Proof of Completeness of Compositional Verification of Interlocking Systems

April 1, 2022

Anne E. Haxthausen[1] and Alessandro Fantechi[2]
[1]DTU Compute, Technical University of Denmark, Lyngby, Denmark
[2]DINFO, University of Florence, Via S. Marta 3, Firenze, Italy

Abstract. This document outlines a proof demonstrating completeness of a compositional method presented in our manuscript [HF22] entitled *Compositional verification of railway interlocking systems.*

1 Introduction

In [HF22] we presented a *compositional method for verification of interlocking systems* using the RobustRailS method and tools [VHP17b,VHP17a,Vu15,VHP14] for automated formal verification of interlocking system. The idea of this compositional method is that safety verification of an interlocking systems controlling a network N can be made by dividing the network into two subnetworks N_1 and N_2 and then verify safety of interlocking systems controlling N_1 and N_2, respectively. In [HF22] it is proved that the compositional method is sound.

In these notes we outline a proof demonstrating that the compositional verification method presented in [HF22] is also complete.

These notes are not standalone, but rely on definitions and theorems given in [HF22].

Throughout these notes, assume given a network N and two subnetworks N_1 and N_2 that have been created by a single cut through N according to our compositional method. Let $m = (S, q_0, R, AP, L)$ and $m_i = (S_i, q_{0_i}, R_i, AP_i, L_i)$ for $i = 1, 2$ be models generated for these networks using the RobustRails tools for interlocking systems with the option without overlaps and without flank and front protection. Let $\overline{m|_i}$ be the reduced projection of m on network N_i and let $\overline{m_i}$ be the reduced model of m_i, for $i = 1, 2$.

1.1 Changes needed in section 2 of [HF22]

In order to make the proof of completeness, in the Background section of [HF22] we first need to generalise Definition 2.4 (Simulation relation) and Theorem 2.5 (Simulation preserves invariants) so they can be used not only for the proof of soundness, but also for the proof of completeness. The new version of Theorem 2.5 is needed in the proof of Corollary 1 shown further below in these notes. (Corollary 1 is reverse of Corollary 4.8 in [HF22].)

Definition 2.4 should be generalised to the following (by replacing the condition $AP_2 \subseteq AP_1$ with the condition $AP_1 \cap AP_2 \neq \emptyset$ and changing the third bullet) as we are focusing on common labels of the two transitions systems:

Definition 1 (Simulation Relation). *Given two transition systems $TS_1 = (S_1, q_{0_1}, R_1, AP_1, L_1)$ and $TS_2 = (S_2, q_{0_2}, R_2, AP_2, L_2)$, with $AP_2 \cap AP_1 \neq \emptyset$, TS_1 is simulated by TS_2 (or, equivalently, TS_2 simulates TS_1), denoted $TS_1 \preceq TS_2$, if there exists a simulation relation $\mathcal{R} \subseteq S_1 \times S_2$ such that:*

- $(q_{0_1}, q_{0_2}) \in \mathcal{R}$
- *for all $(q_1, q_2) \in \mathcal{R}$ it holds that if $(q_1, q_1') \in R_1$, then $\exists q_2' : (q_2, q_2') \in R_2$ and $(q_1', q_2') \in \mathcal{R}$*
- *for all $(q_1, q_2) \in \mathcal{R}$ it holds that $L_1(q_1) \cap AP = L_2(q_2) \cap AP$, where $AP = AP_1 \cap AP_2$*

Throughout these notes, this definition of simulation relation will be used.

Consequently, Theorem 2.5 about invariant preservation by simulations needs then to be changed by removing the condition $AP_2 \subseteq AP_1$ and let the invariant be defined over $AP_1 \cap AP_2$ instead of over AP_2:

Theorem 1 (Simulation preserves invariants). *Given two transition systems as before, let $\mathbf{G}\phi$ be an invariant defined over $AP = AP_1 \cap AP_2$, then: $TS_1 \preceq TS_2$ and $TS_2 \models \mathbf{G}\phi$ implies $TS_1 \models \mathbf{G}\phi$*

Proof. The proof of this theorem is directly derived by Corollary 7.68 of [BK08], by considering that invariants belong to the class of *properties* to which the corollary refers.

2 Proof overview

The proof of completeness is very similar to the proof of soundness. An overview of the two proofs are given in Fig. 1 and Fig. 2.

In the proof of completeness, Theorem 4.7, Corollary 4.8, Theorem 4.9 and Corollary 4.16, and Corollary 4.18, that were used in the proof of soundness in [HF22], are replaced by "reverse" theorems and corollaries as explained in the following.

