Applications of hierarchical reasoning in the verification of complex systems[3]

Swen Jacobs[1] Viorica Sofronie-Stokkermans[2]

Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, Saarbrücken, Germany

Abstract

In this paper we show how hierarchical reasoning can be used to verify properties of complex systems. Chains of local theory extensions are used to model a case study taken from the European Train Control System (ETCS) standard, but considerably simplified. We show how testing invariants and bounded model checking (for safety properties expressed by universally quantified formulae, depending on certain parameters of the systems) can automatically be reduced to checking satisfiability of ground formulae over a base theory.

Keywords: Combinations of decision procedures, Hierarchical reasoning, Verification

1 Introduction

Many problems in computer science can be reduced to proving satisfiability of conjunctions of (ground) literals modulo a background theory. This theory can be a standard theory, the extension of a base theory with additional functions (free or subject to additional conditions), or a combination of theories. In [8] we showed that for special types of theory extensions, which we called *local*, hierarchic reasoning in which a theorem prover for the base theory is used as a "black box" is possible. Many theories important for computer science are local extensions of a base theory. Several examples (including theories of data structures, e.g. theories of lists (or arrays cf. [4]); but also theories of monotone functions or of functions satisfying semi-Galois conditions) are given in [8,9]. Here we present additional examples of local theory extensions occurring in the verification of complex systems.

In this paper we address a case study taken from the specification of the European Train Control System (ETCS) standard (cf. [3]) but considerably simplified, namely an example of a communication device responsible for a given segment of

[1] Email: sjacobs@mpi-sb.mpg.de

[2] Email: sofronie@mpi-sb.mpg.de

[3] 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.

the rail track, where trains may enter and leave. We suppose that, at fixed moments in time, all knowledge about the current positions of the trains is available to a controller which accordingly imposes constraints on the speed of some trains, or allows them to move freely within the allowed speed range on the track. Related problems were tackled before with methods from verification [3].

The approach we use in this paper is different from previously used methods. We use sorted arrays (or monotonely decreasing functions) for storing the train positions. The use of abstract data structures allows us to pass in an elegant way from verification of several finite instances of problems (modeled by finite-state systems) to general verification results, in which sets of states are represented using formulae in first-order logic, by keeping the number of trains as a parameter. We show that for invariant or bounded model checking the specific properties of "position updates" can be expressed in a natural way by using chains of local theory extensions. Therefore we can use results in hierarchic theorem proving both for invariant and for bounded model checking [4]. By using locality of theory extensions we also obtained formal arguments on possibilities of systematic slicing (for bounded model checking): we show that for proving (disproving) the violation of the safety condition we only need to consider those trains which are in a neighborhood of the trains which violate the safety condition [5].

Structure of the paper. Section 2 contains the main theoretical results needed in the paper. In Section 3 we describe the case study we consider. In Section 4 we present a method for invariant and bounded model checking based on hierarchical reasoning. Section 5 contains conclusions and perspectives.

2 Preliminaries

Theories and models. Theories can be regarded as sets of formulae or as sets of models. Let $\mathcal{T}$ be a theory in a (many-sorted) signature $\Pi = (S, \Sigma, \mathsf{Pred})$, where S is a set of sorts, Σ is a set of function symbols and Pred a set of predicate symbols (with given arities). A Π-structure is a tuple

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

where for every $s \in S$, M_s is a non-empty set, for all $f \in \Sigma$ with arity $a(f){=}s_1{\times}\ldots{\times}s_n{\to}s$, $f_{\mathcal{M}} : \prod_{i=1}^{n} M_{s_i}{\to}M_s$ and for all $P \in \mathsf{Pred}$ with arity $a(P) = s_1{\times}\ldots{\times}s_n, P_{\mathcal{M}} \subseteq M_{s_1}\times\ldots\times M_{s_n}$. We consider 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. A model of $\mathcal{T}$ is a Π-structure satisfying all formulae of $\mathcal{T}$. In this paper, whenever we speak about a theory $\mathcal{T}$ we implicitly refer to the set $\mathsf{Mod}(\mathcal{T})$ of all models of $\mathcal{T}$, if not otherwise specified.

Partial structures. Let $\mathcal{T}_0$ be a theory with signature $\Pi_0 = (S_0, \Sigma_0, \mathsf{Pred})$. We consider extensions $\mathcal{T}_1$ of $\mathcal{T}_0$ with signature $\Pi = (S, \Sigma, \mathsf{Pred})$, where $S = S_0 \cup S_1, \Sigma =$

[4] Here we only focus on one example. However, we also used this technique for other case studies (among which one is mentioned – in a slightly different context – in [9]).

[5] In fact, it turns out that slicing (locality) results with a similar flavor presented by Necula and McPeak in [6] have a similar theoretical justification.

$\Sigma_0 \cup \Sigma_1$ (i.e. the signature is extended by new sorts and function symbols) and $\mathcal{T}_1$ is obtained from $\mathcal{T}_0$ by adding a set $\mathcal{K}$ of (universally quantified) clauses. Thus, $\mathsf{Mod}(\mathcal{T}_1)$ consists of all Π-structures which are models of $\mathcal{K}$ and whose reduct to Π_0 is a model of $\mathcal{T}_0$.

