

Applications of hierarchical reasoning in the verification of complex systems

Swen Jacobs and Viorica Sofronie-Stokkermans

Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, Saarbrücken, Germany
e-mail: {sjacobs, sofronie}@mpi-sb.mpg.de

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 can automatically be reduced to checking satisfiability of ground formulae over a base theory.

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. [3]); but also theories of monotone functions or of functions satisfying semi-Galois conditions) are given in [8] and [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 [2], but considerably simplified, namely an example of a communication device responsible for a given segment of 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 [2].

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 checking¹. 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 condition².

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 $\Pi = (S, \Sigma, \text{Pred})$, 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

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

where for every $s \in S$, M_s is a non-empty set, for all $f \in \Sigma$ with arity $a(f) = s_1 \dots s_n \rightarrow s$, $f_{\mathcal{M}} : \prod_{i=1}^n M_{s_i} \rightarrow M_s$ and for all $P \in \text{Pred}$ with arity $a(P) = s_1 \dots s_n$, $P_{\mathcal{M}} \subseteq M_{s_1} \times \dots \times M_{s_n}$. 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 $\text{Mod}(\mathcal{T})$ of all models of \mathcal{T} , if not otherwise specified.

Partial structures. Let \mathcal{T}_0 be a theory with signature $\Pi_0 = (S_0, \Sigma_0, \text{Pred})$. We consider extensions \mathcal{T}_1 of \mathcal{T}_0 with signature $\Pi = (S, \Sigma, \text{Pred})$, where $S = S_0 \cup S_1, \Sigma = \Sigma_0 \cup \Sigma_1$ (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, $\text{Mod}(\mathcal{T}_1)$ 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 $\mathcal{M} = (\{M_s\}_{s \in S}, \{f_{\mathcal{M}}\}_{f \in \Sigma}, \{P_{\mathcal{M}}\}_{P \in \text{Pred}})$, where for every $s \in S$, M_s is a non-empty set and for every $f \in \Sigma$ with arity

¹Here 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]).

²In fact, it turns out that slicing (locality) results with a similar flavor presented by Necula and McPeak in [6] have a similar theoretical justification.

$s_1 \dots s_n \rightarrow s$, $f_{\mathcal{M}}$ is a partial function from $M_{s_1} \times \dots \times M_{s_n}$ 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 this evaluation is undefined if $t = f(t_1, \dots, t_n)$ with $a(f) = (s_1 \dots s_n \rightarrow s)$, 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 $f_{\mathcal{M}}$. 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 assignments β . \mathcal{M} *weakly satisfies (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 $\Pi = (S, \Sigma, \text{Pred})$, 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 $\Pi^c = (S, \Sigma \cup \Sigma_c, \text{Pred})$ where Σ_c is a set of new constants. An extension $\mathcal{T}_0 \subseteq \mathcal{T}_1 = \mathcal{T}_0 \cup \mathcal{K}$ 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 $\text{st}(\mathcal{K}, G)$ of ground terms which already occur in G or \mathcal{K} . Formally,

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

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 $G \models_{\mathcal{T}_1} \perp$ 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 $\text{st}(\mathcal{K}, G)$ 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 $t = f(g_1, \dots, g_n)$ 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 $f(c_1, \dots, c_n) \approx c$, where $f \in \Sigma_1$ and c_1, \dots, 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:

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

Theorem 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 $\text{st}(\mathcal{K}, \mathbf{G})$ are defined).
- (3) $\mathcal{T}_0 \wedge \mathcal{K}_0 \wedge G_0 \wedge D$ has a w.p.model (with all terms in $\text{st}(\mathcal{K}, \mathbf{G})$ 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 (cf. e.g. [2]): 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 Δt , 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 l_{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 following functions and constants:

- $\Delta t > 0$, the time between evaluations of the system.
- \min and \max , the minimum and maximum speed of trains. We assume that $0 \leq \min \leq \max$.
- l_{alarm} , the distance between trains which is deemed secure.
- n , the number of trains.
- pos , a function which maps indices (between 0 and $n - 1$) associated to trains on the track to the positions of those trains on the track. Here $\text{pos}(i)$ denotes the current position of the train with index i .