Theorem 4.7 is replaced by:

Theorem 2 (Model simulates its reduced projection). *For $i = 1, 2$, $\overline{m|_i}$ is simulated by m ($\overline{m|_i} \preceq m$) using the simulation relation $\mathcal{R} \subseteq S_i \times S$ defined by: $(q_i, q) \in \mathcal{R}$ iff $q_i = q|_i$.*

Proof. $\overline{m|_i} = (S|_i, q_0|_i, R|_i, \overline{AP|_i}, \overline{L|_i})$ for $m = (S, q_0, R, AP, L)$, by the definition of model projection and model reduction. We must prove:

- $(q_0|_i, q_0) \in \mathcal{R}$, i.e. $q_0|_i = q_0|_i$: which is true.

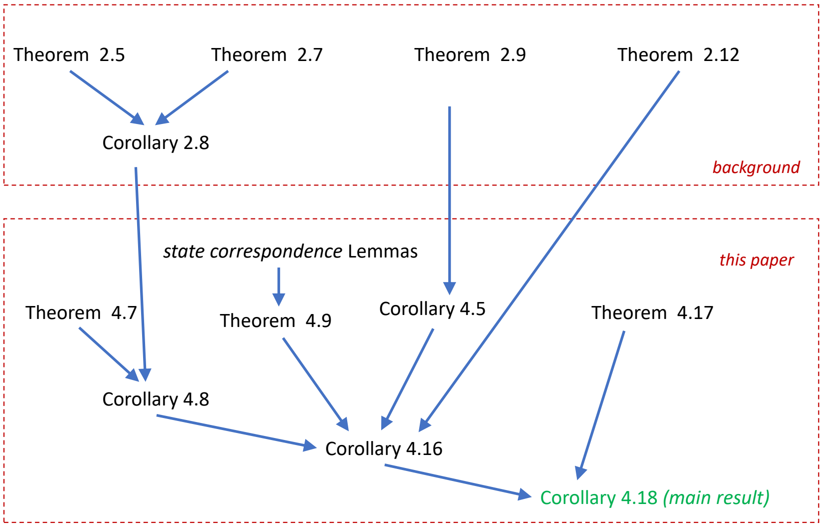

Fig. 1. Proof steps flow in soundness proof in [HF22].

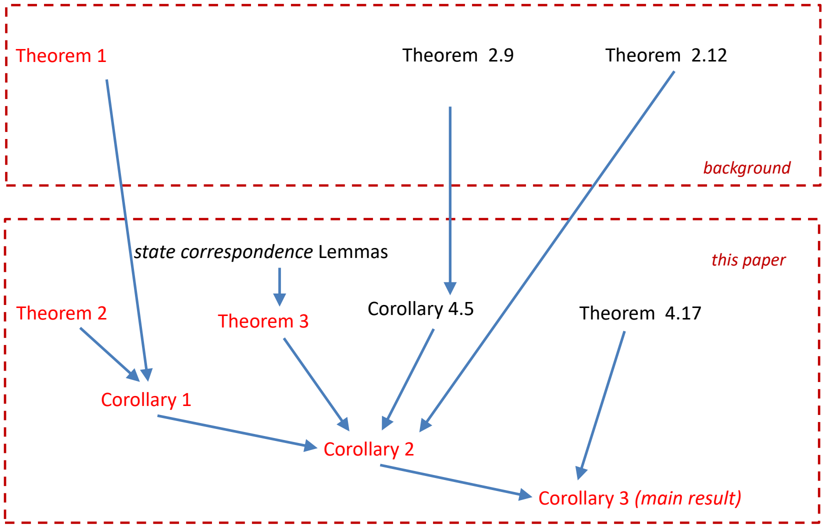

Fig. 2. Proof steps flow in completeness proof in these notes.

