems;
- large (possibly distributed) databases;
- multi-agent systems;
- or very complex systems (e.g. multi-agent systems or reactive or hybrid systems with embedded software, whose behavior is controlled by databases, reasoning about knowledge and belief, planning mechanisms, or programs).

Proving properties of such systems is extremely important. In safety-critical systems (for instance cars, trains, planes, or power-plants) even small mistakes can provoke disasters. Since the amount of data which has to be handled in verification tasks is usually huge, computer support is necessary. The dream of the scientists is to prove such correctness properties automatically. This goal cannot be reached in its full generality - this follows from undecidability results in first order logic which go back to the work of Gödel, Church and Turing. However, for concrete application domains (e.g. for proving specific properties of certain classes of systems) automatic procedures exist. It is therefore very important to identify situations in which automated verification of complex systems is possible. For this, it is essential to identify theories which are decidable, preferably with low complexity. Very often, classical theories do not occur alone in applications:

- In program verification, for instance, one may need to prove properties of lists, arrays, or sets (with elements of a certain type), or of several such datatypes at the same time.
- In distributed ontologies one usually needs to consider various extensions of a common kernel with new concepts and constructors.

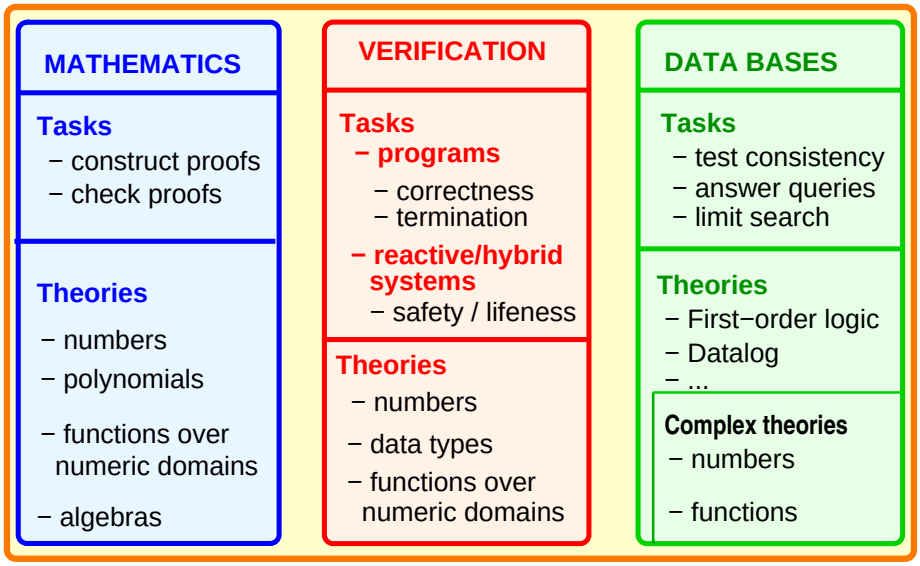
It is therefore very important to obtain methods for efficiently combining decision procedures. For a long time the state of the art in the research in combination of decision procedures was represented by the Nelson-Oppen combination procedure (which can be applied to combine decision procedures for certain theories over disjoint signatures). Only in recent years, significant progress in the area of reasoning in extensions of theories and combinations of possibly non-disjoint theories has been made; some of these results are presented here.

Aims of the course. The goal of this advanced course is to give a comprehensive, in-depth perspective on recent advances in the field of reasoning in complex logical theories, and to present the applications of these results in various areas such as mathematics, formal verification, knowledge representation (data bases and multi-agent systems).

Contents

2 Motivation: Verification of complex systems

The main obstacle in using computers for solving real-life problems is complexity: On the one hand, the systems which are analyzed are complex, therefore their formalization is complex as well. At the same time, the amount of data to be taken into account, even when analyzing relatively simple systems, is huge. In this section we briefly illustrate the type of verification problems in *mathematics*, *verification*, and *databases*.



2.1 Mathematics

The verification tasks needed in mathematics are, on the one hand, proving or disproving theorems about mathematical theories, and also (automatically) checking the correctness of proofs. Theories important in mathematics include:

- theories of real, rational, integer or natural numbers;
- theories of polynomials with real/rational/integer/natural coefficients;
- theories of functions over numerical domains – for instance monotone, continuous, derivable functions, or functions with special properties, e.g. satisfying the Lipschitz conditions – very important in mathematical analysis;
- theories of algebraic structures (groups, rings, fields);
- theories of ordered structures (lattices, Boolean algebras);
- more generally, possibly many-sorted theories of universal algebras;
- logical theories.

To refer to only one of the aspects mentioned above: The task of automatically reasoning in extensions of numerical domains with function symbols whose properties are expressed by first order axioms is highly non-trivial: most existing methods are based on heuristics; very few sound and complete methods or decidability results exist, even for specific fragments [24, 25, 14, 58]. More details are given in Section 9.1 and 9.2.

2.2 Verification of reactive and hybrid systems

Typical examples of reactive systems are programs. In more general reactive or hybrid systems we need to refer to parametric changes which can be described using functions over numerical domains which satisfy certain properties. Depending on the specific application domain, the specifications explicitly refer to certain theories and data types:

- numbers (real/rational/integer/natural);
- functions over numerical domains (e.g. for modeling parametric changes);
- theories of data structures such as lists or arrays – especially for program verification, but also e.g. in the specification of systems with a parametric number of components.

Such theories are usually explicitly mentioned in the specification part. Correspondingly, a *background theory* $\mathcal{T}_S$ – describing the data types used in the specification and their properties – is associated in a canonical way with every specification. More details about the specific problems which are interesting in this context are given in Section 9.3.

2.3 Databases

Among the verification tasks important in relationship with databases are consistency testing and query answering. In addition, when constructing a database it is very important to check incrementally whether after adding new information the truth of formulae w.r.t. the information contained in the database changes or not. Since the information stored in databases is usually huge, it is extremely important to find methods for limiting search without affecting the accuracy of the answers. The type of theories related to the contents of a database depends on the type of information stored in databases. Information can be expressed in:

- (full) first order logic – which has the disadvantage of being undecidable – or even higher-order logic;
- light-weight fragments of first order logic, such as Datalog (sets of (universally quantified) rules consisting of Horn clauses in a language without function symbols) or variants thereof;
- specific theories as used for instance in description logics;
- combinations of theories, involving formulae in (fragments) of first order logic, functions with certain properties and numerical theories.

An issue which is becoming increasingly essential in databases and ontologies is *modularity*: important aspects are modular design and reasoning. Since the component databases or ontologies can be regarded as logical theories, efficient reasoning in combinations of ontologies is extremely important. It is also very important to identify light-weight description logics and possibilities of efficiently integrating numerical domains in light-weight description logics. Such aspects are mentioned in Section 9.6

2.4 Multi-agent systems

Another area of applications in which combinations of theories are important is the area of multi-agent systems. We address here only one aspect, referring to the possibilities of agents of reasoning about knowledge and belief: In a logical formalization, each agent may have its own knowledge about environment, its own way of reasoning about knowledge and belief, and, possibly, also its own knowledge about the knowledge of the other agents.

The way of reasoning about knowledge of an agent A may be represented by a specific type of modal logic, with modal operators $\Box_A, \Diamond_A$ having, for instance, the meaning:

- $\Box_A\phi$: ϕ is known to A (or A believes that ϕ)
- $\Diamond_A\phi$: ϕ is consistent with the knowledge of A (resp. with the beliefs of A).

For proving or disproving statements about the knowledge/belief of a *set of agents* it is important to be able to reason in combinations of modal logics. Since reasoning in many modal logics can be reduced to deciding uniform word problems in corresponding classes of Boolean algebras with operators, the reasoning problem in the combinations of logics can be reduced to the problem of modular reasoning in combinations of Boolean algebras with operators. Such aspects are mentioned in Section 9.7.

We can consider even more complex multi-agent systems whose behavior is controlled by databases, reasoning about knowledge and belief, planning mechanisms, or programs.

2.5 Formalization, reduction to deduction in logical theories

Many of the verification problems mentioned above can be reformulated as follows:

1. Formalize the problems by explicitly pointing out the theories involved and expressing the proof tasks in a logical language.
2. Reduce the verification problems to the task of proving satisfiability, validity, or entailment of formulae in a logical theory – canonically associated with the concrete application.

The main problems which occur in this context are, on the one hand, the size of the formulae obtained this way and, on the other hand, the fact that most real-life problems are heterogeneous in nature, and when formalizing them, complex theories are often needed: typically extensions and combinations of "standard" theories.

2.6 Structure of the course

We start by defining theories and models in Section 3. In Section 4, decidability and undecidability results for theory fragments are presented. After presenting some general decidability and undecidability results for logical theories, we introduce several theory fragments and their associated decision problems, as well as the relationship between them. Methods for proving decidability are mentioned. In Section 5 we present some practically relevant examples of tractable theory fragments.

In Section 6 we present the main problems, challenges and ideas for reasoning in complex theories. The way these ideas can be applied is then explained in detail for combinations of theories over disjoint signatures (Section 7.1), theories over non-disjoint signatures (Section 7.2) as well as certain types of theory extensions (Section 8). We end by providing various examples from mathematics, verification, databases and multi-agent systems where efficient decision procedures exist (Section 9).

3 Preliminaries

In this section we define the main notions used in what follows: theories, models, as well as restrictions, combinations and extensions of theories.

3.1 Formulae, models, theories

For basic definitions in first order logic (formulae, satisfiability, validity, entailment) we refer to any textbook in logic. We will consider formulae and models over many-sorted signatures $\Sigma = (S, \Omega, \mathsf{Pred})$, where S is a set of sorts, Ω is a set of function symbols and Pred a set

of predicate symbols (with given arities). For one-sorted signatures we will omit the set of sorts and use the notation $\Sigma = (\Omega, \mathsf{Pred})$.

Definition 1 *A Σ-structure (or Σ-algebra) is a tuple*

$$\mathcal{M} = (\{M_s\}_{s\in S}, \{f_{\mathcal{M}}\}_{f\in\Omega}, \{P_{\mathcal{M}}\}_{P\in\mathsf{Pred}}),$$

where for every $s \in S$, M_s is a non-empty set, for all $f \in \Omega$ with arity $a(f)=s_1 \ldots s_n \to s$, $f_{\mathcal{M}} : \prod_{i=1}^{n} M_{s_i} \to M_s$ and for all $P \in \mathsf{Pred}$ with arity $a(P) = s_1 \ldots s_n$, $P_{\mathcal{M}} \subseteq M_{s_1} \times \ldots \times M_{s_n}$.

We will denote the set of all Σ-algebras by Σ-Alg. In what follows we will denote the equality predicate by $\approx$. We consider terms and formulae over variables in a (many-sorted) family $X = \{X_s \mid s \in S\}$, where for every $s \in S$, X_s is a set of variables of sort s. The set of all Σ-terms will be denoted by $T_\Sigma(X)$, and if we want to emphasize the set of function symbols used for building the terms also by $T_\Omega(X)$. The set of all Σ-formulae over the variables in X is denoted by $F_\Sigma(X)$. Truth and satisfiability of a first order formula in a given model are defined in the standard way.

Definition 2 *A model of a set $\mathcal{T}$ of formulae is a Σ-structure satisfying all formulae of $\mathcal{T}$.*

We will denote by $\mathsf{Mod}(\mathcal{T})$ the set of all models of a set of formulae.

Definition 3 *If ϕ and ψ be formulae over the signature Σ, we say that:*

(1) ϕ is true *w.r.t. $\mathcal{T}$ (denoted $\models_{\mathcal{T}} \phi$) if ϕ is true in all models of $\mathcal{T}$;*
(2) ϕ entails *ψ w.r.t. $\mathcal{T}$ (denoted $\phi \models_{\mathcal{T}} \psi$) if ψ is true in all models of $\mathcal{T} \wedge \phi$;*
(3) ϕ is satisfiable *w.r.t. $\mathcal{T}$ if there exists a model of $\mathcal{T}$ in which ϕ is true;*
(4) If ϕ is false in all models of $\mathcal{T}$, we say that ϕ is unsatisfiable.

Note that ϕ is unsatisfiable iff $\phi \models_{\mathcal{T}} \bot$, where $\bot$ stands for false.

Theories can be described by sets of formulae or by sets of models. We present the syntactic and semantic views, as well as the link between them.

Definition 4 (Syntactic point of view) *A first order theory can be defined by a set $\mathcal{F}$ of (closed) first order Σ-formulae (its axioms).*

The class of all models of $\mathcal{F}$ is: $\mathsf{Mod}(\mathcal{F}) = \{\mathcal{A} \in \Sigma\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{F}\}$.

Definition 5 (Semantic point of view) *Given a class $\mathcal{M}$ of Σ-algebras, the first order theory of $\mathcal{M}$ is $\mathsf{Th}(\mathcal{M}) = \{G \in F_\Sigma(X)$ closed $\mid \mathcal{M} \models G\}$, where X is a countably infinite set of variables.*

Note that $\mathsf{Th}(\mathsf{Mod}(\mathcal{F}))$, the set of formulae true in all models of $\mathcal{F}$, represents exactly the set of consequences of $\mathcal{F}$. It is easy to show that $\mathcal{F} \subseteq \mathsf{Th}(\mathsf{Mod}(\mathcal{F}))$ (typically strict) and $\mathcal{M} \subseteq \mathsf{Mod}(\mathsf{Th}(\mathcal{M}))$ (also typically strict).

Example 1 (Peano arithmetic) *The language of Peano arithmetic contains constants 0 and 1 and binary functions + and * (addition resp. multiplication). Peano arithmetic is axiomatized by the axioms* PA*:*

$(A1)$	$\forall x \neg(x + 1 \approx 0)$	*(zero)*
$(A2)$	$\forall x \forall y\, (x + 1 \approx y + 1 \to x \approx y)$	*(successor)*
$(A3)$	$F[0] \wedge (\forall x\, (F[x] \to F[x + 1]) \to \forall x F[x])$	*(induction)*
$(A4)$	$\forall x\, (x + 0 \approx x)$	*(plus zero)*
$(A5)$	$\forall x, y\, (x + (y + 1) \approx (x + y) + 1)$	*(plus successor)*
$(A6)$	$\forall x, y\, (x * 0 \approx 0)$	*(times 0)*
$(A7)$	$\forall x, y\, (x * (y + 1) \approx x * y + x)$	*(times successor)*

The inequality and strict inequality relations are definable in this signature. For instance $3{*}y{+}5 > 2{*}y$ *expressed as* $\exists z(z \not\approx 0 \wedge 3{*}y{+}5 \approx 2{*}y{+}z)$. *The intended interpretation for this theory is* $(\mathbb{N}, \{0, 1, +, *\}, \{\approx, \leq\})$ *(but it does not capture true arithmetic by Gödel's incompleteness theorem). Clearly,* $(\mathbb{N}, \{0, 1, +, *\}, \{\approx, \leq\}) \in \mathsf{Mod}(\mathsf{PA})$ *and* $\mathsf{PA} \subset \mathsf{Th}(\mathsf{Mod}(\mathsf{PA}))$.

Example 2 (Presburger arithmetic (Presburger 1929)) *The language of Presburger arithmetic contains constants 0 and 1 and a binary function +. The axioms of Presburger arithmetic are axioms (A1)–(A5) above. The intended interpretation for this theory is* $(\mathbb{N}, \{0, 1, +\}, \{\approx, \leq\})$.

Example 3 (Linear integer arithmetic) *Let* $\Sigma = (\{0/0, s/1, +/2\}, \{\leq /2\})$. *Let* $\mathbb{Z}_+ = (\mathbb{Z}, 0, s, +, \leq)$ *the standard interpretation of integers.* $\{\mathbb{Z}_+\} \subset \mathsf{Mod}(\mathsf{Th}(\mathbb{Z}_+))$.

Example 4 (Uninterpreted function symbols) *Let* $\Sigma = (\Omega, \mathsf{Pred})$ *be an arbitrary signature. Let* $\mathcal{M} = \Sigma\text{-}\mathsf{Alg}$ *be the class of all* Σ*-structures. The theory of uninterpreted function symbols is* $\mathsf{Th}(\Sigma\text{-}\mathsf{Alg})$*, the family of all first order formulae which are true in all* Σ*-algebras. We will denote the theory of uninterpreted function symbols also by* $\mathsf{Free}(\Sigma)$.

Example 5 (Lists) *Let* $\Sigma = (\{\mathsf{car}/1, \mathsf{cdr}/1, \mathsf{cons}/2\}, \emptyset)$ *and* $\mathcal{F}_{\mathsf{lists}}$ *be the set of axioms:*

$$\begin{aligned} \forall x, y\ \mathsf{car}(\mathsf{cons}(x, y)) &\approx x \\ \forall x, y\ \mathsf{cdr}(\mathsf{cons}(x, y)) &\approx y \\ \forall x\ \mathsf{cons}(\mathsf{car}(x), \mathsf{cdr}(x)) &\approx x \end{aligned}$$

Let $\mathsf{Mod}(\mathcal{F}_{\mathsf{lists}})$ *be the class of all models of* $\mathcal{F}$. *Then* $\mathsf{Th}_{\mathsf{Lists}} = \mathsf{Th}(\mathsf{Mod}(\mathcal{F}_{\mathsf{lists}}))$ *is the theory of lists (axiomatized by* $\mathcal{F}_{\mathsf{lists}}$*), i.e. the set of all logical consequences of the axioms in* $\mathcal{F}_{\mathsf{lists}}$.

Example 6 (Acyclic Lists) *Let* $\Sigma = (\{\mathsf{car}/1, \mathsf{cdr}/1, \mathsf{cons}/2\}, \emptyset)$. *The theory of acyclic lists is axiomatized by the set of axioms* $\mathcal{F}_{\mathsf{Alists}}$*:*

$$\begin{aligned} \forall x, y\ \mathsf{car}(\mathsf{cons}(x, y)) &\approx x \\ \forall x, y\ \mathsf{cdr}(\mathsf{cons}(x, y)) &\approx y \\ \forall x\ \mathsf{cons}(\mathsf{car}(x), \mathsf{cdr}(x)) &\approx x \\ \forall x\ t(x) &\not\approx x && \text{for all terms } t \text{ containing only the} \\ & && \text{symbol } \mathsf{cons} \text{ and the variable } x \\ \forall x, y, x', y'\ \mathsf{cons}(x, y) \approx \mathsf{cons}(x', y') &\rightarrow x \approx x' \wedge y \approx y' && \text{(injectivity of constructors)} \end{aligned}$$

(Note that injectivity of constructors is a consequence of the first two axioms, but can be listed also as a separate axiom.) Let $\mathsf{Mod}(\mathcal{F}_{\mathsf{Alists}})$ *be the class of all models of* $\mathcal{A}$. *Then* $\mathsf{Th}_{\mathsf{ALists}} = \mathsf{Th}(\mathsf{Mod}(\mathcal{F}_{\mathsf{Alists}}))$ *is the theory of lists (axiomatized by* $\mathcal{F}_{\mathsf{Alists}}$*), i.e. the set of all logical consequences of the axioms of acyclic lists.*

Example 7 (Arrays) *McCarthy's theory of arrays has three sorts (for arrays (*a*), index (*i*), and elements (*e*)). The signature consists of the function symbols* write *of sort* $\mathsf{a}\times\mathsf{i}\times\mathsf{e} \rightarrow \mathsf{a}$ *and* read *of sort* $\mathsf{a} \times \mathsf{i} \rightarrow \mathsf{e}$. *The theory is axiomatized as follows:*

$$\begin{aligned} \mathsf{read}(\mathsf{write}(a, i, e), i) &\approx e \\ \mathsf{read}(\mathsf{write}(a, i, e), j) &\approx \mathsf{read}(a, j) \vee i \approx j \\ a \approx b &\leftrightarrow \forall i(\mathsf{read}(a, i) \approx \mathsf{read}(b, j)) \end{aligned}$$

3.2 Restriction to a sublanguage

Let $\Sigma = (\Omega, \mathsf{Pred})$ and $\Sigma' = (\Omega', \mathsf{Pred}')$. Suppose $\Sigma \subseteq \Sigma'$, i.e. $\Omega \subseteq \Omega'$ and $\mathsf{Pred} \subseteq \mathsf{Pred}'$.

Definition 6 *Let* $\mathcal{A} = (A, \{f_A\}_{f\in\Omega'}, \{P_A\}_{P\in\mathsf{Pred}'}) \in \Sigma'\text{-}\mathsf{Alg}$. *The* restriction *of* $\mathcal{A}$ *to* Σ *(or the* Σ-reduct *of* $\mathcal{A}$*) is the* Σ*-structure* $\mathcal{A}_{|\Sigma} = (A, \{f_{A_{|\Sigma}}\}_{f\in\Omega}, \{P_{A_{|\Sigma}}\}_{P\in\mathsf{Pred}})$ *obtained by ignoring all functions and predicates associated with symbols in* $\Sigma'\backslash\Sigma$*, i.e. such that:*

$$\begin{aligned} f_{\mathcal{A}_{|\Sigma}} &= f_{\mathcal{A}} \quad \text{for } f \in \Omega \\ P_{\mathcal{A}_{|\Sigma}} &= P_{\mathcal{A}} \quad \text{for } P \in \mathsf{Pred} \end{aligned}$$

Example 8 *Consider the signatures* $\Sigma' = (\{+/2, */2, 1/0\}, \{\leq/2, \mathsf{even}/1, \mathsf{odd}/1\})$ *and* $\Sigma = (\{+/2, 1/0\}, \{\leq/2\}) \subseteq \Sigma'$. *If* $\mathcal{N} = (\mathbb{N}, +, *, 1, \leq, \mathsf{even}, \mathsf{odd})$ *(a* Σ'*-algebra) then the* Σ*-reduct of* $\mathcal{N}$ *is* $\mathcal{N}_{|\Sigma} = (\mathbb{N}, +, 1, \leq)$.

If $\mathcal{M}$ be a class of Σ'-algebras, $\mathcal{M}_{|\Sigma}$ consists of all Σ-reducts of algebras in $\mathcal{M}$:

$$\mathcal{M}_{|\Sigma} = \{\mathcal{A}_{|\Sigma} \mid \mathcal{A} \in \mathcal{M}\}.$$

3.3 Combinations of theories

We can consider combinations of theories again both from a syntactic and from a semantic point of view.

Definition 7 (Syntactic view) *We regard theories as sets of formulae. Let* $\mathcal{T}_1$ *and* $\mathcal{T}_2$ *be described by sets of axioms in the signatures* Σ_1 *resp.* Σ_2 *with variables in a set* X. *The union of the two theories is the theory in the signature* $\Sigma_1 \cup \Sigma_2 = (\Omega_1, \mathsf{Pred}_1) \cup (\Omega_2, \mathsf{Pred}_2) = (\Omega_1 \cup \Omega_2, \mathsf{Pred}_1 \cup \mathsf{Pred}_2)$ *obtained by taking the union of the sets of axioms for* $\mathcal{T}_1$ *and* $\mathcal{T}_2$*:*

$$\mathcal{T}_1 + \mathcal{T}_2 = \mathcal{T}_1 \cup \mathcal{T}_2 \subseteq F_{\Sigma_1\cup\Sigma_2}(X).$$

Note that $\mathsf{Mod}(\mathcal{T}_1 \cup \mathcal{T}_2) = \{\mathcal{A} \in (\Sigma_1 \cup \Sigma_2)\text{-alg} \mid \mathcal{A} \models G, \text{ for all } G \text{ in } \mathcal{T}_1 \cup \mathcal{T}_2\}$.

Definition 8 (Semantic view) *Let* $\mathcal{M}_1, \mathcal{M}_2$ *be two classes of algebras. Let* $\mathcal{M}_1 + \mathcal{M}_2 = \{\mathcal{A} \in (\Sigma_1 \cup \Sigma_2)\text{-}\mathsf{Alg} \mid \mathcal{A}_{|\Sigma_i} \in \mathcal{M}_i \text{ for } i = 1, 2\}$.

The link between the syntactic and the semantic view is given by Theorem 9.

Theorem 9 *Let* $\mathcal{T}_1$ *be a set of* Σ_1*-formulae and* $\mathcal{T}_2$ *a set of* Σ_2*-formulae. Let* $\mathcal{M}_i = \mathsf{Mod}(\mathcal{T}_i), i = 1, 2$. *Then* $\mathsf{Th}(\mathsf{Mod}(\mathcal{T}_1 \cup \mathcal{T}_2)) = \mathsf{Th}(\mathcal{M}_1 + \mathcal{M}_2)$.

Proof: For every $\Sigma_1 \cup \Sigma_2$-algebra $\mathcal{A}$, $\mathcal{A} \in \mathsf{Mod}(\mathcal{T}_1 \cup \mathcal{T}_2)$ iff $\mathcal{A} \models G$ for all G in $\mathcal{T}_1 \cup \mathcal{T}_2$. This happens iff $\mathcal{A}_{|\Sigma_i} \models G$ for all G in $\mathcal{T}_i, i = 1, 2$, i.e. iff $\mathcal{A}_{|\Sigma_i} \in \mathcal{M}_i, i = 1, 2$, hence, by definition, iff $\mathcal{A} \in \mathcal{M}_1 + \mathcal{M}_2$. □

Example 9 *The combination of Presburger arithmetic with the theory* $\mathsf{Free}(\Sigma)$ *of free function symbols in* $\Sigma = (\Omega, \mathsf{Pred})$ *has as models algebras*

$$(A, 0, 1, +, \{f_A\}_{f\in\Omega}, \leq, \{P_A\}_{P\in\mathsf{Pred}}),$$

where $(A, 0, 1, +, \leq)$ *is a model of Presburger arithmetic. The combination of Presburger arithmetic with the theory* $\mathsf{Th}_{\mathsf{Lists}}$ *of lists has as models algebras*

$$(A, 0, 1, +, \mathsf{car}_A, \mathsf{cdr}_A, \mathsf{cons}_A, \leq),$$

where $(A, 0, 1, +, \leq)$ *is a model of Presburger arithmetic and* $(A, \mathsf{car}_A, \mathsf{cdr}_A, \mathsf{cons}_A) \in \mathsf{Mod}(\mathsf{Th}_{\mathsf{Lists}})$.

Example 10 *The combination of the theory of reals with the theory of a monotone function f is the theory $\mathsf{Th}(\mathbb{R}) \cup \mathsf{Mon}(f)$, where*

$$\mathsf{Mon}(f): \qquad \forall x, y(x \leq y \rightarrow f(x) \leq f(y)).$$

Its models are algebras of the form $(A, +, *, f_A, \leq)$, where $(A, +, *, \leq) \in \mathsf{Mod}(\mathsf{Th}(\mathbb{R}))$, and $(A, f_A, \leq) \models \mathsf{Mon}(f)$, i.e. $f_A : A \rightarrow A$ monotone.

Note that in this case the signatures of the two theories share the $\leq$ predicate symbol.

We can also consider the combination of the theory of real numbers with the theory of a monotone function over a poset (these two theories share the common subtheory of posets).

3.4 Theory extensions

Combinations of theories cannot properly capture all types of theories of interest. For instance, as explained in Example 10, the theory of real numbers with a monotone function can be seen as the combination of two theories whose signatures share the predicate symbol $\leq$. Consider now the theory of a λ-Lipschitz function over $\mathbb{R}$ at a given point c. The Lipschitz axiom:

$$\mathsf{L}^{\lambda}_{f}(c) \quad \forall x \ \ |f(x) - f(c)| \leq \lambda * |x - c|$$

contains $\{-, *, \leq\}$ i.e. (almost) the whole signature of the theory of real numbers. Thus, if we would like to represent this theory as a combination of two theories the shared signature would be (almost) the same as the signature of the real numbers. In such situations it is more convenient to talk about *theory extensions*.

Let $\mathcal{T}_0$ be a theory with signature $\Sigma_0 = (S, \Omega_0, \mathsf{Pred})$. We consider extensions $\mathcal{T}_1$ of $\mathcal{T}_0$ with signature $\Sigma = (S, \Omega, \mathsf{Pred})$, where the set of function symbols is $\Sigma = \Sigma_0 \cup \Sigma_1$ (i.e. the signature is extended by new function symbols). We assume that $\mathcal{T}_1$ is obtained from $\mathcal{T}_0$ by adding a set $\mathcal{K}$ of (universally quantified) clauses in the signature Σ. Thus, $\mathsf{Mod}(\mathcal{T}_1)$ consists of all Σ-structures which are models of $\mathcal{K}$ and whose reduct to Σ_0 is a model of $\mathcal{T}_0$.

Example 11 *The combination of the theory of Presburger arithmetic with the theory $\mathsf{Free}(\Sigma)$ of free functions in a set Σ can also be seen as an* extension*:*

- *the base theory is Presburger arithmetic;*
- *the new function symbols are those in Σ; they are not constrained by any axioms.*

Example 12 *The combination of the theory of the reals with the theory of a monotone function f can be also seen as a theory extension:*

- *the base theory $\mathcal{T}_0$ is the theory of real numbers.*
- *Its signature is extended with the function symbol f, subject to the axiom $\mathsf{Mon}(f)$.*

Similarly, we can think of the theory of a Lipschitz function f over the reals as the extension of the theory of reals with a function subject to the Lipschitz condition (which can be expressed as a set of clauses).

4 Decidable theories and theory fragments

Let $\Sigma = (\Omega, \mathsf{Pred})$ be a signature and $\mathcal{T}$ be a Σ-theory (e.g. the class of all models of a class $\mathcal{M}$ of Σ-algebras, or the class of all first order consequences of a set $\mathcal{F}$ of formulae).

Figure 1: Common restrictions on $\mathcal{L}$

	$\mathsf{Pred} = \emptyset$	$\{\phi \in \mathcal{L} \mid \mathcal{T} \models \phi\}$
$\mathcal{L}$=$\{\forall x A(x) \mid A$ atomic$\}$	word problem	
$\mathcal{L}$=$\{\forall x(A_1 \wedge \ldots \wedge A_n \rightarrow B) \mid A_i, B$ atomic$\}$	uniform word problem	$\mathsf{Th}_{\forall\mathsf{Horn}}$
$\mathcal{L}$=$\{\forall x C(x) \mid C(x)$ clause$\}$	clausal validity problem	$\mathsf{Th}_{\forall,\mathsf{cl}}$
$\mathcal{L}$=$\{\forall x \phi(x) \mid \phi(x)$ unquantified$\}$	universal validity problem	$\mathsf{Th}_{\forall}$
$\mathcal{L}$=$\{\exists x A_1 \wedge \ldots \wedge A_n \mid A_i$ atomic$\}$	unification problem	$\mathsf{Th}_{\exists}$
$\mathcal{L}$=$\{\forall x \exists x A_1 \wedge \ldots \wedge A_n \mid A_i$ atomic$\}$	unification with constants	$\mathsf{Th}_{\forall\exists}$

Definition 10 *The theory $\mathcal{T} = \mathsf{Th}(\mathcal{M})$ is* decidable *iff there is an algorithm which, for every closed first order formula ϕ, can decide (after a finite number of steps) whether ϕ is in $\mathcal{T}$ or not. Equivalently, $\mathcal{T} = \mathsf{Th}(\mathcal{M})$ is* decidable *iff there is an algorithm which, for every closed first order formula ϕ, can decide (in finite time) whether $\mathcal{M} \models \phi$ or not. If no such algorithm exists, the theory is said to be* undecidable.

Some examples of undecidable theories are given below:

Example 13 (Peano arithmetic) *Peano arithmetic is undecidable. The related theory of integers (with multiplication): $\mathsf{Th}((\mathbb{Z}, \{0, 1, +, *\}, \{\leq, \approx\}))$ is undecidable as well.*

Example 14 (Uninterpreted function symbols) *The theory $\mathsf{Free}(\Sigma)$ of all Σ-algebras for a given signature Σ is also undecidable: Word problems can be encoded this way; there exist presentations with undecidable word problems.*

Although, as seen above, some theories are undecidable, decision procedures exist if we restrict to certain classes of formulae (for instance only to universally quantified formula, only to existentially quantified formulae, to formulae with alternations of quantifiers of a certain type, or to formulae with a certain syntactic form). Let $\mathcal{T}$ be a first order theory in signature Σ and let $\mathcal{L}$ be a class of (closed) Σ-formulae. Common restrictions on $\mathcal{L}$ can be:

- $\mathcal{L}$=$\{\forall x A(x) \mid A$ atomic$\}$ the class of all universal atomic formulae;
- $\mathcal{L}$=$\{\forall x(A_1 \wedge \ldots \wedge A_n \rightarrow B) \mid A_i, B$ atomic$\}$ the class of universal Horn clauses;
- $\mathcal{L}$=$\{\forall x C(x) \mid C(x)$ clause$\}$ the class of all universally quantified clauses;
- $\mathcal{L}$=$\{\forall x \phi(x) \mid \phi(x)$ quantifier-free$\}$ the class of all universal formulae;
- $\mathcal{L}$=$\{\exists x A_1 \wedge \ldots \wedge A_n \mid A_i$ atomic$\}$ the class of all atomic existential formulae;
- $\mathcal{L}$=$\{\forall x \exists x A_1 \wedge \ldots \wedge A_n \mid A_i$ atomic$\}$ the class of all $\forall\exists$ atomic formulae.