A *partial* Π-*structure* is a structure $\mathcal{M} = (\{M_s\}_{s\in S}, \{f_{\mathcal{M}}\}_{f\in\Sigma}, \{P_{\mathcal{M}}\}_{P\in\mathsf{Pred}})$, where for every $s \in S$, M_s is a non-empty set and for every $f \in \Sigma$ with arity $s_1\times\ldots\times s_n\rightarrow s$, $f_{\mathcal{M}}$ is a partial function from $\prod_{i=1}^{n} M_{s_i}$ to M_s. The notion of evaluating a term t with variables $X = \{X_s \mid s \in S\}$ w.r.t. an assignment $\{\beta_s{:}X_s \rightarrow M_s \mid s \in S\}$ for its variables in a partial structure $\mathcal{M}$ is the same as for total many-sorted algebras, except that the evaluation is undefined if $t = f(t_1, \ldots, t_n)$ with $a(f){=}s_1\times\ldots\times s_n\rightarrow s$, and at least one of $\beta_{s_i}(t_i)$ is undefined, or else $(\beta_{s_1}(t_1), \ldots, \beta_{s_n}(t_n))$ is not in the domain of $f_{\mathcal{M}}$. In what follows we will denote a many-sorted variable assignment $\{\beta_s{:}X_s \rightarrow M_s \mid s \in S\}$ as $\beta : X \rightarrow \mathcal{M}$. Let $\mathcal{M}$ be a partial Π-structure, C a clause and $\beta : X \rightarrow \mathcal{M}$. We say that $(\mathcal{M}, \beta) \models_w C$ iff either (i) for some term t in C, $\beta(t)$ is undefined, or else (ii) $\beta(t)$ is defined for all terms t of C, and there exists a literal L in C s.t. $\beta(L)$ is true in $\mathcal{M}$. $\mathcal{M}$ *weakly satisfies* C (notation: $\mathcal{M} \models_w C$) if $(\mathcal{M}, \beta) \models_w C$ for all $\beta : X \rightarrow \mathcal{M}$. $\mathcal{M}$ *is a weak partial model of a set of clauses* $\mathcal{K}$ (notation: $\mathcal{M} \models_w \mathcal{K}$, $\mathcal{M}$ is a w.p.model of $\mathcal{K}$) if $\mathcal{M} \models_w C$ for all $C \in \mathcal{K}$.

Local theory extensions. Let $\mathcal{K}$ be a set of (universally quantified) clauses in the signature $\Pi = (S, \Sigma, \mathsf{Pred})$, where $S = S_0 \cup S_1$ and $\Sigma = \Sigma_0 \cup \Sigma_1$. In what follows, when referring to sets G of ground clauses we assume they are in the signature $\Pi^c = (S, \Sigma \cup \Sigma_c, \mathsf{Pred})$ where Σ_c is a set of new constants. An extension $\mathcal{T}_0 \subseteq \mathcal{T}_0 \cup \mathcal{K}$ is *local* if satisfiability of a set G of clauses with respect to $\mathcal{T}_0 \cup \mathcal{K}$ only depends on $\mathcal{T}_0$ and those instances $\mathcal{K}[G]$ of $\mathcal{K}$ in which the terms starting with extension functions are in the set $\mathsf{st}(\mathcal{K}, G)$ of ground terms which already occur in G or $\mathcal{K}$. Formally,

$$\begin{aligned}\mathcal{K}[G] = \{C\sigma \mid & C \in \mathcal{K},\ \text{for each subterm } f(t) \text{ of } C,\ \text{with } f \in \Sigma_1,\\ & f(t)\sigma \in \mathsf{st}(\mathcal{K}, G),\ \text{and for each variable } x \text{ which does not}\\ & \text{occur below a function symbol in } \Sigma_1, \sigma(x) = x\},\end{aligned}$$

and $\mathcal{T}_0 \subseteq \mathcal{T}_1{=}\mathcal{T}_0 \cup \mathcal{K}$ is a local extension if it satisfies condition (Loc):

(Loc) For every set G of ground clauses $G \models_{\mathcal{T}_1} \bot$ iff there is no partial Π^c-structure P such that $P_{|\Pi_0}$ is a total model of $\mathcal{T}_0$, all terms in $\mathsf{st}(\mathcal{K}, G)$ are defined in P, and P weakly satisfies $\mathcal{K}[G] \wedge G$.

In [8,9] we gave several examples of local theory extensions: e.g. any extension of a theory with free function symbols; extensions with selector functions for a constructor which is injective in the base theory; extensions of several partially ordered theories with monotone functions. In Section 4.2 we give additional examples which have particular relevance in verification.

Hierarchic reasoning in local theory extensions. Let $\mathcal{T}_0 \subseteq \mathcal{T}_1{=}\mathcal{T}_0 \cup \mathcal{K}$ be a local theory extension. To check the satisfiability of a set G of ground clauses w.r.t. $\mathcal{T}_1$ we can proceed as follows (for details cf. [8]):

Step 1: Use locality. By the locality condition, G is unsatisfiable w.r.t. $\mathcal{T}_1$ iff $\mathcal{K}[G] \wedge G$ has no weak partial model in which all the subterms of $\mathcal{K}[G] \wedge G$ are defined, and whose restriction to Π_0 is a total model of $\mathcal{T}_0$.

Step 2: Flattening and purification. We purify and flatten $\mathcal{K}[G] \wedge G$ by introducing new constants for the arguments of the extension functions as well as for the (sub)terms $t = f(g_1, \ldots, g_n)$ starting with extension functions $f \in \Sigma_1$, together with new corresponding definitions $c_t \approx t$. The set of clauses thus obtained has the form $\mathcal{K}_0 \wedge G_0 \wedge D$, where D is a set of ground unit clauses of the form $f(c_1, \ldots, c_n) \approx c$, where $f \in \Sigma_1$ and $c_1, \ldots, c_n, c$ are constants, and $\mathcal{K}_0, G_0$ are clause sets without function symbols in Σ_1.