- for all $(q_i, q) \in \mathcal{R}$ it holds that if $(q_i, q_i') \in R|_i$, then $\exists q' : (q, q') \in R$ and $(q_i', q') \in \mathcal{R}$: $R|_i$ is defined to be the least relation satisfying that if $(q, q') \in R$ then $(q|_i, q'|_i) \in R|_i$. Therefore, as $(q_i, q_i') \in R|_i$, there must exist a q' such that $(q, q') \in R$ and $q_i' = q'|_i$, i.e. $(q_i', q') \in \mathcal{R}$, (and $q_i = q|_i$ we already know as $(q_i, q) \in \mathcal{R}$).
- for all $(q_i, q) \in \mathcal{R}$ it holds that $\overline{L|_i}(q_i) \cap \overline{AP|_i} \cap AP = L(q) \cap \overline{AP|_i} \cap AP$:
  $(q_i, q) \in \mathcal{R}$ means (0) $q_i = q|_i$.
  By definition of $\overline{AP|_i}$ we have: $\overline{AP|_i} \subseteq AP$. So (1) $\overline{AP|_i} \cap AP = \overline{AP|_i}$.
  By definition of labelling functions for models we have: $\overline{L|_i}(q_i) \subseteq \overline{AP|_i}$.
  So (2) $\overline{L|_i}(q_i) \cap \overline{AP|_i} = \overline{L|_i}(q_i)$.
  By (1), (2) and (0) we get: $\overline{L|_i}(q_i) \cap \overline{AP|_i} \cap AP = \overline{L|_i}(q_i) \cap \overline{AP|_i} = \overline{L|_i}(q_i) = \overline{L|_i}(q|_i)$.
  By (1) we get: $L(q) \cap \overline{AP|_i} \cap AP = L(q) \cap \overline{AP|_i}$.
  So we should prove that $\overline{L|_i}(q|_i) = L(q) \cap \overline{AP|_i}$. This holds as the atomic propositions in $\overline{L|_i}(q|_i)$ are of the form $t = q(t)$ (as $(q|_i)(t) = q(t)$), where $t \in \mathit{sections}(N_i)$, and that is also the case for all atomic propositions in $L(q) \cap \overline{AP|_i}$.

Corollary 4.8 is replaced by the following and the arrow from Corollary 2.8 is replaced by a direct arrow from Theorem 1 (that is replacing Theorem 2.1) to this new Corollary.

Corollary 1 (Safety preserved in reduced projection). $\overline{m|_i} \models \phi_i$ *if* $m \models \phi_i$*, for safety properties (invariants)* $\phi_i, i = 1, 2$.

Proof. Follows from the new version of Theorem 2.5 (= Theorem 1 in these notes) and Theorem 2.

Theorem 4.9 is replaced by:

Theorem 3 (Reduced projection stutter trace includes reduced subnetwork model). $\overline{m_i} \trianglelefteq \overline{m|_i}$*, for* $i = 1, 2$.
which means $\forall\ \overline{\pi_i} \in \mathit{Paths}(\overline{m_i})\ \ \exists\ \overline{\pi|_i} \in \mathit{Paths}(\overline{m|_i})$ *such that* $\overline{\pi|_i}$ *and* $\overline{\pi_i}$ *are stutter equivalent, for* $i = 1, 2$.

Proof. See Sect. 3.

Corollary 4.16 is replaced by:

Corollary 2 (Safety of submodels derived from full model). $m_i \models \phi_i$ *if* $m \models \phi_i$*, for* $i = 1, 2$.

Proof. The proof is similar to that for Corollary 4.3 just with the implications A, B and C in the opposite direction:

$$\text{For } i = 1, 2\text{: } m_i \models \phi_i \xLeftarrow{A} \overline{m_i} \models \phi_i \xLeftarrow{B} \overline{m|_i} \models \phi_i \xLeftarrow{C} m \models \phi_i$$

- A - From Corollary 4.5;
- B - From Theorems 2.12 and 3, and noting that $\overline{AP_i} = \overline{AP|_i}$;

– C - From Corollary 1.

Corollary 4.18 is replaced by:

Corollary 3 (Completeness).
$m_i \models \phi_i$ *for* $i = 1, 2$, *if* $m \models \phi$.

Proof. Follows from Corollary 2 and Theorem 4.17.

3 Proof of Theorem 3

We can prove Theorem 3 by proving that for an arbitrary path π_i in $Paths(m_i)$, it is possible to find a path π in $Paths(m)$, such that $\pi|_i$[1] and π_i are stutter equivalent wrt. the labelling functions of $\overline{m|_i}$ and $\overline{m_i}$, respectively. This approach is valid as by definition of the model projection and reduction operators, we have $Paths(\overline{m|_i}) = Paths(m|_i) = \{\pi|_i \mid \pi \in Paths(m)\}$ and $Paths(\overline{m_i}) = \{\pi_i \mid \pi_i \in Paths(m_i)\}$.

3.1 Construction of π

Given an arbitrary path π_i in $Paths(m_i)$, for a given $i = 1$ or $i = 2$, a corresponding path π is constructed incrementally as follows:

- The first state of π is chosen to be the initial state q_0 of m.
- Then π is obtained by adding more and more states to its path by considering the states in π_i, one by one, in the order they appear. Each transition from a state q_i to a state q_i' in π_i, leads to the addition of 0, 1 or more states to π. What exactly should be added depends on the transition rule r_i that caused the state change from q_i to q_i' in π_i. Below we will explain this systematically by case over various classes of transition rules. As it will be seen, usually one new state q' is added to π – this state is obtained by applying r_i or a rule corresponding to r_i to the latest added state q in π. However, in some cases more than one state is added to π_i by applying several transition rules in m, and in a few cases no state is added.