4.1 Problems and relationships between them

Problems of interest are:

The $\mathcal{T}$-validity problem for formulae in $\mathcal{L}$: Given $\phi \in \mathcal{L}$, does $\mathcal{T} \models \phi$?
The $\mathcal{T}$-satisfiability problem for formulae in $\mathcal{L}$: Given $\phi \in \mathcal{L}$, does $\phi \models_{\mathcal{T}} \bot$?

It is easy to see that $\mathcal{T} \models \phi$ iff $\mathcal{T} \cup \neg\phi$ unsatisfiable.

Theorem 11 *The $\mathcal{T}$-validity problem for formulae in $\mathcal{L}$ can be reduced to the $\mathcal{T}$-satisfiability problem for formulae in $\neg\mathcal{L} = \{\neg\phi \mid \phi \in \mathcal{L}\}$ and vice-versa.*

Example 15 *Let $\mathsf{Free}(\Sigma) = \mathsf{Th}(\Sigma\text{-}\mathsf{Alg})$. The following are equivalent:*

Figure 2: Common restrictions on $\mathcal{L}/\neg\mathcal{L}$

$\mathcal{L}$	$\neg\mathcal{L}$
$\{\forall x A(x) \mid A \text{ atomic}\}$	$\{\exists x \neg A(x) \mid A \text{ atomic}\}$
$\{\forall x (A_1 \wedge \ldots \wedge A_n \rightarrow B) \mid A_i, B \text{ atomic}\}$	$\{\exists x (A_1 \wedge \ldots \wedge A_n \wedge \neg B) \mid A_i, B \text{ atomic}\}$
$\{\forall x \bigvee L_i \mid L_i \text{ literals}\}$	$\{\exists x \bigwedge L'_i \mid L'_i \text{ literals}\}$
$\{\forall x \phi(x) \mid \phi(x) \text{ unquantified}\}$	$\{\exists x \phi'(x) \mid \phi'(x) \text{ unquantified}\}$

- $\mathsf{Free}(\Sigma) \models \forall x \forall y ((x \approx f(y) \wedge y \approx h(z)) \rightarrow g(x) \approx f(h(z)))$;
- $\mathsf{Free}(\Sigma) \wedge \exists x \exists y (x \approx f(y) \wedge y \approx h(z) \wedge g(x) \not\approx f(h(z)))$ *unsatisfiable;*
- $\mathsf{Free}(\Sigma) \wedge c \approx f(d) \wedge d \approx h(e) \wedge g(c) \not\approx f(h(e))$ *unsatisfiable (where c, d, e are new Skolem constants).*

Definition 12 *A first order Σ-theory $\mathcal{T}$ is Σ_0-convex ($\Sigma_0 \subseteq \Sigma$) iff whenever $\mathcal{T} \models \bigwedge_{i=1}^n A_i \rightarrow \bigvee_{j=1}^m B_j$, where $A_1, \ldots, A_n$ are Σ-atoms, and $B_1, \ldots, B_m$ are Σ_0-atoms there exists $k \in \{1, \ldots, m\}$ such that $\mathcal{T} \models \bigwedge_{i=1}^n A_i \rightarrow B_k$.*

Example 16 *Theories axiomatized by sets of Horn clauses are convex. (Any theory $\mathcal{T}$ such that $\mathsf{Mod}(\mathcal{T})$ closed under products is convex.)*

Example 17 *Let $LI(\mathbb{Q})$ be the theory of rational numbers with linear arithmetic. $LI(\mathbb{Q})$ is convex w.r.t. equality atoms: if $LI(\mathbb{Q}) \models \bigwedge_i A_i \rightarrow \bigvee_j t_i \approx t'_i$ then $LI(\mathbb{Q}) \models \bigwedge_{i=1}^n A_i \rightarrow t_k \approx t'_k$ for some k. $LI(\mathbb{Q})$ is not convex w.r.t. inequality atoms: $LI(\mathbb{Q}) \models x \leq y \vee y \leq x$ but $LI(\mathbb{Q}) \not\models x \leq y$ and $LI(\mathbb{Q}) \not\models y \leq x$.*

Theorem 13 *The following reductions hold:*

(1) If $\mathcal{T}$-validity for $\mathsf{Th}_{\forall,\mathsf{cl}}$ is decidable then $\mathcal{T}$-validity for $\mathsf{Th}_{\forall}$ is decidable.

(2) For any convex theory $\mathcal{T}$, if $\mathcal{T}$-validity for $\mathsf{Th}_{\forall,\mathsf{Horn}}$ is decidable then $\mathcal{T}$-validity for $\mathsf{Th}_{\forall,\mathsf{cl}}$ is decidable.

(3) If $\mathcal{T}$-satisfiability for conjunctions of (quantifier-free) literals is decidable, then $\mathsf{Th}_{\forall,\mathsf{cl}}$ is decidable.

4.2 SAT checking modulo a theory

In most cases, decision procedures can efficiently handle simple problems, such as, for instance the problem of checking the satisfiability of (quantifier-free) conjunctions of literals. Before considering more complex satisfiability problems, we recall the methods for proving satisfiability of propositional formulae. The problem can be formulated as follows:

Input	a propositional formula ϕ
Output	YES if there is a Boolean assignment satisfying ϕ
	NO is there is no Boolean assignment satisfying ϕ

The problem is known to be NP-complete. By applying linear time structural transformations we transform any propositional formula in clause form. We can therefore assume that ϕ is a conjunction of clauses.

Modern SAT solvers are based on highly optimized versions of the classical DPLL procedure (Davis-Putnam 1960 + Davis-Logemann-Loveland 1962). The DPLL procedure:

- performs deterministic choices first (unit resolution, backward subsumption, pure literal assignment);

- when this is not possible anymore: choose an atom for case distinction (semantic splitting);
- makes use of appropriate heuristics such as: selection criteria for splitting atoms, non-chronological back-tracking, conflict-driven learning.

State of the art SAT solvers such as for instance MINISAT (`http://minisat.se/`) or Chaff (`http://www.princeton.edu/~chaff/`) currently handle problems with ca. 10K variables. Efficiency of SAT solvers increased their application domains (planning, bounded model checking, security, description logics). However, many applications require solving satisfiability problems for quantifier-free formulae modulo a first order theory $\mathcal{T}$. Systems implementing methods for solving such specialized problems (Yices, Z3, BarcelogicTools, CVC Lite, Math-SAT) are called **S**(atisfiability) **M**(odulo) **T**(heory) solvers. The structure of an SMT solver is roughly as follows:

- we assume w.l.o.g. that the formula is a set of clauses (a set of disjunctions of literals) in the language of the theory $\mathcal{T}$;
- a Boolean assignment for the literals in the formula is found, and is then checked for $\mathcal{T}$-satisfiability (lazy approach);
- the assignment may be partial and checked before splitting (early pruning);
- usual heuristics like non-chronological backtracking and learning are employed.

DPLL($\mathcal{T}$) (cf. e.g. [27]) is an example of a formally structured extension of DPLL which is able to cope with the aspects above.

4.3 Methods for proving decidability

Among the methods for proving decidability of theories or fragments thereof we mention the following:

(1) *Finite/Bounded model property.* Prove that every satisfiable formula has a model of bounded cardinality.

(2) *Show that special calculi provide decision procedures.* Resolution, rewriting, superposition, tableau or sequent calculi can be used as decision procedures for certain fragments of first order logic.

(3) Use *quantifier elimination* to reduce the problem of testing validity/satisfiability of formulae with alternations of quantifiers to the problem of testing validity/satisfiability of formulae without quantifiers.

(4) *Reduction to problems known to be decidable.* For instance:

   (a) encoding to problems in automata theory (used for instance to prove decidability of Presburger arithmetic: validity/satisfiability of formulae is encoded to problems related to emptiness of languages in automata theory).

   (b) encoding into decidable fragments of FOL (this is a method which has been widely used in the study of non-classical logics).

We mention how the methods above allow us to prove decidability of the following theories.

Example 18 (Linear rational arithmetic, ($\mathsf{LI}(\mathbb{Q})$) *Linear rational arithmetic is decidable: this can be shown by giving a decision procedure for the quantifier-free fragment and then a method for eliminating quantifiers. Several methods can be used for this. The most common ones are the Fourier-Motzkin method ([23], cf. also [41]) and the test point method [69].*

Example 19 (Presburger arithmetic) *Presburger arithmetic is decidable [48]. A possibility of proving decidability of Presburger arithmetic uses an encoding of the problem of checking whether a quantifier-free formula is satisfiable to the problem of checking whether the language accepted by an automaton associated with the formula is empty.*

Example 20 (The theory of real numbers) *The theory of real numbers*

$$\mathsf{Th}(\mathbb{R}, \{+/2, */2\}, \{\leq /2, \approx\})$$

is decidable in 2EXPTIME [62]. For proving this, Tarski uses quantifier elimination.

Examples of theories with a decidable $\mathcal{T}$-validity problem for the universal theory (or, equivalently with a decidable satisfiability problem for ground clauses):

Example 21 *The following theories have a decidable universal theory:*

- *The theory* $\mathsf{Free}(\Sigma)$ *of uninterpreted function symbols in a set* Σ*. There exist standard decision procedures for this such as congruence closure.*
- *Linear rational or integer arithmetic.*
- *Theories axiomatizing common datatypes (lists, arrays) in Examples 5,6,7. Several methods can be used: for instance extensions of congruence closure for (acyclic) lists, instantiation-based methods, or superposition [2].*
- *Algebraic counterparts of modal logics (theories axiomatizing varieties of Boolean algebras with operators). Such decidability results can be obtained for instance by proving the bounded model property, or by using tableaux or resolution as decision procedures.*

$\mathcal{T}$-satisfiability vs. constraint solving. The field of constraint solving also deals with satisfiability problems. But one must be careful: In constraint solving one is interested if a formula is satisfiable in a given, fixed model of $\mathcal{T}$, whereas in $\mathcal{T}$-satisfiability one is interested if a formula is satisfiable at all – in some model of $\mathcal{T}$.

5 Tractability

Since in mathematics, databases, and verification the problems to be solved consist of a large number of formulae, a large amount of effort has been dedicated to the study of tractable theories, i.e. theories in which satisfiability or entailment of (ground) Horn clauses can be checked in polynomial time. To address this problem, essentially very similar ideas occurred in various areas: proof theory and automated deduction, databases, mathematics (especially algebra) and verification.

Databases: The inference rules of a deductive database are usually of a special form (known as datalog program): typically a set of universal Horn clauses which do not contain function symbols. Any datalog program defines an inference relation for which entailment of ground clauses is decidable in polynomial time [66, 67].

Proof theory: Possibilities of restricting the search space in inference systems without loss of completeness were studied by McAllester and Givan in [32, 42, 33]. They introduced so-called "local inference systems", which can be modeled by sets of rules (or sets of Horn clauses N) with the property that for any ground Horn clause G, it is guaranteed that if G can be proved using N then G can already be proved by using only those instances $N[G]$ of N containing only ground terms occurring in G or in N. They proved that the notion of locality captures PTIME, i.e. that any problem solvable in PTIME can be encoded as the entailment problem w.r.t. a local inference system.

Mathematics: Similar ideas also occurred in algebra. To prove that the uniform word problem for lattices is decidable in polynomial time, Skolem [50] used the following idea: replace the lattice operations $\vee$ and $\wedge$ by ternary relations $r_\vee$ and $r_\wedge$, required to be functional, but not necessarily total. The lattice axioms were translated to a relational form, by flattening them and then replacing every atom of the form $x \vee y \approx z$ with $r_\vee(x, y, z)$ (similarly for $\wedge$-terms). Additional axioms were added, stating that equality is an equivalence and that the relations are compatible with equality and functional. This new presentation, consisting only of Horn, function-free clauses[1], can be used for deciding in polynomial time the uniform word problem for lattices. The correctness and completeness of the method relies on the fact that every partially-ordered set (where $\vee$ and $\wedge$ are partially defined) embeds into a lattice. A similar idea was used by Evans in the study of classes of algebras with a PTIME decidable word problem [20]. The idea was extended by Burris [13] to quasi-varieties of algebras. He proved that if a quasi-variety axiomatized by a set $\mathcal{K}$ of Horn clauses has the property that *every finite partial algebra which is a partial model of the axioms in $\mathcal{K}$ can be extended to a total algebra model of $\mathcal{K}$* then the uniform word problem for $\mathcal{K}$ is decidable in polynomial time. In [26], Ganzinger established a link between the proof theoretic notion of locality and embeddability of partial into total algebras. In [28, 52] the notion of locality for Horn clauses is extended to the notion of *local extension* of a base theory.

Verification: Apparently independently, similar phenomena were studied in the verification literature, mainly motivated by the necessity of devising methods for efficient reasoning in theories of data structures. In [45], McPeak and Necula investigate local reasoning in pointer data structures, with the goal of efficiently proving invariants in programs dealing with pointers. Locality considerations also occur in the study of a theory of arrays by Bradley, Manna and Sipma [11]. Since the theories from verification mentioned above are in fact combinations of theories, they will be discussed separately in what follows.

Because of the importance of the concept of locality in databases, logic, mathematics and verification, in what follows we give a short introduction to Datalog and we present a summary of results in the area of local theories.

5.1 Datalog

Datalog is a rule-based database query language that syntactically is a subset of Prolog. In contrast to Prolog, it does not allow complex terms as arguments of predicates (e.g. $P(1, 2)$ is admissible but not $P(f(1), 2)$) and it imposes certain restrictions on the use of negation and recursion.

Definition 14 *A* Datalog rule *is an expression of the form $B_1 \wedge \cdots \wedge B_n \rightarrow H$, where H (the head of the rule) is an atomic formula $R(u_1, \ldots, u_n)$, and $B_1 \wedge \cdots \wedge B_n$ (the body of the rule) is a conjunction of literals of the form $S(v_1, \ldots, v_m)$ or $\neg S(v_1, \ldots, v_m)$, where $(u_1, \ldots, u_n)$, $(v_1, \ldots, v_m)$ are tuples of variables or constants. A Datalog rule is* positive *if it does not contain negative literals.*

A Datalog program *is a finite collection of rules such that none of its head predicates occurs negated in the body of any rule. The predicates that appear only in the bodies of the rules are called* input predicates.

In what follows we identify a relational database with a finite relational structure.

[1] The encoding Skolem uses is an encoding to Datalog. A similar encoding is used by Burris in [13].

Definition 15 *A* Datalog query *is a pair* (Prog, R) *consisting of a Datalog program and a designated head predicate. With every finite relational structure* $\mathcal{A}$*, the query* (Prog, R) *associates the result* $(\mathsf{Prog}, R)^{\mathcal{A}}$*, the interpretation of* R *as computed by* Prog *from the input* $\mathcal{A}$*.*

Example 22 *A Datalog program consisting of clauses which describes the rules which define the ancestor relationship,*

$$\begin{array}{rcl} \mathsf{parent}(X,Y) & \rightarrow & \mathsf{ancestor}(X,Y) \\ \mathsf{ancestor}(X,Z) \wedge \mathsf{ancestor}(Z,Y) & \rightarrow & \mathsf{ancestor}(X,Y) \end{array}$$

A database describes a set of facts (unit ground clauses which are supposed to hold):

$$\begin{array}{l} \mathsf{parent}(\mathsf{bill}, \mathsf{mary}) \\ \mathsf{parent}(\mathsf{mary}, \mathsf{john}). \end{array}$$

The query $\mathsf{ancestor}(\mathsf{bill}, X)$ *asks for all ancestors of bill and would return mary and john when posed to a Datalog system containing the facts and rules described above.*

Datalog programs always express queries that are computable in time bound by a polynomial in the size of the underlying database. Moreover, Datalog can capture queries that are complete for polynomial time computations (e.g. the path systems query [16]).

The exact expressive power of Datalog is completely understood in certain cases: if only databases with a successor relation and first and last element are considered, then a query is expressible by a Datalog program if and only if it is computable in polynomial time [68, 38, 47]. (Without this assumption, some PTIME problems cannot be expressed in Datalog [15], see also [40].)

5.2 Local theories

Local theories (or inference systems) were first introduced and studied by McAllester and Givan in [32, 42, 33].

Definition 16 (Local theories) *A* local theory *is a set of Horn clauses* $\mathcal{K}$ *such that, for any ground Horn clause* C*,* $\mathcal{K} \models C$ *only if already* $\mathcal{K}[C] \models C$ *(where* $\mathcal{K}[C]$ *is the set of instances of* $\mathcal{K}$ *in which all terms are subterms of ground terms in either* $\mathcal{K}$ *or* $\mathcal{C}$*).*

The size of $\mathcal{K}[G]$ is polynomial in the size of G for a fixed $\mathcal{K}$. Since satisfiability of sets of ground Horn clauses can be checked in linear time [19], it follows that for local theories, validity of ground Horn clauses can be checked in polynomial time. Givan and McAllester proved that every problem which is decidable in PTIME can be encoded as an entailment problem of ground clauses w.r.t. a local theory [33]. An example of a local theory (cf. [33]) is the set of axioms of a monotone function w.r.t. a transitive relation $\leq$:

$$\mathcal{K} = \{x \leq y \wedge y \leq z \rightarrow x \leq z, \quad x \leq y \rightarrow f(x) \leq f(y)\}.$$

Another example provided in [33] is a local axiom set for reasoning about a lattice (similar to that proposed by Skolem in [50]). In [9, 8], Ganzinger and Basin defined the more general notion of *order locality* and showed how to recognize (order-)local theories and how to use these results for automated complexity analysis. Given a term ordering $\succ$, we say that a set $\mathcal{K}$ of clauses entails a clause C bounded by $\succ$ (notation: $\mathcal{K} \models_{\preceq} C$), if and only if there is a proof of $\mathcal{K} \models C$ from those ground instances of clauses in $\mathcal{K}$ in which (under $\succeq$) each term is smaller than or equal to some term in C.

Definition 17 (Order locality, [9, 8]) *A set of clauses* $\mathcal{K}$ *is local with respect to* $\succ$ *if whenever* $\mathcal{K} \models C$ *for a ground clause* C*, then* $\mathcal{K} \models_{\preceq} C$*.*

Theorem 18 ([9, 8]) *Let $\succ$ be a (possibly partial) term ordering and $\mathcal{K}$ be a set of clauses. Assume that $\mathcal{K}$ is saturated with respect to $\succ$-ordered resolution, and let C be a ground clause. Then $\mathcal{K} \models C$ for a ground clause C if and only if $\mathcal{K} \models_{\preceq} C$, i.e. $\mathcal{K}$ is local with respect to $\succ$.*

The converse of this theorem is not true in general. Ganzinger and Basin established conditions under which the converse holds – they use a hyperresolution calculus and identify conditions when for Horn clauses order locality is equivalent to so-called *peak saturation* (Theorems 4.4–4.7 in [8]). These results are obtained for first order logic without equality. In [26], Ganzinger established a link between proof theoretic and semantic concepts for polynomial time decidability of uniform word problems which had already been studied in algebra [50, 20, 13]. He defined two notions of locality for equational Horn theories, and established relationships between these notions of locality and corresponding semantic conditions, referring to embeddability of partial algebras into total algebras. Theorem 18 also can be used for recognizing equational Horn theories:

Theorem 19 ([26]) *Let $\mathcal{K}$ be a set of Horn clauses. Then $\mathcal{K}$ is a local theory in logic with equality if and only if $\mathcal{K} \cup EQ$ is a local theory in logic without equality, where EQ denotes the set of equality axioms consisting of reflexivity, symmetry, transitivity, and of congruence axioms for each function symbol in the signature.*

Theorems 19 and 18 were used in [26] for proving the locality of the following presentation Int of the set of integers with successor and predecessor by saturation:

$$\begin{array}{llllll} (1) & p(x) \approx y & \rightarrow & s(y) \approx x & (3) & p(x) \approx p(y) \rightarrow y \approx x \\ (2) & s(x) \approx y & \rightarrow & p(y) \approx x & (4) & s(x) \approx s(y) \rightarrow y \approx x \end{array}$$

The presentation Int' of integers with successor and predecessor consisting of the axioms (1) and (2) alone (without the injectivity conditions (3) and (4)) is not local but it is *stably local*: in order to disprove a ground set G of clauses only those ground instances $\mathsf{Int}'^{[G]}$ of Int' are needed where variables are mapped to subterms occurring in G. (Note that $\mathsf{Int}' \cup EQ$ is not saturated under ordered resolution; when saturating it the injectivity axioms are generated.)

6 Efficient reasoning in complex theories

As mentioned before, in most applications theories do not appear alone and combinations or extensions of theories need to be considered. A major challenge is to identify situations where reasoning in complex theories can be done efficiently and accurately. Efficiency can be achieved for instance by:

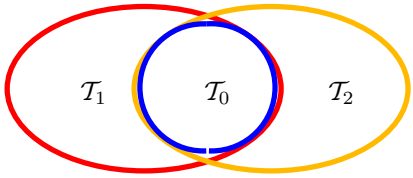
(1) reducing the search space (preferably without losing completeness);
(2) modular reasoning, i.e., delegating some proof tasks which refer to a specific theory to provers specialized in handling formulae of that theory.

Accuracy means that we are interested in situations when *sound and complete* methods exist.

6.1 Modularity

An important aspect in reasoning in complex theories is *modularity*. Given two theories $\mathcal{T}_1$ and $\mathcal{T}_2$, possibly sharing a common subtheory $\mathcal{T}_0$, it is very important to know if we can use provers for the theories $\mathcal{T}_1$ and $\mathcal{T}_2$ as "blackboxes" in order to prove theorems in $\mathcal{T}_1 \cup \mathcal{T}_2$, and if so which amount of communication between the provers for the component theories is necessary in order to guarantee completeness (at least for certain proof tasks). The situation is described in Figure 3.

Figure 3: Modular reasoning in combinations of theories



$\mathcal{T}_0$: Σ_0-theory.
$\mathcal{T}_i$: Σ_i-theory; $\mathcal{T}_0 \subseteq \mathcal{T}_i$ $\Sigma_0 \subseteq \Sigma_i$.

Example:
$\mathbf{lists}(\mathbb{R}) \cup \mathbf{arrays}(\mathbb{R})$

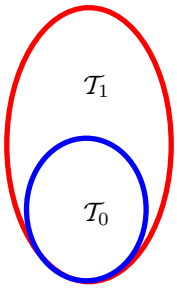
Question: Can we use provers for $\mathcal{T}_1, \mathcal{T}_2$ as blackboxes to prove theorems in $\mathcal{T}_1 \cup \mathcal{T}_2$?

Example 23 *For instance, if we want to prove theorems involving statements about lists of real numbers and arrays of real numbers: is it sufficient to use as black boxes a prover for lists over the reals and one for arrays over the reals? Which information about real numbers needs to be propagated between the two provers in order to guarantee completeness?*

6.2 Hierarchical reasoning in theory extensions

When reasoning in extensions of a theory $\mathcal{T}_0$ with additional functions subject to certain axioms it is important to know: Can we use a prover for $\mathcal{T}_0$ for proving theorems in $\mathcal{T}_1$?

Figure 4: Hierarchical reasoning in theory extensions



$\mathcal{T}_1$: Σ_1-theory; $\mathcal{T}_0 \subseteq \mathcal{T}_1$ $\Sigma_0 \subset \Sigma_1$

$\mathcal{T}_0$: Σ_0-theory.

Example:
$f : \mathbb{R} \to \mathbb{R}$
monotone

$\mathbb{R}$

Question: Can we use a prover for $\mathcal{T}_0$ as a blackbox to prove theorems in $\mathcal{T}_1$?

Example 24 *Assume that we need to prove some properties about monotone functions over real numbers. Can we reduce the proof task to a proof task w.r.t. the real numbers which can be solved by specialized prover for real numbers?*

We start by presenting the type of complex theories we have in mind (combinations of theories and theory extensions) and the main problems when reasoning in extensions and combinations of theories. In Section 7 we talk about theory combinations. In Section 8 we talk about theory extensions.

7 Combinations of theories

We consider possibility of efficient reasoning in combinations of theories. One important property of theories is consistency:

Definition 20 *A theory is consistent if it has at least one model.*

The following natural question arises: Is the union of two consistent theories always consistent? The answer is in general no, even when the two theories have disjoint signatures (everywhere in what follows we assume that equality is built-in and that theories with disjoint signatures share the equality predicate).

Example 25 ([63]) *Let* $\Sigma_1 = (\Omega_1, \emptyset)$, $\Sigma_2 = (\{c/0, d/0\}, \emptyset)$, *such that* $c, d \notin \Omega_1$. *Let* $\mathcal{T}_1$ *be axiomatized by* $\{\exists x, y, z(x \not\approx y \ \wedge \ x \not\approx z \ \wedge \ y \not\approx z)\}$, *and let* $\mathcal{T}_2$ *be axiomatized by* $\{\forall x(x \approx c \ \vee \ x \approx d)\}$. *It is easy to see that*

(1) $\mathcal{A} \in \mathsf{Mod}(\mathcal{T}_1)$ *iff* $|A| \geq 3$ *and*

(2) $\mathcal{B} \in \mathsf{Mod}(\mathcal{T}_2)$ *iff* $|A| \leq 2$.

The second natural question is whether decidability (of a specific fragment) is preserved when combining theories.

Figure 5: The decidability problem for complex theories

Given: For $i = 1, 2$

- $\mathcal{T}_i$ a first order theory in signature Σ_i
- $\mathcal{L}_i$ a class of (closed) Σ_i-formulae

Assumption: the $\mathcal{T}_i$-validity problem for $\mathcal{L}_i$ is decidable

Let $\mathcal{T}_1 \bigoplus \mathcal{T}_2$ be a combination of $\mathcal{T}_1$ and $\mathcal{T}_2$

Let $\mathcal{L}_1 \bigoplus \mathcal{L}_2$ be a combination of $\mathcal{L}_1$ and $\mathcal{L}_2$

Question: Is the $\mathcal{T}_1 \bigoplus \mathcal{T}_2$-validity problem for $\mathcal{L}_1 \bigoplus \mathcal{L}_2$ decidable?

The answer depends on the way the combination of the two theories is defined, and on the way the combination of the fragments $\mathcal{L}_1$ and $\mathcal{L}_2$ is defined.

In what follows we consider combinations of theories as defined in Section 3.3. In general, decidability is not preserved under such combinations. Restrictions are needed for an affirmative answer. Example 26 shows that even if two theories have decidable word problems [2] (i.e. validity of universally quantified unit positive clauses is decidable in the component theories) this problem may be undecidable in the combination.

Example 26 *Let* $\mathcal{A}$ *be the theory of one associative binary operation, and let* $\mathcal{G}$ *consist of a finite set of ground equations over a binary operation such that* $\mathcal{G}$ *is the presentation for a semigroup with an undecidable word problem. (Finitely-presented semigroups with undecidable word problem exist by a result by Matijasevic.) It can be checked that the word problem is decidable for* $\mathcal{A}$ *and for* $\mathcal{G}$, *but it is undecidable for* $\mathcal{A} \cup \mathcal{G}$.

[2] The word problem for a theory $\mathcal{T}$ is the problem of deciding whether $\mathcal{T} \models \forall x(s \approx t)$.

Simpler instances of the combination problem are needed for guaranteeing that the decidability is transfered under composition:

- combinations of theories over disjoint signatures,
- combinations of theories sharing constructors,
- combinations of theories satisfying certain model theoretic compatibility conditions with the shared theory.

In this section we will study situations in which the decidability of the problem of testing satisfiability of ground formulae in the component theories implies the decidability of the problem of testing satisfiability of ground formulae w.r.t. the combination of theories. Important in this context will also the the possibility of finding *modular* decision procedures. Methods for checking satisfiability of conjunctions of ground literals in combinations of theories which have *disjoint signatures* or only share constants are well studied. However, also in this case there cannot be a general, effective, method for reasoning in combinations of theories always leading to a complete algorithm.

Theorem 21 ([10]) *There exist theories $\mathcal{T}_1, \mathcal{T}_2$ with disjoint signatures with decidable ground satisfiability problem such that ground satisfiability in $\mathcal{T}_1 \cup \mathcal{T}_2$ is undecidable.*

Complete methods can only be obtained under additional assumptions on the component theories. The Nelson-Oppen combination procedure [46] for instance, can be applied for combining decision procedures of *stably infinite* theories over disjoint signatures. Resolution-based methods have also been used in this context [2, 1].

7.1 Combinations of theories with disjoint signatures

We present a standard method for reasoning in combinations of theories with disjoint signatures, the Nelson-Oppen method [46]. We then mention some extensions without going into detail.

7.1.1 The Nelson-Oppen combination procedure.

The Nelson-Oppen combination procedure is a method for testing satisfiability of conjunctions of ground literals w.r.t. a combination of theories over a non-disjoint signature.
Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be two theories with signatures Σ_1, Σ_2. Assume that Σ_1 and Σ_2 share only constants and the equality predicate. The problem we consider is the following: Let ϕ be a ground formula over the signature $(\Sigma_1 \cup \Sigma_2)^c = (\Sigma_1 \cup \Sigma_2 \cup C)$ (the extension of the union $\Sigma_1 \cup \Sigma_2$ with a countably infinite set C of constants). We want to test whether ϕ is satisfiable w.r.t. $\mathcal{T}_1 \cup \mathcal{T}_2$.

In order to test the satisfiability of ϕ we first purify it, i.e. we find formulae ϕ_1, ϕ_2 s.t. ϕ_i is a pure Σ_i-formula and $\phi_1 \wedge \phi_2$ is equivalent with ϕ.

Step 1: Purification. We purify the formula ϕ as follows:

(1) Purify all terms by replacing, in a bottom-up manner, the "alien" subterms in ϕ (i.e. terms starting with a function symbol in Σ_i occurring as arguments of a function symbol in Σ_j, $j \neq i$) with new constants (from a countably infinite set C of constants). The transformations are schematically represented as follows:

$$(\neg)P(\dots, g(..., f(t_1, \dots, t_n), ...), \dots) \quad \mapsto \quad (\neg)P(\dots, g(..., u, ...), \dots) \wedge u \approx t$$

where $t \approx f(t_1, \dots, t_n), f \in \Sigma_1, g \in \Sigma_2$ (or vice versa).

(1) Purify mixed equalities and inequalities by adding additional constants and performing the following transformations (where $f \in \Sigma_1$ and $g \in \Sigma_2$ or vice versa):

$$\begin{array}{lcl} f(s_1,\ldots,s_n) \approx g(t_1,\ldots,t_m) & \mapsto & u{\approx}f(s_1,\ldots,s_n) \wedge u{\approx}g(t_1,\ldots,t_m) \\ f(s_1,\ldots,s_n) \not\approx g(t_1,\ldots,t_m) & \mapsto & u{\approx}f(s_1,\ldots,s_n) \wedge v{\approx}g(t_1,\ldots,t_m) \wedge u \not\approx v \end{array}$$

(3) Purify mixed literals by renaming alien terms:

$$(\neg)P(\ldots,s_i,\ldots) \quad \mapsto \quad (\neg)P(\ldots,u,\ldots) \wedge u{\approx}s_i$$

if P is a predicate symbol in Σ_1 and s_i is a Σ_2^c-term (or vice versa).

The purification procedure terminates in linear time in the size of the formula and returns a purified conjunction $\phi_1 \wedge \phi_2$ (of size linear in the size of ϕ) with the following properties:

(i) ϕ_i is a ground Σ_i-formula for $i = 1, 2$;
(ii) $\phi_1 \wedge \phi_2$ and ϕ are satisfiable in exactly the same models of $\mathcal{T}_1 \cup \mathcal{T}_2$ (we regard the constants as existentially quantified variables).

We can now use a decision procedure for $\mathcal{T}_i$ for testing whether $\phi_i \models_{\mathcal{T}_i} \bot$ for $i = 1, 2$. If ϕ_i is unsatisfiable in $\mathcal{T}_i$ for $i = 1$ or $i = 2$ then we know that ϕ must be unsatisfiable. If ϕ_1 and ϕ_2 are satisfiable we do not have the guarantee that ϕ is satisfiable: in a second step we need to exchange some information about the shared constants (or alternatively existentially quantified variables[3]) between the provers.

Step 2: Propagation. We deduce and propagate all possible (disjunctions of) equalities between constants entailed by the components. There are two possibilities for doing this: backtracking/case split or (non-deterministic) guessing.

1. Propagation: backtracking version This is done as follows:

(1) *Propagate equalities between different shared constants.* If ϕ_1 entails an equality between shared constants not entailed by ϕ_2 then add the equality as a new conjunct to ϕ_2 (and vice versa).

(2) *Case split necessary?* If either ϕ_1 or ϕ_2 entails a disjunction $u_1{\approx}v_1 \vee \cdots \vee u_k{\approx}v_k$ without entailing any of the equalities alone, then we apply the procedure recursively to $\phi_1 \wedge \phi_2 \wedge u_i{\approx}v_i$, $1 \leq i \leq k$.

If any of these formulae is satisfiable then return "satisfiable", else return "unsatisfiable".

2. Propagation: guessing version. Let C_0 be the (finite) set of constants common to ϕ_1 and ϕ_2. Guess a possible partition $\mathcal{P}$ on C, and let R be the associated equivalence relation. Let $ar(C_0, R)$ be:

$$\{c \approx d \mid c, d \in C_0, R(c,d)\} \cup \{c \not\approx d \mid c, d \in C_0, \text{ not } R(c,d)\}.$$

Use decision procedures for testing decidability of ground conjunctions of literals in $\mathcal{T}_1$ resp. $\mathcal{T}_2$ to check whether $\phi_1 \wedge ar(C_0, R) \models_{\mathcal{T}_1} \bot$ or $\phi_2 \wedge ar(C_0, R) \models_{\mathcal{T}_2} \bot$.