Step 3: Reduction to testing satisfiability in $\mathcal{T}_0$. We reduce the problem to testing satisfiability in $\mathcal{T}_0$ by replacing D with the following set of clauses:

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

Theorem 2.1 ([8]) *With the notations above, the following are equivalent:*

(1) $\mathcal{T}_0 \wedge \mathcal{K} \wedge G$ *has a model.*

(2) $\mathcal{T}_0 \wedge \mathcal{K}[G] \wedge G$ *has a w.p.model (where all terms in* $\mathsf{st}(\mathcal{K}, G)$ *are defined).*

(3) $\mathcal{T}_0 \wedge \mathcal{K}_0 \wedge G_0 \wedge D$ *has a w.p.model (with all terms in* $\mathsf{st}(\mathcal{K}, G)$ *defined).*

(4) $\mathcal{T}_0 \wedge \mathcal{K}_0 \wedge G_0 \wedge N_0$ *has a (total)* Σ_0*-model.*

3 The RBC Case Study

The case study we discuss here is taken from the specification of the European Train Control System (ETCS) standard [3]: we consider a radio block center (RBC), which communicates with all trains on a given track segment. Trains may enter and leave the area, given 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 decided: if the distance to the preceding train is less than a certain limit $\mathsf{l}_{\mathsf{alarm}}$, the train may only move with minimum speed min (otherwise with any speed between min and the maximum speed max).

3.1 Formal Description of the System Model

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 $\Delta\mathsf{t} >_{\mathbb{R}} 0$ (the time between evaluations of the system) [6], 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 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 pos') is defined by the following set $\mathcal{K}_f = \{\mathsf{F1}, \mathsf{F2}, \mathsf{F3}, \mathsf{F4}\}$ of axioms:

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

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

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

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

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 $\mathsf{F1}$ states that the first train may always move at any speed between min and max. $\mathsf{F2}$ 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 $\mathsf{F3}$). $\mathsf{F4}$ 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 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 $\mathsf{maxTrains}$. 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 $(\mathsf{V1}) - (\mathsf{V9})$:

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

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

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

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

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

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

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

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

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

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

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.

4 Hierarchical reasoning in verification

The safety condition which is important for this type of systems is collision freeness. Intuitively (but in a very simplified model of the system of trains) collision freeness is similar to a bounded strict monotonicity property for the function pos which stores the positions of the trains:

$$\mathsf{Mon}(\mathsf{pos}) \qquad \forall i, j \quad (0 \leq i < j < \mathsf{n} \rightarrow \mathsf{pos}(i) >_{\mathbb{R}} \mathsf{pos}(j))$$

Mon(pos) expresses the condition that for all trains i, j on the track, if i precedes j then i should be positioned strictly ahead of j.

We will also consider a more realistic extension, which allows to express collision-freeness when the maximum length of the trains is known. In both cases, we focus on invariant checking and on bounded model checking.

4.1 Problems: Invariant checking, bounded model checking

In what follows we illustrate the ideas for the simple approach, in which collision-freeness is identified with strict monotonicity of the function which stores the positions of the trains. To check that strict monotonicity of train positions is an invariant, we need to check that:

(a) In the initial state the train positions (expressed by a function pos_0) satisfy the strict monotonicity condition $\mathsf{Mon}(\mathsf{pos}_0)$.

(b) Assuming that at a given state, the function pos (indicating the positions) satisfies the strict monotonicity condition Mon(pos), and the next state positions, stored in pos', satisfy the axioms $\mathcal{K}$, where $\mathcal{K} \in \{\mathcal{K}_f, \mathcal{K}_v\}$, then pos' satisfies the strict monotonicity condition $\mathsf{Mon}(\mathsf{pos}')$.

Checking (a) is not a problem. For (b) we need to show that in the extension $\mathcal{T}$ of a combination $\mathcal{T}_0$ of real arithmetic (sort num) with an index theory describing precedence of trains (sort i), with the two functions pos and pos' (with arity $\mathsf{i} \rightarrow \mathsf{num}$)

the following holds:

$$\mathcal{T} \models \mathcal{K} \wedge \mathsf{Mon}(\mathsf{pos}) \rightarrow \mathsf{Mon}(\mathsf{pos}'), \quad \text{i.e.} \quad \mathcal{T} \wedge \mathcal{K} \wedge \mathsf{Mon}(\mathsf{pos}) \wedge \neg\mathsf{Mon}(\mathsf{pos}') \models \bot .$$

The set of formulae to be proved unsatisfiable w.r.t. $\mathcal{T}$ involves the axioms $\mathcal{K}$ and $\mathsf{Mon}(\mathsf{pos})$, containing universally quantified variables of sort i. Only $\neg\mathsf{Mon}(\mathsf{pos}')$ corresponds to a ground set of clauses G. However, positive results for reasoning in combinations of theories were only obtained for testing satisfiability for ground formulae [7,5], so are not directly applicable.

In bounded model checking the same problem occurs. For a fixed k, one has to show that there are no paths of length at most k from the initial state to an unsafe state. We therefore need to store all intermediate positions in arrays $\mathsf{pos}_0, \mathsf{pos}_1, \ldots, \mathsf{pos}_k$, and – provided that $\mathcal{K}(\mathsf{pos}_{i-1}, \mathsf{pos}_i)$ is defined such that $\mathcal{K} = \mathcal{K}(\mathsf{pos}, \mathsf{pos}')$ – to show:

$$\mathcal{T} \wedge \bigwedge_{i=1}^{j} \mathcal{K}(\mathsf{pos}_{i-1}, \mathsf{pos}_i) \wedge \mathsf{Mon}(\mathsf{pos}_0) \wedge \neg\mathsf{Mon}(\mathsf{pos}_j) \models \bot \qquad \text{for all } 0 \leq j \leq k.$$

4.2 Our solution: locality, hierarchical reasoning

Our idea. In order to overcome the problem mentioned above we proceed as follows. We consider two successive extensions of the base theory $\mathcal{T}_0$ (a many-sorted combination of real or rational arithmetic – for reasoning about positions, sort num – with an index theory – for describing precedence between trains, sort i):

- the extension $\mathcal{T}_1$ of $\mathcal{T}_0$ with a monotone function pos, of arity $\mathsf{i} \rightarrow \mathsf{num}$,
- the extension $\mathcal{T}_2$ of $\mathcal{T}_1$ with a function pos' (arity $\mathsf{i} \rightarrow \mathsf{num}$) satisfying $\mathcal{K} \in \{\mathcal{K}_f, \mathcal{K}_v\}$.

