Compositional model checking of interlocking systems for lines with multiple stations

Compositional Model Checking of Interlocking Systems for Lines with Multiple Stations

Hugo Daniel Macedo[1,2(✉)], Alessandro Fantechi[1,3], and Anne E. Haxthausen[1]

[1] DTU Compute, Technical University of Denmark, Lyngby, Denmark
aeha@dtu.dk
[2] Department of Engineering, Aarhus University, Aarhus, Denmark
hdm@eng.au.dk
[3] DINFO, University of Florence, Firenze, Italy
alessandro.fantechi@unifi.it

Abstract. In the railway domain safety is guaranteed by an interlocking system which translates operational decisions into commands leading to field operations. Such a system is safety critical and demands thorough formal verification during its development process. Within this context, our work has focused on the extension of a compositional model checking approach to formally verify interlocking system models for lines with multiple stations. The idea of the approach is to decompose a model of the interlocking system by applying cuts at the network modelling level. The paper introduces an alternative cut (the linear cut) to a previously proposed cut (border cut). Powered with the linear cut, the model checking approach is then applied to the verification of an interlocking system controlling a real-world multiple station line.

Keywords: Railway interlocking · Compositional verification · Model checking

1 Introduction

A *railway* is a mechanised means of mass movement where diverse vehicles take paths on a shared space/network of tracks. Its main feature is guidance by mechanical contact of wheels on rails. Switch points are introduced to dynamically change the network topology allowing a vehicle to change tracks. Another distinctive feature is the poor braking response time given the physical properties of wheel on rail rolling friction. Such features impose hard restrictions on traffic, vehicle movements, and network configuration.

To regulate traffic, a railway *signalling system* [14] is deployed as an information processing/transmission control loop. The system monitors the status of

H.D. Macedo and A.E. Haxthausen—The authors' research, conducted at DTU Compute, was funded by the RobustRailS project granted by Innovation Fund Denmark.

A. Fantechi—The author's research was funded by Villum Fonden.

C. Barrett et al. (Eds.): NFM 2017, LNCS 10227, pp. 146–162, 2017.
DOI: 10.1007/978-3-319-57288-8_11

vehicles and track elements issuing network re-configuration and vehicle dispatch commands. The usually deployed monitoring scheme assumes that the network under control is divided into sections with train detection equipment and the existence of additional track side elements such as signals. The status (occupied or clear) of train detection sections, position of points, and configuration of track side elements (e.g. the setting of signals) is relayed to the control system. Issued decisions are then transmitted back to each element affecting its configuration (e.g.: issuing a change in point position) and vehicle movements (e.g.: sending dispatch commands to trains through signals).

The technology/operation mode of signalling systems ranges from basic human communication, for instance telecommunications between stakeholders (human controllers, station masters, and vehicle operators), to advanced automation where computers are responsible for the whole control loop. Usually the different systems are used heterogeneously through a network. Several of the recent railway disasters were due to signalling system failures[1] in networks lacking *automated* control.

Automated systems require railway engineers/architects to define the appropriate operation requirements, for instance in the form of routes: each prescribing the path and the required network configuration for safe train traversal along that path. When the system issues a dispatch route command, the network must be reconfigured to comply with such requirements. In addition, the system must ensure the required configuration is maintained during the traversal. And above all, the command must not lead to a safety violation. For that purpose an *interlocking system* takes the responsibility of safely transform each dispatch decision into the control commands that must be executed before a proceed command to a train is issued.

Such responsibility demands for standards in the development of the software controlling interlocking systems. The standard CENELEC 50128 [1] labels such software with the highest safety integrity level (SIL4), and highly recommends the usage of formal methods and formal verification in its development process. However, full formal verification of interlocking systems demands heavy if not infeasible computational resources[2], a phenomenon known as the state explosion problem. The pioneering research in model checking and in applying model checking to the domain of railways [3–5,7,9,20] has developed techniques allowing the verification of models of the interlocking systems controlling larger and highly-complex networks. For example, abstraction techniques can be applied at the domain modelling level before the model checking is performed [9]. Other very efficient techniques applied for real world railways are bounded model checking [8] and k-induction [19]. The state explosion problem can also be tamed using techniques that allow a compositional approach to the model checking task [10]: the model checker must prove that assumptions imply the guarantees of each

[1] For instance the July 2016 rural Southern-Italy head-on train collision would have been prevented if automated train detection equipment had been in place.

[2] A model of the interlocking for a fairly simple network may lead to the potential inspection of an astronomical number of states (e.g. in the order of 10^{51} [11]).

contract of the component. The authors report that this technique allowed the verification of a real world station.

Pursuing the same goal, in a previous work [11] we described a compositional approach to the verification of safety properties of models of interlocking systems controlling lines with multiple stations. The approach was developed in the context of the RobustRailS research project[3] extending an automated method for the formal verification of the new Danish interlocking systems [17–19]. The idea in our previous work was based on the observation that decomposing a network at specific points which satisfy a given topological configuration (called *border cut*, see Fig. 1) generates sub-models corresponding to a complete partition of disjoint, connected components of the state space. It is therefore straightforward to combine the results of checking each sub-model to compute the result of checking the monolithic model. This is the case as the routes that can be set inside one sub-model are completely independent from those in the other sub-model.

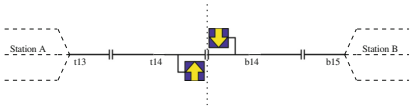

Fig. 1. Border cut dividing the network topology into two parts.

We have then realised that the *border cut* configuration does not occur in some real world networks, but instead a similar configuration (that we call *linear cut*, see Sect. 3.1), in which the routes of the two sub-models partially overlap, is frequent. Inspired by the already cited compositional approach [10], where a similar route overlap is taken into account, we have modified our compositional approach to consider *linear cut* configurations as the points at which to cut a network into sub-models. This requires a finer analysis of the interferences between sub-models, but again we show that checking each sub-model allows the result of checking the monolithic model to be computed, with significant verification time savings.