We use a new function pos' to model the evolution of the system: $\text{pos}'(i)$ denotes the position of i at the next evaluation point (after Δt time units). The way positions change (i.e. the relationship between pos and pos') is defined by the following set $\mathcal{K}_f = \{\text{F1}, \text{F2}, \text{F3}, \text{F4}\}$ of axioms:

- (F1) $\forall i \ (i = 0 \rightarrow \text{pos}(i) + \Delta t * \min \leq_{\mathbb{R}} \text{pos}'(i) \leq_{\mathbb{R}} \text{pos}(i) + \Delta t * \max)$
- (F2) $\forall i \ (0 < i < n \wedge \text{pos}(i-1) >_{\mathbb{R}} 0 \wedge \text{pos}(p(i)) - \text{pos}(i) \geq_{\mathbb{R}} l_{\text{alarm}} \rightarrow \text{pos}(i) + \Delta t * \min \leq_{\mathbb{R}} \text{pos}'(i) \leq_{\mathbb{R}} \text{pos}(i) + \Delta t * \max)$
- (F3) $\forall i \ (0 < i < n \wedge \text{pos}(i-1) >_{\mathbb{R}} 0 \wedge \text{pos}(p(i)) - \text{pos}(i) <_{\mathbb{R}} l_{\text{alarm}} \rightarrow \text{pos}'(i) = \text{pos}(i) + \Delta t * \min)$
- (F4) $\forall i \ (0 < i < n \wedge \text{pos}(i-1) \leq_{\mathbb{R}} 0 \rightarrow \text{pos}'(i) = \text{pos}(i)),$

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 l_{alarm} . If the predecessor of a train has started, but is less than 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 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 $n - 1$. The behavior of this extended system is described by the following set \mathcal{K}_v consisting of axioms (V1) – (V9):

- (V1) $\forall i \ (i = \text{first} \rightarrow \text{pos}(i) + \Delta t * \min \leq_{\mathbb{R}} \text{pos}'(i) \leq_{\mathbb{R}} \text{pos}(i) + \Delta t * \max)$
- (V2) $\forall i \ (\text{first} < i \leq \text{last} \wedge \text{pos}(i-1) >_{\mathbb{R}} 0 \wedge \text{pos}(i-1) - \text{pos}(i) \geq_{\mathbb{R}} l_{\text{alarm}} \rightarrow \text{pos}(i) + \Delta t * \min \leq_{\mathbb{R}} \text{pos}'(i) \leq_{\mathbb{R}} \text{pos}(i) + \Delta t * \max)$

- (V3) $\forall i \quad (\text{first} < i \leq \text{last} \wedge \text{pos}(i-1) >_{\mathbb{R}} 0 \wedge \text{pos}(i-1) - \text{pos}(i) <_{\mathbb{R}} l_{\text{alarm}} \rightarrow \text{pos}'(i) = \text{pos}(i) + \Delta t * \text{min})$
- (V4) $\forall i \quad (\text{first} < i \leq \text{last} \wedge \text{pos}(i-1) \leq_{\mathbb{R}} 0 \rightarrow \text{pos}'(i) = \text{pos}(i))$
- (V5) $\text{last} - \text{first} + 1 < \text{maxTrains} \rightarrow \text{last}' = \text{last} \vee \text{last}' = \text{last} + 1$
- (V6) $\text{last} - \text{first} + 1 = \text{maxTrains} \rightarrow \text{last}' = \text{last}$
- (V7) $\text{last} - \text{first} + 1 > 0 \rightarrow \text{first}' = \text{first} \vee \text{first}' = \text{first} + 1$
- (V8) $\text{last} - \text{first} + 1 = 0 \rightarrow \text{first}' = \text{first},$
- (V9) $\text{last}' = \text{last} + 1 \rightarrow \text{pos}'(\text{last}') <_{\mathbb{R}} \text{pos}'(\text{last})$