We show that both extensions $\mathcal{T}_0 \subseteq \mathcal{T}_1 = \mathcal{T}_0 \cup \mathsf{Mon}(\mathsf{pos})$ and $\mathcal{T}_1 \subseteq \mathcal{T}_2 = \mathcal{T}_1 \cup \mathcal{K}$ are local, where $\mathcal{K} \in \{\mathcal{K}_f, \mathcal{K}_v\}$. This allows us to reduce problem (b) to testing satisfiability of ground clauses in $\mathcal{T}_0$, for which standard methods for reasoning in combinations of theories can be applied. A similar method can be used for bounded model checking.

The base theory. As mentioned before, we assume that $\mathcal{T}_0$ is the many-sorted combination of a theory $\mathcal{T}_0^{\mathsf{i}}$ (sort i) for reasoning about precedence between trains and a theory $\mathcal{T}_0^{\mathsf{num}}$ (sort num) for reasoning about distances between trains. We have several possibilities of choosing $\mathcal{T}_0^{\mathsf{i}}$: we can model the trains on a track by using an (acyclic) list structure, where any train is linked to its predecessor, or using the theory of integers with predecessor. $\mathcal{T}_0^{\mathsf{num}}$ can be the theory of real or rational numbers, or linear real or rational arithmetic.

Notation. As a convention, everywhere in what follows i, j, k denote variables of sort i and c, d denote variables of sort num.

Collision freeness as monotonicity. In what follows let $\mathcal{T}_0^{\mathsf{i}}$ be the theory of (linear) integer arithmetic and $\mathcal{T}_0^{\mathsf{num}}$ be the theory of real or rational numbers. In both these theories satisfiability of ground clauses is decidable. Let $\mathcal{T}_0$ be the (disjoint, many-sorted) combination of $\mathcal{T}_0^{\mathsf{i}}$ and $\mathcal{T}_0^{\mathsf{num}}$. Then classical methods on

combinations of decision procedures for (disjoint, many-sorted) theories can be used to give a decision procedure for satisfiability of ground clauses w.r.t. $\mathcal{T}_0$. Let $\mathcal{T}_1$ be obtained by extending $\mathcal{T}_0$ with a function pos of arity $\mathsf{i} \to \mathsf{num}$ mapping train indices to the real numbers, which satisfies condition $\mathsf{Mon}(\mathsf{pos})$:

$$\mathsf{Mon}(\mathsf{pos}) \qquad \forall i, j \;\; (\mathsf{first} \leq i < j \leq \mathsf{last} \to \mathsf{pos}(i) >_{\mathbb{R}} \mathsf{pos}(j)),$$

where i and j are indices, $<$ is the ordering on indices and $>_{\mathbb{R}}$ is the usual ordering on the real numbers. (For the case of a fixed number n of trains, we can assume that $\mathsf{first} = 0$ and $\mathsf{last} = \mathsf{n} - 1$.)

A more precise axiomatization of collision-freeness. The monotonicity axiom above is, in fact, an oversimplification. A more precise model, in which the length of trains is considered can be obtained by replacing the monotonicity axiom for pos with the following axiom:

$$\forall i, j, k \quad (\mathsf{first} \leq j \leq i \leq \mathsf{last} \wedge i - j = k \to \mathsf{pos}(j) - \mathsf{pos}(i) \geq_{\mathbb{R}} k * \mathsf{LengthTrain}),$$

where $\mathsf{LengthTrain}$ 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 $\mathsf{i} \times \mathsf{num} \to \mathsf{num}$ (multiplication of k with the constant $\mathsf{LengthTrain}$ in the formula above) [7]. Let $\mathcal{T}_1'$ be the theory obtained by extending $\mathcal{T}_0'$ with a function pos satisfying the axiom above.

Theorem 4.1 *The following extensions are local theory extensions:*

(1) The theory extension $\mathcal{T}_0 \subseteq \mathcal{T}_1$.

(2) The theory extension $\mathcal{T}_0' \subseteq \mathcal{T}_1'$.

Proof: (1) The clause which states that pos is strictly decreasing

$$\mathsf{Mon}(\mathsf{pos}) \qquad \forall i, j \;\; (\mathsf{first} \leq i < j \leq \mathsf{last} \to \mathsf{pos}(i) >_{\mathbb{R}} \mathsf{pos}(j))$$

is flat and linear w.r.t. pos, so we can prove the claim by showing that every weak partial model M of $\mathcal{T}_1$ in which everything except pos is totally defined can be extended to a total model of $\mathcal{T}_1$. Locality then follows by results in [8]. To define pos at positions where it is undefined we use the density of real numbers and the fact that between two integers there are only finitely many integers:

Let M be a weak partial model of $\mathcal{T}_1$. We denote by M_{i} the universe of M of sort i (i.e. the set of integers) and by M_{num} the support of M of sort num (i.e. the set of real numbers). Then for all $i, j \in M_{\mathsf{i}}$, if $i < j$ and both $\mathsf{pos}(i)$ and $\mathsf{pos}(j)$ are defined then $\mathsf{pos}(i) >_{\mathbb{R}} \mathsf{pos}(j)$. To extend M to a total model of $\mathcal{T}_1$, we define values for the $\mathsf{pos}(i)$ that are undefined in M. For every $i \in M_{\mathsf{i}}$ we check for the smallest $i^+ > i$ and the greatest $i^- < i$ with $\mathsf{pos}(i^+), \mathsf{pos}(i^-)$ defined in M:

- if neither such an i^+ nor such an i^- exists, then pos is totally undefined. Clearly, one can choose values for all indices such that $\mathsf{Mon}(\mathsf{pos})$ is satisfied.

[7] In the light of locality properties of such extensions (cf. Theorem 4.1), 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.