The exposition of our results is structured as follows: in Sect. 2 we recall some principles of railway interlocking systems and present the RobustRailS verification method and toolkit on the top of which we have built our compositional approach; in Sect. 3 we present our approach using a divide-and-conquer strategy: we introduce the *linear cut* and explain how our method first uses this to divide a network into sub-networks, then generates sub-models and finally conquer the model checking results for these. The soundness and completeness of the approach is proved in Sect. 4, and in Sect. 5 we report on the results given

[3] In Denmark, in the years 2009–2021, new interlocking systems that are compatible with the standardised European Train Control System (ETCS) Level 2 [2] will be deployed in the entire country within the context of the Danish Signalling Programme. In the context of the RobustRailS project accompanying the signalling programme on a scientific level, the approach is applied to the new systems.

by the application of our compositional approach to a typical example and to a real-world line that nearly reached the capacity bounds of the adopted tools when proved as a whole. In both cases the results show that significant gains in verification effort can be achieved. Section 6 summarises the achieved results and discusses possible future extensions and improvements of the work presented here, especially in the direction of addressing interlocking systems that control large stations.

2 The New Danish Route-Based Interlocking Systems

In this section we introduce briefly the new Danish interlocking systems and the domain terminology. The subsequent Sect. 2.1 explains different components of a specification of an interlocking system which is compatible with ERTMS/ETCS Level 2 [2], and Sect. 2.2 explains how the safety properties are verified.

2.1 Specification of Interlocking Systems

The specification of a given route-based interlocking system $I = (N, R)$ consists of two components: (N) a railway network, and (R) an interlocking table.

Railway Networks. A railway network in ETCS Level 2 consists of a number of track and track-side elements of different types[4]: linear sections, points, and marker boards. Figure 2 shows an example layout of a railway network having six linear sections (`b10`,`t10`,`t12`,`t14`,`t20`,`b14`), two points (`t11`,`t13`), and eight marker boards (`mb10`, ..., `mb21`). These terms, and their functionality within the railway network, will be explained in more detail in the next paragraphs.

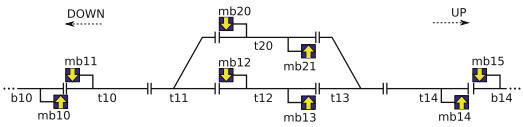

Fig. 2. An example railway network layout.

A *linear section* is a section with up to two neighbours: one in the *up* end, and one in the *down* end. For example, the linear section `t12` in Fig. 2 has `t13` and `t11` as neighbours at its up end and down end, respectively. In Danish railway's terminology, *up* and *down* denote the directions in which the distance from a reference location is *increasing* and *decreasing*, respectively. The reference location is the same for both up and down, e.g., an end of a line. For simplicity, in the examples and figures in the rest of this article, the *up* (*down*) direction is assumed to be the left-to-right (right-to-left) direction.

[4] Here we only show types that are relevant for the work presented in this article.

A *point* can have up to three neighbours: one at the *stem*, one at the *plus* end, and one at the *minus* end, e.g., point `t11` in Fig. 2 has `t10`, `t12`, and `t20` as neighbours at its stem, plus, and minus ends, respectively. The ends of a point are named so that the *stem* and *plus* ends form the straight (main) path, and the *stem* and *minus* ends form the branching (siding) path. A point can be switched between two positions: PLUS and MINUS. When a point is in the PLUS (MINUS) position, its *stem* end is connected to its *plus* (*minus*) end, thus traffic can run from its *stem* end to its *plus* (*minus*) end and vice versa. It is not possible for traffic to run from *plus* end to *minus* end and vice versa.

Linear sections and points are collectively called (train detection) sections, as they are provided with train detection equipment used by the interlocking system to detect the presence of trains. Note that sections are bidirectional, i.e., trains are allowed to travel in both directions (but not at the same time).

Along each linear section, up to two *marker boards* (one for each direction) can be installed. A marker board can only be seen in one direction and is used as reference location (for the start and end of routes) for trains going in that direction. For example, in Fig. 2, marker board `mb13` is installed along section `t12` for travel direction up. Contrary to legacy systems, there are no physical signals in ETCS Level 2, but interlocking systems have a *virtual signal* associated with each marker board. Virtual signals play a similar role as physical signals in legacy systems: a virtual signal can be OPEN or CLOSED, respectively, allowing or disallowing traffic to pass the associated marker board. However, trains (more precisely train drivers) do not see the virtual signals, as opposed to physical signals. Instead, the aspect of virtual signals (OPEN or CLOSED) is communicated to the onboard computer in the train via a radio network. For simplicity, the terms *virtual signals*, *signals*, and *marker boards* are used interchangeably throughout this paper.

Interlocking Tables. An interlocking system constantly monitors the status of track-side elements, and sets them to appropriate states in order to allow trains travelling safely through the railway network under control. The new Danish interlocking systems are route-based. A *route* is a path from a *source* signal to a *destination* signal in the given railway network. A route is called an *elementary route* if there are no signals that are located between its source signal and its destination signal, and that are intended for the same direction as the route.

In railway signalling terminology, *setting* a route denotes the process of allocating the resources – i.e., sections, points, and signals – for the route, and then *locking* it exclusively for only one train when the resources are allocated.

An *interlocking table* specifies the elementary routes in the given railway network and the conditions for setting these routes. The specification of a route r and conditions for setting r include the following information, that will be needed while verifying the expected properties:

- $src(r)$ – the source signal of r,
- $dst(r)$ – the destination signal of r,
- $path(r)$ – the list of sections constituting r's path from $src(r)$ to $dst(r)$,

- $overlap(r)$ – a list of the sections in r's *overlap*[5], i.e., the buffer space after $dst(r)$ that would be used in case trains overshoot the route's path,
- $points(r)$ – a map from points[6] used by r to their required positions,
- $signals(r)$ – a set of protecting signals used for flank or front protection [14] for the route, and
- $conflicts(r)$ – a set of conflicting routes which must not be set while r is set.