Case 1: Some transition rules for m_i **also exist for** m (with the same guards and same variable updates) as they only concern network elements in N for which the projection to N_i is the identity. These rules include:

1. Rules for switching points $p \in points(N_i)$ (i.e. changing $p.POS$).
2. Rules for switching signals s (i.e. changing $s.ACT$) that are inside N_{-i} (i.e. $s \in signals(N_{-i})$[2])
3. Rules describing the train movement of the head or tail of a train from a section t to a neighbouring section t', both inside N_i ($t, t' \in sections(N_i)$).

[1] Here the projection operator on states has been lifted to paths in the obvious way.
[2] Note that $signals(N_{-i})$ does not include any added border signal present in N_i.

4. Rules describing how the head or tail of a train enters/leaves the network at a border which is also in N (i.e. it is not the border that was achieved due to the cut).
5. Rules for controlling routes r_i that are projections of routes r ($r_i = proj_i(r)$) in N that are (1) completely inside N_i (so $proj_i(r) = r$ and therefore $r = r_i$) and (2) not ending at a signal in front of the cut.

A transition from a state q_i to a state q_i' in π caused by such a rule should lead to the same state change in π, here from the last added state q to a new state q' that is obtained by applying the same rule to q (so q' is added to π).

Case 2: For routes r_i that are projections of routes r ($r_i = proj_i(r)$) in N that are (1) completely inside N_i (so $proj_i(r) = r$ and therefore $r = r_i$) and (2) end at a signal in front of the cut:

1. Rules for controlling r in m and r_i in m_i are the same and when applied in π_i they should also be applied in π, with the following exception, see next item.
2. When $release(r_i)$ is applied in π_i, first $release(r_i)$ should applied in π_i and then a whole sequence of rules should be applied to move the train that is now positioned at the first section on the other side of the cut through the other part of the network and out of the network.

Case 3: The remaining transition rules for m_i concern (1) train movements where the head or tail of a train is entering or leaving N_i at the border that was achieved by the cut, and (2) routes r_i that are projections of one or several through routes r in N: $r_i = proj_i(r)$ as illustrated in Figs. 3, 4, and 5. If r_i is the projection of *several* routes, one of these is chosen as its corresponding route r in N. (It is not important which one is chosen, but it should be the same all the time.) Without loss of generality, we assume that the through route r has direction UP, starts at a signal s_1 in N_{-1} and ends at a signal s_2 in N_{-2}, as shown in Fig. 3.

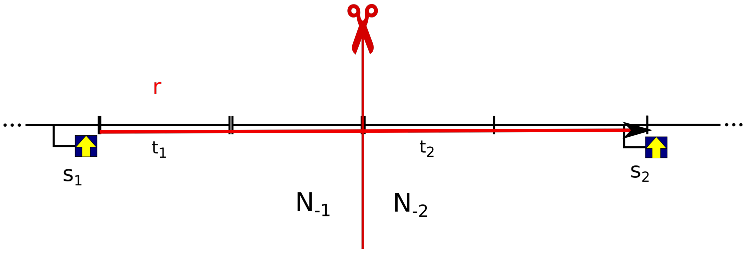

Fig. 3. A cut through a through route r in a network N.

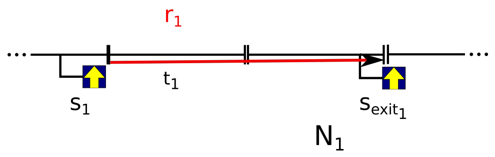

Fig. 4. Projected route r_1 of route r in Fig. 3.

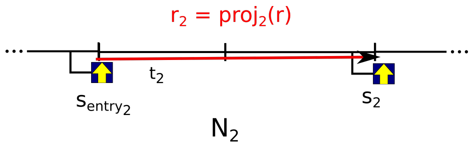

Fig. 5. Projected route r_2 of route r in Fig. 3.