- if there exists an i^+, but not an i^-, we can choose any value for $\mathsf{pos}(i)$ which satisfies $\mathsf{pos}(i) >_{\mathbb{R}} \mathsf{pos}(i^+)$.
- if there exists an i^-, but not an i^+, we can choose any value for $\mathsf{pos}(i)$ which satisfies $\mathsf{pos}(i^-) >_{\mathbb{R}} \mathsf{pos}(i)$.
- if both i^+ and i^- exist, choose $\mathsf{pos}(i)$ such that it satisfies $\mathsf{pos}(i^-) >_{\mathbb{R}} \mathsf{pos}(i) >_{\mathbb{R}} \mathsf{pos}(i^+)$.

The procedure can be repeated until pos is defined at all points between first and last. As there are finitely many positions between these two positions, the procedure terminates after a finite number of steps. We can define pos arbitrarily outside of this range. The result is a total model of $\mathcal{T}_1$.

(2) The proof is similar to the proof of (1). Let M be a weak partial model of $\mathcal{T}_1'$. Let M_{i}, M_{num} as above. Then for all $i, j \in M_{\mathsf{i}}$, if $i - j = k > 0$ and both $\mathsf{pos}(i)$ and $\mathsf{pos}(j)$ are defined then $\mathsf{pos}(j) - \mathsf{pos}(i) \geq_{\mathbb{R}} k * \mathsf{LengthTrain}$. To extend M to a total model of $\mathcal{T}_1'$, we define values for the $\mathsf{pos}(i)$ that are undefined in M, using i^+ and i^- as above:

- if neither such an i^+ nor such an i^- exists, then a is totally undefined. Clearly, one can choose values for all indices such that the condition above is satisfied.
- if there exists an i^+, but not an i^-, we can fill in all the values, starting from i^+ by defining $\mathsf{pos}(i^+ - 1) = \mathsf{pos}(i^+) + \mathsf{LengthTrain}$, and inductively, $\mathsf{pos}(j - 1) = \mathsf{pos}(j) + \mathsf{LengthTrain}$ for all $i^+ \leq j \leq \mathsf{first}$.
- if there exists an i^-, but not an i^+, we proceed similarly.
- if both i^+ and i^- exist, we know that $\mathsf{pos}(i^-) - \mathsf{pos}(i^+) \leq (i^+ - i^-) * \mathsf{LengthTrain}$. Starting with $j = i^- + 1$ we define for every $i^- < j < i^+ - 1$, $\mathsf{pos}(j) = \mathsf{pos}(j - 1) + \mathsf{LengthTrain}$. □

We now extend the resulting theory $\mathcal{T}_1$ again in two different ways, with the axiom sets for one of the two system models, respectively. A similar construction can be done starting from the theory $\mathcal{T}_1'$.

Theorem 4.2 *The following extensions are local theory extensions:*

(1) The extension $\mathcal{T}_1 \subseteq \mathcal{T}_1 \cup \mathcal{K}_f$.

(2) The extension $\mathcal{T}_1 \subseteq \mathcal{T}_1 \cup \mathcal{K}_v$.

Proof: The idea for both proofs is to show that weak partial models can be extended to total ones, which implies locality by the results in [8].

(1) Clauses in $\mathcal{K}_f$ are flat and linear w.r.t. pos', so we can prove locality of the extension by showing that weak partial models can be extended to total ones. Let M be a weak partial model of $\mathcal{T}_1 \cup \mathcal{K}_f$, where everything but pos' is totally defined. We extend M by defining values for all undefined $\mathsf{pos}'(i)$:

- if $i < 0$ or $i \geq n$, $\mathsf{pos}'(i)$ can be chosen arbitrarily;
- if $0 \leq i \leq n - 1$, the left-hand sides of the implications (F1) to (F4) are mutually exclusive, i.e. for any possible valuation of i we only have to satisfy the right-hand side of one implication; the other implications are true because their antecedent

is false. Let i with $0 \leq i \leq n-1$ for which $\mathsf{pos}'(i)$ is undefined in M:
- if the left-hand side of (F1) or (F2) is true in M, choose a $\mathsf{pos}'(i)$ that satisfies $\mathsf{pos}(i) + \mathsf{min} \leq_{\mathbb{R}} \mathsf{pos}'(i) \leq_{\mathbb{R}} \mathsf{pos}(i) + \mathsf{max}$. This is possible, as $\mathsf{min} \leq_{\mathbb{R}} \mathsf{max}$;
- if the the left-hand side of (F3) is true in M, let $\mathsf{pos}'(i) = \mathsf{pos}(i) + \Delta t * \mathsf{min}$;
- if the the left-hand side of (F4) is true in M, let $\mathsf{pos}'(i) = \mathsf{pos}(i)$.

From the construction it is clear that we obtain a total model of $\mathcal{T}_1 \cup \mathcal{K}$.

(2) is proved similarly: As above, the axioms are flat and linear w.r.t. the function symbol pos' (and of course the constants $\mathsf{first}', \mathsf{last}'$), so it is enough to show that the weak partial models of $\mathcal{T}_1 \cup \mathcal{K}_v$ can be extended to total models.

Let M be a partial model of $\mathcal{T}_1 \cup \mathcal{K}_v$ in which everything but $\mathsf{pos}', \mathsf{first}', \mathsf{last}'$ is totally defined. We extend M to a total model of $\mathcal{T}_1 \cup \mathcal{K}_v$ in four steps:

(i) As in (1), we define values for undefined $\mathsf{pos}'(i)$ within the bounds of axioms (V1) to (V4), i.e. between first and last. For values outside of the bounds, we cannot make a statement yet, as (V9) also contains pos'.

(ii) if first' and/or last' are undefined, axioms (V5) to (V8) are satisfied by defining $\mathsf{first}' = \mathsf{first}$ and/or $\mathsf{last}' = \mathsf{last}$.

(iii) if $\mathsf{last}' = \mathsf{last} + 1$ in M and $\mathsf{pos}'(\mathsf{last}')$ is undefined, define $\mathsf{pos}'(\mathsf{last}')$ such that $\mathsf{pos}'(\mathsf{last}') <_{\mathbb{R}} \mathsf{pos}'(\mathsf{last})$.