Table 1 shows an excerpt of an interlocking table for the network shown in Fig. 2. Each row of the table corresponds to a route specification. The column names indicate the information of the route specifications that these columns contain. As can be seen, one of the routes has id `1a`, goes from `mb10` to `mb13` via three sections `t10`, `t11` and `t12` on its path, and has no overlap. It requires point `t11` (on its path) to be in PLUS position, and point `t13` (outside its path) to be in MINUS position (as a protecting point). The route has `mb11`, `mb12` and `mb20` as protecting signals, and it is in conflict with routes `1b`, `2a`, `2b`, `3`, `4`, `5a`, `5b`, `6b`, and `7`.

Table 1. Excerpt of the interlocking table for the network of Fig. 2. The overlap column is omitted as it is empty for all routes. (`p` = PLUS, `m` = MINUS)

Id	*src*	*dst*	*path*	*points*	*signals*	*conflicts*
`1a`	`mb10`	`mb13`	`t10;t11;t12`	`t11:p;t13:m`	`mb11;mb12;mb20`	`1b;2a;2b;3;4;5a;5b;6b;7`
..	...	...	...	...	...	...
`7`	`mb20`	`mb11`	`t11;t10`	`t11:m`	`mb10;mb12`	`1a;1b;2a;2b;3;5b;6a`

2.2 The RobustRailS Verification Method and Toolkit

This section describes shortly the RobustRailS verification method and toolkit that we use as verification technology. For detailed information, see [6,16–19].

The method for modelling and verifying railway interlocking systems is a combination of formal methods and a domain-specific language (DSL) to express network diagrams and interlocking tables. According to this, a toolkit consisting of the following components is provided.

- *An editor and static checker* [6] for editing and checking that a DSL specification $I = (N, R)$ (describing an interlocking system) follows certain well-formedness rules.

[5] An overlap section is needed when, for the short distance of a marker board to the end of the section, there is the concrete danger that a braking train stops after the end of the section, e.g. in adverse atmospheric conditions.

[6] These points include points in the path and overlap, and points used for *flank* and *front protection*. Sometimes it is required to protect tracks occupied by a train from another train not succeeding to brake in due space. For details about flank and front protection, see [14].

- The bounded *model checker* of RT-Tester [12,15] which we use for performing k-induction proofs as explained in [19].
- *Generators* transforming a DSL specification $I = (N, R)$ of an interlocking model into inputs to the model checker:
  - a behavioural model m_I (a Kripke structure) of the interlocking system and its environment, defining the state space and possible state transitions, and
  - the required safety properties given as a state invariant (expressing that there are no hazards like train collisions). The invariant is a conjunction of high-level safety properties $\mathcal{H}$ over the variables of the interlocking system model. An $\mathcal{H}$-property is satisfied by an interlocking specification I, written as $\mathcal{H}(I)$, if it is valid in the model of the interlocking system m_I. $\mathcal{H}(I)$ is valid in the model m_I can be written as $m_I \models \forall e : E_N \cdot \mathcal{P}_{\mathcal{H}}(e)$, where E_N is either the subset of all linear sections or all point sections in N and $\mathcal{P}_{\mathcal{H}}(e)$ is a section property related to $\mathcal{H}$.

  For details of the models and properties, see [19].

The tools can be used to verify the design of an interlocking system in the following steps:

1. A DSL specification of the configuration data (a network layout and its corresponding interlocking table) is constructed in the following order:
   (a) first the network layout,
   (b) and then the interlocking table (this is either done manually or generated automatically from the network layout).
2. The static checker verifies whether the configuration data is statically well-formed according to the static semantics [18] of the DSL.
3. The generators instantiate a generic behavioural model and generic safety properties with the well-formed configuration data to generate the model input of the model checker and the safety properties.
4. The generated model instance is then checked against the generated properties by the bounded model checker performing a k-induction proof.

The static checking in step (2) is intended to catch errors in the network layout and interlocking table, while the model checking in step (4) is intended to catch safety violations in the control algorithm of the instantiated model.

The tool-chain associated with the method has been implemented using the RT-tester framework [12,15]. The bounded model checker in RT-tester uses the SONOLAR SMT solver [13] to compute counterexamples showing the violations of the base case or induction step. Using this SMT solver rather than a SAT solver allowed us to use very efficient bit-vector operations.

As proof technique in step 4, we used k-induction as this was the most promising (cf. the comparison with other techniques in [19]), however, our compositional method could also be used in combination with other proof techniques.

3 Method

We now proceed to describe the details of how we use the locality features of railway networks to verify large interlocking systems in a compositional manner. The idea is to decompose the model into smaller models that are separately verified for safety properties, and to show that under given conditions such separate verifications are enough to guarantee that the whole network satisfies the safety properties as well. We show that a multi-station interlocking system satisfies such conditions if a suitable (and natural) divide strategy is applied. The strategy provides a completely automated method to verify this class of interlocking systems.

3.1 Linear Cuts on Multiple Station Lines

The typical pattern of a railway is a line connecting multiple stations. Without loss of generality, we can consider a line, denoted $\mathcal{A} \vdash\!\dashv \mathcal{B}$, corresponding to a *network* diagram consisting of two stations denoted by A and B, interconnected by one or several linear sections. More complex multi-station layouts can be obtained by concatenation of such elementary lines.

To divide multiple station lines we search for an interface $\mathcal{I}$, which we define as a linear section[7] with an up and down marker board subject to certain conditions described further below. A cut is then applied producing two sub-networks:

- The $\mathcal{A}$ network defined as the A station and the interface $\mathcal{I}$. An entry marker board is added on the up (B) side of this network.
- The $\mathcal{B}$ network defined as the B station and the interface $\mathcal{I}$. An entry marker board is added on the down (A) side of this network.

With the required configuration of marker boards on the interface and the addition of entry marker boards, the two sub-networks fulfil the required marker board configuration at borders of a railway network.