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

Here, axioms V1–V4 are similar to F1–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:

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

`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 `pos0`) satisfy the strict monotonicity condition `Mon(pos0)`.
- (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 `Mon(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 with an index theory describing precedence of trains, with the two functions pos and pos' the following hold:

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

The set of formulae to be proved unsatisfiable w.r.t. \mathcal{T} involves the axioms \mathcal{K} and $\text{Mon}(\text{pos})$, containing universally quantified variables of sort i . Only $\neg \text{Mon}(\text{pos}')$ 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, 4], 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 $\text{pos}_0, \text{pos}_1, \dots, \text{pos}_k$, and – provided that $\mathcal{K}(\text{pos}_{i-1}, \text{pos}_i)$ is defined such that $\mathcal{K} = \mathcal{K}(\text{pos}, \text{pos}')$ – to show:

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

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 many-sorted combination \mathcal{T}_0 of real 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 $i \rightarrow \text{num}$,
- the extension \mathcal{T}_2 of \mathcal{T}_1 with a function pos' satisfying $\mathcal{K} \in \{\mathcal{K}_f, \mathcal{K}_v\}$.

We show that both extensions $\mathcal{T}_0 \subseteq \mathcal{T}_1 = \mathcal{T}_0 \cup \text{Mon}(\text{pos})$ 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^i (sort i) for reasoning about precedence between trains and the theory $\mathcal{T}_0^{\text{num}}$ of real numbers (sort num) for reasoning about distances between trains. As a convention, everywhere in what follows i, j, k denote variables of sort i and c, d denote variables of sort num .

We have several possibilities of choosing \mathcal{T}_0^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.

In what follows let \mathcal{T}_0^i be (a fragment of) integer arithmetic and $\mathcal{T}_0^{\text{num}}$ be the theory of real or rational numbers. In both these theories satisfiability of ground clauses is decidable.

Collision freeness as monotonicity. Let \mathcal{T}_0 be the (disjoint, many-sorted) combination of \mathcal{T}_0^i and $\mathcal{T}_0^{\text{num}}$. 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 $i \rightarrow \text{num}$ mapping train indices to the real numbers, which satisfies condition $\text{Mon}(\text{pos})$:

$$\text{Mon}(\text{pos}) \quad \forall i, j \quad (\text{first} \leq i < j \leq \text{last} \rightarrow \text{pos}(i) >_{\mathbb{R}} \text{pos}(j)),$$

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 of trains, we can assume that $\text{first} = 0$ and $\text{last} = n - 1$.)

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:

$$\forall i, j, k \quad (\text{first} \leq j \leq i \leq \text{last} \wedge i - j = k \rightarrow \text{pos}(j) - \text{pos}(i) \geq k * \text{LengthTrain}),$$

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 $i \times \text{num} \rightarrow \text{num}$ (multiplication of k with the constant LengthTrain in the formula above)³.

Let \mathcal{T}'_1 be the theory obtained by extending the combination \mathcal{T}'_0 of the theory of integers and reals with a function pos satisfying the axiom above.

Theorem 2 *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: We prove that every model of \mathcal{T}_1 in which the function pos is partially defined can be extended to a model in which pos is totally defined. 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 discreteness of the index theory (between two integers there are only finitely many integers). \square

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 3 *The following extensions are local theory extensions:*

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

³In the light of locality properties of such extensions (cf. Theorem 2), 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.