(iv) for all $\mathsf{pos}'(i)$ that are still undefined, arbitrary values can be chosen.

We thus can extend M to a total model of $\mathcal{T}_1 \cup \mathcal{K}_v$. □

4.2.1 Hierarchical reasoning

Let $\mathcal{K} \in \{\mathcal{K}_v, \mathcal{K}_f\}$. By the locality of $\mathcal{T}_1 \subseteq \mathcal{T}_2 = \mathcal{T}_1 \cup \mathcal{K}$ and by Theorem 2.1, the following are equivalent:

(1) $\mathcal{T}_0 \wedge \mathsf{Mon}(\mathsf{pos}) \wedge \mathcal{K} \wedge \neg\mathsf{Mon}(\mathsf{pos}') \models \bot$,

(2) $\mathcal{T}_0 \wedge \mathsf{Mon}(\mathsf{pos}) \wedge \mathcal{K}[G] \wedge G \models_w \bot$, where $G = \neg\mathsf{Mon}(\mathsf{pos}')$,

(3) $\mathcal{T}_0 \wedge \mathsf{Mon}(\mathsf{pos}) \wedge \mathcal{K}_0 \wedge G_0 \wedge N_0(\mathsf{pos}') \models \bot$,

where $\mathcal{K}[G]$ consists of all instances of the rules in $\mathcal{K}$ in which the terms starting with the function symbols pos' are ground subterms already occurring in G or $\mathcal{K}$, $\mathcal{K}_0 \wedge G_0$ is obtained from $\mathcal{K}[G] \wedge G$ by introducing new constants for the arguments of the extension functions as well as for the (sub)terms $t = f(g_1, \ldots, g_n)$ starting with extension functions $f \in \Sigma_1$, and $N_0(\mathsf{pos}')$ is the set of instances of the congruence axioms for pos' which correspond to the definitions for these newly introduced constants.

It is easy to see that, due to the special form of the rules in $\mathcal{K}$ (all free variables in any clause occur as arguments of pos' both in $\mathcal{K}_f$ and in $\mathcal{K}_v$), $\mathcal{K}[G]$ (hence also $\mathcal{K}_0$) is a set of ground clauses. By the locality of $\mathcal{T}_0 \subseteq \mathcal{T}_1 = \mathcal{T}_0 \cup \mathsf{Mon}(\mathsf{pos})$, the following are equivalent:

(1) $\mathcal{T}_0 \wedge \mathsf{Mon}(\mathsf{pos}) \wedge \mathcal{K}_0 \wedge G_0 \wedge N_0(\mathsf{pos}') \models \bot$,

(2) $\mathcal{T}_0 \wedge \mathsf{Mon}(\mathsf{pos})[G'] \wedge G' \models_w \bot$, where $G' = \mathcal{K}_0 \wedge G_0 \wedge N_0(\mathsf{pos}')$,

(3) $\mathcal{T}_0 \wedge \mathsf{Mon}(\mathsf{pos})_0 \wedge G'_0 \wedge N_0(\mathsf{pos}) \models \bot$,

where $\mathsf{Mon}(\mathsf{pos})[G']$ consists of all instances of the rules in $\mathsf{Mon}(\mathsf{pos})$ in which the terms starting with the function symbol pos are ground subterms already occurring in G', $\mathsf{Mon}(\mathsf{pos})_0 \wedge G'_0$ is obtained from $\mathsf{Mon}(\mathsf{pos})[G'] \wedge G'$ by purification and flattening, and $N_0(\mathsf{pos})$ corresponds to the set of instances of congruence axioms for pos which need to be taken into account. The method is illustrated in Section 4.3.

4.2.2 Application: parametric verification

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.)

4.2.3 Bounded model checking

In the example above we restricted attention to the problem of showing that a property of train systems (collision freeness) is an inductive invariant. Similar results can be established for bounded model checking. In this case the arguments are similar, but one needs to consider chains of extensions of length $1, 2, 3, \ldots, k$ for a bounded k, corresponding to the paths from the initial state to be analyzed. An interesting side-effect of our approach (restricting to instances which are similar to the goal) is that it provides a possibility of systematic, goal-directed slicing: for proving (disproving) the violation of the safety condition we only need to consider those trains which are in a neighborhood of the trains which violate the safety condition.

4.3 Illustration

This section contains an illustration of the verification method based on hierarchical reasoning on the case study given in Section 3.

We indicate how to apply hierarchical reasoning on the case study given in Section 3, Model 1 [8]. We follow the steps given at the end of Section 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

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

we check whether $\mathcal{T}_1 \cup \mathcal{K}_f \cup G \models \bot$, where $G = \{0 \leq k < \mathsf{n}-1,\ \ k' = k+1,\ \ \mathsf{pos}'(k) \leq_{\mathbb{R}} \mathsf{pos}'(k')\}$ 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:

[8] We illustrate our approach for the simplest model. For a variable number of trains or the other definition of collision-freeness, the approach is the same.

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 F2 are:

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

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) $\Delta\mathsf{t} >_{\mathbb{R}} 0 \quad \wedge \quad 0 \leq_{\mathbb{R}} \mathsf{min} \quad \wedge \quad \mathsf{min} \leq_{\mathbb{R}} \mathsf{max}$

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 $D = \{\mathsf{pos}'(k) = c_1, \mathsf{pos}'(k') = c_2\}$ and a set $\mathcal{K}_{f_0}$ of clauses which do not contain occurrences of pos', consisting of (Dom) together with:

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

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

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 $\mathcal{T}_1$: $\mathcal{K}_{f_0} \wedge G_0 \wedge N_0$.