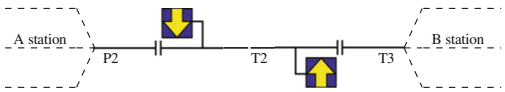

Fig. 3. The multiple station line pattern where sections T2 and T3 connect two stations A and B.

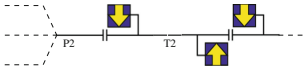

Fig. 4. Resulting $\mathcal{A}$ network.

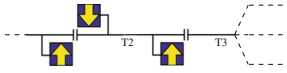

Fig. 5. Resulting $\mathcal{B}$ network.

[7] The extension of the interface to divide networks with parallel tracks is straightforward and defines the interface as a set $\mathcal{I}$ of linear sections dividing a network into disjoint and valid connected sub-networks.

For example in Fig. 3 we depict a highlight of a line network diagram in which T2 connects two stations A and B. In the example A contains element P2 and its down neighbours and B contains elements T3 and its up neighbours. Linear section T2 configures a candidate to a linear cut, which results in the two networks illustrated in Figs. 4 and 5, where the linear section (T2) is kept in both as it defines the interface $\mathcal{I}$.

To guarantee that the compositional approach (to be described in next subsection) is sound, the interface $\mathcal{I}$ must satisfy the following *linear cut conditions (LCCs)*:

1. there is an up marker board on the upper part of the interface section $\mathcal{I}$ and a down marker board on the down part;
2. the two networks ($\mathcal{A}$ and $\mathcal{B}$) resulting from the cut described above must only have $\mathcal{I}$ in common;
3. no *flank/front protection* requirements for routes in the up (down) sub-network $\mathcal{B}$ ($\mathcal{A}$) depends on elements outside $\mathcal{B}$ ($\mathcal{A}$), except for routes in down (up) direction with destination marker board mounted in $\mathcal{I}$ (i.e. routes that end at the entrance of the A (B) station).

3.2 A Compositional Model Checking Approach

In the division process a network is inspected in search for regions that present candidate patterns to be cut, that is, linear sections of the form T2 of Fig. 3. The search is then recursively applied to the created sub-networks, until either no more suitable cut points can be found or the sub-networks produced are already sufficiently small.

The linear cut allows to automate the compositional verification of multi-station interlocking systems by dividing the network in sub-networks by means of four steps:

1. Search the network for suitable interfaces satisfying the LCCs. For each interface instantiate the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ pattern and divide recursively the network into sub-networks as described in Subsect. 3.1.
2. For each of the resulting sub-networks N_i, complete the specification of a sub-interlocking system using the interlocking table generator mentioned in item 1 of Sect. 2.2. The resulting specifications are called the N_i interlocking specifications.
3. Statically check each of the resulting N_i specifications and generate the models m_{N_i} (called the N_i models) and properties to be verified using the checker and generator mentioned in item 2 and item 3, respectively, of Sect. 2.2.
4. Verify the m_{N_i} models following item 4 of Sect. 2.2.

4 Soundness and Completeness of the Approach

To prove that the decomposition approach is *sound* and *complete* one needs to show that the result of checking any of the high-level safety properties $\mathcal{H}$

(as defined in Subsect. 2.2) for the $\mathcal{A}$ and $\mathcal{B}$ sub-models implies the result of checking the same property $\mathcal{H}$ for the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ monolithic model, and vice versa. (The extension to more than one sub-model is then straightforward). First we prove soundness and then completeness.

4.1 Soundness

Soundness can be rephrased in terms of $\mathcal{H}$'s related invariant $\mathcal{P}_{\mathcal{H}}$. If the invariant holds for every section in the $\mathcal{A}$ interlocking specification and for every section in the $\mathcal{B}$ interlocking specification we can conclude the whole interlocking specification $\mathcal{A} \vdash\!\dashv \mathcal{B}$ satisfies $\mathcal{H}$, meaning its related invariant $\mathcal{P}_{\mathcal{H}}$ holds for every section in the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ interlocking specification.

Given that $\mathcal{H}$-properties are universal quantifications over the sets of linear/point sections[8], a natural strategy to produce such a proof is to decompose the property in terms of the disjoint sets of sections defining the A and B stations, and the interface $\mathcal{I}$. That is, the $\mathcal{H}$ related property $\mathcal{P}_{\mathcal{H}}$ holds for every section in the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ network, if $\mathcal{P}_{\mathcal{H}}$ holds for every section of the network containing the A station, for the interface section $\mathcal{I}$, and for every section of the network containing the B station. In mathematical terms, if we denote by $E_{\mathcal{A}}$ the set of sections of an interlocking specification $\mathcal{A}$, it corresponds to rewrite the formulation of the satisfiability of $\mathcal{H}$ by the model of $\mathcal{A} \vdash\!\dashv \mathcal{B}$, i.e. $m_{\mathcal{A}\vdash\!\dashv\mathcal{B}} \models \forall e : E_{\mathcal{A}\vdash\!\dashv\mathcal{B}} \cdot \mathcal{P}_{\mathcal{H}}(e)$, into:

$$m_{\mathcal{A}\vdash\!\dashv\mathcal{B}} \models (\forall e : A \cdot \mathcal{P}_{\mathcal{H}}(e)) \wedge \mathcal{P}_{\mathcal{H}}(\mathcal{I}) \wedge (\forall e : B \cdot \mathcal{P}_{\mathcal{H}}(e)) \tag{1}$$

The aforementioned rewrite leads one to decompose the proof into three lemmas. The first two relate the local properties satisfied by $\mathcal{A} \vdash\!\dashv \mathcal{B}$ and $\mathcal{A}$ and similarly by $\mathcal{A} \vdash\!\dashv \mathcal{B}$ and $\mathcal{B}$.