If one of the formulae is unsatisfiable, then return: "ϕ is unsatisfiable". If both formulae are satisfiable then return "ϕ is satisfiable".

[3]We emphasize the formulation in which one mentions shared variables due to historical reasons

Remark. The advantage of the first option (backtracking) is that the procedure can be made deterministic if the theories $\mathcal{T}_i$ are *convex*. It can be in fact seen that if $\mathcal{T}_i$ are *convex* and if satisfiability of conjunctions of ground literals in $\mathcal{T}_i$ can be checked in PTIME then satisfiability of conjunctions of ground literals in $\mathcal{T}_1 \cup \mathcal{T}_2$ can be checked in PTIME.
The advantage of the second option (guessing) is that whenever constraints are represented as boolean combinations of atoms one may combine heuristics of SMT solvers for the specific theories in order to efficiently produce the right arrangement.

Termination, Soundness and Completeness. Termination is obvious, since the number of shared constants is finite. We prove soundness and completeness for the second variant of the propagation step (the guessing version). Let C_0 be the (finite) set of constants common to ϕ_1 and ϕ_2. For every partition $\mathcal{P}$ on C_0, let R be the associated equivalence relation, and $ar(C_0, R)$ be:

$$\{c \approx d \mid c, d \in C_0, R(c,d)\} \cup \{c \not\approx d \mid c, d \in C_0, \text{ not } R(c,d)\}.$$

We first prove soundness.

Theorem 22 *If for any partition of C_0, $\phi_i \wedge ar(C_0, R) \models_{\mathcal{T}_i} \bot$ for some $i \in \{1,2\}$ then ϕ is unsatisfiable.*

Proof: Assume that ϕ is satisfiable. There there exists a satisfying assignment v for $\phi_1 \wedge \phi_2$. Let R be relation on the set C of shared constants defined by $R(c,d)$ iff $v(c) = v(d)$. Then $\phi_1 \wedge ar(C_0, R)$ and $\phi_2 \wedge ar(C_0, R)$ are both satisfiable w.r.t. the theories $\mathcal{T}_1$ w.r.t. $\mathcal{T}_2$. □

In order to prove completeness we need to show that if there exists a partition of C_0 such that for $i = 1, 2$, $\phi_i \wedge ar(C_0, R)$ is satisfiable w.r.t. $\mathcal{T}_i$ then ϕ satisfiable. However in general this is not true as the following example shows:

Example 27 ([63]) *Let $\mathcal{T}_1$ be the theory axiomatized by:*

$$\begin{aligned} f(g(x), g(y)) &\approx x \\ f(g(x), h(y)) &\approx y \end{aligned}$$

and $\mathcal{T}_2$ be the free theory of a unary function k. Let $\phi = g(c) \approx h(c) \wedge k(c) \not\approx c$. ϕ is in purified form. It is easy to see that $g(c) \approx h(c)$ is satisfiable in $\mathcal{T}_1$ and $k(c) \not\approx c$ is satisfiable in $\mathcal{T}_2$. There are no entailed non-trivial equations between the shared constants. However, ϕ is unsatisfiable in $\mathcal{T}_1 \cup \mathcal{T}_2$: if $A \in \mathsf{Mod}(\mathcal{T}_1 \cup \mathcal{T}_2)$ is a model of ϕ then $A \models g(c) \approx h(c)$, hence for all $a \in A$, $a = f_A(g_A(a), g_A(c_A)) = f_A(g_A(a), h_A(c_A)) = c_A$. Then A has only one element, so it cannot satisfy $k(c) \not\approx c$.

The problem in Example 27 is that the formula ϕ_1 has only models of bounded cardinality. In order to avoid to this type of problems – which lead to loss of completeness – we consider in what follows theories in which every formula which has a model has also an infinite model.

Definition 23 *A theory $\mathcal{T}$ is* stably infinite *iff for every quantifier-free formula ϕ, ϕ is satisfiable in $\mathcal{T}$ iff ϕ is satisfiable in an infinite model of $\mathcal{T}$.*

Under the additional hypothesis that $\mathcal{T}_1$ and $\mathcal{T}_2$ are stably infinite we can prove that the Nelson-Oppen combination method is complete.

Theorem 24 *Assume that $\mathcal{T}_1$ and $\mathcal{T}_2$ are stably infinite. With the notations used previously, assume that there exists a partition of C_0 such that for $i = 1, 2$, $\phi_i \wedge ar(C_0, R)$ is satisfiable w.r.t. $\mathcal{T}_i$. Then ϕ satisfiable.*

Proof: Assume that for $i = 1, 2$, $\phi_i \wedge ar(C_0, R)$ is satisfiable w.r.t. $\mathcal{T}_i$. By stable infinity, $\phi_i \wedge ar(C_0, R)$ have both an infinite (hence a countably infinite) model. Let A_i be a countably infinite model of $\phi_i \wedge ar(C_0, R)$. For every i, let $\{v_i(c) \mid c \in C_0\} \subseteq A_i$ be the interpretation of the shared constants in C_0. Let $i : A_1 \to A_2$ be a bijection with the property that $i(v_1(c)) = v_2(c)$ for every $c \in C_0$. We transfer the Σ_1-structure, as well as the constants occurring in ϕ_1, from A_1 to A_2 via the bijection i as follows: if $f \in \Sigma_1$, and $a_1, \ldots, a_n \in A_2$, we define

$$f_{A_2}(a_1, \ldots, a_n) := i(f_{A_1}(i^{-1}(a_1), \ldots, i^{-1}(a_n))).$$

Let A be the $(\Sigma_1 \cup \Sigma_2)$-algebra obtained this way. It is not difficult to check that $A \models_{\mathcal{T}_1 \cup \mathcal{T}_2} \phi$. □

Example 28 *We can use the Nelson-Oppen method for checking the satisfiability of conjunctions of ground literals in the combination of any stably infinite theory $\mathcal{T}_1$ with signature Σ_1 (with decidable ground satisfiability) with the theory of free (or uninterpreted) functions symbols $\mathsf{Free}(\Sigma_2)$ (cf. definition in Example 4), where $\Sigma_1 \cap \Sigma_2 = \emptyset$. In Section 8.1 we will show (using an alternative approach) that the condition that $\mathcal{T}_1$ needs to be stably infinite can be safely dropped without influencing the decidability of ground satisfiability in the combination $\mathcal{T}_1 \cup \mathsf{Free}(\Sigma_2)$.*

Example 29 *We illustrate the Nelson-Oppen procedure on the example given in the original paper of Nelson and Oppen [46]. Consider the following theories with mutually disjoint signatures:*

(1) $LI(\mathbb{Q})$ linear rational arithmetic;
(2) The theory $\mathcal{T}_{\mathsf{lists}}$ of lists in Example 5;
(3) $\mathsf{Free}(\Sigma)$ the theory of uninterpreted function symbols in Σ.

We want to check whether

$$LI(\mathbb{Q}) \cup \mathcal{T}_{\mathsf{lists}} \cup \mathsf{Free}(\Sigma) \models \forall x, y (x \leq y \wedge y \leq x + \mathsf{car}(\mathsf{cons}(0, x)) \wedge P(h(x) - h(y)) \to P(0))$$

or, equivalently, whether the following conjunction (the negation of the formula above):

$$\phi : \quad c \leq d \ \wedge \ d \leq c + \mathsf{car}(\mathsf{cons}(0, c)) \ \wedge \ P(h(c) - h(d)) \ \wedge \ \neg P(0)$$

is satisfiable w.r.t. the conbination $LI(\mathbb{Q}) \cup \mathcal{T}_{\mathsf{lists}} \cup \mathsf{Free}(\Sigma)$. We proceed in two steps:

Step 1: Purification. *We replace alien terms with new constants and introduce the new definitions. The purified formula can be represented as follows:*

$LI(\mathbb{Q})$	$\mathcal{T}_{\mathsf{lists}}$	$\mathsf{Free}(\Sigma)$
$c \leq d$	$c_1 \approx \mathsf{car}(\mathsf{cons}(c_5, c))$	$P(c_2)$
$d \leq c + c_1$		$\neg P(c_5)$
$c_2 \approx c_3 - c_4$		$c_3 \approx h(c)$
$c_5 \approx 0$		$c_4 \approx h(d)$
ϕ_1	ϕ_2	ϕ_3

Step 2: Propagation. *Since ϕ_1, ϕ_2, ϕ_3 are satisfiable in the corresponding theories, we need to propagate information about shared constants between theories. Since all the theories involved are convex, it is sufficient if we propagate atomic equality atoms between the shared constants. Since $\phi_2 \models c_1 \approx c_5$, we add $c_1 \approx c_5$ also to the set of formulae corresponding to $LI(\mathbb{Q})$. As a consequence, the new set of the formulae corresponding to $LI(\mathbb{Q})$ now entails $c \approx d$. We propagate this equality to the $\mathsf{Free}(\Sigma)$. The set of formulae corresponding to $\mathsf{Free}(\Sigma)$ entails $c_3 \approx c_4$, equality which is propagated to the $LI(\mathbb{Q})$ part. In $LI(\mathbb{Q})$ we thus*

deduce that $c_2 \approx c_5$*, which is propagated to* $\mathsf{Free}(\Sigma)$ *and there leads to a contradiction. The procedure ends with the following configuration:*

$LI(\mathbb{Q})$	$\mathcal{T}_{\mathsf{lists}}$	$\mathsf{Free}(\Sigma)$
$c \leq d$	$c_1 \approx \mathsf{car}(\mathsf{cons}(c_5, c))$	$P(c_2)$
$d \leq c + c_1$		$\neg P(c_5)$
$c_2 \approx c_3 - c_4$		$c_3 \approx h(c)$
$c_5 \approx 0$		$c_4 \approx h(d)$
$c_1 \approx c_5$	$c_1 \approx c_5$	$c \approx d$
$c \approx d$		$c_3 \approx c_4$
$c_3 \approx c_4$		
$c_2 \approx c_5$		$\bot$

Hence, ϕ *is unsatisfiable.*

7.1.2 Extensions of the Nelson-Oppen procedure.

Recently, attempts have been made to extend the Nelson-Oppen combination procedure to more general theories. Extensions have been achieved either by relaxing the requirement that the theories to be combined are stably-infinite [65] or by relaxing the requirement that the theories to be combined have disjoint signatures (cf. e.g. [7, 64] and [29]). In what follows we will discuss a generalization of the Nelson-Oppen combination method to theories with non-disjoint signatures, in which the restrictions on the component theories are of a model theoretic nature given by Ghilardi in [29].

7.2 Combinations of theories with non-disjoint signatures

Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be theories with signatures Σ_1, Σ_2. The common subsignature $\Sigma_0 = \Sigma_1 \cap \Sigma_2$ is not assumed to be empty. The goal is to apply the Nelson-Oppen combination schema for reasoning in the combination $\mathcal{T}_1 \cup \mathcal{T}_2$:

Step 1: Purification can be done as in the disjoint case.

Step 2: Propagation We need to ensure that only a finite number of formulae over the shared signature needs to be propagated.

Soundness: is obvious.

Completeness: We need to impose additional conditions on the theories such that completeness is guaranteed.

The most simple way of avoiding non-termination in Step 2 is to assume that:

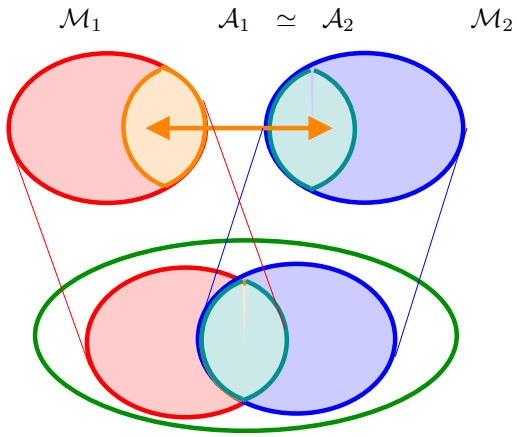
Assumption I: the theories $\mathcal{T}_1$ and $\mathcal{T}_2$ contain a Σ_0-theory $\mathcal{T}_0$ which is *locally finite*, i.e. has the property that for any finite set X of variables only finitely many Σ_0-terms with variables in X (up to $\mathcal{T}_0$-equivalence) exist.

Assumption II: Representative terms for each equivalence class are computable.

Step 2 (Propagation) can still be implemented by guessing (guess a maximal set of representative literals over Σ_0 over the shared constants) or by backtracking (make case-split on disjunctions of Σ_0-literals over the shared constants which are not entailed by the purified formulae).

We need to identify sufficient conditions for completeness. In the completeness proof of the Nelson-Oppen procedure (Theorem 24) the main idea is: given $\mathcal{T}_i$-models A_i of ϕ_i – construct a $\mathcal{T}_1 \cup \mathcal{T}_2$-model A of $\phi_1 \wedge \phi_2$. Stable infinity guarantees that this is possible. Ghilardi [29] noticed that the condition that $\mathcal{T}_i$ is stably infinite (used in the Nelson-Oppen

Figure 6: Amalgamations of models for non-disjoint signatures



procedure) means that every model of $\mathcal{T}_i$ embeds into an infinite model, and that the theory of an infinite model is the model completion $\mathcal{T}_0^*$ of the theory $\mathcal{T}_0$ of pure theory of equality (for definitions cf. Appendix A). Thus, every model of $\mathcal{T}_i$ embeds into a model of $\mathcal{T}_i \cup \mathcal{T}_0^*$. He adapted this definition to the non-disjoint case, and proved that the completeness proof of the disjoint case works (with small modifications) also in the non-disjoint case.

Definition 25 *Let $\mathcal{T}_0 \subseteq \mathcal{T}$ be theories in the signatures $\Sigma_0 \subseteq \Sigma$. Assume that $\mathcal{T}_0$ is universal and has a model completion $\mathcal{T}_0^*$. We say that $\mathcal{T}$ is $\mathcal{T}_0$-compatible if every model of $\mathcal{T}$ embeds into a model of $\mathcal{T} \cup \mathcal{T}_0^*$.*

We make the additional assumptions:

Assumption III: $\mathcal{T}_0$ is universal and has a model completion $\mathcal{T}_0^*$.

Assumption IV: Every model of $\mathcal{T}_i$ embeds into a model of $\mathcal{T}_i \cup \mathcal{T}_0^*$, for $i = 1, 2$.

Theorem 26 ([29]) *Under Assumptions I, II, III, IV on the theories $\mathcal{T}_1$ and $\mathcal{T}_2$, the Nelson-Oppen procedure transfers decidability of ground satisfiability from $\mathcal{T}_1, \mathcal{T}_2$ to $\mathcal{T}_1 \cup \mathcal{T}_2$.*

Proof: Assumptions I and II ensure that Step 2 (propagation) terminates. It is easy to see that all steps of the algorithm are satisfiability preserving. Thus, if at the end of the algorithm the prover for one of the component theories returns "unsatisfiable" on the corresponding clauses the initial formula was unsatisfiable.

The idea of the completeness proof is presented below. Assume that the Nelson-Oppen style procedure ends without returning unsatisfiable. It is shown that in this case in the propagation step exhaustive information between the theories has been exchanged (in the original paper [29] this infomation exchange is modeled using so-called residue chains) - a saturation set (consisting of Σ_0-formulae) is obtained in the process. This guarantees that we can find models $\mathcal{M}_1 \in \mathsf{Mod}(\mathcal{T}_1), \mathcal{M}_2 \in \mathsf{Mod}(\mathcal{T}_2)$ of the pure formulae Γ_1, Γ_2 obtained at the end of the propagation phase which share a common substructure $\mathcal{A}$. By Assumption IV, the models $\mathcal{M}_i$ embed into models $\mathcal{M}_i^*$ of $\mathcal{T}_i \cup \mathcal{T}_0^*$ for $i = 1, 2$; we can still assume that $\mathcal{M}_1^*$ and $\mathcal{M}_2^*$ share $\mathcal{A}$ as a common substructure.

By Assumption III, $\mathcal{T}_0$ is universal. Since truth of universal formulae is preserved in substructures, $\mathcal{A} \in \mathsf{Mod}(\mathcal{T}_0)$. Moreover, since $\mathcal{T}_0$ has model completion, $\mathcal{T}_0^* \cup \Delta(A)$ is a complete theory (the diagram $\Delta(A)$ of A is defined in Appendix A). This allows us to use Robinsons Joint Consistency theorem to infer that $\Delta^e(\mathcal{M}_1^*) \cup \Delta^e(\mathcal{M}_2^*)$ is consistent and its

model is a model of $(\mathcal{T}_1 \cup \Gamma_1) \cup (\mathcal{T}_2 \cup \Gamma_2)$, and thus is a model of $\mathcal{T}_1 \cup \mathcal{T}_2$ and of ϕ. (For the definition of the elementary diagram of a structure $\mathcal{A}$, $\Delta^e(\mathcal{A})$ see Appendix A). □

Example 30 *Let $\mathcal{T}_1 = LI(\mathbb{Q})$ be linear arithmetic, $\mathcal{T}_2$ be the theory of total orders with a strictly monotone function f, i.e. a function satisfying*

$$\forall x, y \ (x < y \rightarrow f(x) < f(y)).$$

Let $\mathcal{T}_0$ be the theory of total orders. This theory has no function symbols and it is therefore effectively locally finite. Thus, Assumptions I and II hold. Both in $\mathcal{T}_1$ and in $\mathcal{T}_2$ the ground satisfiability problem is decidable. Since the model completion $\mathcal{T}_0^*$ of the theory of total orders is the theory of dense total orders without endpoints, obviously $\mathcal{T}_0^* \subseteq \mathcal{T}_1 = LI(\mathbb{Q})$. It can be shown that every model of the theory of total orders with a strictly monotone function f embeds into a model of the theory of dense total orders without endpoints with a strictly monotone function f. This shows that the Nelson-Oppen style procedure presented above yields a decision procedure for the ground satisfiability problem w.r.t. $\mathcal{T}_1 \cup \mathcal{T}_2$.

Note that the argument above cannot be used for the combination of Presburger arithmetic with the theory of total orders with a strictly monotone function f. A different, orthogonal, method which provides decidability results for many theories of monotone functions over numerical and non-numeric domains is presented in Section 9.1.1 (cf. also Section 8.1).

Example 31 *Let $\mathcal{T}_i = \mathsf{Th}(BAO(\Sigma_i))$ be the theories of Boolean algebras with operators in $\Sigma_i, i = 1, 2$ such that $\Sigma_1 \cap \Sigma_2 = \emptyset$. Let $\mathcal{T}_0$ be the theory of Boolean algebras. This theory is effectively locally finite, hence satisfies assumptions I and II. Both in $\mathcal{T}_1$ and in $\mathcal{T}_2$ the ground satisfiability problem is decidable. $\mathcal{T}_0$ also satisfies Assumption III. The model completion $\mathcal{T}_0^*$ of the theory of Boolean algebras is the theory of atomless Boolean algebras. It can be shown that every Boolean algebra with operators in Σ_i embeds into an atomless Boolean algebra with operators in Σ_i, so Assumption IV is fulfilled. This shows that the Nelson-Oppen style procedure presented above yields a decision procedure for the ground satisfiability problem w.r.t. $\mathcal{T}_1 \cup \mathcal{T}_2$.*

7.3 Further comments

Note that the extension mentioned above is still quite restrictive, as the conditions imposed on the base theory and on the component theories are very strong, and of a model theoretic nature. For instance, decidability transfer in combinations of theories studied in [29], is guaranteed only under strong conditions assumptions on the shared theory (only locally finite or noetherian shared theories (hence only a small class of numerical domains) can be handled when applying these results to verification in [31]. In contrast, the notion of local extensions we studied [52] imposes no restrictions on using numerical domains as a base theory. Additional results concerning reasoning in combinations of theories will be given in Section 8.5.

8 Theory extensions

We can consider the extension of a given theory with new function or predicate symbols subject to certain axioms.

In this section we present a class of extensions of a "base" theory (which we will call *local*) in which hierarchic reasoning is possible (i.e. proof tasks in the extension can be hierarchically reduced to proof tasks w.r.t. the base theory). Many theories important for computer science or mathematics fall into this class (typical examples are theories of data structures, theories of free or monotone functions, but also functions occurring in mathematical analysis). In

fact, it is often necessary to consider complex extensions, in which various types of functions or data structures need to be taken into account at the same time. We show how such local theory extensions can be identified and under which conditions locality is preserved when combining theories, and we investigate possibilities of efficient modular reasoning in such theory combinations.

In the study of local theory extensions we will refer to *total* models of a theory and to *partial models* of a theory. The necessary definitions are given in Appendix B.

8.1 Local theory extensions

The notion of local theories introduced and studied by Givan and McAllester [32, 42, 33] can be extended in a natural way to extensions of a base theory with a set of additional function symbols constrained by a set $\mathcal{K}$ of clauses.

Let $\mathcal{K}$ be a set of clauses in the signature $\Sigma = (S, \Omega, \mathsf{Pred})$, where $\Omega = \Omega_0 \cup \Omega_1$. In what follows, when we refer to sets G of ground clauses we assume that they are in the signature $\Sigma^c = (S, \Omega \cup \Omega_c, \mathsf{Pred})$, where Ω_c is a set of new constants. If T is a set of ground $\Omega_0 \cup \Omega_1 \cup \Omega_c$-terms, we denote by $\mathcal{K}[T]$ the set of all instances of $\mathcal{K}$ in which all terms starting with a Ω_1-function symbol are ground terms in the set T. We denote by $\mathcal{K}^{[T]}$ the set of all instances of $\mathcal{K}$ in which all variables occurring below a Ω_1-function symbol are instantiated with ground terms in the set $T_{\Omega_0}(T)$ of Ω_0-terms generated by T.

If G is a set of ground clauses and $T = \mathsf{st}(\mathcal{K}, G)$ is the set of ground subterms occurring in either $\mathcal{K}$ or G then we write $\mathcal{K}[G] := \mathcal{K}[T]$, and $\mathcal{K}^{[G]} := \mathcal{K}^{[T]}$.

We will focus on the following type of locality of a theory extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$, where $\mathcal{T}_1 = \mathcal{T}_0 \cup \mathcal{K}$ with $\mathcal{K}$ a set of (universally quantified) clauses:

(Loc) For every set G of ground clauses $\mathcal{T}_1 \cup G \models \bot$ iff $\mathcal{T}_0 \cup \mathcal{K}[G] \cup G$ has no weak partial model in which all terms in $\mathsf{st}(\mathcal{K}, G)$ are defined.

(SLoc) For every set G of ground clauses $\mathcal{T}_1 \cup G \models \bot$ iff $\mathcal{T}_0 \cup \mathcal{K}^{[G]} \cup G$ has no partial model in which all terms in $\mathsf{st}(\mathcal{K}, G)$ are defined.

Weaker notions ($\mathsf{Loc}^{\mathsf{f}}$), resp. ($\mathsf{SLoc}^{\mathsf{f}}$) can be defined if we require that the respective conditions only hold for *finite* sets G of ground clauses. An intermediate notion of locality ($\mathsf{Loc}^{\mathsf{fd}}$) can be defined if we require that the respective conditions only hold for sets G of ground clauses containing only a finite set of terms starting with a function symbol in Ω_1.

An extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ is *local* (*stably local*) if it satisfies condition ($\mathsf{Loc}^{\mathsf{f}}$) (resp. ($\mathsf{SLoc}^{\mathsf{f}}$)). A local (stably local) theory [26] is a local extension of the empty theory. In (stably) local theory extensions hierarchical reasoning is possible.

In [37] we study a natural generalization of the notion of locality. Let Ψ be a function associating with a set $\mathcal{K}$ of axioms and a set T_0 of ground terms a set $\Psi(\mathcal{K}, T_0)$ of ground terms such that (i) all ground subterms in $\mathcal{K}$ and T_0 are in $\Psi(\mathcal{K}, T_0)$; (ii) for all sets of ground terms T_0 and T_0' if $T_0 \subseteq T_0'$ then $\Psi(\mathcal{K}, T_0) \subseteq \Psi(\mathcal{K}, T_0')$; (iii) Ψ defines a closure operation, i.e. for all sets of ground terms T_0, $\Psi(\mathcal{K}, \Psi(\mathcal{K}, T_0)) \subseteq \Psi(\mathcal{K}, T_0)$. Let $\mathcal{K}[\Psi(\mathcal{K}, G)]$ be the set of instances of $\mathcal{K}$ in which the extension terms are in $\Psi(\mathcal{K}, \mathsf{st}(G))$, which here will be denoted by $\Psi(\mathcal{K}, G)$. We say that an extension $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathcal{K}$ satisfies a locality condition w.r.t. Ψ if it satisfies condition (Loc^{Ψ}):

(Loc^{Ψ}) for every set G of ground clauses, $\mathcal{T}_1 \cup G \models \bot$ iff $\mathcal{T}_0 \cup \mathcal{K}[\Psi(\mathcal{K}, G)] \cup G$ has no weak partial model in which all terms in $\Psi(\mathcal{K}, G)$ are defined.

8.2 Hierarchical reasoning in local theory extensions

Consider a local theory extension $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathcal{K}$. The locality conditions defined above require that, for every set G of ground clauses, $\mathcal{T}_1 \cup G$ is satisfiable if and only if $\mathcal{T}_0 \cup \mathcal{K}*[G] \cup G$ has a (Evans, weak, finite) partial model with additional properties, where, depending on the notion of locality, $\mathcal{K}*[G]$ is $\mathcal{K}[G]$ or $\mathcal{K}^{[G]}$. All clauses in $\mathcal{K}*[G] \cup G$ have the property that the function symbols in Ω_1 have as arguments only ground terms. Therefore, $\mathcal{K}*[G] \cup G$ can be flattened and purified (i.e. the function symbols in Ω_1 are separated from the other symbols) by introducing, in a bottom-up manner, new constants c_t for subterms $t = f(g_1, \ldots, g_n)$ with $f \in \Omega_1$, g_i ground $\Omega_0 \cup \Omega_c$-terms (where Ω_c is a set of constants which contains the constants introduced by flattening, resp. purification), together with corresponding definitions $c_t \approx t$. The set of clauses thus obtained has the form $\mathcal{K}_0 \cup G_0 \cup D$, where D is a set of ground unit clauses of the form $f(g_1, \ldots, g_n) \approx c$, where $f \in \Omega_1$, c is a constant, $g_1, \ldots, g_n$ are ground terms without function symbols in Ω_1, and $\mathcal{K}_0$ and G_0 are clauses without function symbols in Ω_1. Flattening and purification preserve both satisfiability and unsatisfiability w.r.t. total algebras, and also w.r.t. partial algebras in which all ground subterms which are flattened are defined [52]. For the sake of simplicity in what follows we will always flatten and then purify $\mathcal{K} * [G] \cup G$. Thus we ensure that D consists of ground unit clauses of the form $f(c_1, \ldots, c_n) \approx c$, where $f \in \Omega_1$, and $c_1, \ldots, c_n, c$ are constants.

Lemma 27 ([52]) *Let $\mathcal{K}$ be a set of clauses and G a set of ground clauses, and let $\mathcal{K}_0 \cup G_0 \cup D$ be obtained from $\mathcal{K} * [G] \cup G$ by flattening and purification, as explained above. Assume that $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathcal{K}$ is a local theory extension. Then the following are equivalent:*

(1) $\mathcal{T}_0 \cup \mathcal{K} * [G] \cup G$ has a partial model in which all terms in $\mathsf{st}(\mathcal{K}, G)$ are defined.

(2) $\mathcal{T}_0 \cup \mathcal{K}_0 \cup G_0 \cup D$ has a partial model with all terms in $\mathsf{st}(\mathcal{K}_0, G_0, D)$ defined.

(3) $\mathcal{T}_0 \cup \mathcal{K}_0 \cup G_0 \cup N_0$ has a (total) model, where

$$N_0 = \{ \bigwedge_{i=1}^{n} c_i \approx d_i \rightarrow c \approx d \mid f(c_1, \ldots, c_n) \approx c, f(d_1, \ldots, d_n) \approx d \in D \}.$$

Theorem 28 ([52]) *Assume that the theory extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ either (1) satisfies condition ($\mathsf{Loc}^{\mathsf{f}}$), or (2) satisfies condition ($\mathsf{SLoc}^{\mathsf{f}}$) and $\mathcal{T}_0$ is locally finite. Then:*

(a) If all variables in the clauses in $\mathcal{K}$ occur below some function symbol from Ω_1 and if the universal theory of $\mathcal{T}_0$ is decidable, then the universal theory of $\mathcal{T}_1$ is decidable.

(b) Assume some variables in $\mathcal{K}$ do not occur below any function symbol in Ω_1. If the $\forall\exists$ theory of $\mathcal{T}_0$ is decidable then the universal theory of $\mathcal{T}_1$ is decidable.

In case (a) above locality allows to reduce reasoning in $\mathcal{T}_1$ to reasoning in an extension of $\mathcal{T}_0$ with free function symbols (for this an SMT procedure can be used). In case (b) this is not possible, as $\mathcal{K} * [G]$ is not a set of ground clauses.

Similar results hold also for Ψ-locality.

We will illustrate the applicability of Lemma 27 and Theorem 28 for specific examples of local theory extensions in Section 8.3 (Examples 32 and 33), as well as Sections 9.1, 9.5, 9.6.3 and 9.6.4.

We discuss two different ways of recognizing the locality of a theory extension. The first is semantical, based on possibilities of embedding partial models of a theory extension into total models. The second is proof theoretical, and at the moment part of work in progress: we present some results based on possibilities of saturating the extension axioms with respect to ordered resolution.

8.3 Recognizing local extensions: 1. Semantical criterion

We will use the following notation:

- $\mathsf{PMod}(\Omega_1, \mathcal{T}_1)$ is the class of all partial models of $\mathcal{T}_1$ in which the functions in Ω_1 are partial, and all other function symbols are total;
- $\mathsf{PMod}_\mathsf{w}(\Omega_1, \mathcal{T}_1)$ is the class of all weak partial models of $\mathcal{T}_1$ in which the Ω_1-functions are partial and all the other function symbols are total;
- $\mathsf{Mod}(\mathcal{T}_1)$ denotes the class of all total models of $\mathcal{T}_1$.

We will also consider small variations of the notion of weak partial model:

- $\mathsf{PMod}^\mathsf{f}_\mathsf{w}(\Omega_1, \mathcal{T}_1)$ is the class of all finite weak partial models of $\mathcal{T}_1$ in which the Ω_1-functions are partial and all other functions are total;
- $\mathsf{PMod}^\mathsf{fd}_\mathsf{w}(\Omega_1, \mathcal{T}_1)$ is the class of all weak partial models of $\mathcal{T}_1$ in which the Ω_1-functions are partial and their definition domain is a finite set, and all other functions are total,

and similar variations $\mathsf{PMod}^\mathsf{f}(\Omega_1, \mathcal{T}_1)$, $\mathsf{PMod}^\mathsf{fd}(\Omega_1, \mathcal{T}_1)$ of the notion of partial model.

Embeddability. For theory extensions $\mathcal{T}_0 \subseteq \mathcal{T}_1 = \mathcal{T}_0 \cup \mathcal{K}$, where $\mathcal{K}$ is a set of clauses, we consider the following conditions:

(Emb)	Every $A \in \mathsf{PMod}(\Omega_1, \mathcal{T}_1)$ weakly embeds into a total model of $\mathcal{T}_1$.
$(\mathsf{Emb_w})$	Every $A \in \mathsf{PMod}_\mathsf{w}(\Omega_1, \mathcal{T}_1)$ weakly embeds into a total model of $\mathcal{T}_1$.

We also define a stronger notion of embeddability, which we call *completability*:

$(\mathsf{Comp_w})$	Every $A \in \mathsf{PMod}_\mathsf{w}(\Omega_1, \mathcal{T}_1)$ weakly embeds into a total model B of $\mathcal{T}_1$ such that $A_{\lvert\Sigma_0}$ and $B_{\lvert\Sigma_0}$ are isomorphic.

(Comp) is defined analogously (w.r.t. $\mathsf{PMod}(\Omega_1, \mathcal{T}_1)$).

Conditions which only refer to embeddability of *finite* partial models are denoted by $(\mathsf{Emb}^\mathsf{f}_\mathsf{w})$, $(\mathsf{Comp}^\mathsf{f}_\mathsf{w})$, resp. $(\mathsf{Emb}^\mathsf{f})$, $(\mathsf{Comp}^\mathsf{f})$. Conditions referring to embeddability of partial models in which the extension functions have a finite definition domain (i.e. in $\mathsf{PMod}^\mathsf{fd}_\mathsf{w}(\Omega_1, \mathcal{T}_1)$) are denoted by $(\mathsf{Emb}^\mathsf{fd}_\mathsf{w})$, resp. $(\mathsf{Comp}^\mathsf{fd}_\mathsf{w})$.