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 do 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 the theory 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_i for $\mathsf{pos}(k), \mathsf{pos}(k'), \mathsf{pos}(k''), \mathsf{pos}(k''')$. Taking into account only the corresponding instances of the monotonicity axiom for pos we obtain a set of clauses consisting of (Dom) together with:

$$(\mathsf{G}'_0) \qquad k'' = k - 1 \quad \wedge \quad k''' = k' - 1$$

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

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

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

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

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

$\mathsf{Mon(pos)[G']}$ $k < k' \rightarrow d_1 >_{\mathbb{R}} d_2 \ \wedge \ k' < k \rightarrow d_2 >_{\mathbb{R}} d_1 \ \wedge \ k'' < k''' \rightarrow d_3 >_{\mathbb{R}} d_4$
$k < k'' \rightarrow d_1 >_{\mathbb{R}} d_3 \ \wedge \ k' < k''' \rightarrow d_2 >_{\mathbb{R}} d_4 \ \wedge \ k''' < k \rightarrow d_4 >_{\mathbb{R}} d_1$
$k < k''' \rightarrow d_1 >_{\mathbb{R}} d_4 \ \wedge \ k'' < k \rightarrow d_3 >_{\mathbb{R}} d_1 \ \wedge \ k''' < k' \rightarrow d_4 >_{\mathbb{R}} d_2$
$k' < k'' \rightarrow d_2 >_{\mathbb{R}} d_3 \ \wedge \ k'' < k' \rightarrow d_3 >_{\mathbb{R}} d_2 \ \wedge \ k''' < k'' \rightarrow d_4 >_{\mathbb{R}} d_3$

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

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

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 of clauses equivalent to $\mathsf{Mon(pos)}[G'] \wedge N_0(\mathsf{pos}) \wedge N_0(\mathsf{pos'})$ 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} (sort num)	C_{Mixed}
$\mathsf{pos'}(k) = c_1 \ \wedge \ \mathsf{pos}(k') = d_2$	$k' = k + 1$	$d_1 >_{\mathbb{R}} d_2 \wedge d_3 >_{\mathbb{R}} d_4$	$(\mathsf{GF1_0})$
$\mathsf{pos'}(k') = c_2 \ \wedge \ \mathsf{pos}(k'') = d_3$	$k'' = k - 1$	$d_3 >_{\mathbb{R}} d_2 \wedge d_4 >_{\mathbb{R}} d_2$	$(\mathsf{GF2_0})$
$\mathsf{pos}(k) = d_1 \ \wedge \ \mathsf{pos}(k''') = d_4$	$k''' = k' - 1$	$d_3 >_{\mathbb{R}} d_1 \wedge d_1 = d_4$	$(\mathsf{GF3_0})$
	$0 \leq k, k' < n - 1$	$c_1 \leq_{\mathbb{R}} c_2 \ \wedge \ (\mathsf{Dom})$	$(\mathsf{GF4_0})$

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.

We present below an alternative method, somewhat similar to DPLL($\mathcal{T}_0$), but which uses only branching on the literals containing terms of sort i, and thus reduces the verification problem to the problem of checking the satisfiability of a set of linear constraints over the reals. This idea, we think, may be used to simplify automated verification for a whole class of problems in which the axioms are guarded by simple, mutually disjoint and exhaustive premises expressed in a specific theory. Such examples occur very often in verification where several disjoint cases need to be taken into account.

$k = 0$: Due to the constraints in C_{Indices}, the premises of all the other rules in $\mathsf{GF1_0} - GF4_0$ become false, so all the rules except for the first rule in $\mathsf{GF1_0}$ become trivially true. We thus only need to check the satisfiability (in the theory $\mathcal{T}_0^{\mathsf{num}}$) of $d_1+\Delta t*\mathsf{min} \leq_{\mathbb{R}} c_1 \leq_{\mathbb{R}} d_1+\Delta t*\mathsf{max} \ \wedge \ C_{\mathsf{Reals}}$. This is obviously satisfiable.

$k \neq 0$: We distinguish the following possibilities:

$k < 0$**:** unsatisfiable because of $0 \leq k$ in G_0.
$k > 0$**:** We distinguish the following possibilities:
$k > n-1$**:** unsatisfiable because of $k \leq n-1$ in G_0.
$k \leq n-1$**:** We distinguish the following possibilities:
$k' = 0$**:** We need to check the satisfiability of the following set of constraints over the reals:

$$\begin{cases} \mathsf{GF1}_0 : d_2 + \Delta t{*}\mathsf{min} \leq_{\mathbb{R}} c_2 \leq_{\mathbb{R}} d_2 + \Delta t{*}\mathsf{max} \\ \mathsf{GF2}_0 : d_3{>}_{\mathbb{R}}0 \ \wedge \ d_3{-}d_1 \geq_{\mathbb{R}} \mathsf{l}_{\mathsf{alarm}} \rightarrow d_1{+}\Delta t{*}\mathsf{min} \leq_{\mathbb{R}} c_1 \leq_{\mathbb{R}} d_1{+}\Delta t{*}\mathsf{max} \\ \mathsf{GF3}_0 : d_3{>}_{\mathbb{R}}0 \ \wedge \ d_3{-}d_1 <_{\mathbb{R}} \mathsf{l}_{\mathsf{alarm}} \rightarrow c_1 = d_1{+}\Delta t{*}\mathsf{min} \\ \mathsf{GF4}_0 : d_3 \leq_{\mathbb{R}} 0 \ \rightarrow \ c_1{=}d_1 \\ \\ C_{\mathsf{Reals}} \ c_1 \leq_{\mathbb{R}} c_2 \ \wedge \ d_1 >_{\mathbb{R}} d_2 \ \wedge \ d_3 >_{\mathbb{R}} d_1 \ \wedge \ d_3 >_{\mathbb{R}} d_2 \ \wedge \ d_4 >_{\mathbb{R}} d_2 \wedge \\ \qquad d_3 >_{\mathbb{R}} d_4 \ \wedge \ d_1 = d_2 \end{cases}$$