Proof: (1) Clauses in \mathcal{K}_f are flat and linear w.r.t. \mathbf{pos}' , so we again prove locality of the extension by showing that weak partial models can be extended to total ones. The proof proceeds by a case distinction. We use the fact that the left-hand sides of the implications in \mathcal{K}_f are mutually exclusive. (2) is proved similarly. \square

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 1, the following are equivalent:

- (1) $\mathcal{T}_0 \wedge \text{Mon}(\mathbf{pos}) \wedge \mathcal{K} \wedge \neg \text{Mon}(\mathbf{pos}') \models \perp$,
- (2) $\mathcal{T}_0 \wedge \text{Mon}(\mathbf{pos}) \wedge \mathcal{K}[G] \wedge G \models_w \perp$, where $G = \neg \text{Mon}(\mathbf{pos}')$,
- (3) $\mathcal{T}_0 \wedge \text{Mon}(\mathbf{pos}) \wedge \mathcal{K}_0 \wedge G_0 \wedge N_0(\mathbf{pos}') \models \perp$,

where $\mathcal{K}[G]$ consists of all instances of the rules in \mathcal{K} in which the terms starting with the function symbols \mathbf{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 $t = f(g_1, \dots, g_n)$ starting with extension functions $f \in \Sigma_1$, and $N_0(\mathbf{pos}')$ is the set of instances of the congruence axioms for \mathbf{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 \mathbf{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 $\mathcal{T}_0 \subseteq \mathcal{T}_1 = \mathcal{T}_0 \cup \text{Mon}(\mathbf{pos})$, the following are equivalent:

- (1) $\mathcal{T}_0 \wedge \text{Mon}(\mathbf{pos}) \wedge \mathcal{K}_0 \wedge G_0 \wedge N_0(\mathbf{pos}') \models \perp$,
- (2) $\mathcal{T}_0 \wedge \text{Mon}(\mathbf{pos})[G'] \wedge G' \models_w \perp$, where $G' = \mathcal{K}_0 \wedge G_0 \wedge N_0(\mathbf{pos}')$,
- (3) $\mathcal{T}_0 \wedge \text{Mon}(\mathbf{pos})_0 \wedge G'_0 \wedge N_0(\mathbf{pos}) \models \perp$,

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

This allows us to use hierarchical reasoning on properties of the system, i.e. to reduce the verification of system properties to deciding satisfiability of constraints in \mathcal{T}_0 . An advantage is that, after the reduction of the problem to a satisfiability problem in the base theory, one can automatically determine which constraints on the parameters (e.g. $\Delta t, \min, \max, \dots$) guarantee truth of the invariant. This can be achieved, e.g. using quantifier elimination. The method is illustrated in Section 4.3; more details can be found in [5].

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 $1, 2, 3, \dots, k$ 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 “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

In this section we indicate how to apply hierarchical reasoning on the case study given in Section 3, Model 1. 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 \mathcal{T}_0 .

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

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

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

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

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:

$$(\text{Dom}) \quad \Delta t > 0 \quad \wedge \quad 0 \leq \min \quad \wedge \quad \min \leq \max$$

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

$$\begin{aligned} (\text{G}_0) \quad & 0 \leq k < n - 1 \quad \wedge \quad k' = k + 1 \quad \wedge \quad c_1 \leq_{\mathbb{R}} c_2 \\ (\text{F2}_0) \quad & (0 < k < n \wedge \text{pos}(k - 1) >_{\mathbb{R}} 0 \wedge \text{pos}(k - 1) - \text{pos}(k) \geq_{\mathbb{R}} l_{\text{alarm}} \\ & \rightarrow \text{pos}(k) + \Delta t * \min \leq_{\mathbb{R}} c_1 \leq_{\mathbb{R}} \text{pos}(k) + \Delta t * \max) \\ & (0 < k' < n \wedge \text{pos}(k' - 1) >_{\mathbb{R}} 0 \wedge \text{pos}(k' - 1) - \text{pos}(k') \geq_{\mathbb{R}} l_{\text{alarm}} \\ & \rightarrow \text{pos}(k') + \Delta t * \min \leq_{\mathbb{R}} c_2 \leq_{\mathbb{R}} \text{pos}(k') + \Delta t * \max) \end{aligned}$$