In what follows we say that a non-ground clause is Ω_1-*flat* if function symbols (including constants) do not occur as arguments of function symbols in Ω_1. A Ω_1-flat non-ground clause is called Ω_1-*linear* if whenever a variable occurs in two terms in the clause which start with function symbols in Ω_1, the two terms are identical, and if no term which starts with a function symbol in Ω_1 contains two occurrences of the same variable.

For sets of Ω_1-flat clauses locality implies embeddability. This generalizes results presented in the case of local theories in [26].

Theorem 29 ([54]) *Assume that $\mathcal{K}$ is a family of Ω_1-flat clauses in the signature Σ.*

(1) If the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1 := \mathcal{T}_0 \cup \mathcal{K}$ satisfies (Loc) then it satisfies $(\mathsf{Emb_w})$.
(2) If the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1 := \mathcal{T}_0 \cup \mathcal{K}$ satisfies $(\mathsf{Loc}^\mathsf{f})$ then it satisfies $(\mathsf{Emb}^\mathsf{f}_\mathsf{w})$.
(3) If the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1 := \mathcal{T}_0 \cup \mathcal{K}$ satisfies $(\mathsf{Loc}^\mathsf{fd})$ then it satisfies $(\mathsf{Emb}^\mathsf{fd}_\mathsf{w})$.
(4) If $\mathcal{T}_0$ is compact and $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies $(\mathsf{Loc}^\mathsf{f})$, then $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies $(\mathsf{Emb_w})$.

Conversely, embeddability implies locality. The following results appear in [52], [56] and allow us to give several examples of local theory extensions (cf. Examples 32 and 33, as well as Sections 9.1, 9.5, 9.6.3 and 9.6.4).

Theorem 30 ([52, 56]) *Let $\mathcal{K}$ be a set of Ω-flat and Ω-linear clauses.*

(1) If the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies ($\mathsf{Emb_w}$) *then it satisfies* (Loc).
(2) Assume that $\mathcal{T}_0$ is a locally finite universal theory, and that $\mathcal{K}$ contains only finitely many ground subterms. If the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies ($\mathsf{Emb_w^f}$), *then $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies* ($\mathsf{Loc^f}$).
(3) $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies ($\mathsf{Emb_w^{fd}}$). *Then $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies* ($\mathsf{Loc^{fd}}$).

Theorem 31 ([52]) *Let $\mathcal{T}_0$ be a universal theory and $\mathcal{K}$ be a set of clauses. Then:*

(1) If the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies (Emb) *then it satisfies* (SLoc).
(2) Assume that $\mathcal{T}_0$ is a locally finite universal theory, and that $\mathcal{K}$ contains only finitely many ground subterms. If $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies ($\mathsf{Emb^f}$), *then $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies* ($\mathsf{SLoc^f}$).

Analyzing the proofs of Theorems 30 and 31 we notice that the embeddability conditions (Comp) and ($\mathsf{Comp_w}$) imply, in fact, stronger locality conditions. Consider a theory extension $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathcal{K}$ with a set $\mathcal{K}$ of formulae of the form $\forall x_1 \dots x_n(\Phi(x_1, \dots, x_n) \vee C(x_1, \dots, x_n))$, where $\Phi(x_1, \dots, x_n)$ is an *arbitrary first order formula* in the base signature Σ_0 with free variables $x_1, \dots, x_n$, and $C(x_1, \dots, x_n)$ is a *clause* in the signature Σ.
We can extend the notion of locality of an extension accordingly:

(ELoc) For every formula $\Gamma = \Gamma_0 \cup G$, where Γ_0 is a Σ_0-sentence and G is a set of ground clauses, $\mathcal{T}_1 \cup \Gamma \models \bot$ iff $\mathcal{T}_0 \cup \mathcal{K}[\Gamma] \cup \Gamma$ has no weak partial model in which all terms in $\mathsf{st}(\mathcal{K}, G)$ are defined.

A stable locality condition (ESLoc) and a Ψ-locality condition (ELoc^{Ψ}) can be defined similarly. The proofs of Theorems 30 and 31 can be adapted with minimal changes to prove a stronger result:

Theorem 32 ([52]) *(1) Assume all terms of $\mathcal{K}$ starting with a Ω_1-function are flat and linear. If the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies* ($\mathsf{Comp_w}$) *then it satisfies* (ELoc).
(2) Assume that $\mathcal{T}_0$ is a universal theory. If the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies (Comp) *then it satisfies* (ESLoc).

Similar results hold also for Ψ-locality.

Example 32 *Let $\mathcal{T}_0$ be the theory $\mathsf{LI}(\mathbb{Q})$ of linear rational arithmetic, and let $\mathcal{T}_1 = \mathsf{LI}(\mathbb{Q}) \cup \mathsf{Free}(\{f, g, h\})$ be the extension of $\mathcal{T}_0$ with the free functions f, g, h, and let $G = g(a) \approx c + 5 \ \wedge \ f(g(a)) \geq c + 1 \ \wedge \ h(b) \approx d + 4 \ \wedge \ d \approx c + 1 \ \wedge \ f(h(b)) < c + 1$. We show that G is unsatisfiable in $LI(\mathbb{Q}) \cup \mathsf{Free}(\{f, g, h\})$ as follows:*

Step 1: Flattening; purification. *G is purified and flattened by replacing the terms starting with f, g, h with new constants from a countably infinite set Ω_c of constants. We obtain the following purified form:*

$$\begin{array}{lllllllllll} G_0: & a_1 \approx c+5 & \wedge & a_2 \geq c+1 & \wedge & b_1 \approx d+4 & \wedge & d \approx c+1 & \wedge & b_2 < c+1, \\ \mathsf{Def}: & a_1 \approx g(a) & \wedge & a_2 \approx f(a_1) & \wedge & b_1 \approx h(b) & \wedge & b_2 \approx f(b_1). & & \end{array}$$

Step 2: Hierarchical reasoning. *By Lemma 27, G is unsatisfiable in $LI(\mathbb{Q}) \cup \mathsf{Free}(\{f, g, h\})$ iff $G_0 \wedge N_0$ is unsatisfiable in $LI(\mathbb{Q})$, where N_0 corresponds to the consequences of the congruence axioms for those ground terms which occur in the definitions* Def *for the newly introduced variables.*

Def	G_0	N_0
$a_1 \approx g(a) \wedge a_2 \approx f(a_1)$	$a_1 \approx c+5 \wedge a_2 \geq c+1$	$N_0: \ b_1 \approx a_1 \rightarrow b_2 \approx a_2$
$b_1 \approx h(b) \wedge b_2 \approx f(b_1)$	$b_1 \approx d+4 \wedge d \approx c+1 \wedge b_2 < c+1$	

To prove that $G_0 \wedge N_0$ is unsatisfiable in $\mathsf{LI}(\mathbb{Q})$, note that $G_0 \models_{\mathsf{LI}(\mathbb{Q})} a_1 \approx b_1$. Hence, $G_0 \wedge N_0$ entails $a_2 \approx b_2 \wedge a_2 \geq c + 1 \wedge b_2 < c + 1$, which is inconsistent.

Remark: It is important to see that the hierarchical calculus for local extensions gives a decision procedure for the extension with free functions symbols of *any* theory. A similar result can be obtained using the Nelson-Oppen combination procedure, but only for extensions of *stably infinite theories* with free function symbols.

Example 33 *Let $\mathcal{T}_0$ be a theory (with a binary predicate $\leq$), and $\mathcal{T}_1$ a local extension of $\mathcal{T}_0$ with two monotone functions f, g. Consider the following problem:*

$$\mathcal{T}_0 \cup \mathsf{Mon}_f \cup \mathsf{Mon}_g \models \forall x, y, z, u, v(x \leq y \wedge f(y \vee z) \leq g(u \wedge v) \rightarrow f(x) \leq g(v)).$$

The problem reduces to the problem of checking whether $\mathcal{T}_0 \cup \mathsf{Mon}_f \cup \mathsf{Mon}_g \cup G \models \bot$, where $G = c_0 \leq c_1 \wedge f(c_1 \vee c_2) \leq g(c_3 \wedge c_4) \wedge f(c_0) \not\leq g(c_4)$.

The locality of the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ means that, in order to test if $\mathcal{T}_0 \cup \mathsf{Mon}_f \cup \mathsf{Mon}_g \cup G \models \bot$, it is sufficient to test whether $\mathcal{T}_0 \cup \mathsf{Mon}_f[G] \cup \mathsf{Mon}_g[G] \cup G \models_w \bot$, where $\mathsf{Mon}_f[G], \mathsf{Mon}_g[G]$ consist of those instances of the monotonicity axioms for f and g in which the terms starting with f and g already occur in G:

$$\begin{array}{lcl} \mathsf{Mon}_f[G] & = & c_0 \leq c_1 \vee c_2 \rightarrow f(c_0) \leq f(c_1 \vee c_2) \\ & & c_1 \vee c_2 \leq c_0 \rightarrow f(c_1 \vee c_2) \leq f(c_0) \end{array} \qquad \begin{array}{lcl} \mathsf{Mon}_g[G] & \approx & c_4 \leq c_3 \wedge c_4 \rightarrow g(c_4) \leq g(c_3 \wedge c_4) \\ & & c_3 \wedge c_4 \leq c_4 \rightarrow g(c_3 \wedge c_4) \leq g(c_4) \end{array}$$

In order to check the satisfiability of the latter formula, we purify it, introducing definitions for the terms below the extension functions $d_1 \approx c_1 \vee c_2, d_2 \approx c_3 \wedge c_4$ as well as for the terms starting with the extension functions themselves: $f(d_1) \approx e_1, f(c_0) \approx e_3, g(c_4) \approx e_4, g(d_2) \approx e_2$, and add the following (purified) instances of the congruence axioms: $d_1 \approx c_0 \rightarrow e_1 \approx e_3$ and $c_4 \approx d_2 \rightarrow e_4 \approx e_2$. We obtain the following set of clauses:

Def	G_0	N_0	$\mathcal{K}_0$
$f(d_1) \approx e_1$	$c_0 \leq c_1$	$d_1 \approx c_0 \rightarrow e_1 \approx e_3$	$d_1 \leq c_0 \rightarrow e_1 \leq e_3$
$f(c_0) \approx e_3$	$d_1 \approx c_1 \vee c_2$	$d_2 \approx c_4 \rightarrow e_2 \approx e_4$	$c_0 \leq d1 \rightarrow e_3 \leq e_1$
$g(c_4) \approx e_4$	$d_2 \approx c_3 \wedge c_4$		$d_2 \leq c_4 \rightarrow e_2 \leq e_4$
$g(d_2) \approx e_2$	$e_1 \leq e_2 \quad \wedge \quad e_3 \not\leq e_4$		$d_4 \leq d_2 \rightarrow e_4 \leq e_2$

We illustrate the hierarchical reduction to testing satisfiability in the base theory for the following examples of local extensions:

(1) Let $\mathcal{T}_0 = \mathsf{DLat}$, the theory of distributive lattices or $\mathcal{T}_0 = \mathsf{Bool}$, the theory of Boolean algebras. The universal clause theory of DLat (resp. Bool) is the theory of the two element lattice (resp. two element Boolean algebra), so testing Boolean satisfiability is sufficient (this is in NP); any SAT solver can be used for this.

(2) If $\mathcal{T}_0 = \mathsf{Lat}$, the theory of lattices, we can reduce the problem above to the problem of checking the satisfiability of a set of ground Horn clauses. This can be checked in PTIME.

(3) If $\mathcal{T}_0 = \mathbb{R}$ we first need to explain what $\vee$ and $\wedge$ are. For this, we replace $d_1 \approx c_1 \vee c_2$ with $(c_1 \leq c_2 \rightarrow d_1 \approx c_2) \wedge (c_2 < c_1 \rightarrow d_1 \approx c_2)$ and similarly for $d_2 \approx c_3 \wedge c_4$. We proved unsatisfiability using REDLOG *[18].*

We can therefore conclude that in all cases above:
$\mathcal{T}_1 \models \forall x, y, z, u, v(x \leq y \wedge f(y \vee z) \leq g(u \wedge v) \rightarrow f(x) \leq g(v))$. □

8.4 Recognizing local extensions: 2. Locality and saturation

In Section 5.2, we mentioned the results of Basin and Ganzinger [9, 8] in which links between saturation w.r.t. ordered resolution of a set of clauses $\mathcal{K}$ and the (order)-locality of $\mathcal{K}$ are established. A first result which establishes a link between the locality of the set $\mathcal{K}$ of clauses and the locality of the extension $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathcal{K}$ is given below:

Theorem 33 ([54]) *Let $\mathcal{T}_0$ be a theory with signature $\Sigma_0 = (S_0, \Omega_0, \mathsf{Pred})$. Let $\mathcal{T}_1 = \mathcal{T}_0 \cup \mathcal{K}$ with signature $\Sigma = (S_0 \cup S_1, \Omega_0 \cup \Omega_1, \mathsf{Pred})$. Assume that:*

- *all functions in Ω_1 occurring in $\mathcal{K}$ have their output sort in S_1;*
- *$\mathcal{K}$ is a set of clauses which only contain function symbols in Ω_1;*
- *the set $\mathcal{K}$ of clauses is local (resp. stably local).*

Then the extension $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathcal{K}$ is also local (resp. stably local).

Proof: (Sketch) Let $P = (\{P_s\}_{s\in S_0\cup S_1}, \{f_P\}_{f\in\Omega_0\cup\Omega_1}, \{R_P\}_{R\in\mathsf{Pred}})$ be a weak partial model of $\mathcal{T}_0 \cup \mathcal{K}$ in which all Ω_0-functions are totally defined. We will denote by $P_{|\Sigma_1}$ the partial structure obtained from P by forgetting all operation symbols in Ω_0. P (hence also $P_{|\Sigma_1}$) is a weak partial model of $\mathcal{K}$. By the locality of $\mathcal{K}$, $P_{|\Sigma_1} = (\{P_s\}_{s\in S_0\cup S_1}, \{f_P\}_{f\in\Omega_1}, \{R_A\}_{R\in\mathsf{Pred}})$ weakly embeds (via embedding i) into a total model $A = (\{A_s\}_{s\in S_0\cup S_1}, \{f_A\}_{f\in\Omega_1}, \{R_A\}_{R\in\mathsf{Pred}})$ of $\mathcal{K}$. Let A^* be the substructure of A having the same supports as A for the sorts in S_1 and support $i_s(P_s)$ for each sort $s \in S_0$. (Since we assumed that all function symbols in Ω_1 have output sort in S_1, A^* is closed under all Ω_1-operations.)

Let $B = (\{B_s\}_{s\in S_0\cup S_1}, \{f_B\}_{f\in\Omega_0\cup\Omega_1}, \{R_B\}_{R\in\mathsf{Pred}})$, where:

- for $s \in S_0$, $B_s = i_s(P_s)$, for $s \in S_1$, $B_s = A_s$;
- for $f \in \Omega_0$, f_B coincides with f_P;
- for $f \in \Omega_1$, f_B coincides with f_{A^*},

and all predicate symbols coincide with those in A^*. Then $B_{|\Sigma_0}$ is isomorphic to $P_{|\Sigma_0}$, hence is a model of $\mathcal{T}_0$ and $B_{|\Sigma_1} = A^*$, hence $B \models \mathcal{K}$. □

The locality of $\mathcal{K}$ can be checked e.g. by testing whether $\mathcal{K} \cup EQ$ is saturated under ordered resolution (w.r.t. the (strict) subterm ordering) using Theorems 18 and 19 (cf. also [9, 8, 26]) but now extended to a many-sorted framework. The advantage is that even if $\mathcal{K}$ is not saturated, if $\mathcal{K}^*$ is a finite saturation of $\mathcal{K} \cup EQ$ under ordered resolution, then $\mathcal{T}_0 \cup \mathcal{K}^*$ can be used to extract a presentation which defines a (local) theory extension which has the same total models as $\mathcal{T}_0 \cup \mathcal{K}$.

Note: The idea in the proof of Theorem 33 can also be used to show that (under the assumptions in Theorem 33) if $\mathcal{K}$ satisfies Comp (resp. (Comp_w)) then the extension $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathcal{K}$ also satisfies Comp (resp. (Comp_w)).

8.5 Combinations of local theory extensions

In this section we study the locality of combinations of local theory extensions. In the light of the results in Section 8.3 we concentrate on studying which embeddability properties are preserved under combinations of theories. For the sake of simplicity, in what follows we only consider conditions $(\mathsf{Emb}_\mathsf{w})$ and $(\mathsf{Comp}_\mathsf{w})$. Analogous results can be given for conditions $(\mathsf{Emb}^\mathsf{f}_\mathsf{w})$, $(\mathsf{Comp}^\mathsf{f}_\mathsf{w})$, resp. $(\mathsf{Emb}^\mathsf{fd}_\mathsf{w})$, $(\mathsf{Comp}^\mathsf{fd}_\mathsf{w})$ and combinations thereof. We first consider the situation when both components satisfy the embeddability condition $(\mathsf{Comp}_\mathsf{w})$.

Theorem 34 ([54]) *Let $\mathcal{T}_0$ be a first order theory with signature $\Sigma_0 = (\Omega_0, \mathsf{Pred})$ and (for $i \in \{1, 2\}$) $\mathcal{T}_i = \mathcal{T}_0 \cup \mathcal{K}_i$ be an extension of $\mathcal{T}_0$ with signature $\Sigma_i = (\Omega_0 \cup \Omega_i, \mathsf{Pred})$. Assume that both extensions $\mathcal{T}_0 \subseteq \mathcal{T}_1$ and $\mathcal{T}_0 \subseteq \mathcal{T}_2$ satisfy condition $(\mathsf{Comp}_\mathsf{w})$, and that $\Omega_1 \cap \Omega_2 = \emptyset$. Then the extension $\mathcal{T}_0 \subseteq \mathcal{T} = \mathcal{T}_0 \cup \mathcal{K}_1 \cup \mathcal{K}_2$ satisfies condition $(\mathsf{Comp}_\mathsf{w})$. If, additionally, in $\mathcal{K}_i$ all terms starting with a function symbol in Ω_i are flat and linear, for $i = 1, 2$, then the extension is local.*

Example 34 *The following combinations of theories (seen as extensions of a first order theory $\mathcal{T}_0$) satisfy condition $(\mathsf{Comp}_\mathsf{w})$ (in case (4) condition $(\mathsf{Comp}^\mathsf{fd}_\mathsf{w})$)[4]:*

(1) $\mathcal{T}_0 \cup \mathsf{Free}(\Sigma_1)$ and $\mathcal{T}_0 \cup \mathsf{Sel}_c$ if $\mathcal{T}_0$ is a theory and $c \in \Omega_0$ is injective in $\mathcal{T}_0$.

[4]We always assume that the signatures are $\Sigma_i = (\Omega_i, \mathsf{Pred})$.

(2) $\mathbb{R} \cup \mathsf{Free}(\Sigma_1)$ *and* $\mathbb{R} \cup \mathsf{L}_f^{\lambda}(c)$*, where* $f \notin \Omega_1$*. Here* $\mathsf{L}_f^{\lambda}(c)$ *is the* λ*-Lipschitz condition*[5] *for* f *at point* $c \in \mathbb{R}$ *(for* $\lambda > 0$*):* $\forall x\ |f(x) - f(c)| \leq \lambda \cdot |x - c|$.

(3) $\mathbb{R} \cup \mathsf{L}_f^{\lambda_1}(c_1)$ *and* $\mathbb{R} \cup \mathsf{L}_g^{\lambda_2}(c_2)$*, where* $f \neq g$.

(4) $\mathcal{T}_0 \cup \mathsf{Free}(\Sigma_1)$ *and* $\mathcal{T}_0 \cup \mathsf{Mon}_f^{\sigma}$*, where* $f \notin \Omega_1$ *has arity* n*,* $\sigma : \{1, \ldots, n\} \to \{-1, 1, 0\}$*, if* $\mathcal{T}_0$ *is, e.g., a theory of algebras with a bounded semilattice reduct.*

This result can be extended to the more general situation in which one extension satisfies condition $(\mathsf{Emb_w})$ and the other satisfies $(\mathsf{Comp_w})$ or $(\mathsf{Emb_w})$.

Theorem 35 ([54]) *Let* $\mathcal{T}_0$ *be a first order theory with signature* $\Sigma_0 = (\Omega_0, \mathsf{Pred})$*, and let* $\mathcal{T}_1 = \mathcal{T}_0 \cup \mathcal{K}_1$ *and* $\mathcal{T}_2 = \mathcal{T}_0 \cup \mathcal{K}_2$ *be two extensions of* $\mathcal{T}_0$ *with signatures* $\Sigma_1 = (\Omega_0 \cup \Omega_1, \mathsf{Pred})$ *and* $\Sigma_2 = (\Omega_0 \cup \Omega_2, \mathsf{Pred})$*, respectively. Assume that:*

(1) $\mathcal{T}_0 \subseteq \mathcal{T}_1$ *satisfies condition* $(\mathsf{Comp_w})$*;*

(2) $\mathcal{T}_0 \subseteq \mathcal{T}_2$ *satisfies condition* $(\mathsf{Emb_w})$*,*

(3) $\mathcal{K}_1$ *is a set of* Ω_1*-flat clauses in which all variables occur below a* Ω_1*-function.*

Then the extension $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathcal{K}_1 \cup \mathcal{K}_2$ *satisfies* $(\mathsf{Emb_w})$*. If in* $\mathcal{K}_i$ *all terms starting with a function symbol in* Ω_i *are flat and linear (for* i=1, 2*) the extension is local.*

Theorem 36 ([54]) *Let* $\mathcal{T}_0$ *be an arbitrary theory in signature* $\Sigma_0 = (\Omega_0, \mathsf{Pred})$*. Let* $\mathcal{K}_1$ *and* $\mathcal{K}_2$ *be two sets of clauses over signatures* $\Sigma_i = (\Omega_0 \cup \Omega_i, \mathsf{Pred})$*, where* Ω_1 *and* Ω_2 *are disjoint. We make the following assumptions:*

(A1) The class of models of $\mathcal{T}_0$ *is closed under direct limits of diagrams in which all maps are embeddings (or, equivalently,* $\mathcal{T}_0$ *is a* $\forall\exists$ *theory).*

(A2) $\mathcal{K}_i$ *is* Ω_i*-flat and* Ω_i*-linear for* $i = 1, 2$*, and* $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathcal{K}_i$*,* $i = 1, 2$ *are both local extensions of* $\mathcal{T}_0$*.*

(A3) For all clauses in $\mathcal{K}_1$ *and* $\mathcal{K}_2$*, every variable occurs below some extension function.*

Then $\mathcal{T}_0 \cup \mathcal{K}_1 \cup \mathcal{K}_2$ *is a local extension of* $\mathcal{T}_0$*.*

Example 35 *The following combinations of theories (seen as extensions of the theory* $\mathcal{T}_0$*) satisfy condition* $(\mathsf{Emb_w})$*, hence are local:*

(1) $\mathcal{E}q \subseteq \mathsf{Free}(\Sigma_1) \cup \mathcal{L}$*, where* $\mathcal{E}q$ *is the pure theory of equality, without function symbols, and* $\mathcal{L}$ *the theory of lattices.*

(2) $\mathcal{T}_0 \subseteq (\mathcal{T}_0 \cup \mathsf{Free}(\Sigma_1)) \cup (\mathcal{T}_0 \cup \mathsf{Mon}(\Sigma_2))$*, where* $\Sigma_1 \cap \Sigma_2 = \emptyset$*,* $\mathsf{Mon}(\Sigma_2) = \bigwedge_{f \in \Omega_2} \mathsf{Mon}_f^{\sigma(f)}$ *and* $\mathcal{T}_0$ *is, e.g., the theory of posets.*

(3) The combination of the theory of lattices and the theory of integers with injective successor and predecessor is local (local extension of the theory of pure equality).

In what follows we discuss some issues related to modular reasoning in combinations of local theory extensions. We analyze, in particular, the form of information which needs to be exchanged between provers for the component theories when reasoning in combinations of local theory extensions.

Let $\mathcal{T}_1$ and $\mathcal{T}_2$ be theories with signatures $\Sigma_1 = (\Omega_1, \mathsf{Pred})$ and $\Sigma_2 = (\Omega_2, \mathsf{Pred})$, and G a set of ground clauses in the joint signature with additional constants $\Sigma^c = (\Omega_0 \cup \Omega_1 \cup \Omega_2 \cup \Omega_c, \mathsf{Pred})$. We want to decide whether $\mathcal{T}_1 \cup \mathcal{T}_2 \cup G \models \bot$.

[5] We proved in [52] that for every function f and constants c and λ with $\lambda > 0$ the extension $\mathbb{R} \subseteq \mathbb{R} \cup (\mathsf{Lip}_c^{\lambda}(\mathsf{f}))$ satisfies $(\mathsf{Comp_w})$, hence it is local.

The set G of ground clauses can be flattened and purified as explained above. For the sake of simplicity, everywhere in what follows we will assume w.l.o.g. that $G = G_1 \wedge G_2$, where G_1, G_2 are flat and linear sets of clauses in the signatures Σ_1, Σ_2 respectively, i.e. for $i = 1, 2$, $G_i = G_i^0 \wedge G_0 \wedge D_i$, where G_i^0 and G_0 are clauses in the base theory and D_i is a conjunction of unit clauses of the form $f(c_1, \ldots, c_n) \approx c, f \in \Omega_i$.

Corollary 37 *Assume that $\mathcal{T}_1 \approx \mathcal{T}_0 \cup \mathcal{K}_1$ and $\mathcal{T}_2 = \mathcal{T}_0 \cup \mathcal{K}_2$ are local extensions of a theory $\mathcal{T}_0$ with signature $\Sigma_0 = (\Omega_0, \mathsf{Pred})$, where $\Omega_0 = \Omega_1 \cap \Omega_2$, and that the extension $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathcal{K}_1 \cup \mathcal{K}_2$ is local. Let $G = G_1 \wedge G_2$ be a set of flat, linear are purified ground clauses, such that $G_i = G_i^0 \wedge G_0 \wedge D_i$ are as explained above. Then the following are equivalent:*

(1) $\mathcal{T}_1 \cup \mathcal{T}_2 \cup (G_1 \wedge G_2) \models \bot$,

(2) $\mathcal{T}_0 \cup (\mathcal{K}_1 \cup \mathcal{K}_2)[G_1 \wedge G_2] \cup (G_1^0 \wedge G_0 \wedge D_1) \wedge (G_2^0 \wedge G_0 \wedge D_2) \models \bot$,

(3) $\mathcal{T}_0 \cup \mathcal{K}_1^0 \cup \mathcal{K}_2^0 \cup (G_1^0 \cup G_0) \cup (G_2^0 \cup G_0) \cup N_1 \cup N_2 \models \bot$, *where*

$$N_i \quad = \quad \{\bigwedge_{i=1}^{n} c_i \approx d_i \rightarrow c \approx d \mid f(c_1, \ldots, c_n) \approx c, f(d_1, \ldots, d_n) \approx d \in D_i\}, i = 1, 2,$$

and $\mathcal{K}_i^0$ is the formula obtained from $\mathcal{K}_i[G_i]$ after purification and flattening, taking into account the definitions from D_i.

9 Applications

We now present various examples from mathematics, verification, databases and multi-agent systems where efficient decision procedures exist.

9.1 Mathematics: numerical functions

We give several examples of theories from mathematics for which hierarchic and modular theorem proving is possible. For the sake of simplicity, we here restrict to unary functions, but most of the results also hold for functions $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$. The results presented here were first given in [52, 56, 39, 58].

9.1.1 Monotonicity and boundedness conditions

Any extension of a theory with free function symbols is local. In addition the following theory extensions have been proved to be local in [52, 56, 39]:

Monotonicity. Any extension of the theory of reals, rationals or integers with functions satisfying $\mathsf{Mon}^\sigma(f)$ is local ($(\mathsf{Comp_w})$ holds [52, 56]) [6]:

$$\mathsf{Mon}^\sigma(f) \quad \bigwedge_{i \in I} x_i \leq_i^{\sigma_i} y_i \wedge \bigwedge_{i \notin I} x_i \approx y_i \rightarrow f(x_1, .., x_n) \leq f(y_1, .., y_n).$$

The extension $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathsf{SMon}(f)$ is local if $\mathcal{T}_0$ is the theory of reals (and $f : \mathbb{R} \rightarrow \mathbb{R}$) or the disjoint combination of the theories of reals and integers (and $f : \mathbb{Z} \rightarrow \mathbb{R}$) [37]. The extension of the theory of integers with ($\mathsf{SMon}_{\mathbb{Z}}(f)$) is local.

$$\mathsf{SMon}(f) \quad \forall i, j (i < j \rightarrow f(i) < f(j)) \qquad \mathsf{SMon}_{\mathbb{Z}}(f) \quad \forall i, j (i < j \rightarrow (j - i) < f(j) - f(i)).$$

[6] For $i \in I$, $\sigma_i \in \{-, +\}$, and for $i \notin I$, $\sigma_i = 0$; $\leq^+ = \leq, \leq^- = \geq$.

Boundedness. Assume $\mathcal{T}_0$ contains a reflexive binary predicate $\leq$, and $f \notin \Omega_0$. Let $m \in \mathbb{N}$. For $1 \leq i \leq m$ let $t_i(x_1, \ldots, x_n)$ and $s_i(x_1, \ldots, x_n)$ be terms in the signature Σ_0 and $\phi_i(x_1, \ldots, x_n)$ be Σ_0-formulae with (free) variables among $x_1, \ldots, x_n$, such that $\mathcal{T}_0 \models \forall \overline{x}(\phi_i(\overline{x}) \rightarrow s_i(\overline{x}) \leq t_i(\overline{x}))$, and if $i \neq j$, $\phi_i \wedge \phi_j \models_{\mathcal{T}_0} \bot$.
Let $\mathsf{GB}(f) = \bigwedge_{i=1}^{m} \mathsf{GB}^{\phi_i}(f)$ and $\mathsf{Def}(f) = \bigwedge_{i=1}^{n} \mathsf{Def}^{\phi_i}(f)$, where:

$$\mathsf{GB}^{\phi_i}(f) \quad \forall \overline{x}(\phi_i(\overline{x}) \rightarrow s_i(\overline{x}) \leq f(\overline{x}) \leq t_i(\overline{x})) \qquad \mathsf{Def}^{\phi_i}(f) \quad \forall \overline{x}(\phi_i(\overline{x}) \rightarrow f(\overline{x}) \approx t_i(\overline{x})).$$

(i) The extensions $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathsf{GB}(f)$ and $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathsf{Def}(f)$ are both local [56, 37].

(ii) Any extension of a theory for which $\leq$ is a partial order (or at least reflexive) with functions satisfying $\mathsf{Mon}^{\sigma}(f)$ and $\mathsf{Bound}^{t}(f)$ is local [56, 37].

$\mathsf{Bound}^{t}(f) \qquad \forall x_1, \ldots, x_n(f(x_1, \ldots, x_n) \leq t(x_1, \ldots, x_n))$

where $t(x_1, \ldots, x_n)$ is a Σ_0-term with variables among $x_1, \ldots, x_n$ whose associated function has the same monotonicity as f in any model. Similar results hold for strictly monotone functions.

Injectivity. An extension $\mathcal{T}_0 \subseteq \mathcal{T}_1 = \mathcal{T}_0 \cup \mathsf{Inj}(f)$ with a function f of arity $\mathsf{i} \rightarrow \mathsf{e}$ satisfying $\mathsf{Inj}(f)$ is local provided that in all models of $\mathcal{T}_1$ the cardinality of the support of sort i is lower or equal to the cardinality of the support of sort e.

$$\mathsf{Inj}(f) \quad \forall i, j(i \not\approx j \rightarrow f(i) \not\approx f(j)).$$

9.1.2 Inverse conditions

Consider the following inverse condition:

$$\mathsf{Inv}(f, g) \qquad \forall x, y(y \approx f(x) \rightarrow g(y) \approx x).$$

Such conditions often occur in mathematics and are important in verification (e.g. to model direct and inverse links between certain objects).

Theorem 38 *Let $\mathcal{T}_0$ be a theory and $f, g \notin \Omega_0$. Assume that $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathcal{K}(f)$ satisfies Comp_w, and that $\mathcal{T}_0 \cup \mathcal{K}(f) \models \mathsf{Inj}(f)$. Then $\mathcal{T}_0 \cup (\mathcal{K}(f) \cup \mathsf{Inv}(f, g)) \cup G \models \bot$ iff $\mathcal{T}_0 \cup (\mathcal{K}(f) \cup \mathsf{Inv}(f, g))[G] \cup G \models \bot$ for all sets G of ground clauses with the property that if $g(c)$ occurs in G then also some $f(a) \approx c$ occurs in G.*

9.1.3 Convexity/concavity

Let f be a unary function, and $I = [a, b]$ a subset of the domain of definition of f. We consider the axiom:

$$\mathsf{Conv}^{I}(f) \qquad \forall x, y, z \left(x, y \in I \wedge x \leq z \leq y \rightarrow \frac{f(z) - f(x)}{z - x} \leq \frac{f(y) - f(x)}{y - x} \right).$$

Theorem 39 *Let $\mathcal{T}_0$ be one of the theories: (i) $\mathbb{R}$, (ii) $\mathbb{Z}$ (a theory of integers), (iii) the many-sorted combination of the theories of reals (sort real) and integers (sort int). Let f be a new, unary function (for (iii) assume f has arity $\mathsf{int} \rightarrow \mathsf{real}$).*
$\mathcal{T}_0 \cup \mathsf{Conv}^{I}_{f}$ and $\mathcal{T}_0 \cup \mathsf{Conc}^{I}_{f}$ are local extensions of $\mathcal{T}_0$, where $\mathsf{Conc}^{I}(f) = \mathsf{Conv}^{I}(-f)$.