1. When the $dispatch(r_i)$ is applied in π_i, the corresponding $dispatch(r)$ rule should be applied in π.
2. When the $allocate(r_i)$ is applied in π_i, the corresponding $allocate(r)$ rule should be applied in π.
3. When the $lock(r_i)$ is applied in π_i, first, for all points p of r that are outside r_i ($p \in points(r) \setminus points(r_i)$), the rules for switching the actual position of p to the commanded position of p should be applied in π, and then the corresponding $lock(r)$ rule should be applied in π.
4. For $i = 1$: When the entry signal s_1 is opened in π_1, the same rule for opening s_1 (changing $s_1.ACT$ to OPEN) should be applied in π.
5. For $i = 2$: When the added entry signal $s_{entry_2}(= proj_2(s_1))$ is opened in π_2, the corresponding rule for opening s_1 (changing $s_1.ACT$ to OPEN) should be applied in π.
6. For $i = 1$: When the $route\_in\_use(r_1)$ rule is applied in π_1, the corresponding rule $route\_in\_use(r)$ should be applied in π.
7. For $i = 2$: When the $route\_in\_use(r_2)$ rule is applied in π_2, then $element\_in\_use(r, t_2)$ should be applied in π, where t_2 is the first section in r_2.
8. For $i = 1$: When the entry signal s_1 is closed in π_1, the same rule for closing s_1 (changing $s_1.ACT$ to CLOSED) should be applied in π.
9. For $i = 2$: When the added entry signal s_{entry_2} is closed in π_2, the corresponding rule for closing s_1 (changing $s_1.ACT$) should NOT be applied in π (as it has already been applied).
10. When the $element\_in\_use(r_i, e)$ rule for a section e in the path of r_i (which is not the first section in the path of r_i) is applied in π_i, then the corresponding $element\_in\_use(r, e)$ rule should be applied in π.

11. When the $sequential\_release\_e(r_i, e)$ rule for a section e (which is not the last section in the path of r_i) is applied in π_i, then the corresponding $sequential\_release\_e(r, e)$ rule should be applied in π.
12. For $i = 1$: When the $release(r_1)$ rule is applied in π_1, then in π first the $sequential\_release\_e(r, last(r_1))$ rule should be applied in π, and then a whole sequence of rule applications should take place, dispatching, allocating, locking and opening a sequence of routes out to an exit border in N_{-2}[3] and moving the train along r and these routes and leaving N, while releasing r and the other routes one by one after use. (So this sequence of events will in particular include $release(r)$.)

Train movements passing a cut where there was no signal in N, for $i = 1$:

1. For $i = 1$: When the train movement rule for the head **leaving** the network N_1 via section t at the cut border is applied in π_1, then in π, first the rule describing the movement of the head of the **train passing a cut** from section t in $sections(N_1)$ to section t_2 in $sections(N_2)$ should be applied, and then the $element\_in\_use(r, t_2)$ rule should be applied.
2. For $i = 1$: When the train movement rule for the tail **leaving** the network N_1 via section t at the cut border is applied in π_1, then the rule describing the movement of the tail of **train passing a cut** from section t in $sections(N_{-1})$ to section t_2 in $sections(N_{-2})$ should be applied in π.

Train movements passing a cut where there was no signal in N, for $i = 2$:

1. For $i = 2$: When the train movement rule for the head **entering** the network N_2 via section t_2 at the cut border is applied in π_2, then first a whole sequence of rule applications should take place in π, dispatching, allocating, locking and opening a sequence of routes from an entry border in the down end of N_{-1} up to s_1[4] and entering a train N from that border and moving it along these routes (and releasing them one by one after use) and r down to the section t just before the cut (note that these steps include the application of $route\_in\_use(r)$ and $close(s_1)$), and finally the rule describing the movement of the head **passing the cut** from section t in $sections(N_1)$ to section t_2 in $sections(N_2)$ should be applied in π.
2. For $i = 2$: When the train movement rule for the tail **entering** the network N_2 via section t_2 at the cut border is applied in π_2, then in π, first the rule describing the movement of the tail of **train passing a cut** from section t in $sections(N_1)$ to section t_2 in $sections(N_2)$ should be applied, and then $sequential\_release\_e(r, t)$ should be applied.

Train movements passing a cut where there was a signal in N: see handwritten notes.

[3] Such a sequence of routes exist according to Lemma 5.

[4] Such a sequence of routes exist according to Lemma 5.

Validity of π For $i = 1, 2$, it should then be demonstrated that the constructed path π is a path of m. For this to hold, the first state in π should be q_0, and each time a state transition rule for m should be applied to a state q in the construction of π, it should be ensured that the rule was actually applicable in q, i.e. its guard was true in q.

The first is clearly the case. In order to prove the latter, some state correspondence lemmas are needed expressing relations $\rho(q, q_i)$ that at any point in the construction process hold between the last considered state q_i in π_i and the last added state q in π. Furthermore, some more lemmas and corollaries about projection of network elements and about network traversability are needed. Examples of such lemmas and corollaries are presented in the next subsections.