Lemma 1. *Consider a line interlocking specification with A and B stations satisfying the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ pattern, the $\mathcal{A}$ and $\mathcal{B}$ interlocking specifications resulting from the application of a linear cut, a high-level safety property $\mathcal{H}$ and its related invariant $\mathcal{P}_{\mathcal{H}}$. We relate the outcome of evaluating $\mathcal{H}(\mathcal{A})$ and $\mathcal{H}(\mathcal{A} \vdash\!\dashv \mathcal{B})$ through the following implication:*

$$m_{\mathcal{A}\vdash\!\dashv\mathcal{B}} \models \forall e : A \cdot \mathcal{P}_{\mathcal{H}}(e) \Leftarrow m_{\mathcal{A}} \models \forall e : A \cdot \mathcal{P}_{\mathcal{H}}(e)$$

Proof. By contradiction. Let us assume that in $m_{\mathcal{A}}$ the property $\mathcal{P}_{\mathcal{H}}$ holds for every section in A and there is a section e in A such that $\mathcal{P}_{\mathcal{H}}(e)$ does not hold in the $m_{\mathcal{A}\vdash\!\dashv\mathcal{B}}$ model. Then, as detailed in [19], there is a state s of $m_{\mathcal{A}\vdash\!\dashv\mathcal{B}}$, where $\mathcal{P}_{\mathcal{H}}(e)$ is false, reachable from the initial state by a sequence of transitions (trace) that we denote as t^*. The state s is characterised by an assignment of values to a vector of variables referring to the elements (sections, signals etc.) of the network. Due to the linear cut definition, such variables refer to elements that

[8] In the following, for simplicity, we just quantify over the whole set of sections of a network, intending that we are referring either only to point or only to linear sections according to the nature of $\mathcal{H}$.

are in the $\mathcal{A}$ or in the $\mathcal{B}$ network. Any transition in t^* changes such assignments: following t^* we can find in $m_{\mathcal{A}}$ a corresponding trace $t^{*\prime}$ that makes the same changes to the variables in the state vector of $m_{\mathcal{A}}$, skipping those transitions in t^* that do not change variables in $m_{\mathcal{A}}$. The trace $t^{*\prime}$ therefore ends in a reachable state s' in which the assignments to variables in $m_{\mathcal{A}}$ are the same of those of s, and hence $\mathcal{P}_{\mathcal{H}}(e)$ does not hold, contradicting the hypothesis.

Lemma 2. *The dual case of Lemma 1. Given by substitution of the interlocking specification $\mathcal{A}$ by $\mathcal{B}$, $\mathcal{H}(\mathcal{A})$ by $\mathcal{H}(\mathcal{B})$ and A by B.*

The two lemmas above allow us to transfer checking results on the sections of the two stations A and B to the check of the whole line; however, we still miss the contribution of the interface section, which is copied in both the $\mathcal{A}$ and $\mathcal{B}$ networks. The next lemma has this purpose.

Lemma 3. *(Interfacing lemma) Consider the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ interlocking specification, the $\mathcal{A}$ interlocking specification and the $\mathcal{B}$ interlocking specification resulting from applying a linear cut, a high-level safety property $\mathcal{H}$ and its related invariant $\mathcal{P}_{\mathcal{H}}$. For the interface $\mathcal{I} \in E_{\mathcal{A}} \cap E_{\mathcal{B}}$ we have:*

$$m_{\mathcal{A}\vdash\!\dashv\mathcal{B}} \models \mathcal{P}_{\mathcal{H}}(\mathcal{I}) \Leftarrow m_{\mathcal{A}} \models \mathcal{P}_{\mathcal{H}}(\mathcal{I}) \wedge m_{\mathcal{B}} \models \mathcal{P}_{\mathcal{H}}(\mathcal{I})$$

Proof. By contradiction. Assume $\mathcal{P}_{\mathcal{H}}(\mathcal{I})$ is true in both the $m_{\mathcal{A}}$ and $m_{\mathcal{B}}$ models, but false in the $m_{\mathcal{A}\vdash\!\dashv\mathcal{B}}$ model. Furthermore assume s is the state of $m_{\mathcal{A}\vdash\!\dashv\mathcal{B}}$ falsifying $\mathcal{P}_{\mathcal{H}}(\mathcal{I})$. Thus, there is a trace t^* in $m_{\mathcal{A}\vdash\!\dashv\mathcal{B}}$ leading from the model's initial state to the variable assignment in s. Similarly to what said for Lemma 1 it is then possible to form a trace $t^{*\prime}$ in $m_{\mathcal{A}}$ and a trace $t^{*\prime\prime}$ in $m_{\mathcal{B}}$ from the initial states to two states s' and s'' such that the state vector has an assignment falsifying $\mathcal{P}_{\mathcal{H}}(\mathcal{I})$ in s' or s''. Thus arriving at a contradiction.

Given the proofs of Lemmas 1, 2, and 3, one is in the position to relate the result of the monolithic checking of the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ interlocking specification with the results of the compositional approach in which the $\mathcal{A}$ and $\mathcal{B}$ interlocking specifications are checked.

Theorem 1. *(Soundness) Consider the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ interlocking specification, the $\mathcal{A}$ and $\mathcal{B}$ interlocking specifications resulting from the application of a linear cut, and a high-level safety property $\mathcal{H}$. Then*

$$\mathcal{H}(\mathcal{A} \vdash\!\dashv \mathcal{B}) \Leftarrow \mathcal{H}(\mathcal{A}) \wedge \mathcal{H}(\mathcal{B})$$

which means that if $\mathcal{H}$ is satisfied by $\mathcal{A}$ and by $\mathcal{B}$, one can conclude that it is satisfied by $\mathcal{A} \vdash\!\dashv \mathcal{B}$.