9.1.4 Lipschitz conditions

Consider the following conditions:

$(\mathsf{L}_f^\lambda(c_0))$	$\forall x(\|f(x)-f(c_0)\| \leq \lambda\|x-c_0\|)$	Lipschitz condition at c_0
(L_f^λ)	$\forall x,y(\|f(x)-f(y)\| \leq \lambda\|x-y\|)$	(uniform) Lipschitz condition
(BL_f^λ)	$\forall x,y(\frac{1}{\lambda}\|x-y\| \leq \|f(x)-f(y)\| \leq \lambda\|x-y\|)$	bi-Lipschitz condition

Such conditions occur in the verification of hybrid systems when specifying (by universal axioms) that the derivative of a function is bounded by a given value.

Theorem 40 *$\mathbb{R} \cup (\mathsf{L}_f^\lambda(c_0))$, $\mathbb{R} \cup (\mathsf{L}_f^\lambda)$, and $\mathbb{R} \cup (\mathsf{BL}_f^\lambda)$ are local extensions of $\mathbb{R}$.*

Theorem 41 *The extension $\mathbb{R} \cup \mathsf{BL}_f^\lambda \cup \mathsf{Inv}(f,g)$ of $\mathbb{R}$ has the property that for all sets G of ground clauses such that if $g(c)$ occurs in G then also $f(a) \approx c$ occurs in G, $\mathbb{R} \cup (\mathsf{BL}_f^\lambda \cup \mathsf{Inv}(f,g)) \cup G \models \bot$ iff $\mathbb{R} \cup (\mathsf{BL}_f^\lambda \cup \mathsf{Inv}(f,g))[G] \cup G$ has no weak partial model in which all subterms of G (and only those) are defined.*

9.1.5 Continuity, derivability

We consider the following continuity conditions for a function $f : \mathbb{R} \to \mathbb{R}$:

$\mathsf{Cont}_f(c_0)$	$\forall\epsilon(\epsilon{>}0 \to \exists\delta(\delta{>}0 \wedge \forall x(\|x-c_0\|{<}\delta \to \|f(x)-f(c_0)\|{<}\epsilon)))$	continuity at c_0
Cont_f	$\forall x(\mathsf{Cont}_f(x))$	continuity

and the following derivability conditions for a (continuous) function f:

$$\mathsf{Der}(f,f')(c_0): \quad \forall\epsilon(\epsilon{>}0 \to \exists\delta(\delta{>}0 \wedge \forall x(\|x-c_0\|{<}\delta \to |\tfrac{f(x)-f(c_0)}{x-c_0} - f'(c_0)|{<}\epsilon)))$$

$$\mathsf{Der}^{\leq n}(f,f^1,\ldots,f^n)(c_0): \quad \bigwedge_{i=1}^{n} \mathsf{Cont}_{f^{i-1}}(c_0) \wedge \mathsf{Der}(f^{i-1},f^i)(c_0)$$

$\mathsf{Der}(f,f') := \forall x \mathsf{Der}(f,f')(x)$; $\mathsf{Der}^{\leq n}(f,f^1,\ldots,f^n) = \forall x \mathsf{Der}^{\leq n}(f,f^1,\ldots,f^n)(x)$ (axiomatizing derivability – resp. n-times derivability – at every point, where $n \in \mathbb{N} \cup \{\infty\}$, $f^0 = f$ and f^i is the i-th derivative of f).

Theorem 42 *(1) Any partial function over the reals with a finite domain of definition extends to a total continuous function over the reals.*

(2) $\mathbb{R} \cup \mathsf{Cont}_f(c_0) \cup \mathsf{Der}(f,f')(c_0)$ and $\mathbb{R} \cup \mathsf{Cont}_f \cup \mathsf{Der}(f,f')$ are Ψ-local extensions of $\mathbb{R}$, where $\Psi(T) = T \cup \{f(c) \mid f'(c) \in T\} \cup \{f'(c) \mid f(c) \in T\}$.

The extensions $\mathbb{R} \subseteq \mathbb{R} \cup \mathsf{Der}^{\leq n}(f,f^1,\ldots,f^n)(c_0)$ and $\mathbb{R} \subseteq \mathbb{R} \cup \mathsf{Der}^{\leq n}(f,f^1,\ldots,f^n)$ are Ψ^n-local extensions, where $\Psi^n(T) = T \cup \{f^k(c) \mid 0{\leq}k{\leq}n$ if $f^i(c){\in}T$ for some $0{\leq}i{\leq}n\}$.

Proof: We can use any polynomial interpolation theorem to compute a total model from any partial model (e.g. the Hermite interpolation theorem). □

9.1.6 Combinations of axioms

Analyzing the proofs in the previous sections we notice that the same completion for the partial functions can be used for (i) monotone, strictly monotone, convex/concave, Lipschitz and continuous functions over $\mathbb{R}$. The same completion (possibly different from that in (i)) is used (ii) for Lipschitz, and for (uniformly) continuous and n-derivable functions over $\mathbb{R}$.

Theorem 43 *The following axiom combinations define local extensions of $\mathbb{R}$:*

(1) Arbitrary combinations of $[\mathsf{S}]\mathsf{Mon})(f)$, Conv_f, $\mathsf{L}_f^\lambda[(c_0)]$, $\mathsf{BL}_f^\lambda, \mathsf{Cont}_f[(c_0)]$;

(2) Arbitrary combinations of $\mathsf{L}_f^\lambda, \mathsf{L}_f^\lambda(c_0), \mathsf{BL}_f^\lambda, \mathsf{Cont}_f, \mathsf{Cont}_f(c_0), \mathsf{Cont}_f \wedge \mathsf{Der}(f,f'), \mathsf{Cont}_f(c_0) \wedge \mathsf{Der}(f,f')(c_0)$ $\mathsf{Der}^{\leq n}(f,f^1,..,f^n)(c_0)$, *and* $\mathsf{Der}^{\leq n}(f,f^1,..,f^n)$.

However, care is needed when combining $\mathsf{Der}(f,f')$ with boundedness or monotonicity conditions on f', or with convexity/concavity conditions on f or f'.

9.1.7 Example of proof task

We present an example which illustrates the type of deduction problems in mathematics we may need to consider: Assume that $f{:}\mathbb{R}\to\mathbb{R}$ satisfies the bi-Lipschitz condition (BL_f^λ) with constant λ and g is the inverse of f. We want to determine whether g satisfies the bi-Lipschitz condition on the codomain of f, and if so with which constant λ_1, i.e. to determine under which conditions the following holds:

$$\mathbb{R} \cup (\mathsf{BL}_f^\lambda) \cup (\mathbf{Inv}(f,g)) \models \phi, \tag{1}$$

where $\phi : \forall x,x',y,y' (y \approx f(x) \wedge y' \approx f(x') \to \frac{1}{\lambda_1}|y-y'| \leq |g(y)-g(y')| \leq \lambda_1 |y-y'|)$;

$$(\mathsf{BL}_f^\lambda) \qquad \forall x,y (\frac{1}{\lambda}|x-y| \leq |f(x)-f(y)| \leq \lambda|x-y|);$$

$$\mathsf{Inv}(f,g) \qquad \forall x,y (y \approx f(x) \to g(y) \approx x).$$

Entailment (1) is true iff $\mathbb{R} \cup (\mathsf{BL}_f^\lambda) \cup (\mathbf{Inv}(f,g)) \cup G$ is unsatisfiable, where $G = (c_1 \approx f(a_1) \wedge c_2 \approx f(a_2) \wedge (\frac{1}{\lambda_1}|c_1-c_2| > |g(c_1)-g(c_2)| \vee |g(c_1)-g(c_2)| > \lambda_1|c_1-c_2|))$ is the formula obtained by Skolemizing the negation of ϕ.

Standard theorem provers for first order logic cannot be used in such situations. Provers for reals do not know about additional functions. The Nelson-Oppen method [46] for reasoning in combinations of theories cannot be used either.

The method we propose reduces the task of checking whether formula (1) holds to the problem of checking the satisfiability of a set of constraints over $\mathbb{R}$. We first note that for any set G of ground clauses with the property that "*if* $g(c)$ *occurs in* G *then* G *also contains a unit clause of the form* $f(a) \approx c$" every partial model P of G – where (i) f and g are partial and defined exactly on the ground subterms occurring in G and (ii) P satisfies $\mathsf{BL}_f^\lambda \cup \mathbf{Inv}(f,g)$ at all points where f and g are defined – can be completed to a total model of $\mathbb{R} \cup (\mathsf{BL}_f^\lambda) \cup (\mathbf{Inv}(f,g)) \cup G$ (cf. Theorem 40 and Corollary 41). Therefore, problem (1) is equivalent to

$$\mathbb{R} \cup (\mathsf{BL}_f^\lambda \cup \mathbf{Inv}(f,g))[G] \cup G \models \bot,$$

where $(\mathsf{BL}_f^\lambda \cup \mathbf{Inv}(f,g))[G]$ is the set of those instances of $\mathsf{BL}_f^\lambda \cup \mathbf{Inv}(f,g)$ in which the terms starting with g or f are ground terms occurring in G, i.e.

$$\begin{aligned}(\mathsf{BL}_f^\lambda \cup \mathbf{Inv}(f,g))[G] \quad = \quad & \frac{1}{\lambda}|a_1-a_2| \leq |f(a_1)-f(a_2)| \leq \lambda|a_1-a_2| \wedge \\ & (c_1 \approx f(a_1) \to g(c_1) \approx a_1) \ \wedge \ (c_2 \approx f(a_1) \to g(c_2) \approx a_1) \wedge \\ & (c_1 \approx f(a_2) \to g(c_1) \approx a_2) \ \wedge \ (c_2 \approx f(a_2) \to d(c_2) \approx a_2).\end{aligned}$$

We separate the numerical symbols from the non-numerical ones by introducing new names for the extension terms, together with their definitions $D = (f(a_1) \approx e_1 \wedge f(a_2) \approx e_2 \wedge g(c_1) \approx d_1 \wedge g(c_2) \approx d_2)$ and replacing them in $(\mathsf{BL}_f^\lambda \cup \mathbf{Inv}(f,g))[G] \cup G$. The set of formulae obtained this way is $\mathsf{BL}_0 \cup \mathbf{Inv}_0 \cup G_0$. We then use – instead of these definitions – only the instances $\mathsf{Con}[G]_0$ of the congruence axioms for f and g which correspond to these terms. We obtain:

$$\begin{aligned}\mathsf{BL}_0 : \quad & \frac{1}{\lambda}|a_1-a_2| \leq |e_1-e_2| \leq \lambda|a_1-a_2| \\ \mathsf{Inv}_0 : \quad & (c_1 \approx e_1 \to d_1 \approx a_1) \wedge (c_2 \approx e_1 \to d_2 \approx a_1) \wedge (c_1 \approx e_2 \to d_1 \approx a_2) \ \wedge \ (c_2 \approx e_2 \to d_2 \approx a_2) \\ G_0 : \quad & c_1 \approx e_1 \wedge c_2 \approx e_2 \wedge (|d_1-d_2| < \frac{|c_1-c_2|}{\lambda_1} \vee |d_1-d_2| > \lambda_1|c_1-c_2|) \\ \mathsf{Con}[G]_0 \quad & c_1 \approx c_2 \to d_1 \approx d_2 \wedge a_1 \approx a_2 \to e_1 \approx e_2\end{aligned}$$

Thus, entailment (1) holds iff $\mathsf{BL}_0 \wedge \mathsf{Inv}_0 \wedge G_0 \wedge \mathsf{Con}[G]_0$ is unsatisfiable, i.e. iff

$$\exists a_1, a_2, c_1, c_2, d_1, d_2, e_1, e_2 (\mathsf{BL}_0 \wedge \mathsf{Inv}_0 \wedge G_0 \wedge \mathsf{Con}[G]_0) \text{ is false.}$$

The quantifiers can be eliminated with any QE system for $\mathbb{R}$. We used REDLOG [18]; after simplification (w.r.t. $\lambda{>}1, \lambda_1{>}1$ and some consequences) we obtained:

$$\begin{array}{l} \lambda_1\lambda^2 - \lambda < 0 \ \vee \ \lambda_1\lambda - \lambda^2 < 0 \ \vee \ \lambda_1 - \lambda < 0 \ \vee \ (\lambda_1\lambda - \lambda^2 > 0 \wedge \lambda_1 \approx \lambda) \vee \\ (\lambda_1^2\lambda - \lambda_1 > 0 \ \wedge \ (\lambda_1^2 - \lambda_1\lambda < 0 \ \vee \ (\lambda_1^2 - \lambda_1\lambda > 0 \ \wedge \ \lambda_1 \approx \lambda))) \vee \\ (\lambda_1^2\lambda - \lambda_1 > 0 \ \wedge \ \lambda_1^2 - \lambda_1\lambda < 0 \ \wedge \ \lambda_1 - \lambda > 0) \ \vee \ (\lambda_1^2\lambda - \lambda_1 > 0 \ \wedge \ \lambda_1^2 - \lambda_1\lambda < 0). \end{array}$$

If $\lambda{>}1, \lambda_1{>}1$, this formula is equivalent to $\lambda_1{<}\lambda$. Hence, if $\lambda{>}1, \lambda_1{>}1$ we have:

$$\mathbb{R} \cup (\mathsf{BL}_f^{\lambda}) \wedge (\mathbf{Inv}(f, g)) \models \phi, \quad \text{iff } \lambda_1 \geq \lambda. \tag{2}$$

The constraints we obtain can be used for optimization (e.g. we can show that the smallest value of λ_1 for which g satisfies the bi-Lipschitz condition is λ).

9.2 Mathematics: partial-ordered structures

We now present various of examples of local extensions in mathematics, involving possibly non-numeric functions. Some of the results below can be seen as a natural generalization of the results in Section 9.1.

9.2.1 Monotone functions

The extensions of any (possibly many-sorted) theory whose models are posets with functions satisfying the axioms $\mathsf{Mon}^{\sigma}(f)$ satisfy condition ($\mathsf{Comp}_{\mathsf{w}}$) if the codomains of the functions have a bounded semilattice reduct or are totally ordered [52, 56]

Theorem 44 ([56]) *The extensions with functions satisfying monotonicity axioms of the following (possibly many-sorted) classes of algebras are local:*

(1) Any class of algebras with a bounded (semi)lattice reduct, a bounded distributive lattice reduct, or a Boolean algebra reduct.

(2) Any extension of a class of algebras with a semilattice reduct, a (distributive) lattice reduct, or a Boolean algebra reduct, with monotone functions into an infinite numeric domain[7]*.*

(3) $\mathcal{T}$, the class of totally-ordered sets; $\mathcal{DO}$, the theory of dense totally-ordered sets.

Consider now the following conditions:

$\mathsf{SMon}(f) \quad \forall i, j (i < j \rightarrow f(i) < f(j)) \qquad$ and $\qquad \mathsf{Inj}(f) \quad \forall i, j (i \not\approx j \rightarrow f(i) \not\approx f(j))$

Theorem 45 ([37]) *Assume that in all models of $\mathcal{T}_0$ the support of sort i has an underlying strict total order relation $<$. Let $\mathcal{T}_1 = \mathcal{T}_0 \cup \mathsf{SMon}(f)$, where f is a new function of arity $\mathsf{i} \rightarrow \mathsf{e}$ (e may be a new or an old sort), in all models of $\mathcal{T}_1$ the support of sort e has an underlying strict total order $<$, and there exist injective order-preserving maps from any interval of the support of sort i to any interval of the support e. Then the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ satisfies ($\mathsf{Comp}_{\mathsf{w}}$), hence it is local.*

Theorem 46 ([37]) *A theory extension $\mathcal{T}_0 \subseteq \mathcal{T}_1 = \mathcal{T}_0 \cup \mathsf{Inj}(f)$ with a function f of arity $\mathsf{i} \rightarrow \mathsf{e}$ satisfying $\mathsf{Inj}(f)$ is local provided that in all models of $\mathcal{T}_1$ the cardinality of the support of sort i is lower or equal to the cardinality of the support of sort e.*

[7] Of interest in non-classical logics (e.g. description logics) [56, 59].

9.2.2 Extensions with definitions and boundedness conditions

Let $\mathcal{T}_0$ be a theory containing a binary predicate $\leq$ which is reflexive, and $f \notin \Sigma_0$.

Guarded boundedness. Let $m \in \mathbb{N}$. For $1 \leq i \leq m$ let $t_i(x_1, \ldots, x_n)$ and $s_i(x_1, \ldots, x_n)$ be terms in the signature Π_0 with variables among $x_1, \ldots, x_n$, and let $\phi_i(x_1, \ldots, x_n)$, $i \in \{1, \ldots, m\}$ be Π_0-formulae with free variables among $x_1, \ldots, x_n$, such that (i) for every $i \neq j$, $\phi_i \wedge \phi_j \models_{\mathcal{T}_0} \bot$, and (ii) for every i, $\mathcal{T}_0 \models \forall \overline{x}(\phi_i(\overline{x}) \rightarrow s_i(\overline{x}) \leq t_i(\overline{x}))$. Let $\mathsf{GBound}(f) = \bigwedge_{i=1}^{m} \mathsf{GBound}^{\phi_i}(f)$, where:

$\mathsf{GBound}^{\phi_i}(f) \quad \forall \overline{x}(\phi_i(\overline{x}) \rightarrow s_i(\overline{x}) \leq f(\overline{x}) \leq t_i(\overline{x}))$.

The extension $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathsf{GBound}(f)$ is local.

Boundedness for (strictly) monotone and injective functions. Any extension of a theory for which $\leq$ is a partial order (or at least reflexive) with functions satisfying $\mathsf{Mon}^{\sigma}(f)$ and boundedness $\mathsf{Bound}^{t}(f)$ conditions is local [53, 56].

$\mathsf{Bound}^{t}(f) \qquad \forall x_1, \ldots, x_n(f(x_1, \ldots, x_n) \leq t(x_1, \ldots, x_n))$

where $t(x_1, \ldots, x_n)$ is a Π_0-term with variables among $x_1, \ldots, x_n$ whose associated function has the same monotonicity as f in any model. Similar results hold for strictly monotone/injective functions (under the conditions in Thm. 45, 46).

Extensions with (guarded) definitions are defined similarly.

9.3 Verification

In the verification of reactive and hybrid systems the most important problems are to prove safety and lifeness.

9.3.1 System specification.

In the *specification of complex systems*, one needs to take several aspects into account: control flow, data changes, and timing aspects. An example is the specification language CSP-OZ-DC (COD) [34, 36] which integrates three well-investigated formalisms: *Communicating Sequential Processes* [35], *Object-Z* [51], and *Duration Calculus* [70], allowing the compositional and declarative specification of each aspect by means of the best-suited formalism. In particular, data and data changes can be specified in a constraint-based representation (using OZ). In particular, depending on the specific application domain, the specifications explicitly refer to certain theories and data types:

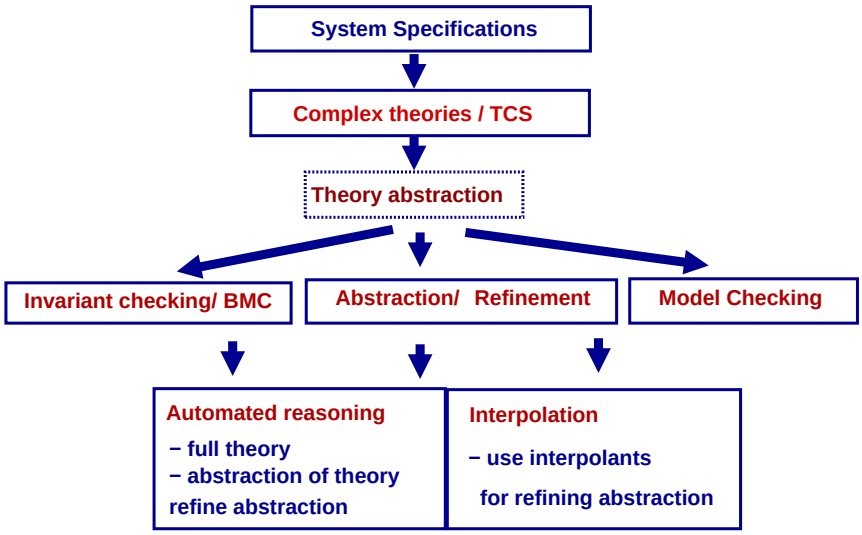
- numbers (real/rational/integer/natural);
- functions over numerical domains (e.g. for modeling parametric changes);
- theories of data structures such as lists or arrays – especially for program verification, but also e.g. in the specification of systems with a parametric number of components.

Such theories are usually explicitly mentioned in the specification part. Correspondingly, a *background theory* $\mathcal{T}_S$ – describing the data types used in the specification and their properties – is associated in a canonical way with every specification.

9.3.2 System verification.

Starting from the specifications of a reactive and real time system we can build a formal model for the system expressed as a transition constraint system $T = (V, \Omega, \mathsf{Init}, \mathsf{Update})$ which specifies:

- the variables (V) and function symbols (Ω) whose values change over time;
- a formula Init specifying the properties of initial states;



- a formula Update with variables in $V \cup V'$ and function symbols in $\Omega \cup \Omega'$ (where V' and Ω' are copies of V resp. Ω, denoting the variables resp. functions after the transition) which specifies the relationship between the values of variables x and function symbols f before a transition and their values (x', f') after the transition.

Such descriptions can be obtained from system specifications (for an example cf. [22]).

Verification problems usually considered are *invariant checking*, *bounded model checking* (BMC), verification using *abstraction/refinement* and *full verification*. The first two problems are usually much simpler than the last one. The use of abstraction and refinement can help provide additional safety proofs compared to the first two ones.

Invariant checking. We can check whether a formula Ψ is an inductive invariant of a transition constraint system $T=(V, \Omega, \mathsf{Init}, \mathsf{Update})$ in two steps:

(1) prove that $\mathcal{T}_S, \mathsf{Init} \models \Psi$;

(2) prove that $\mathcal{T}_S, \Psi, \mathsf{Update} \models \Psi'$, where Ψ' results from Ψ by replacing each $x \in \mathcal{V}$ by x' and each $f \in \Omega$ by f'.

Failure to prove (2) means that Ψ is not an invariant, or Ψ is not inductive w.r.t. T.[8]

Bounded model checking. We check whether, for a fixed k, unsafe states are reachable by runs of length at most k. Formally, we check whether:

$$\mathcal{T}_S \wedge \mathsf{Init}_0 \wedge \bigwedge_{i=1}^{j} \mathsf{Update}_i \wedge \neg\Psi_j \models \bot \qquad \text{for all } 0 \leq j \leq k,$$

where Update_i is obtained from Update by replacing all variables $x \in V$ by x_i and any $f \in \Omega$ by f_i, and all $x' \in V'$, $f' \in \Omega'$ by x_{i+1}, f_{i+1}; Init_0 is Init with x_0 replacing $x \in V$ and f_0 replacing $f \in \Omega$; Ψ_i is obtained from Ψ similarly.

We are interested in checking whether a safety property (expressed by a suitable formula) is an invariant, or holds for paths of bounded length, *for given instances of the parameters*, or *under given constraints on parameters*. We aim at identifying situations in which decision procedures exist. We will show that this is often the case, by investigating locality phenomena in verification. As a by-product, this will allow us to consider problems more

[8] Proving that Ψ is an invariant of the system in general requires to find a stronger formula Γ (i.e., $\mathcal{T}_S \models \Gamma \rightarrow \Psi$) and prove that Γ is an inductive invariant.

general than usual tasks in verification, namely to *derive constraints between parameters* which guarantee safety. These constraints may also be used to solve optimization problems (maximize/minimize some of the parameters) such that safety is guaranteed.

Abstraction/Refinement. In [43], McMillan proposed a method for abstraction-based verification in which interpolation (e.g. for linear arithmetic + free functions) is used for abstraction refinement. The idea is the following: Starting from a concrete, precise description of a (possibly infinite-state) system one can obtain a finite abstraction by merging the states into equivalence classes. A transition exists between two abstract states if there exists a transition in the concrete systems between representatives in the corresponding equivalence classes. Literals describing the relationships between the state variables at the concrete level are represented – at the abstract level – by predicates on the abstract states (equivalence classes of concrete states). Classical methods (e.g. BDD-based methods) can be used for checking whether there is a path in the abstract model from an initial state to an unsafe state. We distinguish the following cases:

(1) No unsafe state is reachable from an initial state in the abstract model. Then, due to the way transitions are defined in the abstraction, this is the case also at the concrete level. Hence, the concrete system is guaranteed to be safe.

(2) There exists a path in the abstract model from an initial state to an unsafe state. This path may or may not have a correspondent at the concrete level. In order to check this, we analyse the counterpart of the counterexample in the concrete model. This can be reduced to testing the satisfiability of a set of constraints:

$$\mathsf{Init}(s_0) \wedge \mathsf{Tr}(s_0, s_1) \wedge \cdots \wedge \mathsf{Tr}(s_{n-1}, s_n) \wedge \neg \mathsf{Safe}(s_n).$$

(2.1) If the set of constraints is satisfiable then an unsafe state is reached from the initial state also in the concrete system. Thus, the concrete system is not safe.

(2.2) If the set of constraints is unsatisfiable, then the counterexample obtained due to the abstraction was spurious. This means that the abstraction was too coarse. In order to refine it we need to take into account new predicates or relationships between the existing predicates. Interpolants provide information about which new predicates need to be used for refining the abstraction. Interpolation in local theory extensions has been studied in [53] and will not be discussed in this tutorial.

Full verification. Methods for the verification of finite-state systems are well-studied. In the presence of complex data types however, full verification is often impossible. Positive results have been obtained for locally finite theories of data structures by Ghilardi, Ranise, Nicolini and Zuccheli. The main idea is that local finiteness for the data structures ensures that the state space of the systems is finite, so ideas from finite model checking can be used. We will not present these results in detail here.

9.4 Some theories important in verification

In what follows we present methods for proving decidability of certain theories important in verification such as the theory of pointer data structures and a fragment of the theory of arrays.

9.4.1 Pointer data structures à la McPeak and Necula

In [45], McPeak and Necula investigate reasoning in pointer data structures. The language used has sorts p (pointer) and s (scalar). Sets Ω_p and Ω_s of pointer resp. scalar fields are

modeled by functions of sort $\mathsf{p} \to \mathsf{p}$ and $\mathsf{p} \to \mathsf{s}$, respectively. A constant null of sort p exists. The only predicate of sort p is equality; predicates of sort s can have any arity. The axioms considered in [45] are of the form

$$\forall p \quad \mathcal{E} \vee \mathcal{C} \tag{3}$$

where $\mathcal{E}$ contains disjunctions of pointer equalities and $\mathcal{C}$ contains scalar constraints (sets of both positive and negative literals). It is assumed that for all terms $f_1(f_2(\ldots f_n(p)))$ occurring in the body of an axiom, the axiom also contains the disjunction $p \approx \mathsf{null} \vee f_n(p) \approx \mathsf{null} \vee \cdots \vee f_2(\ldots f_n(p)) \approx \mathsf{null}$.[9] Examples of axioms (for doubly linked data structures with priorities) considered there are:

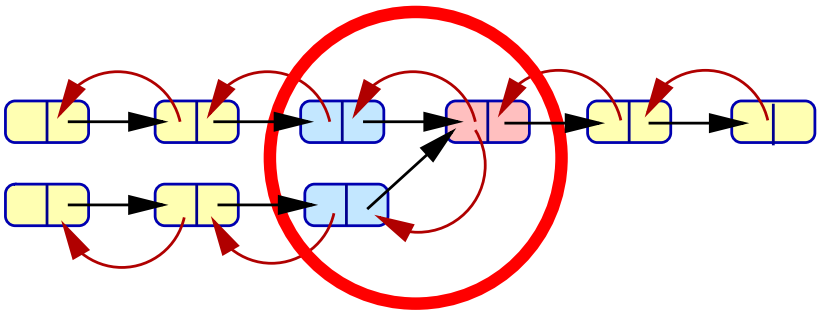
$$\forall p \quad p \not\approx \mathsf{null} \wedge \mathsf{next}(p) \not\approx \mathsf{null} \quad \rightarrow \quad \mathsf{prev}(\mathsf{next}(p)) \approx p \tag{4}$$
$$\forall p \quad p \not\approx \mathsf{null} \wedge \mathsf{prev}(p) \not\approx \mathsf{null} \quad \rightarrow \quad \mathsf{next}(\mathsf{prev}(p)) \approx p \tag{5}$$
$$\forall p \quad p \not\approx \mathsf{null} \wedge \mathsf{next}(p) \not\approx \mathsf{null} \quad \rightarrow \quad \mathsf{priority}(p) \geq \mathsf{priority}(\mathsf{next}(p)) \tag{6}$$

(the first two axioms state that prev is a left inverse for next, the third axiom is a monotonicity condition on the function $\mathsf{priority}$). Let $\Psi_{\mathcal{K}}(T) = \mathsf{st}(\mathcal{K}) \cup T \cup \{f(t) \mid t \in \mathsf{st}(\mathcal{K}) \cup T, f \in \Omega_s\}$ for any set of ground terms T.

Theorem 47 ([37]) *Let $\mathcal{T}_0$ be a Σ_0-theory, where $S_0 = \{\mathsf{s}\}$, and $\mathcal{T}_1 = \mathcal{T}_0 \cup \mathcal{K}$ be the extension of $\mathcal{T}_0$ with signature $\Sigma = (\{\mathsf{p}, \mathsf{s}\}, \Omega, \mathsf{Pred})$ – where $\Omega = \Omega_p \cup \Omega_s \cup \Omega_0$, and $\mathcal{K}$ is a set of axioms $\forall p(\mathcal{E} \vee \mathcal{C})$ of type (3). Then every partial model A of $\mathcal{K}$ with total Ω_0-functions such that the definition domain of A is closed under $\Psi_{\mathcal{K}}$ (i.e. if $f \in \Omega_s$ and the p-term t is defined in A then $f(t)$ is defined in A) weakly embeds into a total model of $\mathcal{K}$. Hence $\mathcal{T}_0 \subseteq \mathcal{T}_1$ is a Ψ-stably local extension.*

Ψ-stable locality is not harmful in this case, since all universally quantified variables in the axioms in $\mathcal{K}$ are of sort p, and the number of instances of these variables with subterms in $\Psi_{\mathcal{K}}(G)$ which need to be considered is polynomial in the size of $\mathsf{st}(\mathcal{K}, G)$ (no operations with output sort s generate such terms).

Figure 7: Properties of lists



Example: In order to prove that in a theory of doubly-linked lists it is not possible for two fields to have the same "next" field, we need to consider only instances of the axioms for "next" and "prec" in a neighborhood of a possible counterexample.

[9] This excludes null pointer errors.

9.4.2 Extensions of the fragment of Necula and McPeak.

We are interested in pointer structures which can be changed during execution of a program (a cell of a list can be removed, or a new subtree added into a tree structure). The general remarks above also apply for such situations.

Theorem 48 ([37]) *Assume that the update axioms* $\mathsf{Update}(\Omega, \Omega')$ *describe how the values of the* Ω*-functions change, depending on a finite set* $\{\phi_i \mid i \in I\}$ *of mutually exclusive conditions, expressed as formulae over the base signature and the* Ω*-functions (axioms of type (7) below represent precise ways of defining the updated functions, whereas axioms of type (8) represent boundedness properties on the updated scalar fields, assuming the scalar domains are partially ordered):*

$$\forall \overline{x}(\phi_i(\overline{x}) \rightarrow f_i'(\overline{x}) \approx s_i(\overline{x})) \qquad i \in I, \text{ where } \phi_i(\overline{x}) \wedge \phi_j(\overline{x}) \models_{\mathcal{T}_0} \bot \text{ for } i \neq j \tag{7}$$

$$\forall \overline{x}(\phi_i(\overline{x}) \rightarrow t_i(\overline{x}) \leq f_i'(\overline{x}) \leq s_i(\overline{x})) \qquad i \in I, \text{ where } \phi_i(\overline{x}) \wedge \phi_j(\overline{x}) \models_{\mathcal{T}_0} \bot \text{ for } i \neq j \tag{8}$$

where s_i, t_i *are terms over the signature* Ω *such that* $\mathcal{T}_0 \models \forall \overline{x}(\phi_i(\overline{x}) \rightarrow t_i(\overline{x}) \leq s_i(\overline{x}))$ *for all* $i \in I$*. They define local theory extensions. This holds for any extension of disjoint combinations of various pointer structures with such update axioms.*

9.4.3 The theory of arrays à la Bradley, Manna and Sipma

In [11] the *array property fragment* is studied, a fragment of the theory of arrays with Presburger arithmetic as index theory and parametric element theories. Consider the extension of the combination $\mathcal{T}_0$ of the index and element theories with functions read, write and axioms:

$$\text{read}(\text{write}(a, i, e), i) \approx e \qquad j \not\approx i \rightarrow \text{read}(\text{write}(a, i, e), j) \approx \text{read}(a, j).$$

The array property fragment is defined as follows[10]:

An *index guard* is a positive Boolean combination of atoms of the form $t \leq u$ or $t \approx u$ where t and u are either a variable of index sort or a ground term (of index sort) constructed from (Skolem) constants and integer numbers using addition and multiplication with integers. A formula of the form $(\forall i)(\varphi_I(i) \rightarrow \varphi_V(i))$ is an *array property* if φ_I is an index guard and if any universally quantified variable of index sort i only occurs in a direct array read $\mathsf{read}(a, x)$ in φ_V. Array reads may not be nested. The *array property fragment* consists of all existentially-closed Boolean combinations of array property formulae and quantifier-free formulae.