Here is an example of how we have proved rule applicability for the case considered in item 12 of Case 3 in Sec. 3.1. (This is one of the more complicated cases.)

Example 1. Let q_1 is that last considered state in π_1 and q is the last state added to π.

First it should be proved that when the transition rule $release(r_1)$ is applicable in q_1 then $sequential\_release\_e(r, last(r_1))$ is applicable in q. I.e. we should prove that $r.MODE = OCCUPIED$ and $t.MODE = USED$ and $vacant(t)$ in q, where $t = last(r_1)$ is the last section of r_1 in N_1 (i.e., in N, t is the last section of r before the cut). This follows from (1) the corresponding properties in q_1: $r_1.MODE = OCCUPIED$ and $t.MODE = USED$ and $vacant(t)$ that hold as $release(r_1)$ was applicable, and (2) the state correspondence Lemma 9 and Lemma 12.

Let q' be the state in π that is achieved by applying $sequential\_release\_e(r, last(r_1))$ in q. It should now be proved that we can prepare routes in N_2 and move a train present on t_2 through r and these routes out of the network. This is possible due to Corollary 7 and state correspondence Lemma 14.

3.2 Lemmas about projection of network elements

In the following, let r be a route, $sections(r)$ be the set of sections in the path of r, $points(r)$ be the set of points in the path of r, $first(r)/last(r)$ be the first/last section in the path of r, $src(r)/dst(r)$ be the entry/exit signal of r, $req(r, p)$ be the required position of a point p in $points(r)$, and $conflicts(r)$ be the set of conflicting routes of r.

Lemma 1 (projection gives subsets of network elements). *$sections(proj_i(r)) \subseteq sections(r)$ and $points(proj_i(r)) \subseteq points(r)$when $proj_i(r)$ is defined.*

Lemma 2 (projection of entry/exit signals of a route). *$scr(proj_i(r)) = proj_i(src(r))$ and $dst(proj_i(r)) = proj_i(dst(r))$ when $proj_i(r)$ is defined.*

Lemma 3 (projection preserves required point settings). *Assume $proj_i(r)$ is defined. The required point setting for any point $p \in points(proj_i(r))$ in the path of $proj_i(r)$ in N_i is the same as for p in the path of r in N: $req(proj_i(r), p) = req(r, p)$.*

Lemma 4 (projection preserves and reflects conflicts).
$proj_i(cr) \in conflicts(proj_j(r))$ in N_i if and only if $cr \in conflicts(r)$ in N, when $proj_i(r), proj_i(cr)$ are defined and $proj_i(cr) \neq proj_i(r)$.

The lemmas follow from the definition of the projection function on network elements.

3.3 Network Traversability

In RobustRailS, any legal railway network N has the following properties:

Lemma 5 (Network traversability).
Any internal signal s of a network N is reachable along some consecutive routes from some entrance of the network, and from s some exit of the network is reachable along some consecutive routes.

Proof. This lemma follows from the fact that any entrance/exit to/from the network is covered by an entry/exit signal, and for any internal signal s in direction UP/DOWN, there exists a path from at least one entry signal in the DOWN/UP end of the network to s, and there exists a path from s to at least one exit signal in the UP/DOWN end of the network. Since we in this work include all possible elementary signal-to-signal routes in a network, each of these paths are made of one or several consecutive routes (going from one signal along the path to the next signal along the path).

The model m generated for network N has the following route traversability properties:

Lemma 6 (Route traversability 0).
If a route r has an open entry signal s, but is not yet occupied, and a train in the direction of the route is present on the last section before the entry signal s, it is possible for the train to traverse the route up to any section t in the route: it can enter the route, and then the route_in_use event followed by the closing of the entry signal will take place, whereupon a series of train movement events and (element_in_use and sequential_release) reactions to these by the route controller can take place, until the train is on section t.

Proof. Inspecting the guards of transition rules for these events, it is easy to see that the suggested sequence of events is actually possible.

Lemma 7 (Route traversability 1).
If a route r is occupied by a train going along the route and those sections of the route that are in front of the train are vacant, then the train can move to the last section of the route.

Proof. It is easy to see that the train movement rules for bringing the train to the end of the route (and the corresponding reactions by the route controller) can be applied.