Proof. Assume $\mathcal{H}(\mathcal{A}) \wedge \mathcal{H}(\mathcal{B})$ is true, our goal is to prove $\mathcal{H}(\mathcal{A} \vdash\!\dashv \mathcal{B})$, i.e. (cf. Formula (1)): $m_{\mathcal{A}\vdash\!\dashv\mathcal{B}} \models (\forall e : A \cdot \mathcal{P}_{\mathcal{H}}(e)) \wedge \mathcal{P}_{\mathcal{H}}(\mathcal{I}) \wedge (\forall e : B \cdot \mathcal{P}_{\mathcal{H}}(e))$ which is equivalent to:

$$m_{\mathcal{A}\vdash\!\dashv\mathcal{B}} \models (\forall e : A \cdot \mathcal{P}_{\mathcal{H}}(e)) \wedge m_{\mathcal{A}\vdash\!\dashv\mathcal{B}} \models \mathcal{P}_{\mathcal{H}}(\mathcal{I}) \wedge m_{\mathcal{A}\vdash\!\dashv\mathcal{B}} \models (\forall e : B \cdot \mathcal{P}_{\mathcal{H}}(e))$$

Applying Lemma 1, Lemma 2, and Lemma 3, one obtains:

$$m_{\mathcal{A}} \models (\forall e : A \cdot \mathcal{P}_{\mathcal{H}}(e)) \wedge m_{\mathcal{A}} \models \mathcal{P}_{\mathcal{H}}(\mathcal{I}) \wedge m_{\mathcal{B}} \models \mathcal{P}_{\mathcal{H}}(\mathcal{I}) \wedge m_{\mathcal{B}} \models (\forall e : B \cdot \mathcal{P}_{\mathcal{H}}(e))$$

which is equivalent to: $\mathcal{H}(\mathcal{A}) \wedge \mathcal{H}(\mathcal{B})$.

4.2 Completeness

The following theorem states that the method is complete.

Theorem 2. *(Completeness) Consider the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ interlocking specification, the $\mathcal{A}$ and $\mathcal{B}$ interlocking specifications resulting from the application of a linear cut at an interface $\mathcal{I}$, and a high-level safety property $\mathcal{H}$. Assume that for each internal section b of $\mathcal{A} \vdash\!\dashv \mathcal{B}$ which appears as a border section in one of the subnetworks $\mathcal{A}/\mathcal{B}$ (i.e. b is an $\mathcal{B}/\mathcal{A}$ neighbour to $\mathcal{I}$), there exists a finite trace prefix in $m_{\mathcal{A}\vdash\!\dashv\mathcal{B}}$ leading a train to b from some outer border of the $\mathcal{B}/\mathcal{A}$ network without changing any of the variables that only exist in $m_{\mathcal{A}}/m_{\mathcal{B}}$. Then*

$$\mathcal{H}(\mathcal{A} \vdash\!\dashv \mathcal{B}) \Rightarrow \mathcal{H}(\mathcal{A}) \wedge \mathcal{H}(\mathcal{B})$$

which means that if $\mathcal{H}$ is dissatisfied by $\mathcal{A}$ or by $\mathcal{B}$, one can conclude that it is dissatisfied by $\mathcal{A} \vdash\!\dashv \mathcal{B}$.

Proof. Assume that $\mathcal{H}$ is dissatisfied by $\mathcal{A}/\mathcal{B}$, and let t be the associated counter example (trace). t can now be lifted to a counter example $t_{\mathcal{A}\vdash\!\dashv\mathcal{B}}$ in $m_{\mathcal{A}\vdash\!\dashv\mathcal{B}}$ by first extending the states of t with the additional variables of $m_{\mathcal{A}\vdash\!\dashv\mathcal{B}}$ mapped to their initial states, and then, if the t trace involves a train entering $\mathcal{I}$ from the border b at the $\mathcal{B}/\mathcal{A}$ side of $\mathcal{I}$, this extended trace should be preceded by a trace prefix from $m_{\mathcal{A}\vdash\!\dashv\mathcal{B}}$ leading the train to b from some outer border of $\mathcal{B}/\mathcal{A}$ without changing any of the variables that only exist in $m_{\mathcal{A}}/m_{\mathcal{B}}$.

5 Experiments

In this section we present the results of applying our decomposition approach to an invented line ($\mathcal{A} \vdash\!\dashv \mathcal{B}$) with two stations and to a real world case study with eight stations. Both lines exhibit the pattern of a line with multiple stations which cannot be divided using the *border cut* defined in our previous work [11].

5.1 Experimental Approach

For each of the case studies, we put the method described in Sect. 3.2 in practice by first obtaining sub-networks (in XML format) according to the divide strategy. Then for each sub-network, we use the RobustRailS verification tool [17–19] to generate a model instance and safety properties, and then to verify that the generated safety properties hold in the model.

We also use the RobustRailS verification tool to monolithically verify the railway network (without decomposing it) such that we can compare verification metrics for the compositional approach with verification metrics for the monolithic approach.

While verifying each instance we measure (in seconds) the real time taken to obtain the verification result and what was the total memory (in MB) used by the verification tool. In addition we collect some statistics about the network and model instances as presented in Tables 2 and 3. Such statistics provide a basis for complexity comparison and include: the number of linear and point sections, the number of marker boards (signals), routes, and the potential state space dimension (in logarithmic scale).

All the experiments for both case studies have been performed on a machine with an Intel(R) Xeon(R) CPU E5-1650 @ 3.6 GHz, 125 GB RAM, and running Linux 4.4.0–47.x86_64 kernel.

5.2 Two Stations Case Study

Let us consider as an example the railway line of Fig. 6 denoted $\mathcal{A} \vdash\!\dashv \mathcal{B}$. In it we find two stations: the set of elements $A = \{T1, P1, A1, A2, P2\}$ defines the A station, whereas the set $B = \{T3, P3, B1, B2, P4, T4\}$ defines the B station. The linear section T2 connects A and B.