By using quantifier elimination for the variables $c_1, c_2, d_1, d_2, d_3, d_4$, we can obtain a direct relationship between $\Delta t, \mathsf{min}, \mathsf{max}, l_{\mathsf{alarm}}$ which guarantees satisfiability. We did this by using the REDLOG system [2].
$k' < 0$**:** unsatisfiable because of C_{Indices}.
$k' > 0$**:** We distinguish the following possibilities:
$k' > n-1$**:** unsatisfiable because of C_{Indices}.
$k' \leq n$**:** We need to check the satisfiability of the following set of constraints over the reals:

$$\begin{cases} \mathsf{GF2}_0 : d_3{>}_{\mathbb{R}}0 \ \wedge \ d_3{-}d_1 \geq_{\mathbb{R}} \mathsf{l}_{\mathsf{alarm}} \rightarrow d_1{+}\Delta t{*}\mathsf{min} \leq_{\mathbb{R}} c_1 \leq_{\mathbb{R}} d_1{+}\Delta t{*}\mathsf{max} \\ \qquad d_4{>}_{\mathbb{R}}0 \ \wedge \ d_4{-}d_2 \geq_{\mathbb{R}} \mathsf{l}_{\mathsf{alarm}} \rightarrow d_2{+}\Delta t{*}\mathsf{min} \leq_{\mathbb{R}} c_2 \leq_{\mathbb{R}} d_2{+}\Delta t{*}\mathsf{max} \\ \mathsf{GF3}_0 : d_3{>}_{\mathbb{R}}0 \ \wedge \ d_3{-}d_1 <_{\mathbb{R}} \mathsf{l}_{\mathsf{alarm}} \rightarrow c_1 = d_1{+}\Delta t{*}\mathsf{min} \\ \qquad d_4{>}_{\mathbb{R}}0 \ \wedge \ d_4{-}d_2 <_{\mathbb{R}} \mathsf{l}_{\mathsf{alarm}} \rightarrow c_2 = d_2{+}\Delta t{*}\mathsf{min} \\ \mathsf{GF4}_0 : d_3 \leq_{\mathbb{R}} 0 \ \rightarrow \ c_1{=}d_1 \\ \qquad d_4 \leq_{\mathbb{R}} 0 \ \rightarrow \ c_2{=}d_2 \\ C_{\mathsf{Reals}} \ c_1 \leq_{\mathbb{R}} c_2 \ \wedge \ d_1 >_{\mathbb{R}} d_2 \ \wedge \ d_3 >_{\mathbb{R}} d_1 \ \wedge \ d_3 >_{\mathbb{R}} d_2 \ \wedge \ d_4 >_{\mathbb{R}} d_2 \wedge \\ \qquad d_3 >_{\mathbb{R}} d_4 \ \wedge \ d_1 = d_2 \end{cases}$$

Again, by using quantifier elimination we can obtain a relationship between $\Delta t, \mathsf{min}, \mathsf{max}, l_{\mathsf{alarm}}$ which guarantees satisfiability.

Although the proofs above are generated by hand, the method is easy to implement. An implementation of the hierarchical method described in Section 2 is in progress.

5 Conclusions

In this paper we described a case study concerning a system of trains on a rail track, where trains may enter and leave the area. An example of a safety condition for such a system (collision freeness) was considered. The problem above can be reduced to testing satisfiability of *quantified formulae* in complex theories. However, the existing results on reasoning in combinations of theories are restricted to testing satisfiability for *ground formulae*.

This paper shows that, in the example considered, we can reduce satisfiability checking of universally quantified formulae to the simpler task of satisfiability

checking for ground clauses. For this, we identify corresponding chains of theory extensions $\mathcal{T}_0 \subseteq \mathcal{T}_1 \subseteq \cdots \subseteq \mathcal{T}_i$, such that $\mathcal{T}_j = \mathcal{T}_{j-1} \cup \mathcal{K}_j$ is a local extension of $\mathcal{T}_{j-1}$ by a set $\mathcal{K}_j$ of (universally quantified) clauses. This allows us to reduce, for instance, testing collision freeness in theories containing arrays to represent the train positions, to checking the satisfiability of a set of sets of ground clauses over the combination of the theory of reals with a theory which expresses precedence between trains. However, the applicability of the method is far more general: the challenge is, at the moment, to recognize classes of local theories occurring in various areas of application. The method can be used for parametric verification: after the reduction of the problem to a satisfiability problem in the base theory, one can automatically determine constraints on the parameters $(\Delta\mathsf{t}, \mathsf{min}, \mathsf{max}, ...)$ which guarantee that the property is an inductive invariant. The implementation of the procedure described here is in progress; the method is clearly easy to implement. Our results also open a possibility of using abstraction-refinement deductive model checking in a whole class of applications including the examples presented here – these aspects are not discussed in this paper, and rely on results we obtained in [9].

The results we present here also have theoretical implications: In one of the models we considered here, collision-freeness is expressed as a monotonicity condition. Limits of decidability in reasoning about sorted arrays were explored in [1]. The decidability of satisfiability of ground clauses in the fragment of the theory of sorted arrays which we consider here is an easy consequence of the locality of extensions with monotone functions.

Acknowledgement. Many thanks to Johannes Faber for sending us a case study taken from the specification of the European Train Control System which we used as a starting point for the specification considered in this paper.

References

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

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

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

[4] 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.

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

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

[7] G. Nelson and D. Oppen. Simplification by cooperating decision procedures. *ACM Trans. on Programming Languages and Systems*, 1(2):245–257, 1979.

[8] V. Sofronie-Stokkermans. Hierarchic reasoning in local theory extensions. In R. Nieuwenhuis, editor, *Automated deduction - CADE-20. Proceedings of the 20th International Conference on Automated Deduction*, LNCS 3632, pp. 219–234. Springer, 2005.

[9] V. Sofronie-Stokkermans. Interpolation in local theory extensions. In U. Furbach and N. Shankar editors, *Automated Reasoning. Third International Joint Conference, IJCAR 2006*, LNAI 4130, pp. 235–250. Springer, 2006.