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 : $\mathcal{K}_{f_0} \wedge G_0 \wedge N_0$. 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 **pos**(k), **pos**(k'), **pos**(k''), **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:

$$\begin{aligned}
(G_0) \quad & k'' = k - 1 \quad \wedge \quad k''' = k' - 1 \\
(G_0) \quad & 0 \leq k < n - 1 \quad \wedge \quad k' = k + 1 \quad \wedge \quad c_1 \leq_{\mathbb{R}} c_2 \\
(GF2_0) \quad & 0 < k < n \quad \wedge \quad d_3 >_{\mathbb{R}} 0 \quad \wedge \quad d_3 - d_1 \geq_{\mathbb{R}} l_{\text{alarm}} \rightarrow d_1 + \Delta t * \min \leq_{\mathbb{R}} c_1 \leq_{\mathbb{R}} d_1 + \Delta t * \max \\
& 0 < k' < n \quad \wedge \quad d_4 >_{\mathbb{R}} 0 \quad \wedge \quad d_4 - d_2 \geq_{\mathbb{R}} l_{\text{alarm}} \rightarrow d_2 + \Delta t * \min \leq_{\mathbb{R}} c_2 \leq_{\mathbb{R}} d_2 + \Delta t * \max
\end{aligned}$$

and $\text{Mon}(\text{pos})[G']$. After making some simplifications we obtain the following set of (many-sorted) constraints:

$C_{\text{Definitions}}$	C_{Indices}	C_{Reals}	C_{Mixed}
$\text{pos}'(k) = c_1 \quad \text{pos}(k') = d_2$	$k'' = k - 1$	$c_1 \leq_{\mathbb{R}} c_2 \quad d_3 >_{\mathbb{R}} d_4$	(GF1 ₀)
$\text{pos}'(k') = c_2 \quad \text{pos}(k'') = d_3$	$k''' = k' - 1$	$d_1 >_{\mathbb{R}} d_2 \quad d_4 >_{\mathbb{R}} d_2$	(GF2 ₀)
$\text{pos}(k) = d_1 \quad \text{pos}(k''') = d_4$	$0 \leq k < n - 1$	$d_3 >_{\mathbb{R}} d_1 \quad d_1 = d_4$	(GF3 ₀)
	$k' = k + 1$	$d_3 >_{\mathbb{R}} d_2 \quad (\text{Dom})$	(GF4 ₀)

For checking the satisfiability of $C_{\text{Indices}} \wedge C_{\text{Reals}} \wedge C_{\text{Mixed}}$ 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, somewhat similar to $\text{DPLL}(\mathcal{T}_0)$, would be to use only branching on the literals containing terms of index sort – this reduces the verification problem to the problem of checking the satisfiability of a set of linear constraints over the reals.

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 $\mathcal{T}_0 \subseteq \mathcal{T}_1 \subseteq \dots \subseteq \mathcal{T}_i$, 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. The applicability of the method is however general: the challenge is, at the moment, to recognize classes of local theories occurring in various areas of application. The implementation of the procedure described here is in progress, the method is clearly easy to implement. At a different level, our results 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.

Acknowledgements. 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. 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] 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, October 2005.
- [3] H. Ganzinger, V. Sofronie-Stokkermans, and U. Waldmann. Modular proof systems for partial functions with weak equality. In D. Basin and M. Rusinowitch, editors, *Automated reasoning : 2nd Int. Joint Conference, IJCAR 2004*, LNAI 3097, pp. 168–182. Springer, 2004. An extended version will appear in *Information and Computation*.
- [4] S. Ghilardi. Model theoretic methods in combined constraint satisfiability. *Journal of Automated Reasoning*, 33(3–4):221–249, 2004.
- [5] S. Jacobs and V. Sofronie-Stokkermans. Applications of hierarchical reasoning in the verification of complex systems (extended version). Available online at <http://www.mpi-sb.mpg.de/~sofronie/papers/jacobs-sofronie-pdpar-extended.ps>
- [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. *CADE’2005: Int. Conf. on Automated Deduction*, LNCS 3632, pp. 219–234. Springer, 2005.
- [9] V. Sofronie-Stokkermans. Interpolation in local theory extensions. In *Proceedings of IJCAR 2006*, LNAI 4130, pp. 235–250. Springer, 2006.