The decision procedure proposed in [11] decides satisfiability of formulae in negation normal form in the array property fragment in the following steps.

1. Replace all existentially quantified array variables with Skolem constants; replace all terms of the form $\mathsf{read}(a, i)$ with $a(i)$; eliminate all terms of the form $\mathsf{write}(a, i, e)$ by replacing the formula $\phi(\mathsf{write}(a, i, e))$ with the conjunction of the formula $\phi(b)$ (obtained by introducing a fresh array name b for $\mathsf{write}(a, i, e)$) with $(b(i) \approx e) \wedge \forall j(j \leq i - 1 \vee i + 1 \leq j \rightarrow b(j) \approx a(j))$.[11]

2. Existentially quantified index variables are replaced with Skolem constants.

3. Universal quantification over index variables is replaced by conjunction of suitably chosen instances of the variables.

[10] The considerations below are for arrays of dimension 1, the general case is similar.

[11] Note that, by the definition of array property formulae, if a term $\mathsf{write}(a, i, e)$ occurs in the array property fragment then i is an existentially quantified index variable.

For determining the set of ground instances to be used in Step 3, the authors prove that certain partial "minimal" models can be completed to total ones.

Theorem 49 ([11, 37]) *Let $\mathcal{K}$ be the clause part and G the ground part (after the transformation steps (1)–(3)), and $\mathcal{I}$ be the set of index terms defined in [11]. Let $\Psi(\mathcal{K}, \mathsf{st}(G)) = \{f(i_1, \ldots, i_n) \mid f$ array name $, i_1, \ldots, i_n \in I\}$. Every partial model of $\mathcal{T}_0 \cup \mathcal{K}[\Psi(\mathcal{K}, G)] \cup G$ in which all terms in $\Psi(\mathcal{K}, G)$ are defined can be transformed into a (total) model of $\mathcal{T}_0 \cup \mathcal{K} \cup G$. This criterion entails (ELoc^{Ψ}).*

9.4.4 Extending the array property fragment.

Let $\mathcal{T}_0$ be the array property fragment in [11] (set of arrays Ω_0). There are several ways of extending $\mathcal{T}_0$:

Theorem 50 ([37]) *Let $\mathcal{T}_1 = \mathcal{T}_0 \cup \mathcal{K}$ be an extension of $\mathcal{T}_0$ with new arrays in a set Ω_1.*

(1) If $\mathcal{K}$ consists of guarded boundedness axioms, or guarded definitions (cf. Section 9.1.1) for the Ω_1-function symbols, then the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ is local. [12]

(2) If $\mathcal{K}$ consists of injectivity or (strict) monotonicity (and possibly boundedness axioms) for the function symbols in Ω_1 then the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ is local if the assumptions about the element theory specified in Section 9.1.1 hold.

(3) Any combination of extensions of $\mathcal{T}_0$ as those mentioned in (1),(2) with disjoint sets of new array constants leads to a local extension of $\mathcal{T}_0$.

If the guards ϕ_i of the axioms in $\mathcal{K}$ are clauses then the result of the hierarchical reasoning method in Theorem 27 is a formula in $\mathcal{T}_0$, hence satisfiability of ground clauses w.r.t. $\mathcal{T}_0 \cup \mathcal{K}$ is decidable. Similarly for chains of extensions. The same holds for testing satisfiability of goals $\Gamma_0 \cup G$ where Γ_0 and $(\mathcal{K}[G])_0$ belong to the array property fragment. For general guards and chains of extensions decidability depends on the form of the formulae obtained by hierarchical reduction(s).

9.5 Case studies in verification

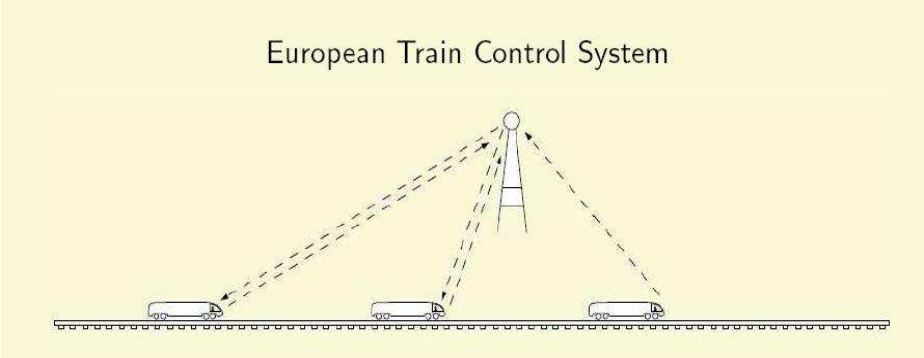
We present three case studies in verification which illustrate the use of decision procedures for the theories previously mentioned for invariant checking and bounded model checking. The examples below only refer to invariant checking. For the sake of simplicity, in this section we use the notation "=" for the equality predicate.

9.5.1 A RBC Case Study

The case study we discuss here is taken from the specification of the European Train Control System (ETCS) standard [21] and presented in detail in [39] and [57]. We consider a radio block center (RBC), which communicates with all trains on a given track segment. Trains may enter and leave the area, provided that a certain maximum number of trains on the track is not exceeded. Every train reports its position to the RBC in given time intervals and the RBC communicates to every train how far it can safely move, based on the position of the preceding train. It is then the responsibility of the trains to adjust their speed between given minimum and maximum speeds.

For a first try at verifying properties of this system, we have considerably simplified it: we abstract from the communication issues in that we always evaluate the system after a certain time $\Delta\mathsf{t}$, and at these evaluation points the positions of all trains are known. Depending on these positions, the possible speed of every train until the next evaluation is

[12] An example is defining new arrays by writing x at a (constant) index c, axiomatized by $\{\forall i(i \not\approx c \rightarrow a'(i) = a(i)), \quad \forall i(i = c \rightarrow a'(i) = x)\}$.



decided: if the distance to the preceding train is less than a certain limit $\mathsf{l_{alarm}}$, the train may only move with minimum speed min (otherwise with any speed between min and the maximum speed max).

We present two formal system models. In the first one we have a fixed number of trains; in the second we allow for entering and leaving trains.

Model 1: Fixed Number of Trains. In this simpler model, any state of the system is characterized by the real-valued constants $\Delta\mathsf{t} > 0$ (the time between evaluations of the system), min and max (the minimum and maximum speed of trains), $\mathsf{l_{alarm}}$ (the distance between trains which is deemed secure), the integer constant n (the number of trains) and the function pos (mapping integers between 0 and $n-1$ to real values representing the position of the corresponding train).

We use an additional function $\mathsf{pos'}$ to model the evolution of the system: $\mathsf{pos'}(i)$ denotes the position of i at the next evaluation point (after $\Delta\mathsf{t}$ time units). The way positions change (i.e. the relationship between pos and $\mathsf{pos'}$) is defined by the following set $\mathcal{K}_f = \{\mathsf{F1}, \mathsf{F2}, \mathsf{F3}, \mathsf{F4}\}$ of axioms[13]:

$$\mathsf{(F1)} \quad \forall i \quad (i = 0 \quad \rightarrow \quad \mathsf{pos}(i) + \Delta\mathsf{t}{*}\mathsf{min} \leq_{\mathbb{R}} \mathsf{pos'}(i) \leq_{\mathbb{R}} \mathsf{pos}(i) + \Delta\mathsf{t}{*}\mathsf{max})$$

$$\mathsf{(F2)} \quad \forall i \quad (0 < i < \mathsf{n} \quad \wedge \quad \mathsf{pos}(i-1) >_{\mathbb{R}} 0 \quad \wedge \quad \mathsf{pos}(p(i)) - \mathsf{pos}(i) \geq_{\mathbb{R}} \mathsf{l_{alarm}} \rightarrow \mathsf{pos}(i) + \Delta\mathsf{t} * \mathsf{min} \leq_{\mathbb{R}} \mathsf{pos'}(i) \leq_{\mathbb{R}} \mathsf{pos}(i) + \Delta\mathsf{t}{*}\mathsf{max})$$

$$\mathsf{(F3)} \quad \forall i \quad (0 < i < \mathsf{n} \quad \wedge \quad \mathsf{pos}(i-1) >_{\mathbb{R}} 0 \quad \wedge \quad \mathsf{pos}(p(i)) - \mathsf{pos}(i) <_{\mathbb{R}} \mathsf{l_{alarm}} \rightarrow \mathsf{pos'}(i) = \mathsf{pos}(i) + \Delta\mathsf{t}{*}\mathsf{min})$$

$$\mathsf{(F4)} \quad \forall i \quad (0 < i < \mathsf{n} \quad \wedge \quad \mathsf{pos}(i-1) \leq_{\mathbb{R}} 0 \quad \rightarrow \quad \mathsf{pos'}(i) = \mathsf{pos}(i))$$

Note that the train with number 0 is the train with the greatest position, i.e. we count trains from highest to lowest position.

Axiom F1 states that the first train may always move at any speed between min and max. F2 states that the other trains can do so if their predecessor has already started and the distance to it is larger than $\mathsf{l_{alarm}}$. If the predecessor of a train has started, but is less than $\mathsf{l_{alarm}}$ away, then the train may only move at speed min (axiom F3). F4 requires that a train may not move at all if its predecessor has not started.

Model 2: Incoming and leaving trains. If we allow incoming and leaving trains, we additionally need a measure for the number of trains on the track. This is given by additional constants first and last, which at any time give the number of the first and

[13] Inequality over integers is displayed without subscript, inequality over reals is marked with an $\mathbb{R}$

last train on the track (again, the first train is supposed to be the train with the highest position). Furthermore, the maximum number of trains that is allowed to be on the track simultaneously is given by a constant maxTrains. These three values replace the number of trains n in the simpler model, the rest of it remains the same except that the function pos is now defined for values between first and last, where before it was defined between 0 and $\mathsf{n}-1$. The behavior of this extended system is described by the following set $\mathcal{K}_v$ consisting of axioms (V1) – (V9):

(V1) $\forall i \quad (i = \mathsf{first} \rightarrow \mathsf{pos}(i) + \Delta\mathsf{t} * \mathsf{min} \leq_{\mathbb{R}} \mathsf{pos}'(i) \leq_{\mathbb{R}} \mathsf{pos}(i) + \Delta\mathsf{t} * \mathsf{max})$

(V2) $\forall i \quad (\mathsf{first} < i \leq \mathsf{last} \wedge \mathsf{pos}(i-1) >_{\mathbb{R}} 0 \wedge \mathsf{pos}(i-1) - \mathsf{pos}(i) \geq_{\mathbb{R}} \mathsf{l_{alarm}}$
$\rightarrow \mathsf{pos}(i) + \Delta\mathsf{t} * \mathsf{min} \leq_{\mathbb{R}} \mathsf{pos}'(i) \leq_{\mathbb{R}} \mathsf{pos}(i) + \Delta\mathsf{t} * \mathsf{max})$

(V3) $\forall i \quad (\mathsf{first} < i \leq \mathsf{last} \wedge \mathsf{pos}(i-1) >_{\mathbb{R}} 0 \wedge \mathsf{pos}(i-1) - \mathsf{pos}(i) <_{\mathbb{R}} \mathsf{l_{alarm}}$
$\rightarrow \mathsf{pos}'(i) = \mathsf{pos}(i) + \Delta\mathsf{t} * \mathsf{min})$

(V4) $\forall i \quad (\mathsf{first} < i \leq \mathsf{last} \wedge \mathsf{pos}(i-1) \leq_{\mathbb{R}} 0 \rightarrow \mathsf{pos}'(i) = \mathsf{pos}(i))$

(V5) $\mathsf{last} - \mathsf{first} + 1 < \mathsf{maxTrains} \rightarrow \mathsf{last}' = \mathsf{last} \vee \mathsf{last}' = \mathsf{last} + 1$

(V6) $\mathsf{last} - \mathsf{first} + 1 = \mathsf{maxTrains} \rightarrow \mathsf{last}' = \mathsf{last}$

(V7) $\mathsf{last} - \mathsf{first} + 1 > 0 \rightarrow \mathsf{first}' = \mathsf{first} \vee \mathsf{first}' = \mathsf{first} + 1$

(V8) $\mathsf{last} - \mathsf{first} + 1 = 0 \rightarrow \mathsf{first}' = \mathsf{first}$

(V9) $\mathsf{last}' = \mathsf{last} + 1 \quad \rightarrow \quad \mathsf{pos}'(\mathsf{last}') <_{\mathbb{R}} \mathsf{pos}'(\mathsf{last})$

where primed symbols denote the state of the system at the next evaluation.

Here, axioms $\mathsf{V1}-\mathsf{V4}$ are similar to $\mathsf{F1}-\mathsf{F4}$, except that the fixed bounds are replaced by the constants first and last. V5 states that if the number of trains is less than maxTrains, then a new train may enter or not. V6 says that no train may enter if maxTrains is already reached. V7 and V8 are similar conditions for leaving trains. Finally, V9 states that if a train enters, its position must be behind the train that was last before.

The safety condition which is important for this type of systems is collision freeness. In [39] we use a very simplified model of the system of trains, where collision freeness is modeled by a 'strict monotonicity' property for the function pos which stores the positions of the trains. We now consider a more realistic axiomatization – assuming the maximum length of the trains is known – expressed by the following axiom:

SdMon(pos) $\quad \forall\, i,j,k \quad (0 \leq j < i < \mathsf{n} \ \wedge \ i - j = k \rightarrow \mathsf{pos}(j) - \mathsf{pos}(i) \geq_{\mathbb{R}} k * \mathsf{LengthTrain}),$

where LengthTrain is the standard (resp. maximal) length of a train.

As base theory we consider the combination $\mathcal{T}_0$ of the theory of integers and reals with a multiplication operation $*$ of arity $\mathsf{i} \times \mathsf{num} \rightarrow \mathsf{num}$ (multiplication of k with the constant LengthTrain in the formula above)[14]. Let $\mathcal{T}_1$ be the theory obtained by extending $\mathcal{T}_0$ with a function pos satisfying the axiom above. By the results presented in Section 9.1.1 on locality of extensions with functions satisfying strict monotonicity and related properties, the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$ is local.

We now extend the resulting theory $\mathcal{T}_1$ in two different ways, with the axiom sets for one of the two system models, respectively. Both extensions are extensions with guarded boundedness axioms. By the remarks in Section 9.1 both $\mathcal{T}_1 \subseteq \mathcal{T}_1 \cup \mathcal{K}_f$ and $\mathcal{T}_1 \subseteq \mathcal{T}_1 \cup \mathcal{K}_v$ are

[14] In the light of locality properties of such extensions, k will always be instantiated by values in a finite set of *concrete* integers, all within a given, *concrete* range; thus the introduction of this many-sorted multiplication does not affect decidability.

local extensions. The method for hierarchical reasoning described above allows us to reduce the problem of checking whether system properties such as collision freeness are inductive invariants to deciding satisfiability of corresponding constraints in $\mathcal{T}_0$. As a side effect, after the reduction of the problem to a satisfiability problem in the base theory, one can automatically determine constraints on the parameters (e.g. $\Delta\mathsf{t}, \mathsf{min}, \mathsf{max}, ...$) which guarantee that the property is an inductive invariant, and are sufficient for this. (This can be achieved for instance using quantifier elimination.) These constraints can be used, for instance, for optimization problems (e.g. maximize speed and Δt while guaranteeing safety).

Illustration: We indicate how to apply hierarchical reasoning on the case study given in Section 9.5.1, Model 1[15]. We follow the steps given in Section 8.2 and show how the sets of formulas are obtained that can finally be handed to a prover of the base theory.

To check whether $\mathcal{T}_1 \cup \mathcal{K}_f \models \mathsf{ColFree}(\mathsf{pos}')$, where

$$\mathsf{ColFree}(\mathsf{pos}') \quad \forall i \quad (0 \leq i < \mathsf{n} - 1 \quad \rightarrow \quad \mathsf{pos}'(i) - \mathsf{pos}'(i+1) >_{\mathbb{R}} \mathsf{LengthTrain}),$$

we check whether $\mathcal{T}_1 \cup \mathcal{K}_f \cup G \models \bot$, where $G = \{0 \leq k < \mathsf{n} - 1, \quad k' = k + 1, \quad \mathsf{pos}'(k) - \mathsf{pos}'(k') \leq_{\mathbb{R}} \mathsf{LengthTrain}\}$ is the (skolemized) negation of $\mathsf{ColFree}(\mathsf{pos}')$, flattened by introducing a new constant k'.

Reduction from $\mathcal{T}_1 \cup \mathcal{K}_f$ to $\mathcal{T}_1$. This problem is reduced to a satisfiability problem over $\mathcal{T}_1$ as follows:

Step 1: Use locality. We construct the set $\mathcal{K}_f[G]$: There are no ground subterms with pos' at the root in $\mathcal{K}_f$, and only two ground terms with pos' in G, $\mathsf{pos}'(k)$ and $\mathsf{pos}'(k')$. This means that $\mathcal{K}_f[G]$ consists of two instances of $\mathcal{K}_f$: one with i instantiated to k, the other instantiated to k'. E.g., the two instances of $\mathsf{F2}$ are:

$$\begin{aligned}(\mathsf{F2[G]})\ & (0 < k < \mathsf{n} \quad \wedge \quad \mathsf{pos}(k-1) >_{\mathbb{R}} 0 \quad \wedge \quad \mathsf{pos}(k-1) - \mathsf{pos}(k) \geq_{\mathbb{R}} \mathsf{l_{alarm}} \\ & \quad \rightarrow \mathsf{pos}(k) + \Delta\mathsf{t} * \mathsf{min} \leq_{\mathbb{R}} \mathsf{pos}'(k) \leq_{\mathbb{R}} \mathsf{pos}(k) + \Delta\mathsf{t}{*}\mathsf{max}) \\ & (0 < k' < \mathsf{n} \quad \wedge \quad \mathsf{pos}(k'-1) >_{\mathbb{R}} 0 \quad \wedge \quad \mathsf{pos}(k'-1) - \mathsf{pos}(k') \geq_{\mathbb{R}} \mathsf{l_{alarm}} \\ & \quad \rightarrow \mathsf{pos}(k') + \Delta\mathsf{t} * \mathsf{min} \leq_{\mathbb{R}} \mathsf{pos}'(k') \leq_{\mathbb{R}} \mathsf{pos}(k') + \Delta\mathsf{t}{*}\mathsf{max})\end{aligned}$$

The construction of (F1[G]), (F3[G]) and (F4[G]) is similar. In addition, we specify the known relationships between the constants of the system:
(Dom) $\Delta\mathsf{t} > 0 \quad \wedge \quad 0 \leq \mathsf{min} \quad \wedge \quad \mathsf{min} \leq \mathsf{max}$

Step 2: Flattening and purification. $\mathcal{K}_f[G] \wedge G$ is already flat w.r.t. pos'. We replace all ground terms with pos' at the root with new constants: we replace $\mathsf{pos}'(k)$ by c_1 and $\mathsf{pos}'(k')$ by c_2. We obtain a set of definitions $\mathsf{Def} = \{\mathsf{pos}'(k) = c_1, \mathsf{pos}'(k') = c_2\}$ and a set $(\mathsf{Dom}) \cup \mathsf{G_0} \cup \mathcal{K}_{f_0}$ of clauses which do not contain occurrences of pos', consisting of (Dom) together with:

$$(\mathsf{G_0}) \qquad 0 \leq k < \mathsf{n} - 1 \quad \wedge \quad k' = k + 1 \quad \wedge \quad c_1 - c_2 \leq_{\mathbb{R}} \mathsf{LengthTrain}$$

$$\begin{aligned}(\mathsf{F2_0})\ & (0 < k < \mathsf{n} \quad \wedge \quad \mathsf{pos}(k-1) >_{\mathbb{R}} 0 \quad \wedge \quad \mathsf{pos}(k-1) - \mathsf{pos}(k) \geq_{\mathbb{R}} \mathsf{l_{alarm}} \\ & \quad \rightarrow \mathsf{pos}(k) + \Delta\mathsf{t} * \mathsf{min} \leq_{\mathbb{R}} c_1 \leq_{\mathbb{R}} \mathsf{pos}(k) + \Delta\mathsf{t}{*}\mathsf{max}) \\ & (0 < k' < \mathsf{n} \quad \wedge \quad \mathsf{pos}(k'-1) >_{\mathbb{R}} 0 \quad \wedge \quad \mathsf{pos}(k'-1) - \mathsf{pos}(k') \geq_{\mathbb{R}} \mathsf{l_{alarm}} \\ & \quad \rightarrow \mathsf{pos}(k') + \Delta\mathsf{t} * \mathsf{min} \leq_{\mathbb{R}} c_2 \leq_{\mathbb{R}} \mathsf{pos}(k') + \Delta\mathsf{t}{*}\mathsf{max})\end{aligned}$$

[15] We illustrate our approach for the simplest model. For a variable number of trains the approach is the same.

The construction can be continued similarly for F1, F3 and F4.

Step 3: Reduction to satisfiability in $\mathcal{T}_1$. We add the functionality clause $N_0 = \{k = k' \rightarrow c_1 = c_2\}$ and obtain a satisfiability problem in $\mathcal{T}_1$: $\mathcal{K}_{f_0} \wedge G_0 \wedge N_0$.

The reduction is schematically represented in the following diagram:

Def	$\mathsf{Dom} \cup \mathsf{G_0} \cup \mathcal{K}_{f_0} \cup N_0$
$\mathsf{pos}'(k) = c_1$ $\mathsf{pos}'(k') = c_2$	$(\mathsf{Dom}) \cup (\mathsf{G_0}) \cup (\mathsf{F1_0}) \cup (\mathsf{F2_0}) \cup (\mathsf{F3_0}) \cup (\mathsf{F4_0})$ $N_0 : k = k' \rightarrow c_1 = c_2$

Reduction from $\mathcal{T}_1$ to $\mathcal{T}_0$. To decide satisfiability of $\mathcal{T}_1 \wedge \mathcal{K}_{f_0} \wedge G_0 \wedge N_0$, we have to perform another transformation w.r.t. the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$. The resulting set of ground clauses can directly be handed to a decision procedure for the combination of the theory of indices and that of reals. We flatten and purify the set $\mathcal{K}_{f_0} \wedge G_0 \wedge N_0$ of ground clauses w.r.t. pos by introducing new constants denoting $k-1$ and $k'-1$, together with their definitions $k'' = k-1, k''' = k'-1$; as well as constants d_1 for $\mathsf{pos}(k)$, d_2 for $\mathsf{pos}(k')$, d_3 for $\mathsf{pos}(k'')$ and d_4 for $\mathsf{pos}(k''')$ together with their definitions. Taking into account only the corresponding instances of the collision freeness axiom for pos we obtain a set of clauses consisting of (Dom) together with:

(G_0') $\quad k'' = k-1 \;\wedge\; k''' = k'-1$

$(\mathsf{G_0})$ $\quad 0 \leq k < n-1 \;\wedge\; k' = k+1 \;\wedge\; c_1 - c_2 \leq_{\mathbb{R}} \mathsf{LengthTrain}$

$(\mathsf{GF1_0})$ $\quad k = 0 \;\rightarrow\; d_1 + \Delta t{*}\mathsf{min} \leq_{\mathbb{R}} c_1 \leq_{\mathbb{R}} d_1 + \Delta t{*}\mathsf{max}$
$k' = 0 \;\rightarrow\; d_2 + \Delta t{*}\mathsf{min} \leq_{\mathbb{R}} c_2 \leq_{\mathbb{R}} d_2 + \Delta t{*}\mathsf{max}$

$(\mathsf{GF2_0})$ $\quad 0<k<n \;\wedge\; d_3 >_{\mathbb{R}} 0 \;\wedge\; d_3 - d_1 \geq_{\mathbb{R}} \mathsf{l_{alarm}} \rightarrow d_1 + \Delta\mathsf{t}{*}\mathsf{min} \leq_{\mathbb{R}} c_1 \leq_{\mathbb{R}} d_1 + \Delta\mathsf{t}{*}\mathsf{max}$
$0<k'<n \;\wedge\; d_4 >_{\mathbb{R}} 0 \;\wedge\; d_4 - d_2 \geq_{\mathbb{R}} \mathsf{l_{alarm}} \rightarrow d_2 + \Delta\mathsf{t}{*}\mathsf{min} \leq_{\mathbb{R}} c_2 \leq_{\mathbb{R}} d_2 + \Delta\mathsf{t}{*}\mathsf{max}$

$(\mathsf{GF3_0})$ $\quad 0<k<n \;\wedge\; d_3 >_{\mathbb{R}} 0 \;\wedge\; d_3 - d_1 <_{\mathbb{R}} l_{\mathsf{alarm}} \rightarrow c_1 = d_1 + \Delta t{*}\mathsf{min}$
$0<k'<n \;\wedge\; d_4 >_{\mathbb{R}} 0 \;\wedge\; d_4 - d_2 <_{\mathbb{R}} l_{\mathsf{alarm}} \rightarrow c_2 = d_2 + \Delta t{*}\mathsf{min}$

$(\mathsf{GF4_0})$ $\quad (0<k<n \;\wedge\; d_3 \leq_{\mathbb{R}} 0 \;\rightarrow\; c_1 = d_1) \wedge (0<k'<n \;\wedge\; d_4 \leq_{\mathbb{R}} 0 \;\rightarrow\; c_2 = d_2)$

$\mathsf{SdMon(pos)[G']}$

$$
\begin{array}{rrrl}
0 \leq k < & k' < \mathsf{n} & \rightarrow & d_1 - d_2 >_{\mathbb{R}} (k' - k) * \mathsf{LengthTrain} \\
0 \leq k' < & k < \mathsf{n} & \rightarrow & d_2 - d_1 >_{\mathbb{R}} (k - k') * \mathsf{LengthTrain} \\
0 \leq k < & k'' < \mathsf{n} & \rightarrow & d_1 - d_3 >_{\mathbb{R}} (k'' - k) * \mathsf{LengthTrain} \\
0 \leq k'' < & k < \mathsf{n} & \rightarrow & d_3 - d_1 >_{\mathbb{R}} (k - k'') * \mathsf{LengthTrain} \\
0 \leq k < & k''' < \mathsf{n} & \rightarrow & d_1 - d_4 >_{\mathbb{R}} (k''' - k) * \mathsf{LengthTrain} \\
0 \leq k''' < & k < \mathsf{n} & \rightarrow & d_4 - d_1 >_{\mathbb{R}} (k - k''') * \mathsf{LengthTrain} \\
0 \leq k' < & k'' < \mathsf{n} & \rightarrow & d_2 - d_3 >_{\mathbb{R}} (k'' - k') * \mathsf{LengthTrain} \\
0 \leq k'' < & k' < \mathsf{n} & \rightarrow & d_3 - d_2 >_{\mathbb{R}} (k' - k'') * \mathsf{LengthTrain} \\
0 \leq k' < & k''' < \mathsf{n} & \rightarrow & d_2 - d_4 >_{\mathbb{R}} (k''' - k') * \mathsf{LengthTrain} \\
0 \leq k''' < & k' < n & \rightarrow & d_4 - d_2 >_{\mathbb{R}} (k' - k''') * \mathsf{LengthTrain} \\
0 \leq k'' < & k''' < n & \rightarrow & d_3 - d_4 >_{\mathbb{R}} (k''' - k'') * \mathsf{LengthTrain} \\
0 \leq k''' < & k'' < n & \rightarrow & d_4 - d_3 >_{\mathbb{R}} (k'' - k''') * \mathsf{LengthTrain}
\end{array}
$$

$N_0(\mathsf{pos}')$ $\quad k = k' \rightarrow c_1 = c_2$

$N_0(\mathsf{pos})$ $\quad k = k' \rightarrow d_1 = d_2 \;\wedge\; k = k'' \rightarrow d_1 = d_3 \;\wedge\; k' = k''' \rightarrow d_2 = d_4$
$k = k''' \rightarrow d_1 = d_4 \;\wedge\; k' = k'' \rightarrow d_2 = d_3 \;\wedge\; k'' = k''' \rightarrow d_3 = d_4$

In fact, the constraints on indices can help to further simplify the instances of monotonicity of $\mathsf{Mon(pos)}[G'] \wedge N_0(\mathsf{pos}) \wedge N_0(\mathsf{pos}')$: $k' > k, k'' < k, k'' < k', k''' < k', k''' = k$. The set

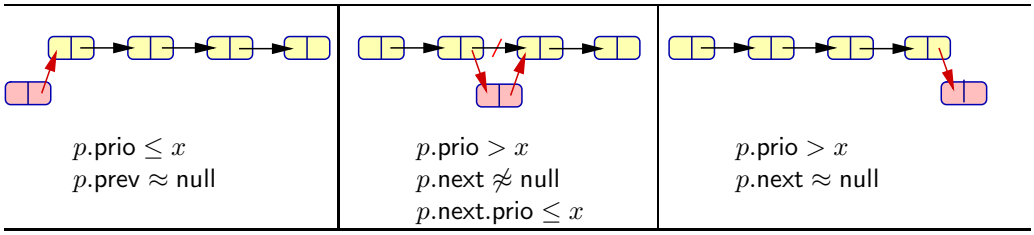


Figure 8: Insertion in lists

of clauses equivalent to $\mathsf{SdMon}(\mathsf{pos})[G'] \wedge N_0(\mathsf{pos}) \wedge N_0(\mathsf{pos}')$ is given below. (Here we do these simplifications by hand; this can be done as well by a pre-simplification program which detects obviously true relationships between the premises of these rules.) After making these simplifications we obtain the following set of (many-sorted) constraints:

$C_{\mathsf{Definitions}}$	C_{Indices} (sort i)	C_{Reals}	C_{Mixed}
$\mathsf{pos}'(k) = c_1$	$k' = k + 1$	$d_1 - d_2 >_{\mathbb{R}} \mathsf{LengthTrain}$	$0 \leq k'' \rightarrow d_3 - d_2 >_{\mathbb{R}} 2 * \mathsf{LengthTrain}$
$\mathsf{pos}(k') = d_2$	$k'' = k - 1$	$d_3 - d_4 >_{\mathbb{R}} \mathsf{LengthTrain}$	$0 \leq k'' \rightarrow d_3 - d_1 >_{\mathbb{R}} \mathsf{LengthTrain}$
$\mathsf{pos}'(k') = c_2$	$k''' = k' - 1$	$d_3 - d_2 >_{\mathbb{R}} \mathsf{LengthTrain}$	$(\mathsf{GF1}_0)$
$\mathsf{pos}(k'') = d_3$		$d_4 - d_2 >_{\mathbb{R}} \mathsf{LengthTrain}$	$(\mathsf{GF2}_0)$
$\mathsf{pos}(k) = d_1$	$0 \leq k < n - 1$	$d_1 = d_4 \wedge c_1 - c_2 \leq_{\mathbb{R}}$	$(\mathsf{GF3}_0)$
$\mathsf{pos}(k''') = d_4$	$0 \leq k' < n - 1$	(Dom)	$(\mathsf{GF4}_0)$

For checking the satisfiability of $C_{\mathsf{Indices}} \wedge C_{\mathsf{Reals}} \wedge C_{\mathsf{Mixed}}$ we can use a prover for the two-sorted combination of the theory of integers and the theory of reals, possibly combined with a DPLL methodology for dealing with full clauses. An alternative method is discussed in [39].

9.5.2 Verification of a program changing pointer structures

Consider the following algorithm for inserting an element c with priority field $c.\mathsf{prio} = x$ into a doubly-linked list sorted w.r.t. the priority fields.