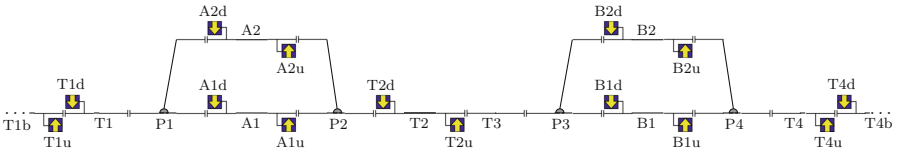

Fig. 6. $\mathcal{A} \vdash\!\dashv \mathcal{B}$ Network

The RobustRailS tool allows the automatic generation of interlocking tables from a given network layout, and for the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ network it generates 24 routes. A thorough inspection of the table shows that routes can be categorised into three blocks, partitioning the network into two disjoint networks and a common interface (linear section T2). The inspection of the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ route table reveals that it makes sense to divide the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ network into two networks, choosing the linear section T2 as an interface between a network containing the A station and a network containing the B station.

As planned, we have verified the model both compositionally and monolithically; Table 2 shows the verification metrics, first separately for the $\mathcal{A}$ and $\mathcal{B}$ networks. The metrics for the compositional analysis ($\mathcal{A} + \mathcal{B}$) are obtained by summing the corresponding metrics for the networks, except for the state space and the memory usage, which are calculated as the respective maximum between the two sub-networks. The table also shows the verification metrics for the monolithic analysis of the network ($\mathcal{A} \vdash\!\dashv \mathcal{B}$).

Table 2. Verification metrics for the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ case study.

	Linears	Points	Signals	Routes	$log_{10}(\|S\|)$	Time	Memory
$\mathcal{A}$	6	2	9	13	38	10	186
$\mathcal{B}$	7	2	9	13	41	16	234
$\mathcal{A} + \mathcal{B}$	13	4	18	26	41	26	234
$\mathcal{A} \vdash\!\dashv \mathcal{B}$	10	4	14	24	68	68	556

In all cases the verification tool succeeded to verify the safety properties. As it can be observed the verification time and memory usage of the compositional analysis ($\mathcal{A} + \mathcal{B}$) is, as expected, much better than for the monolithic analysis of ($\mathcal{A} \vdash\!\dashv \mathcal{B}$): The verification time is approximately three times faster and the memory usage (234 MB) is more than halved.

Moreover, if the verification for the $\mathcal{A}$ and $\mathcal{B}$ networks were run in parallel, our compositional approach would achieve a running time of just 16 s. Even though memory consumption would increase in this case, the parallelisation would still use less memory resources (the sum of individual memory usages: 420 MB) than the monolithic case (556 MB).

5.3 EDL: The Real World Case Study

The **EDL** is the first regional line in Denmark to be commissioned in the Danish Signalling Programme. The line spreads over 55 km from the station in Roskilde to Næstved's station, with 8 small to medium sized stations, and the statistics shown in Table 3 gives insight into its composition.

With the definition of the linear cut it is now directly possible to cut the **EDL** network into eight sub-networks, each corresponding to an **EDL** station. Six of the sub-networks (Gadstrup, Havdrup, Herfølge, Tureby, Haslev, and Holme-Olstrup) are of fairly similar complexity, while two (L. Skensved and Køge) are more complex. With such a division we decompose the verification of the interlocking system for **EDL** into the separate verification of the eight stations.

As in the $\mathcal{A} \vdash\!\dashv \mathcal{B}$ case study, the verification tool succeeded to verify the safety properties for the eight sub-interlocking systems and the verification metrics show that for the compositional analysis (see the entry Compositional in Table 3) the verification time is approximately a third (approx. 1.5 h) of that for the monolithic analysis (approx. 4 h). Furthermore, the compositional analysis uses less than half of the memory resources (9243 MB) because we only need as much as the maximum value of memory used to verify each sub-interlocking. Although we are still far from the memory bounds of the used machine in this experiment, such memory reduction is important when checking real world interlocking systems where a single station with a complex network may quickly exhaust the amount of memory available. As already discussed, if run in parallel our compositional approach would achieve a much better running time. Even though memory consumption would increase, the parallelisation would only use

Table 3. Verification metrics for the **EDL** case study.

	Linears	Points	Signals	Routes	$log_{10}(\|S\|)$	Time	Memory
Gadstrup	14	3	16	21	73	62	567
Havdrup	10	2	12	14	51	19	264
L. Skensved	15	3	16	21	75	72	616
Køge	58	23	62	75	337	5170	9243
Herfølge	6	2	10	14	39	13	210
Tureby	6	2	10	14	39	11	203
Haslev	10	2	12	14	51	14	256
Holme-Ol	12	2	16	20	63	22	352
Compositional	131	39	154	193	337	5383	9243
EDL	110	39	126	179	651	14352	22476

roughly 50% (the sum of the individual memory usages: 11711 MB) of the memory resources than the monolithic case. The parallel verification time is dominated by the time to verify the Køge station, which is the largest of the network: actually, the internal layouts of the stations do not present candidates for linear cuts, so they are not further decomposed in this approach.

6 Conclusion

We have presented a compositional approach to the problem of model checking large railway interlocking systems. This approach, built on top of tools providing support for efficient verification of this kind of systems, is tailored to the characteristics of multi-station interlocking systems, that is, systems that control a line connecting several stations. The approach extends our previous work [11], by a new, realistic division process which can be applied in cases where the previous, simpler approach is not applicable. The approach has successfully been applied to a real world line with eight stations in which case it achieved significant improvements in verification time and memory usage compared to the previous non compositional verification process.

In order to compositionally address more general network layouts the *linear cut* concept put forward in this paper needs to be generalised. An immediate extension is to combine it with the border cut concept introduced in our previous work [11]: such interesting strategy should not demand any special efforts beyond the practicalities involved. But the generalisation of the concepts to the application to interlocking systems controlling large stations, which exhibit highly complex and densely connected networks, requires a novel cut concept, which is the subject of some of our new, ongoing work. In that case the main source of difficulty stems from the fact that a division of a large station into smaller areas implies that some routes have to go through the operated cuts, a situation that is not exhibited by the multiple station lines we have addressed till now.

Actually, we have seen that the interface elements in the linear cut have the destination signals of routes coming from both sides of the interface: in the cut the added markerboard behaves as an abstraction of the removed subnetwork. We are currently studying a similar abstraction principle to support the more complex cut configuration required to address large station interlocking systems.