Lemma 8 (Route traversability 2).
If a route r with entry signal s is in mode FREE and all its track elements are vacant and in mode FREE and the same holds for all routes cr that are in conflict with r, it is possible to make a sequence of events preparing r for being used (i.e. dispatching r, allocating r, switching the points in the route as commanded in allocation step, locking r, and finally opening the entry signal s of r as commanded in locking step). Then, if a train in the direction of the route is present on the last section before the entry signal s of r, it is possible for the train to traverse the route: it can enter the route, and then the route_in_use event followed by the closing of the entry signal will take place, whereupon a series of train movement events and (element_in_use and sequential_release) reactions to these by the route controller can take place, until the train is on the last section of the route. If that section is at a border, the train can then leave the network and the route will be released.

Proof. Inspecting the guards of transition rules for these events, it is easy to see that the suggested sequence of events is actually possible.

The following corollary expresses that under certain conditions it is possible to let a train enter a network N and move along some routes to the front of an internal signal s:

Corollary 4. *For any internal signal s of a network N, consider a sequence of consecutive routes from some entrance of the network up/down to s (note such a sequence exists according to Lemma 5). If these routes are in mode FREE and all their track elements are vacant and in mode FREE and the same holds for all routes cr that are in conflict with these route, then it is possible to let a train traverse from the entrance along the routes up/down to the last section before s.*

Proof. Follows from Lemma 8: One by one each of the routes are first being prepared for being used and then the train is traversing that route until its last section.

Similarly, the following corollary expresses that under certain conditions it is possible to move a train from the section before an internal signal s to an exit and let it leave the network:

Corollary 5. *For any internal signal s of a network N, consider a sequence of consecutive routes from s up/down to an exit of the network N (note such a sequence exists according to Lemma 5). If these routes are in mode FREE and all their track elements are vacant and in mode FREE and the same holds for all routes cr that are in conflict with these route, then it is possible to let a train present at the last section before s traverse the routes one by one and leave the network.*

Proof. Follows from Lemma 8: One by one each of the routes are first being prepared for being used and then the train is traversing that route until its last section.

The following corollary expresses conditions under which it is possible to let a train enter a network N and move up the last section before the cut in N:

Corollary 6 (Moving a train through opposite network up to the cut). *Given a through route r (passing a cut) from one signal s_1 to another signal s_2 in a network N. Let R_1 be a sequence of consecutive routes from some entrance of the network up/down to s_1. (Note that such a sequence exists according to Lemma 5.)*

If (1) the routes in R_1 are in mode FREE and all their track elements are vacant and in mode FREE and the same holds for all routes cr that are in conflict with these routes, and (2) r has an open entry signal s_1, but is not yet occupied, it is possible to let a train enter the network and go along the routes in R_1 and r up to the last section before the cut.

Proof. Follows from Corollary 4 and Lemma 6.

The following corollary expresses conditions under which it is possible to move a train from the first section after a cut through routes leading it out of the network:

Corollary 7 (Moving a train in opposite network out of N). *Given a through route r (passing a cut) from one signal s_1 to another signal s_2 in a network N. Let R_2 be a sequence of consecutive routes from s_2 up/down to an exit border of N. (Note that such a sequence exists according to Lemma 5.)*

If r is occupied by a train going along the route and (1) the train has reached the first section t_2 after the cut and those sections of the route r that are in front of the train are vacant and (2) the routes of R_2 are in mode FREE and all their track elements are vacant and in mode FREE and the same holds for all routes cr that are in conflict with these routes, then it is possible to let a train go along the rest of r and the routes in R_2 and leave the network.

Proof. Follows from Lemma 7 and Corollary 5.

3.4 State correspondence lemmas

Lemma 9 (State correspondence for track sections in N_i). *At any point in the construction process of π from π_i, it holds that $\overline{L_i}(q_i) = \overline{L|}_i(q|_i)$, where q_i is that last considered state in π_i and q is the last state added to π. Note that this also means that $q_i(t) = q(t)$ for sections $t \in sections(N_i)$.*

Proof. This can be proved by induction. It is easy to see that the relation holds for the initial states. It is also easy to see that any step in the construction process of π from π_i preserves this relation: Either a concurrent step makes the same changes to variables for sections in N_i or they make no changes to these variables. (Note that the state changes made in a step from q to q' in π may change variables for sections outside N_i, but these variables are removed by the projection to $q|_i$ and $q'|_i$.)

Lemma 10 (State correspondence for routes totally inside N_{-i}). *At any point in the construction process of π from π_i, $q(r) = q_i(r)$ for those routes $r \in routes(N)$ that are completely inside N_i, where q_i is that last considered state in π_i and q is the last state added to π.*

Proof. This can be proved by induction: It is easy to see that the relation holds for the initial states. It is also easy to see that any step in the construction process of π from π_i preserves this relation.