$c.\mathsf{prio} = x, c.\mathsf{next} = \mathsf{null}$
for all $p \neq c$ **do**
if $p.\mathsf{prio} \leq x$ **then if** $p.\mathsf{prev} = \mathsf{null}$ **then** $c.\mathsf{next}' = p$, **endif**; $p.\mathsf{next}' = p.\mathsf{next}$
$p.\mathsf{prio} > x$ **then** **case** $p.\mathsf{next} = \mathsf{null}$ **then** $p.\mathsf{next}' := c, c.\mathsf{next}' = \mathsf{null}$
$p.\mathsf{next} \neq \mathsf{null} \wedge p.\mathsf{next.prio} > x$ **then** $p.\mathsf{next}' = p.\mathsf{next}$
$p.\mathsf{next} \neq \mathsf{null} \wedge p.\mathsf{next.prio} \leq x$ **then** $p.\mathsf{next}' = c$, $c.\mathsf{next}' = p.\mathsf{next}$

The update rules $\mathsf{Update}(\mathsf{next}, \mathsf{next}')$ can be read from the program above, and are represented in Fig. 8:

$\forall p(p{\neq}\mathsf{null} \wedge p{\neq}c \wedge \mathsf{prio}(p){\leq}x \wedge (\mathsf{prev}(p) = \mathsf{null}) \rightarrow \mathsf{next}'(c){=}p \wedge \mathsf{next}'(p){=}\mathsf{next}(p))$
$\forall p(p{\neq}\mathsf{null} \wedge p{\neq}c \wedge \mathsf{prio}(p){\leq}x \wedge (\mathsf{prev}(p) \neq \mathsf{null}) \rightarrow \mathsf{next}'(p){=}\mathsf{next}(p))$
$\forall p(p{\neq}\mathsf{null} \wedge p{\neq}c \wedge \mathsf{prio}(p){>}x \wedge \mathsf{next}(p){=}\mathsf{null} \rightarrow \mathsf{next}'(p){=}c \wedge \mathsf{next}'(c){=}\mathsf{null})$
$\forall p(p{\neq}\mathsf{null} \wedge p{\neq}c \wedge \mathsf{prio}(p){>}x \wedge \mathsf{next}(p){\neq}\mathsf{null} \wedge \mathsf{prio}(\mathsf{next}(p)){>}x \rightarrow \mathsf{next}'(p){=}\mathsf{next}(p))$
$\forall p(p{\neq}\mathsf{null} \wedge p{\neq}c \wedge \mathsf{prio}(p){>}x \wedge \mathsf{next}(p){\neq}\mathsf{null} \wedge \mathsf{prio}(\mathsf{next}(p)){\leq}x \rightarrow \mathsf{next}'(p){=}c{\wedge}\mathsf{next}'(c){=}\mathsf{next}(p))$

We prove that if the list is sorted, it remains so after insertion, i.e. the formula:

$$d \neq \mathsf{null} \wedge \mathsf{next}'(d) \neq \mathsf{null} \wedge \neg \mathsf{prio}(d) \geq \mathsf{prio}(\mathsf{next}'(d))$$

is unsatisfiable in the extension $\mathcal{T}_1 = \mathcal{T}_0 \cup \mathsf{Update}(\mathsf{next}, \mathsf{next}')$ of the theory $\mathcal{T}_0$ of doubly linked lists with a monotone field prio. $\mathcal{T}_0$ is axiomatized by the axioms $\mathcal{K} = \{(4), (5), (6)\}$ in Section 9.4.1. The update rules are guarded boundedness axioms, so the extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$

is local. Hence, the satisfiability task above w.r.t. $\mathcal{T}_1$ can be reduced to a satisfiability task w.r.t. $\mathcal{T}_0$ as follows:

Def		$\mathsf{Update}_0 \wedge G_0 \wedge N_0$
	Update_0	$d{\neq}\mathsf{null} \wedge d{\neq}c \wedge \mathsf{prio}(d){\leq}x \wedge \mathsf{prev}(d){=}\mathsf{null} \rightarrow c_1{=}d$
$\mathsf{next}'(d){=}d_1$		$d{\neq}\mathsf{null} \wedge d{\neq}c \wedge \mathsf{prio}(d){\leq}x \wedge \mathsf{prev}(d){=}\mathsf{null} \rightarrow d_1{=}\mathsf{next}(d)$
$\mathsf{next}'(c){=}c_1$		$d{\neq}\mathsf{null} \wedge d{\neq}c \wedge \mathsf{prio}(d){\leq}x \wedge \mathsf{prev}(d){\neq}\mathsf{null} \rightarrow d_1{=}\mathsf{next}(d)$
		$d{\neq}\mathsf{null} \wedge d{\neq}c \wedge \mathsf{prio}(d){>}x \wedge \mathsf{next}(d){=}\mathsf{null} \rightarrow d_1{=}c \wedge c_1{=}\mathsf{null}$
		$d{\neq}\mathsf{null} \wedge d{\neq}c \wedge \mathsf{prio}(d){>}x \wedge \mathsf{next}(d){\neq}\mathsf{null} \wedge \mathsf{prio}(\mathsf{next}(d)){<}x \rightarrow d_1{=}\mathsf{next}(d)$
		$d{\neq}\mathsf{null} \wedge d{\neq}c \wedge \mathsf{prio}(d){>}x \wedge \mathsf{next}(d){\neq}\mathsf{null} \wedge \mathsf{prio}(\mathsf{next}(d)){\leq}x \rightarrow d_1{=}c$
		$d{\neq}\mathsf{null} \wedge d{\neq}c \wedge \mathsf{prio}(d){>}x \wedge \mathsf{next}(d){\neq}\mathsf{null} \wedge \mathsf{prio}(\mathsf{next}(d)){\leq}x \rightarrow c_1{=}\mathsf{next}(d)$
	G_0	$d \neq \mathsf{null} \wedge \mathsf{next}'(d) \neq \mathsf{null} \wedge \neg\mathsf{prio}(d) \geq \mathsf{priority}(\mathsf{next}'(d))$
	N_0	$d{=}c \rightarrow d_1{=}c_1$

To check the satisfiability of $G' = \mathsf{Update}_0 \wedge G_0 \wedge N_0$ w.r.t. $\mathcal{T}_0$ we use the Ψ-stable locality of the theory defined by the axioms $\mathcal{K} = \{(4), (5), (6)\}$ of doubly linked lists with decreasing priorities in Section 9.4.1 or the instantiation method in [45].

9.5.3 Verification of a program handling arrays

The following example illustrates the extension of the fragment in [11] we consider. Consider a parametric number m of processes. The priorities associated with the processes (non-negative real numbers) are stored in an array p. The states of the process – enabled (1) or disabled (0) – are stored in an array a. At each step only the process with maximal priority is enabled, its priority is set to x and the priorities of the waiting processes are increased by y. This can be expressed with the following set of axioms which we denote by $\mathsf{Update}(a, p, a', p')$:

$$\begin{array}{l}
\forall i(1 \leq i \leq m \wedge \quad (\forall j(1 \leq j \leq m \wedge j{\neq}i \rightarrow p(i){>}p(j))) \rightarrow a'(i) = 1) \\
\forall i(1 \leq i \leq m \wedge \quad (\forall j(1 \leq j \leq m \wedge j{\neq}i \rightarrow p(i){>}p(j))) \rightarrow p'(i){=}x) \\
\forall i(1 \leq i \leq m \wedge \neg(\forall j(1 \leq j \leq m \wedge j{\neq}i \rightarrow p(i){>}p(j))) \rightarrow a'(i){=}0) \\
\forall i(1 \leq i \leq m \wedge \neg(\forall j(1 \leq j \leq m \wedge j{\neq}i \rightarrow p(i){>}p(j))) \rightarrow p'(i){=}p(i){+}y)
\end{array}$$

where x and y are considered to be parameters.

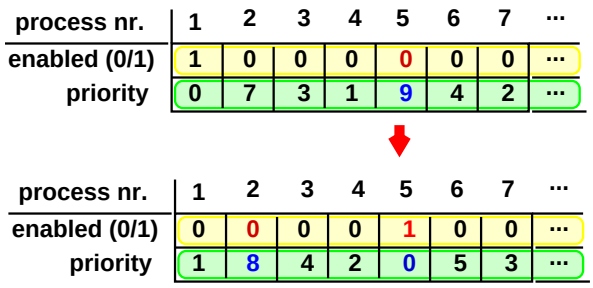


Figure 9: Array update: priority lists

The task is to check the unsatisfiability of the formula $G = (1 \leq c \leq m \wedge 1 \leq d \leq m \wedge c \neq d \wedge p'(c) = p'(d))$ in the extension of the many sorted combination $\mathcal{T}_0$ of $\mathbb{Z}, \mathbb{R}_+, \{0, 1\}$ with the axioms $\forall i, j(1 \leq i \leq m \wedge 1 \leq j \leq m \wedge i \neq j \rightarrow p(i) \neq p(j)) \wedge \mathsf{Update}(a, p, a', p')$.
The extension can be expressed as a chain: $\mathcal{T}_0 \subseteq \mathcal{T}_1 = \mathcal{T}_0 \cup \mathsf{Inj}(p) \subseteq \mathcal{T}_2 = \mathcal{T}_1 \cup \mathsf{Update}(a, p, a', p')$.
By the locality of the second extension (with guarded boundedness axioms) we obtain the following reduction of the task of proving $\mathcal{T}_2 \wedge G \models \bot$ to a satisfiability problem w.r.t. $\mathcal{T}_1$.
We take into account only the instances of $\mathsf{Update}(a, p, a', p')$ which contain ground terms

occurring in G. This means that the axioms containing a' do not need to be considered. After purification and skolemization of the existentially quantified variables we obtain:

Def		$\mathsf{Update}_0 \wedge G_0 \wedge N_0$
$p'(c) = c_1$	Update_0	$1 \leq c \leq m \wedge (1 \leq k_c \leq m \wedge k_c{\neq}c \rightarrow p(c){>}p(k_c)) \rightarrow c_1{=}x$
$p'(d) = d_1$		$1 \leq d \leq m \wedge (1 \leq k_d \leq m \wedge k_d{\neq}d \rightarrow p(d){>}p(k_d)) \rightarrow d_1{=}x$
		$\forall j(1 \leq j \leq m \wedge j{\neq}c \rightarrow p(c){>}p(j)) \vee (1 \leq c \leq m \rightarrow c_1{=}p(c){+}y)$
		$\forall j(1 \leq j \leq m \wedge j{\neq}d \rightarrow p(d){>}p(j)) \vee (1 \leq d \leq m \rightarrow d_1{=}p(d){+}y)$
	G_0	$1 \leq c \leq m \wedge 1 \leq d \leq m \wedge c \neq d \wedge c_1 = d_1$
	N_0	$c = d \rightarrow c_1 = d_1$

We reduced the problem to checking satisfiability of $G_1 = \mathsf{Update}_0 \wedge G_0 \wedge N_0$ (which contains universal quantifiers) w.r.t. $\mathcal{T}_1$. Let $G_1 = G_g \wedge G_\forall$, where G_g is the ground part of G and $G_\forall$ the part of G containing universally quantified variables. We now have to check whether $\mathcal{T}_0 \wedge \mathsf{Inj}(p) \wedge G_\forall \wedge G_g \models \bot$. Note that extensions of injectivity axioms and boundedness are local, and thus $\mathcal{T}_0 \subseteq \mathcal{T}_0 \wedge \mathsf{Inj} \wedge G_\forall$ is a local extension. This makes the following reduction possible:

Def_1		$\mathsf{Inj}_0 \wedge G_{\forall 0} \wedge G_g \wedge N_0'$	
$p(c){=}c_2$	Inj_0	$1{\leq}i{\neq}j{\leq}m \rightarrow p(i){\neq}p(j)$	where i, j are instan-
$p(d){=}d_2$	$G_{\forall 0}$	$(1{\leq}j{\leq}m \wedge j{\neq}c \rightarrow c_2{>}p(j)) \vee (1{\leq}c{\leq}m \rightarrow c_1{=}c_2{+}y)$	tiated with c, d, k_c, k_d
$p(k_c){=}c_3$		$(1{\leq}j{\leq}m \wedge j{\neq}d \rightarrow d_2{>}p(j)) \vee (1 \leq d \leq m \rightarrow d_1{=}d_2{+}y)$	+ purification
$p(k_d){=}d_3$	G_g	$1{\leq}c{\leq}m \wedge (1{\leq}k_c{\leq}m \wedge k_c{\neq}c \rightarrow c_2{>}c_3) \rightarrow c_1{=}x$	
		$1{\leq}d{\leq}m \wedge (1{\leq}k_d{\leq}m \wedge k_d{\neq}d \rightarrow d_2{>}d_3) \rightarrow d_1{=}x$	
		$1{\leq}c{\leq}m \wedge 1{\leq}d{\leq}m \wedge c{\neq}d \wedge c_1{=}d_1$	
		$c{=}d \rightarrow c_1{=}d_1$	
	N_0'	$c{=}d \rightarrow c_2{=}d_2,\ c{=}k_c \rightarrow c_2{=}c_3,\ c{=}k_d \rightarrow c_2{=}d_3,$	
		$d{=}k_c \rightarrow d_2{=}c_3, d{=}k_d \rightarrow d_2{=}d_3, k_c{=}k_d \rightarrow c_3{=}d_3$	

We can use a prover for a combination of integers and reals to determine whether the conjunction of formulae above is satisfiable or symbolic computation packages performing quantifier elimination over the combined theory to derive constraints between x and y which guarantee injectivity after update.

9.6 Databases

In what follows we present some results on hierarchical and modular reasoning in description logic. In particular, we present results which relate locality results for semilattices with monotone oiperators to the PTIME description logic $\mathcal{EL}$ and allow us to give an extension of $\mathcal{EL}$ with numerical domains.

9.6.1 Description logics: generalities

The central notions in description logics are concepts and roles. In any description logic a set N_C of *concept names* and a set N_R of *roles* is assumed to be given. Complex concepts are defined starting with the concept names in N_C, with the help of a set of *concept constructors*. The available constructors determine the expressive power of a description logic. The semantics of description logics is defined in terms of interpretations $\mathcal{I} = (D^\mathcal{I}, \cdot^\mathcal{I})$, where $D^\mathcal{I}$ is a non-empty set, and the function $\cdot^\mathcal{I}$ maps each concept name $C \in N_C$ to a set $C^\mathcal{I} \subseteq D^\mathcal{I}$ and each role name $r \in N_R$ to a binary relation $r^\mathcal{I} \subseteq D^\mathcal{I} \times D^\mathcal{I}$. Table 1 shows the constructor names used in $\mathcal{ALC}$ and their semantics. The extension of $\cdot^\mathcal{I}$ to concept descriptions is inductively defined using the semantics of the constructors.

Terminology. A *terminology* (or TBox, for short) is a finite set consisting of *primitive concept definitions* of the form $C \equiv D$, where C is a concept name and D a concept description; and *general concept inclusions* (GCI) of the form $C \sqsubseteq D$, where C and D are concept descriptions.

Table 1: Constructors and their semantics

Constructor name	Syntax	Semantics
negation	$\neg C$	$D^{\mathcal{I}} \backslash C^{\mathcal{I}}$
conjunction	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
disjunction	$C_1 \sqcup C_2$	$C_1^{\mathcal{I}} \cup C_2^{\mathcal{I}}$
existential restriction	$\exists r.C$	$\{x \mid \exists y((x,y) \in r^{\mathcal{I}} \text{ and } y \in C^{\mathcal{I}})\}$
universal restriction	$\forall r.C$	$\{x \mid \forall y((x,y) \in r^{\mathcal{I}} \implies y \in C^{\mathcal{I}})\}$

Interpretations. An interpretation $\mathcal{I}$ is a model of a TBox $\mathcal{T}$ if it satisfies:

- all concept definitions in $\mathcal{T}$, i.e. $C^{\mathcal{I}}{=}D^{\mathcal{I}}$ for all definitions $C{\equiv}D \in \mathcal{T}$;
- all general concept inclusions in $\mathcal{T}$, i.e. $C^{\mathcal{I}}{\subseteq}D^{\mathcal{I}}$ for every $C{\sqsubseteq}D \in \mathcal{T}$.

Since definitions can be expressed as double inclusions, in what follows we will only refer to TBoxes consisting of general concept inclusions (GCI) only.

Definition 51 *Let $\mathcal{T}$ be a TBox, and C_1, C_2 two concept descriptions. C_1 is subsumed by C_2 w.r.t. $\mathcal{T}$ (for short, $C_1 \sqsubseteq_{\mathcal{T}} C_2$) if and only if $C_1^{\mathcal{I}} \subseteq C_2^{\mathcal{I}}$ for every model $\mathcal{I}$ of $\mathcal{T}$.*

A translation of concept descriptions into terms in a signature naturally associated with the set of constructors can be defined as follows. For every role name r, we introduce unary function symbols, $f_{\exists r}$ and $f_{\forall r}$. The renaming is inductively defined by:

- $\overline{C} = C$ for every concept name C;
- $\overline{\neg C} = \neg\overline{C}$; $\quad \overline{C_1 \sqcap C_2} = \overline{C}_1 \wedge \overline{C}_2$, $\quad \overline{C_1 \sqcup C_2} = \overline{C}_1 \vee \overline{C}_2$;
- $\overline{\exists r.C} = f_{\exists r}(\overline{C})$, $\quad \overline{\forall r.C} = f_{\forall r}(\overline{C})$.

Set theoretical semantics. There exists a one-to-one correspondence between interpretations of description logics, $\mathcal{I} = (D, \cdot^{\mathcal{I}})$ and Boolean algebras of sets with additional operators $(\mathcal{P}(D), \cup, \cap, \neg, \emptyset, D, \{f_{\exists r}, f_{\forall r}\}_{r \in N_R})$, together with valuations $v : N_C \to \mathcal{P}(D)$, where $f_{\exists r}, f_{\forall r}$ are defined, for every $U \subseteq D$, by:

$$\begin{aligned} f_{\exists r}(U) &= \{x \mid \exists y((x,y) \in r^{\mathcal{I}} \text{ and } y \in U)\}; \\ f_{\forall r}(U) &= \{x \mid \forall y((x,y) \in r^{\mathcal{I}} \Rightarrow y \in U)\}. \end{aligned}$$

Let $v : N_C \to \mathcal{P}(D)$ with $v(A) = A^{\mathcal{I}}$ for all $A \in N_C$, and let $\overline{v}$ be the (unique) homomorphic extension of v to terms. Let C be a concept description and $\overline{C}$ be its associated term. Then $C^{\mathcal{I}} = \overline{v}(\overline{C})$ (denoted by $\overline{C}^{\mathcal{I}}$).

Boolean algebras with operators. Let BAO_{N_R} be the class of all Boolean algebras with operators $(B, \vee, \wedge, \neg, 0, 1, \{f_{\exists r}, f_{\forall r}\}_{r \in N_R})$, where

- $f_{\exists r}$ is a join hemimorphism, i.e. $f_{\exists r}(x \vee y) = f_{\exists r}(x) \vee f_{\exists r}(y)$, $f_{\exists r}(0) = 0$;
- $f_{\forall r}$ is a meet hemimorphism, i.e. $f_{\forall r}(x \wedge y){=}f_{\forall r}(x) \wedge f_{\forall r}(y)$, $f_{\forall r}(1){=}1$;
- $f_{\forall r}(x) = \neg f_{\exists r}(\neg x)$ for every $x \in B$.

It is known that the TBox subsumption problem for $\mathcal{ALC}$ can be expressed as uniform word problem for Boolean algebras with suitable operators.

Theorem 52 *If $\mathcal{T}$ is an $\mathcal{ALC}$ TBox consisting of general concept inclusions between concept terms formed from concept names $N_C = \{C_1, \ldots, C_n\}$, and D_1, D_2 are concept descriptions, the following are equivalent:*

(1) $D_1 \sqsubseteq_{\mathcal{T}} D_2$.

(2) $\mathcal{P}(\mathbf{D}) \models \forall C_1 ... C_n \left(\left(\bigwedge_{C \sqsubseteq D \in \mathcal{T}} \overline{C} \leq \overline{D} \right) \rightarrow \overline{D_1} \leq \overline{D_2} \right)$ *for all interpretations* $\mathcal{I} = (D, \cdot^{\mathcal{I}})$, *where* $\mathcal{P}(\mathbf{D}) = (\mathcal{P}(D), \cup, \cap, \neg, \emptyset, D, \{f_{\exists r}, f_{\forall r}\}_{r \in N_R})$.

(3) $\mathsf{BAO}_{N_R} \models \forall C_1 ... C_n \left(\left(\bigwedge_{C \sqsubseteq D \in \mathcal{T}} \overline{C} \leq \overline{D} \right) \rightarrow \overline{D_1} \leq \overline{D_2} \right)$.

9.6.2 Combinations of $\mathcal{ALC}$-ontologies

Let T_1 and T_2 be two $\mathcal{ALC}$-TBoxes with sets of constructors N_R^1, N_R^2 which only share the Boolean constructors. We are interested in possibilities of modularily checking TBox subsumption w.r.t. $T = T_1 \cup T_2$. By Theorem 52, this problem can be expressed as the problem of testing satisfiability of conjunctions of ground literals in the combination of theories of $\mathsf{BAO}_{N_R^1}$ and $\mathsf{BAO}_{N_R^2}$. The results by Ghilardi [29] in Section 7.2 (and especially Example 31) can be used to show that this is possible. Similar results can be obtained for description logics with nominals (i.e. with unrestricted use of individuals) under the additional condition that the logics are *nominal-closed*, i.e. that all the definable nominals are already explicitly named [30].

9.6.3 The description logic $\mathcal{EL}$

By restricting the type of allowed concept constructors less expressive but tractable description logics can be defined. If we only allow $\bot$ and intersection and existential restriction as concept constructors, we obtain the description logic $\mathcal{EL}$ [4], a logic used in terminological reasoning in medicine [60, 61].

In [4] it was shown that subsumption w.r.t. TBoxes in $\mathcal{EL}$ can be reduced in linear time to subsumption w.r.t. *normalized* TBoxes, in which all GCIs have one of the forms: $C \sqsubseteq D, C_1 \sqcap C_2 \sqsubseteq D, C \sqsubseteq \exists r.D, \exists r.C \sqsubseteq D$, where C, C_1, C_2, D are concept names.

The algebraic semantics of $\mathcal{EL}$ is much simpler [55], it is given by the class $\mathsf{SLO}^{\exists}_{N_R}$ of all $\wedge$-semilattices with 0 and with operators $(S, \wedge, 0, \{f_{\exists R}\}_{R \in N_R})$, such that $f_{\exists R}$ is monotone and $f_{\exists R}(0) = 0$. [16] The following results are presented in [59].

Theorem 53 ([59]) *If the only concept constructors are intersection and existential restriction, then for all concept descriptions* D_1, D_2 *and every TBox* $\mathcal{T}$, *with concept names* $N_C = \{C_1, \ldots, C_n\}$ *the following are equivalent:*

(1) $D_1 \sqsubseteq_{\mathcal{C}} D_2$.

(2) $\mathsf{SLO}^{\exists}_{N_R} \models \forall C_1 \ldots C_n \left(\left(\bigwedge_{C \sqsubseteq D \in \mathcal{T}} \overline{C} \leq \overline{D} \right) \rightarrow \overline{D_1} \leq \overline{D_2} \right)$.

We regard $\mathsf{SLO}^{\exists}_{N_R}$ as the extension of the theory SL of semilattices with monotone operators $\{f_{\exists r} \mid r \in N_R\}$. This extension is local.

Theorem 54 ([56]) *Let* G *be a set of ground clauses. The following are equivalent:*

(1) $\mathsf{SL} \cup \mathsf{Mon}(\Omega) \wedge G \models \bot$.

(2) $\mathsf{SL} \cup \mathsf{Mon}(\Omega)[G] \wedge G$ *has no partial model* A *such that its* $\{\wedge\}$*-reduct is a (total) semilattice and the functions in* Ω *are partially defined, their domain of definition is finite and all terms in* G *are defined in* A.

[16] For the sake of simplicity, in what follows we assume that the description logics $\mathcal{EL}$ contain the additional constructors $\bot, \top$, which will be interpreted as 0 and 1. Similar considerations can be used to show that the algebraic semantics for variants of $\mathcal{EL}$ having only $\top$ (or $\bot$) is given by semilattices with 1 (resp. 0).

Let $\mathsf{Mon}(\Omega)[G]_0 \wedge G_0 \wedge \mathsf{Def}$ *be obtained from* $\mathsf{Mon}(\Omega)[G] \wedge G$ *by purification, i.e. by replacing, in a bottom-up manner, all subterms* $f(g)$ *with* $f \in \Omega$*, with newly introduced constants* $c_{f(g)}$ *and adding the definitions* $f(g) = c_t$ *to the set* Def*. The following are equivalent (and equivalent to (1) and (2)):*

(3) $\mathsf{Mon}(\Omega)[G]_0 \wedge G_0 \wedge \mathsf{Def}$ *has no partial model* $(A, \wedge, \{f_A\}_{f\in\Omega})$ *such that* $(A, \wedge)$ *is a semilattice and for all* $f\in\Omega$*,* f_A *is partially defined, its domain of definition is finite and all terms in* Def *are defined in* A*.*

(4) $\mathsf{Mon}(\Omega)[G]_0 \wedge G_0$ *is unsatisfiable in* SL*.*

(Note that in the presence of $\mathsf{Mon}(\Omega)$ *the instances* N_0 *of the congruence axioms for the functions in* Ω *are not necessary*

$$N_0 = \{g{\approx}g' \rightarrow c_{f(g)}{\approx}c_{f(g')} \mid f(g){\approx}c_{f(g)}, f(g'){\approx}c_{f(g')} \in \mathsf{Def}\}).$$

This equivalence allows us to hierarchically reduce, in polynomial time, proof tasks in $\mathsf{SL} \cup \mathsf{Mon}(\Omega)$ to proof tasks in SL (cf. e.g. [56]) which can then be solved in polynomial time.

Example 36 *We illustrate the method on an example first considered in [4]. Consider the* $\mathcal{EL}$ *TBox* $\mathcal{T}$ *consisting of the following definitions:*

$$\begin{array}{lcl} A_1 & = & P_1 \sqcap A_2 \sqcap \exists r_1.\exists r_2.A_3 \\ A_2 & = & P_2 \sqcap A_3 \sqcap \exists r_2.\exists r_1.A_1 \\ A_3 & = & P_3 \sqcap A_2 \sqcap \exists r_1.(P_1 \sqcap P_2) \end{array}$$

We want to prove that $P_3 \sqcap A_2 \sqcap \exists r_1.(A_1 \sqcap A_2) \sqsubseteq_{\mathcal{T}} A_3$*. We translate this subsumption problem to the following satisfiability problem:*

$$\begin{array}{lcl} \mathsf{SL} \cup \mathsf{Mon}(f_1, f_2) & \cup & \{\, a_1 = (p_1 \wedge a_2 \wedge f_1(f_2(a_3))), \\ & & \quad a_2 = (p_2 \wedge a_3 \wedge f_2(f_1(a_1))), \\ & & \quad a_3 = (p_3 \wedge a_2 \wedge f_1(p_1 \wedge p_2)), \\ & & \quad \neg(p_3 \wedge a_2 \wedge f_1(a_1 \wedge a_2) \leq a_3)\} \models \bot\ . \end{array}$$

We proceed as follows: We flatten and purify the set G *of ground clauses by introducing new names for the terms starting with the function symbols* f_1 *or* f_2*. Let* Def *be the corresponding set of definitions. We then take into account only those instances of the monotonicity and congruence axioms for* f_1 *and* f_2 *which correspond to the instances in* Def*, and purify them as well, by replacing the terms themselves with the constants which denote them. We obtain the following separated set of formulae:*

Def	G_0	$(\mathsf{Mon}(f_1, f_2)[G])_0$
$f_2(a_3) \approx c_1$	$(a_1 \approx p_1 \wedge a_2 \wedge c_2)$	$a_1Rc_1 \rightarrow c_3Rc_2, \quad R \in \{\leq, \geq\}$
$f_1(c_1) \approx c_2$	$(a_2 \approx p_2 \wedge a_3 \wedge c_4)$	$a_3Rc_3 \rightarrow c_1Rc_4, \quad R \in \{\leq, \geq\}$
$f_1(a_1) \approx c_3$	$(a_3 \approx p_3 \wedge a_2 \wedge d_1)$	$a_1Re_1 \rightarrow c_3Rd_1, \quad R \in \{\leq, \geq\}$
$f_2(c_3) \approx c_4$	$(p_3 \wedge a_2 \wedge d_2 \not\leq a_3)$	$a_1Re_2 \rightarrow c_3Rd_2, \quad R \in \{\leq, \geq\}$
$f_1(e_1) \approx d_1$	$p_1 \wedge p_2 \approx e_1$	$c_1Re_1 \rightarrow c_2Rd_1, \quad R \in \{\leq, \geq\}$
$f_1(e_2) \approx d_2$	$a_1 \wedge a_2 \approx e_2$	$c_1Re_2 \rightarrow c_2Rd_2, \quad R \in \{\leq, \geq\}$
		$e_1Re_2 \rightarrow d_1Rd_2, \quad R \in \{\leq, \geq\}$

The subsumption is true iff $G_0 \wedge (\mathsf{Mon}(f_1, f_2)[G])_0$ *is unsatisfiable in the theory of semilattices. We can see this as follows: note that* $a_1 \wedge a_2 \leq p_1 \wedge p_2$*, i.e.* $e_2 \leq e_1$*. Then (using an instance of monotonicity)* $d_2 \leq d_1$*, so* $p_3 \wedge a_2 \wedge d_2 \leq p_3 \wedge a_2 \wedge d_1 = a_3$*.*

This can also be checked automatically in PTIME *either by using the fact that there exists a local presentation of* SL *or using the fact that* $\mathsf{SL} = ISP(S_2)$ *(i.e. every semilattice is isomorphic with a sublattice of a power of* S_2*), where* S_2 *is the semilattice with two elements, hence* SL *and* S_2 *satisfy the same Horn clauses. Since the theory of semilattices is convex, satisfiability of ground clauses w.r.t.* SL *can be reduced to SAT solving.*

Table 2: Constructors for $\mathcal{EL}$ with n-ary roles and their semantics

Constructor	Syntax	Semantics
conjunction	$C_1 \sqcap C_2$	$C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}}$
existential	$\exists R.(C_1, \ldots C_n)$	$\{x \mid \exists y_1, \ldots, y_n\, (x, y_1, \ldots, y_n) \in R^{\mathcal{I}}$ and $y_i \in C_i^{\mathcal{I}}\}$

9.6.4 $\mathcal{EL}$ with numerical domains

The results described in Section 9.6.3 can easily be generalized to semilattices with n-ary monotone functions. This allows us to define natural generalizations of $\mathcal{EL}$. We start by presenting a generalization of $\mathcal{EL}$ in which n-ary roles are allowed.

The semantics is defined in terms of interpretations $\mathcal{I} = (D^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $D^{\mathcal{I}}$ is a non-empty set, concepts are interpreted as usual, and each n-ary role $R \in N_R$ is interpreted as an n-ary relation $R^{\mathcal{I}} \subseteq (D^{\mathcal{I}})^n$ (cf. Table 2). A further extension is obtained by allowing for certain concrete sorts – having the same support in all interpretations; or additionally assuming that there exist specific concrete concepts which have a fixed semantics (or additional fixed properties) in all interpretations. The extensions we consider are different from the extensions with concrete domains and those with n-ary quantifiers studied in the description logic literature (cf. e.g. [6, 5].

Example 37 *Consider a description logic having a usual (*$\mathsf{concept}$*) sort and a 'concrete' sort* num *with fixed domain* $\mathbb{N}$*. We may be interested in general concrete concepts of sort* num *(interpreted as subsets of* $\mathbb{R}$*) or in special concepts of sort* num *such as* $\uparrow n$*,* $\downarrow n$*, or* $[n, m]$ *for* $m, n \in \mathbb{R}$*. For any interpretation* $\mathcal{I}$*,* $\uparrow n^{\mathcal{I}} = \{x \in \mathbb{R} \mid x \geq n\}$*,* $\downarrow n^{\mathcal{I}} = \{x \in \mathbb{R} \mid x \leq n\}$*, and* $[n, m]^{\mathcal{I}} = \{x \in \mathbb{R} \mid n \leq x \leq m\}$*. We will denote the arities of roles using a many-sorted framework. Let* $(D, \mathbb{R}, \cdot^{\mathcal{I}})$ *be an interpretation with two sorts* $\mathsf{concept}$ *and* num*. A role with arity* $(s_1, \ldots, s_n)$ *is interpreted as a subset of* $D_{s_1} \times \cdots \times D_{s_n}$*, where* $D_{\mathsf{concept}} = D$ *and* $D_{\mathsf{num}} = \mathbb{R}$*.*

1. *Let* price *be a binary role or arity* $(\mathsf{concept}, \mathsf{num})$*, which associates with every element of sort* $\mathsf{concept}$ *its possible prices. The concept*

$$\exists \mathsf{price}.{\uparrow}n = \{x \mid \exists k \geq n : \mathsf{price}(x, k)\}$$

represents the class of all individuals with some price *greater than* n*.*

2. *Let* $\mathsf{has\text{-}weight\text{-}price}$ *be a role of arity* $(\mathsf{concept}, \mathsf{num}, \mathsf{num})$*. The concept*

$$\exists\ \mathsf{has\text{-}weight\text{-}price}.(\uparrow \mathsf{y}, \downarrow \mathsf{p}) = \{x \mid \exists y' \geq \mathsf{y}, \exists p' \leq \mathsf{p}\ \textit{and}\ \mathsf{has\text{-}weight\text{-}price}(x, y', p')\}$$

denotes the family of individuals for which a weight above y *and a price below* p *exist.*

The example below can be generalized by allowing a set of concrete sorts. We discuss the algebraic semantics of this type of extensions of $\mathcal{EL}$.

Let $\mathsf{SLO}^{\exists}_{N_R,S}$ denote the class of all structures $(S, \mathcal{P}(A_1), \ldots, \mathcal{P}(A_n), \{f_{\exists r} \mid r \in N_R\})$, where S is a bounded semilattice[17], $A_1, \ldots, A_n$ are concrete domains, and $\{f_{\exists r} \mid r \in N_R\}$ are n-ary monotone operators. We may allow constants of concrete sort, interpreted as sets in $\mathcal{P}(A_i)$.