Another topic for future work could be to formalise the proofs done in Sect. 4 by using a proof assistant like Coq or Isabelle.

Acknowledgement. The authors would like to express their gratitude to Jan Peleska and Linh Hong Vu with whom Anne Haxthausen developed the RobustRailS verification method and tools used in the presented work.

References

1. CENELEC European Committee for Electrotechnical Standardization. EN 50128:2011 - Railway applications - Communications, signalling and processing systems - Software for railway control and protection systems (2011)
2. European Railway Agency. ERTMS - System Requirements Specification - UNISIG SUBSET-026, April 2014. http://www.era.europa.eu/Document-Register/Pages/Set-2-System-Requirements-Specification.aspx
3. Ferrari, A., Magnani, G., Grasso, D., Fantechi, A.: Model checking interlocking control tables. In: Schnieder, E., Tarnai, G. (eds.) FORMS/FORMAT 2010 - Formal Methods for Automation and Safety in Railway and Automotive Systems, pp. 107–115. Springer, Heidelberg (2010)
4. Hvid Hansen, H., Ketema, J., Luttik, B., Mousavi, M.R., Pol, J., Santos, O.M.: Automated verification of executable UML models. In: Aichernig, B.K., Boer, F.S., Bonsangue, M.M. (eds.) FMCO 2010. LNCS, vol. 6957, pp. 225–250. Springer, Heidelberg (2011). doi:10.1007/978-3-642-25271-6_12
5. Haxthausen, A.E., Bliguet, M., Kjær, A.A.: Modelling and verification of relay interlocking systems. In: Choppy, C., Sokolsky, O. (eds.) Monterey Workshop 2008. LNCS, vol. 6028, pp. 141–153. Springer, Heidelberg (2010). doi:10.1007/978-3-642-12566-9_8
6. Haxthausen, A.E., Østergaard, P.H.: On the use of static checking in the verification of interlocking systems. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016. LNCS, vol. 9953, pp. 266–278. Springer, Cham (2016). doi:10.1007/978-3-319-47169-3_19
7. Haxthausen, A.E., Peleska, J., Kinder, S.: A formal approach for the construction and verification of railway control systems. Form. Asp. Comput. **23**(2), 191–219 (2011)
8. Haxthausen, A.E., Peleska, J., Pinger, R.: Applied bounded model checking for interlocking system designs. In: Counsell, S., Núñez, M. (eds.) SEFM 2013. LNCS, vol. 8368, pp. 205–220. Springer, Cham (2014). doi:10.1007/978-3-319-05032-4_16
9. James, P., Moller, F., Nguyen, H.N., Roggenbach, M., Schneider, S., Treharne, H.: Techniques for modelling and verifying railway interlockings. Int. J. Softw. Tools Technol. Transf. **16**(6), 685–711 (2014)
10. Limbrée, C., Cappart, Q., Pecheur, C., Tonetta, S.: Verification of railway interlocking - compositional approach with OCRA. In: Lecomte, T., Pinger, R., Romanovsky, A. (eds.) RSSRail 2016. LNCS, vol. 9707, pp. 134–149. Springer, Cham (2016). doi:10.1007/978-3-319-33951-1_10

11. Macedo, H.D., Fantechi, A., Haxthausen, A.E.: Compositional verification of multi-station interlocking systems. In: Margaria, T., Steffen, B. (eds.) ISoLA 2016. LNCS, vol. 9953, pp. 279–293. Springer, Cham (2016). doi:10.1007/978-3-319-47169-3_20
12. Peleska, J.: Industrial-strength model-based testing - state of the art and current challenges. In: Petrenko, A.K., Schlingloff, H. (eds.) 8th Workshop on Model-Based Testing, Rome, Italy, vol. 111, Electronic Proceedings in Theoretical Computer Science, pp. 3–28. Open Publishing Association (2013)
13. Peleska, J., Vorobev, E., Lapschies, F.: Automated test case generation with SMT-solving and abstract interpretation. In: Bobaru, M., Havelund, K., Holzmann, G.J., Joshi, R. (eds.) NFM 2011. LNCS, vol. 6617, pp. 298–312. Springer, Heidelberg (2011). doi:10.1007/978-3-642-20398-5_22
14. Theeg, G., Vlasenko, S.V., Anders, E.: Railway Signalling & Interlocking: International Compendium. Eurailpress, Hamburg (2009)
15. Verified Systems International GmbH. RT-Tester Model-Based Test Case and Test Data Generator - RTT-MBT - User Manual (2013). http://www.verified.de
16. Vu, L.H., Haxthausen, A.E., Peleska, J.: A domain-specific language for railway interlocking systems. In: Schnieder, E., Tarnai, G. (eds.) FORMS/FORMAT 2014–10th Symposium on Formal Methods for Automation and Safety in Railway and Automotive Systems, pp. 200–209. Institute for Traffic Safety and Automation Engineering, Technische Universität Braunschweig (2014)
17. Vu, L.H., Haxthausen, A.E., Peleska, J.: Formal modeling and verification of interlocking systems featuring sequential release. In: Artho, C., Ölveczky, P.C. (eds.) Formal Techniques for Safety-Critical Systems. Communications in Computer and Information Science, vol. 476, pp. 223–238. Springer International Publishing, Cham (2015)
18. Vu, L.H.: Formal development and verification of railway control systems. In the context of ERTMS/ETCS Level 2. Ph.D. thesis, Technical University of Denmark, DTU Compute (2015)
19. Linh Hong, V., Haxthausen, A.E., Peleska, J.: Formal modelling and verification of interlocking systems featuring sequential release. Sci. Comput. Program. **133**, 91–115 (2017)
20. Winter, K.: Symbolic model checking for interlocking systems. In: Flammini, F. (ed.) Railway Safety, Reliability, and Security: Technologies and Systems Engineering. IGI Global (2012)