Lemma 11 (State correspondence for signals totally inside N_{-i}). *At any point in the construction process of π from π_i, $q(s) = q_i(s)$ for those signals $s \in signals(N)$ that are completely inside N_{-i} (and are especially not an entry signal of a through route), where q_i is that last considered state in π_i and q is the last state added to π.*

Proof. This can be proved by induction: It is easy to see that the relation holds for the initial states. It is also easy to see that any step in the construction process of π from π_i preserves this relation.

For any route r_i in N_i that is the projection of one or more through routes in N, let r be the chosen corresponding through route in N. The state correspondence between r and r_i is as described in the following lemma.

Lemma 12 (State correspondence for through routes). *If the through route starts in N_{-i}: At any point in the construction process of π from π_i, $q(r) = q_i(r_i)$, where q_i is that last considered state in π_i and q is the last state added to π.*
If the through route ends in N_{-i}: See hand-written notes.

Proof. This can be proved by induction: It is easy to see that the relation holds for the initial states. It is also easy to see that any step in the construction process of π from π_i preserves this relation.

For any route r_i in N_i that is the projection of one or more through routes in N and for which its entry signal s_i is an added entry signal, let r be the chosen corresponding through route in N and let s be the entry signal of r. (I.e. $proj_i(s) = s_i$ and $proj_i(r) = r_i$.) The state correspondence between s and s_i is as as described in the following lemma.

Lemma 13 (State correspondence for added entry signals). *See hand-written notes.*

Proof. This can be proved by induction: It is easy to see that the relation holds for the initial states. It is also easy to see that any step in the construction process of π from π_i preserves this relation.

Let $opposite(1) = 2$ and $opposite(2) = 1$.

Lemma 14 (States of track sections and routes in opposite subnetwork). *At any point in the construction process of π from π_i, it holds for the last added state q in π after a step in π_i [5]:*

[5] This does not necessarily hold for intermediate states in cases where a step in π_i leads to several added states in π.

- *$vacant(t) = true$ and $t.MODE = FREE$ for sections $t \in sections(N)$ that are in the opposite subnetwork (i.e. $t \in sections(N_{opposite(i)})$), except if t is neighbour to the cut.*
- *$r.MODE = FREE$ for those routes $r \in routes(N)$ that are completely inside the opposite subnetwork $N_{opposite(i)}$.*

Proof. This can be proved by induction: It is easy to see that the relation holds for the initial states. It is also easy to see that any step in the construction process of π from π_i preserves these properties.

3.5 Proof of stutter equivalence

We have now proved for $i = 1, 2$ that for an arbitrary path $\pi_i \in Path(m_i) = Path(\overline{m_i})$, we can construct a path $\pi \in Path(m)$ such that it satisfies Lemma 9, i.e. $\overline{L_i}(q_i) = \overline{L|}_i(q|_i)$, in any step of the construction process. Hence, π_i and $\pi|_i$ are stutter equivalent (as $\pi \in Path(m)$ implies $\pi|_i \in Path(\overline{m_i})$).

References

BK08. Christel Baier and Joost-Pieter Katoen. *Principles of model checking.* MIT Press, 2008.

HF22. Anne E. Haxthausen and Alessandro Fantechi. Compositional verification of railway interlocking systems. *Submitted for publication*, 2022.

VHP14. Linh H. Vu, Anne E. Haxthausen, and Jan Peleska. A Domain-Specific Language for Railway Interlocking Systems. In Eckehard Schnieder and Géza Tarnai, editors, *FORMS/FORMAT 2014 - 10th Symposium on Formal Methods for Automation and Safety in Railway and Automotive Systems*, pages 200–209. Institute for Traffic Safety and Automation Engineering, Technische Universität Braunschweig, 2014.

VHP17a. Linh H. Vu, Anne E. Haxthausen, and Jan Peleska. A domain-specific language for generic interlocking models and their properties. In Alessandro Fantechi, Thierry Lecomte, and Alexander Romanovsky, editors, *Reliability, Safety, and Security of Railway Systems. Modelling, Analysis, Verification, and Certification: Second International Conference, RSSRail 2017, Pistoia, Italy, November 14-16, 2017, Proceedings*, pages 99–115, Cham, 2017. Springer International Publishing.

VHP17b. Linh Hong Vu, Anne E. Haxthausen, and Jan Peleska. Formal modelling and verification of interlocking systems featuring sequential release. *Science of Computer Programming*, 133, Part 2:91 – 115, 2017. http://dx.doi.org/10.1016/j.scico.2016.05.010.

Vu15. Linh Hong Vu. *Formal Development and Verification of Railway Control Systems - In the context of ERTMS/ETCS Level 2.* PhD thesis, Technical University of Denmark, DTU Compute, 2015.