Theorem 55 ([59]) *If the only concept constructors are intersection and existential restriction, then for all concept descriptions* D_1, D_2*, and every TBox* $\mathcal{T}$ *consisting of general concept inclusions GCI the following are equivalent:*

[17] Again, we assume that the variant of $\mathcal{EL}$ we consider has $\bot$ and $\top$. The other situations can be treated analogoulsy.

(1) $D_1 \sqsubseteq_{\mathcal{T}} D_2$.

(2) $\mathsf{SLO}^{\exists}_{N_R,S} \models \forall C_1,\ldots,C_n \left(\left(\bigwedge_{C\sqsubseteq D\in\mathcal{T}} \overline{C} \leq \overline{D}\right) \rightarrow \overline{D_1} \leq \overline{D_2}\right)$.

Proof: Analogous to the proof of Theorem 53. □

Let SL_S be the class of all structures $\mathcal{A} = (A, \mathcal{P}(A_1),\ldots,\mathcal{P}(A_n))$, with signature $\Sigma = (S, \{\wedge, 0, 1\}\cup\Omega, \mathsf{Pred})$ with $S=\{\mathsf{concept}, \mathsf{s}_1,\ldots,\mathsf{s}_n\}$, $\mathsf{Pred}=\{\leq\}\cup\{\subseteq_i \mid 1 \leq i \leq n\}$, where $A \in \mathsf{SL}$, the support of sort concept of $\mathcal{A}$ is A, and for all i the support sort s_i of $\mathcal{A}$ is $\mathcal{P}(A_i)$.

Theorem 56 ([56]) *Every structure* $(A, \mathcal{P}(A_1),\ldots,\mathcal{P}(A_n), \{f_A\}_{f\in\Omega})$, *where*

(i) $(A, \mathcal{P}(A_1),\ldots,\mathcal{P}(A_n)) \in \mathsf{SL}_S$, *and*

(ii) for every $f\in\Omega$ *of arity* $s_1\ldots s_n\rightarrow s$, f_A *is a partial function from* $\prod_{i=1}^n U_{s_i}$ *to* U_s *with a finite definition domain on which it is monotone,*

weakly embeds into a total model of $\mathsf{SL}_S\cup\mathsf{Mon}(\Omega)$.

Corollary 57 ([59]) *Let* $G = \bigwedge_{i=1}^n s_i(\overline{c})\leq s'_i(\overline{c}) \wedge s(\overline{c})\not\leq s'(\overline{c})$ *be a set of ground unit clauses in the extension* Σ^c *of* Σ *with new constants* Ω_c. *The following are equivalent:*

(1) $\mathsf{SL}_S \cup \mathsf{Mon}(\Omega) \wedge G \models \bot$.

(2) $\mathsf{SL}_S \cup \mathsf{Mon}(\Omega)[G] \wedge G$ *has no partial model with a total* $\{\wedge_{SL}, 0, 1\}$*-reduct in which all terms in* G *are defined.*

Let $\bigcup_{i=0}^n \mathsf{Mon}(\Omega)[G]_i \wedge G_i \wedge \mathsf{Def}$ *be obtained from* $\mathsf{Mon}(\Omega)[G] \wedge G$ *by purification, i.e. by replacing, in a bottom-up manner, all subterms* $f(g)$ *of sort* s *with* $f \in \Omega$, *with newly introduced constants* $c_{f(g)}$ *of sort* s *and adding the definitions* $f(g) = c_t$ *to the set* Def. *We thus separate* $\mathsf{Mon}(\Omega)[G] \wedge G$ *into a conjunction of constraints* $\Gamma_i = \mathsf{Mon}(\Omega)[G]_i \wedge G_i$, *where* Γ_0 *is a constraint of sort* semilattice *and for* $1 \leq i \leq n$, Γ_i *is a set of constraints over terms of sort* i *(*i *being the concrete sort with fixed support* $\mathcal{P}(A_i)$*). Then the following are equivalent (and are also equivalent to (1) and (2)):*

(3) $\bigcup_{i=0}^n \mathsf{Mon}(\Omega)[G]_i \wedge G_i \wedge \mathsf{Def}$ *has no partial model with a total* $\{\wedge_{\mathsf{SL}}, 0, 1\}$*-reduct in which all terms in* Def *are defined.*

(4) $\bigcup_{i=0}^n \mathsf{Mon}(\Omega)[G]_i \wedge G_i$ *is unsatisfiable in the many-sorted disjoint combination of* SL *and the concrete theories of* $\mathcal{P}(A_i)$, $1 \leq i \leq n$.

The complexity of the uniform word problem of $\mathsf{SL}_S \cup \mathsf{Mon}(\Omega)$ depends on the complexity of the problem of testing the satisfiability — in the many-sorted disjoint combination of SL with the concrete theories of $\mathcal{P}(A_i)$, $1 \leq i \leq n$ — of sets of clauses $C_{\mathsf{concept}} \cup \bigcup_{i=1}^n C_i \cup \mathsf{Mon}$, where C_{concept} and C_i are unit clauses of sort concept resp. s_i, and Mon consists of possibly mixed ground Horn clauses.

Specific extensions of the logic $\mathcal{EL}$ can be obtained by imposing additional restrictions on the interpretation of the "concrete"-type concepts within $\mathcal{P}(A_i)$. (For instance, we can require that numerical concepts are always interpreted as intervals, as in Example 37.)

Theorem 58 ([59]) *Consider the following extensions of* $\mathcal{EL}$ *with* n*-ary roles:*

(1) The one-sorted extension of $\mathcal{EL}$ *with* n*-ary roles.*

(2) The extension of $\mathcal{EL}$ *with two sorts,* concept *and* num, *where the semantics of classical concepts is the usual one, and the concepts of sort* num *are interpreted as elements in the ORD-Horn, convex fragment of Allen's interval algebra [44], where any CBox can contain many-sorted GCI's over concepts, as well as constraints over the numerical data expressible in the ORD-Horn fragment.*

In both cases, CBox subsumption is decidable in PTIME.

Example 38 *Consider the special case described in Example 37. Assume that the concepts of sort* num *used in any TBox are of the form* $\uparrow n, \downarrow m$ *and* $[n, m]$*. Consider the TBox* $\mathcal{T}$ *consisting of the following GCIs:*

$$\begin{array}{rcl}
\exists \mathsf{price}(\downarrow n_1) & \sqsubseteq & \mathsf{affordable}, \\
\exists \mathsf{weight}(\uparrow m_1) \sqcap \mathsf{car} & \sqsubseteq & \mathsf{truck}, \\
\mathsf{has\text{-}weight\text{-}price}(\uparrow m, \downarrow n) & \sqsubseteq & \exists \mathsf{price}(\downarrow n) \sqcap \exists \mathsf{weight}(\uparrow m), \\
C & \sqsubseteq & \exists\ \mathsf{has\text{-}weight\text{-}price}(\uparrow m, \downarrow n) \\
\downarrow n & \sqsubseteq & \downarrow n_1, \\
\uparrow m & \sqsubseteq & \uparrow m_1, \\
C & \sqsubseteq & \mathsf{car}.
\end{array}$$

In order to prove that $C \sqsubseteq_{\mathcal{T}} \mathsf{affordable} \sqcap \mathsf{truck}$ *we proceed as follows. We refute* $\bigwedge_{D \sqsubseteq D' \in \mathcal{T}} \overline{D} \leq \overline{D}' \wedge \overline{C} \not\leq \mathsf{affordable} \wedge \mathsf{truck}$*. We purify the problem introducing definitions for the terms starting with existential restrictions, and express the interval constraints using constraints over* $\mathbb{Q}$ *and obtain the following set of constraints:*

Def	C_{num}	C_{concept}	Mon
$f_{\mathsf{price}}(\downarrow n_1) \approx c_1$	$n \leq n_1$	$c_1 \leq \mathsf{affordable}$	$n_1 \leq n \rightarrow c_1 \leq c$
$f_{\mathsf{price}}(\downarrow n) \approx c$	$m \geq m_1$	$d_1 \wedge \mathsf{car} \leq \mathsf{truck}$	$n_1 \geq n \rightarrow c_1 \geq c$
$f_{\mathsf{weight}}(\uparrow m_1) \approx d_1$		$e \leq c \wedge d$	$m_1 \geq m \rightarrow d_1 \leq d$
$f_{\mathsf{weight}}(\uparrow m) \approx d$		$C \leq \mathsf{car}$	$m_1 \leq m \rightarrow d_1 \geq d$
$f_{h\text{-}w\text{-}p}(\uparrow m, \downarrow n) \approx e$		$C \leq e$	
		$C \not\leq \mathsf{affordable} \wedge \mathsf{truck}$	

The task of proving $C \sqsubseteq_{\mathcal{T}} \mathsf{affordable} \sqcap \mathsf{truck}$ *can therefore be reduced to checking if* $C_{\mathsf{num}} \wedge C_{\mathsf{concept}} \wedge \mathsf{Mon}$ *is satisfiable w.r.t. the combination of* SL *(sort* concept*) with* $LI(\mathbb{Q})$ *(sort* num*). For this, we note that* C_{num} *entails the premises of the first, second, and fourth monotonicity rules. Thus, we can add* $c \leq c_1$ *and* $d \leq d_1$ *to* C_{concept}*. Thus, we deduce that* $C \leq e \wedge \mathsf{car} \leq (c \wedge d) \wedge \mathsf{car} \leq c_1 \wedge (d_1 \wedge \mathsf{car}) \leq \mathsf{affordable} \wedge \mathsf{truck}$*, which contradicts the last clause in* C_{concept}*.*

A similar procedure can be used in general for testing (in PTIME) the satisfiability of mixed constraints in the many-sorted combination of SL of semilattices with concrete domains of sort num, assuming that all concepts of sort num are interpreted as intervals and the constraints C_{num} are expressible in a PTIME, convex fragment of Allen's interval algebra.

9.7 Multi-agent systems

As mentioned at the beginning, the way of reasoning about knowledge of an agent may be represented by a specific type of modal logic. In order to model and reason about the knowledge of a *set* of agents it is important to be able to prove theorems in combinations of modal logics. Reasoning in many modal logics can be reduced to deciding uniform word problems in corresponding classes of Boolean algebras with operators, the reasoning problem in the combinations of logics can be reduced to the problem of modular reasoning in combinations of Boolean algebras with operators. In such cases, the model theoretical results on combinations of theories by Ghilardi presented in Section 7.2, and their application to modular reasoning in combinations of Boolean algebras with operators in Theorem 31 can be used also in this case for giving modular methods for checking validity of statements in combinations of modal logics associated with the individual agents (cf. also [30]).

10 Conclusions

We presented an in-depth perspective on recent advances in the field of reasoning in complex logical theories. Our presentation focused on possibilities of efficient, sound and complete reasoning in standard theories, as well as in combinations of theories and theory extensions. We illustrated both the relevance and the applicability of results on several case studies in mathematics, verification, databases, and multi-agent systems.

Acknowledgement: This work was partly supported by the German Research Council (DFG) as part of the Transregional Collaborative Research Center "Automatic Verification and Analysis of Complex Systems" (SFB/TR 14 AVACS). See `www.avacs.org` for more information.

References

[1] A. Armando, M.P. Bonacina, S. Ranise, and St. Schulz. On a rewriting approach to satisfiability procedures: extension, combination of theories and an experimental appraisal. In *Proceedings of the 5th International Workshop Frontiers of Combining Systems (FroCos'05)*, LNCS 3717, pages 65–80. Springer Verlag, 2005.

[2] A. Armando, S. Ranise, and M. Rusinowitch. A rewriting approach to satisfiability procedures. *Information and Computation*, 183(2):140–164, 2003.

[3] F. Baader. Restricted role-value-maps in a description logic with existential restrictions and terminological cycles. In *Proc. of the 2003 International Workshop on Description Logics* (DL2003), CEUR-WS, 2003.

[4] F. Baader. Terminological cycles in a description logic with existential restrictions. In: G. Gottlob and T. Walsh, editors, *Proc. of the 18th International Joint Conference in Artificial Intelligence*, pages 325–330, Morgan Kaufmann, 2003.

[5] F. Baader, C. Lutz, E. Karabaev, and M. Theien. A new n-ary existential quantifier in description logics. In *Proc. 28th Annual German Conference on Artificial Intelligence* (KI 2005), *LNAI 3698*, pages 18-033, Springer, 2005.

[6] F. Baader, S. Brandt, and C. Lutz. Pushing the EL Envelope. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence* IJCAI-05, Morgan-Kaufmann Publishers, 2005.

[7] F. Baader and C. Tinelli. Deciding the word problem in the union of equational theories sharing constructors. *Information and Computation*, 178(2):346–390, 2002.

[8] D. Basin and H. Ganzinger. Automated complexity analysis based on ordered resolution. *Journal of the ACM*, 48(1):70–109, 2001.

[9] D.A. Basin and H. Ganzinger. Complexity analysis based on ordered resolution. In *Proc. 11th IEEE Symposium on Logic in Computer Science (LICS'96)*, pages 456–465. IEEE Computer Society Press, 1996.

[10] M.P. Bonacina, S. Ghilardi, E. Nicolini, S. Ranise, D. Zucchelli. Decidability and Undecidability Results for Nelson-Oppen and Rewrite-Based Decision Procedures. In *Proceedings of IJCAR 2006*, LNAI 4130, pages 513-527, Springer 2006.

[11] A.R. Bradley, Z. Manna, and H.B. Sipma. What's decidable about arrays? In E.A. Emerson and K.S. Namjoshi, editors, *Verification, Model-Checking, and Abstract-Interpretation, 7th Int. Conf. (VMCAI 2006)*, LNCS 3855, pages 427–442. Springer, 2006.

[12] P. Burmeister. *A Model Theoretic Oriented Approach to Partial Algebras: Introduction to Theory and Application of Partial Algebras, Part I*, volume 31 of *Mathematical Research*. Akademie-Verlag, Berlin, 1986.

[13] S. Burris. Polynomial time uniform word problems. *Mathematical Logic Quarterly*, 41:173–182, 1995.

[14] D. Cantone, G. Cincotti, and G. Gallo. Decision algorithms for fragments of real analysis. I. Continuous functions with strict convexity and concavity predicates. *Journal of Symbolic Computation*, 41:763–789, 2006.

[15] A. Chandra and D. Harel. Horn clause queries and generalizations. *Journal of Logic Programming*, 1:1–15, 1985.

[16] S.A. Cook. An observation of time-storage trade-off. *Journal of Computer and System Sciences*, 9:308–316, 1974.

[17] W. Craig. Linear reasoning. A new form of the Herbrand-Gentzen theorem. *J. Symb. Log.*, 22(3):250–268, 1957.

[18] A. Dolzmann and T. Sturm. Redlog: Computer algebra meets computer logic. *ACM SIGSAM Bulletin*, 31(2):2–9, 1997.

[19] W.F. Dowling and J.H. Gallier. Linear-time algorithms for testing the satisfiability of propositional Horn formulae. *J. Logic Programming*, 1(3):267–284, 1984.

[20] T. Evans. The word problem for abstract algebras. *J. London Math. Soc.*, 26:64–71, 1951.

[21] Faber, J.: Verifying real-time aspects of the European Train Control System. In *Proceedings of the 17th Nordic Workshop on Programming Theory*, University of Copenhagen, Denmark (2005) 67–70.

[22] J. Faber, S. Jacobs, and V. Sofronie-Stokkermans. Verifying CSP-OZ-DC specifications with complex data types and timing parameters. In *Integrated Formal Methods (IFM 2007), LNCS 4591*, pages 233–252. Springer, 2007.

[23] J.-B.J. Fourier. Reported in *Analyse des travaux de l'Academie Royale de Sciences, pendant l'année 1824, Partie Mathematique*.

[24] H. Friedman and A. Serres. Decidability in elementary analysis, I. *Adv. in Mathematics*, 76(1):94–115, 1989.

[25] H. Friedman and A. Serres. Decidability in elementary analysis, II. *Adv. in Mathematics*, 70(2):1–17, 1990.

[26] H. Ganzinger. Relating semantic and proof-theoretic concepts for polynomial time decidability of uniform word problems. In *Proc. 16th IEEE Symposium on Logic in Computer Science (LICS'01)*, pages 81–92. IEEE Computer Society Press, 2001.

[27] H. Ganzinger, G. Hagen, R. Nieuwenhuis, A. Oliveras, C. Tinelli. DPLL(T): Fast Decision Procedures. In *Proc. CAV 2004*, LNCS 3114, pages 175-188. Springer, 2004.

[28] H. Ganzinger, V. Sofronie-Stokkermans, and U. Waldmann. Modular proof systems for partial functions with Evans equality. *Information and Computation*, 204(10):1453–1492, 2006.

[29] S. Ghilardi. Model theoretic methods in combined constraint satisfiability. *Journal of Automated Reasoning*, 33(3-4):221–249, 2004.

[30] S. Ghilardi and L. Santocanale. Algebraic and model theoretic techniques for fusion decidability in modal logics. In *Proceedings of LPAR 2003*, LNCS 2850, pages 152-166, Springer 2003.

[31] S Ghilardi, E. Nicolini, S. Ranise, and D. Zucchelli. Combination methods for satisfiability and model-checking of infinite-state systems. In *Proceedings of the International Conference on Automated Deduction (CADE 21)*, 2007.

[32] R. Givan and D. McAllester. New results on local inference relations. In *Principles of Knowledge Representation and reasoning: Proceedings of the Third International Conference (KR'92)*, pages 403–412. Morgan Kaufmann Press, 1992.

[33] R. Givan and D.A. McAllester. Polynomial-time computation via local inference relations. *ACM Transactions on Computational Logic*, 3(4):521–541, 2002.

[34] J. Hoenicke and E.-R. Olderog. CSP-OZ-DC: A combination of specification techniques for processes, data and time. *Nordic Journal of Computing*, 9(4):301–334, 2002. Appeared March 2003.

[35] C. A. R. Hoare. *Communicating Sequential Processes.* Prentice-Hall, 1985.

[36] J. Hoenicke. *Combination of Processes, Data, and Time.* PhD thesis, University of Oldenburg, Germany, 2006.

[37] C. Ihlemann, S. Jacobs, and V. Sofronie-Stokkermans. On local reasoning in verification. In *Proc. TACAS 2008, LNCS 4963*, pages 265–281, Springer 2008.

[38] N. Immerman. Relational queries computable in polynomial time. *Information and Control*, 68:86–104, 1986.

[39] S. Jacobs and V. Sofronie-Stokkermans. Applications of hierarchical reasoning in the verification of complex systems. *Electronic Notes in Theoretical Computer Science*, 174(8):39–54, 2007.

[40] P. Kolaitis and M. Vardi. On the Expressive Power of Datalog: Tools and a Case Study. In *Proc. 9th ACM Symp. on Pronciples of Database Systems*, pages 61–71, 1990.

[41] J.-L. Lassez and M.J. Maher. On Fourier's Algorithm for Linear Arithmetic Constraints. *Journal of Automated Reasoning* 9:373-379, 1992.

[42] D. McAllester. Automatic recognition of tractability in inference relations. *Journal of the ACM*, 40(2):284–303, 1993.

[43] K.L. McMillan. An interpolating theorem prover. *Theoretical Computer Science*, 345(1):101–121, 2005.

[44] B.Nebel and H.-J.Bürckert. Reasoning about temporal relations: A maximal tractable subclass of Allen's interval algebra. *Journal of the ACM*, 42 (1): 43–66, 1995.

[45] S. McPeak and G.C. Necula. Data structure specifications via local equality axioms. In K. Etessami and S.K. Rajamani, editors, *Computer Aided Verification, 17th International Conference, CAV 2005*, LNCS 3576, pages 476–490, 2005.

[46] G. Nelson and D.C. Oppen. Simplification by cooperating decision procedures. *ACM Transactions on Programming Languages and Systems*, 1979.

[47] C. Papadimitriou. A note on the expressive power of Prolog. *Bulletin of the EATCS*, 26:21–23, 1985.

[48] M. Presburger. ber die Vollstndigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. in *Comptes Rendus du I congrès de Mathématiciens des Pays Slaves*. Warszawa: 92-101, 1929.

[49] A. Rybalchenko and V. Sofronie-Stokkermans. Constraint solving for interpolation. In B. Cook and A. Podelski, editors, *8th International Conference on Verification, Model Checking and Abstract Interpretation (VMCAI 2007)*, LNCS 4349, pages 346-362. Springer, 2007.

[50] T. Skolem. Logisch-kombinatorische Untersuchungen über die Erfüllbarkeit und Beweisbarkeit mathematischen Sätze nebst einem Theoreme über dichte Mengen. *Skrifter utgit av Videnskabsselskapet i Kristiania, I. Matematisk-naturvidenskabelig klasse, 4*, pages 1–36, 1920.

[51] G. Smith. *The Object Z Specification Language*. Kluwer Academic, 2000.

[52] V. Sofronie-Stokkermans. Hierarchic reasoning in local theory extensions. In R. Nieuwenhuis, editor, *20th International Conference on Automated Deduction (CADE-20), LNAI 3632*, pages 219–234, Tallinn, Estonia, 2005. Springer.

[53] V. Sofronie-Stokkermans. Interpolation in local theory extensions. In U. Furbach and N. Shankar, editors, *Proceedings of the International Joint Conference on Automated Reasoning (IJCAR 2006)*, volume 4130 of *Lecture Notes in Artificial Intelligence*, pages 235–250. Springer, 2006.

[54] V. Sofronie-Stokkermans. Hierarchical and modular reasoning in complex theories: The case of local theory extensions. In *Proceedings of FroCos 2007*, LNCS 4721, pages 47–71, Springer 2007.

[55] V. Sofronie-Stokkermans. Automated theorem proving by resolution in non-classical logics. *Annals of Mathematics and Artificial Intelligence* (Special issue "Knowledge Discovery and Discrete Mathematics: Dedicated to the Memory of Peter L. Hammer"), 49 (1-4): 221-252, 2007.

[56] V. Sofronie-Stokkermans and C. Ihlemann. Automated reasoning in some local extensions of ordered structures. In *Proceedings of ISMVL-2007*. IEEE Computer Society, 2007. `http://dx.doi.org/10.1109/ISMVL.2007.10`.

[57] V. Sofronie-Stokkermans, C. Ihlemann, and S. Jacobs. Local Theory Extensions, Hierarchical Reasoning and Applications to Verification. In *Deduction and Decision Procedures*, Dagstuhl Seminar Proceedings 07401, Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany, 2007.

[58] V. Sofronie-Stokkermans. Efficient Hierarchical Reasoning about Functions over Numerical Domains. In *Proceedings of KI 2008*. LNAI 5243, pages 135-143, Springer 2008.

[59] V. Sofronie-Stokkermans. Locality and subsumption testing in $\mathcal{EL}$ and some of its extensions. In *Proceedings of AiML 2008*, To appear. 2008.

[60] K.A. Spackman, K.E. Campbell, R.A. Cote. SNOMED RT: A reference terminology for health care. *Journal of the Americal Medical Informatics Association*, pages 640-644, 1997. Fall Symposium Supplement.

[61] K.A. Spackman. Normal forms for description logic expression of clinical concepts in SNOMED RT. *Journal of the Americal Medical Informatics Association*, pages 627–631, 2001. Symposium Supplement.

[62] A. Tarski. A Decision Method for Elementary Algebra and Geometry. Manuscript. Santa Monica, CA: RAND Corp., 1948. Republished as "A Decision Method for Elementary Algebra and Geometry", 2nd ed. Berkeley, CA: University of California Press, 1951.

[63] C. Tinelli and M. Harandi. A New Correctness Proof of the Nelson-Oppen Combination Procedure. In Proc. FroCos 1996, Kluwer Academic Publishers, Applied Logic Series, Vol. 3, pages 103-119, 1996.

[64] C. Tinelli and C. Ringeissen. Unions of non-disjoint theories and combinations of satisfiability procedures. *Theoretical Computer Science*, 290(1):291–353, 2003.

[65] C. Tinelli and C. Zarba. Combining nonstably infinite theories. *Journal of Automated Reasoning*, 34(3):209–238, 2005.

[66] J. Ullman. *Principles of Database and Knowledge-Base Systems*. Computer Science Press, 1988.

[67] J. Ullman. Bottom-up beats top-down for datalog. In *Proceedings of the 8th ACM SIGACT-SIGMOD-SIGART Symposium on the Principles of Database Systems*, pages 140–149, 1989.

[68] M. Vardi. The complexity of relational query languages. In *Proc. 14th. ACM Symp. on Theory of Computing*, pages 137–146, 1982.

[69] R. Loos and V. Weispfenning. Applying Linear Quantifier Elimination. *The Computer Journal*, 36(5):450-461, 1993.

[70] C. Zhou and M. R. Hansen. *Duration Calculus*. Springer, 2004.

A Appendix A. Some notions in model theory

We give some definitions in model theory needed for the results presented in Section 7.2.

Definition 59 (Embedding) *An* embedding *between two Σ-structures $\mathcal{A}$ and $\mathcal{B}$ with supports A and B is a mapping $i : A \to B$ such that for all Σ^A-atoms Φ, $\mathcal{A} \models \Phi$ iff $\mathcal{B} \models \Phi$ where we view $\mathcal{B}$ as a Σ^A-structure by interpreting every constant $a \in \mathcal{A}$ as $i(a)$ in $\mathcal{B}$.*

Definition 60 (Diagram) *The diagram $\Delta(\mathcal{A})$ of a Σ-structure $\mathcal{A}$ is the set of all ground Σ^A-sentences that are true in $\mathcal{A}$.*

Definition 61 (Elementary diagram) *The elementary diagram $\Delta^e(\mathcal{A})$ of a Σ-structure $\mathcal{A}$ is the set of all first order Σ^A-sentences that are true in $\mathcal{A}$.*

Definition 62 (Model Completion) *Let $\mathcal{T}$ and $\mathcal{T}^*$ be Σ-theories such that $\mathcal{T} \subseteq \mathcal{T}^*$. Then $\mathcal{T}^*$ is a* model completion *of $\mathcal{T}$ iff:*

(1) every model of $\mathcal{T}$ can be embedded into a model of $\mathcal{T}^*$;
(2) for every model $\mathcal{M}$ of $\mathcal{T}$ we have that $\mathcal{T}^* \cup \Delta(\mathcal{M})$ is a complete theory (in Σ^M).

Theorem 63 *For Σ-theories $\mathcal{T}$ and $\mathcal{T}^*$ with $\mathcal{T} \subseteq \mathcal{T}^*$, $\mathcal{T}^*$ is a model completion of $\mathcal{T}$ if:*

(a) every model of $\mathcal{T}$ can be embedded into a model of $\mathcal{T}^*$;
(b) $\mathcal{T}^*$ eliminates quantifiers.

If $\mathcal{T}$ is a universal theory conditions (a) and (b) above are equivalent with conditions (1),(2) in Definition 62.

B Appendix B. Partial algebras

Let $\Sigma = (S, \Omega, \mathsf{Pred})$ be an S-sorted signature where Ω is a set of function symbols and Pred a set of predicate symbols. A *partial Σ-structure* is a structure

$$A = (\{A_s\}_{s \in S}, \{f_A\}_{f \in \Omega}, \{P_A\}_{P \in \mathsf{Pred}}),$$

where for every $s \in S$, A_s is a non-empty set and for every $f \in \Omega$ with arity $s_1 \dots s_n \to s$, f_A is a partial function from $\prod_{i=1}^{n} A_{s_i}$ to A_s. A is called a *total structure* if all functions f_A are total. (In the one-sorted case we will denote both an algebra and its support with the same symbol.) Details on partial algebras can be found in [12]. The notion of evaluating a term t with variables $X = \{X_s \mid s \in S\}$ w.r.t. an assignment $\{\beta_s{:}X_s \to A_s \mid s \in S\}$ for its variables in a partial structure A is the same as for total many-sorted algebras, except that the evaluation is undefined if $t = f(t_1, \dots, t_n)$ with $a(f) = (s_1 \dots s_n \to s)$, and at least one of $\beta_{s_i}(t_i)$ is undefined, or else $(\beta_{s_1}(t_1), \dots, \beta_{s_n}(t_n))$ is not in the domain of f_A.

A *weak Σ-embedding* between the partial structures $A = (\{A_s\}_{s \in S}, \{f_A\}_{f \in \Omega}, \{P_A\}_{P \in \mathsf{Pred}})$ and $B = (\{B_s\}_{s \in S}, \{f_B\}_{f \in \Omega}, \{P_B\}_{P \in \mathsf{Pred}})$ is a (many-sorted) family $i = (i_s)_{s \in S}$ of total maps $i_s : A_s \to B_s$ such that

- if $f_A(a_1, \dots, a_n)$ is defined then $f_B(i_{s_1}(a_1), \dots, i_{s_n}(a_n))$ is defined and $i_s(f_A(a_1, \dots, a_n)) = f_B(i_{s_1}(a_1), \dots, i_{s_n}(a_n))$, provided $a(f) = s_1 \dots s_n \to s$;
- for each s, i_s is injective and an embedding w.r.t. Pred i.e. for every $P \in \mathsf{Pred}$ with arity $s_1 \dots s_n$ and every $a_1, \dots, a_n$ where $a_i \in A_{s_i}$, $P_A(a_1, \dots, a_n)$ if and only if $P_B(i_{s_1}(a_1), \dots, i_{s_n}(a_n))$.

In this case we say that A *weakly embeds* into B.

In what follows we will denote a many-sorted variable assignment $\{\beta_s{:}X_s \to A_s \mid s \in S\}$ as $\beta : X \to \mathcal{A}$. For the sake of simplicity all definitions below are given for the one-sorted case. They extend in a natural way to many-sorted structures.

Definition 64 (Weak validity) *Let $(A, \{f_A\}_{f\in\Omega}, \{P_A\}_{P\in\mathsf{Pred}})$ be a partial structure and $\beta : X{\to}A$.*

(1) $(A,\beta) \models_w t \approx s$ *iff (a) $\beta(t)$ and $\beta(s)$ are both defined and equal; or (b) at least one of $\beta(s)$ and $\beta(t)$ is undefined.*

(2) $(A,\beta) \models_w t \not\approx s$ *iff (a) $\beta(t)$ and $\beta(s)$ are both defined and different; or (b) at least one of $\beta(s)$ and $\beta(t)$ is undefined.*

(3) $(A,\beta) \models_w P(t_1,\ldots,t_n)$ *iff (a) $\beta(t_1),\ldots,\beta(t_n)$ are defined and $(\beta(t_1),\ldots,\beta(t_n)){\in}P_A$; or (b) at least one of $\beta(t_1)$, …, $\beta(t_n)$ is undefined.*

(4) $(A,\beta){\models_w}\neg P(t_1,\ldots,t_n)$ *iff (a) $\beta(t_1)$, …, $\beta(t_n)$ are defined and $(\beta(t_1),\ldots,\beta(t_n)){\notin}P_A$; or (b) at least one of $\beta(t_1),\ldots,\beta(t_n)$ is undefined.*

(A,β) weakly satisfies a clause C *(notation: $(A,\beta) \models_w C$) if $(A,\beta) \models_w L$ for at least one literal L in C. A* weakly satisfies C *(notation: $A \models_w C$) if $(A,\beta) \models_w C$ for all assignments β. A* weakly satisfies a set of clauses $\mathcal{K}$ *(notation: $A \models_w \mathcal{K}$) if $A \models_w C$ for all $C \in \mathcal{K}$.*

Definition 65 (Evans validity) *Evans validity is defined similarly, with the difference that (1) is replaced with:*

(1') $(A,\beta) \models t \approx s$ *if and only if*

- *(a) $\beta(t)$ and $\beta(s)$ are both defined and equal; or*
- *(b) $\beta(s)$ is defined, $t = f(t_1,\ldots,t_n)$ and $\beta(t_i)$ is undefined for at least one of the direct subterms of t; or*
- *(c) both $\beta(s)$ and $\beta(t)$ are undefined.*

Evans validity extends to (sets of) clauses in the usual way. We use the notation: $(A,\beta) \models L$ for a literal L; $(A,\beta) \models C$ and $A \models C$ for a clause C, etc.

Example 39 *Let A be a partial Ω-algebra, where $\Omega = \{\mathsf{car}/1, \mathsf{nil}/0\}$. Assume that nil_A is defined and $\mathsf{car}_A(\mathsf{nil}_A)$ is not defined. Then:*

- $A \not\models \mathsf{car}(\mathsf{nil}) \approx \mathsf{nil}$ *(since $\mathsf{car}_A(\mathsf{nil})$ is undefined in A, but nil is defined in A);*
- $A \models \mathsf{car}(\mathsf{nil}) \not\approx \mathsf{nil}$*;*
- $A \models_w \mathsf{car}(\mathsf{nil}) \approx \mathsf{nil}$*, $A \models_w \mathsf{car}(\mathsf{nil}) \not\approx \mathsf{nil}$ (since $\mathsf{car}(\mathsf{nil})$ is not defined in A).*

Definition 66 *A partial Σ-algebra A is a* weak partial model *(resp. partial model) of $\mathcal{T}_1$ with totally defined Ω_0-function symbols if (i) $A_{|\Sigma_0}$ is a model of $\mathcal{T}_0$ and (ii) $A \models_w \mathcal{K}$ (resp. $A \models \mathcal{K}$).*