CSP||B modelling for railway verification: the Double Junction case study

Faron Moller
Hoang Nga Nguyen
Markus Roggenbach
Steve Schneider
Helen Treharne

March 30th 2012

University of Surrey

Department of Computing

Computing Sciences Report

CS-12-03

CSP||B modelling for railway verification: the double junction case study

Faron Moller[1], Hoang Nga Nguyen[1], Markus Roggenbach[1],
Steve Schneider[2] and Helen Treharne[2]

[1]Swansea University, Wales [2]University of Surrey, England

Abstract. This paper extends recent work in verifying railway systems through CSP ‖ B modelling and analysis. In particular we consider the Double Junction case study, a more complex example than we have considered previously, which involves a crossover of two tracks, two related sets of points, and open ends where trains enter and exit the system. We are able to apply the general control system previously developed, and instantiate the model with the track topology and control tables of this particular example. We are able to verify safety (collision-freedom) properties automatically using ProB, and to identify sequences of events that lead to safety violations in alternative models.

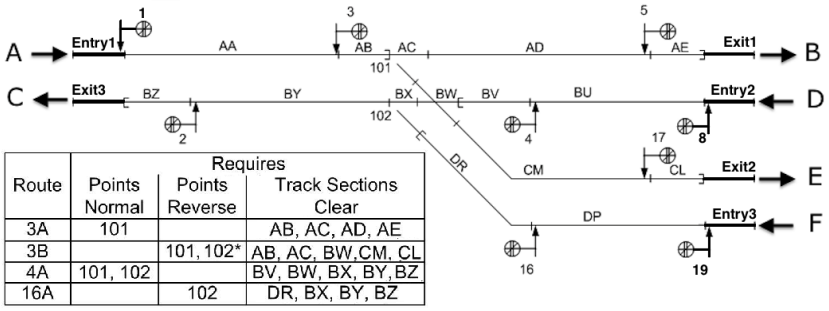
1 Introduction

CSP||B [7] has recently been demonstrated to be applicable to modelling railway interlockings and verifying their safety properties [4]. This paper shows that the methodology developed is easily reusable, and extends to more complex track features and their associated safety requirements. Our safety properties are proved using model-checking [3], whereas other approaches [1, 6] establish similar properties through formal proof, or through a combination of proof and model-checking [5]. We also demonstrate that the model-checking approach provides traceable feedback in situations where errors in track plans mean that safety does not hold.

2 The Double Junction example

We model the Double Junction track plan shown in Figure 1, taken from [2]. This plan contains a number of features that have not been present in our previous CSP||B railway modelling: the crossing, the related points, and the fact that the system is open (i.e., contains entries and exits). These aspects provide a number of new modelling challenges for our application of CSP||B.

An interlocking system gathers train locations, and sends out commands to control signal aspects and point positions. The *control table* which dictates the behaviour of the Double Junction interlocking system appears in Figure 1. For each signal, there is one row describing the condition under which the signal can show proceed. There are two rows for signal 3: one for the main line (Route 3A)



Route	Requires: Points Normal	Requires: Points Reverse	Requires: Track Sections Clear
3A	101		AB, AC, AD, AE
3B		101, 102*	AB, AC, BW,CM, CL
4A	101, 102		BV, BW, BX, BY,BZ
16A		102	DR, BX, BY, BZ

Fig. 1: The track plan of the *Double Junction*.

and one for the side line (Route 3B). For example, signal 3 for the main line can only show proceed when point 101 is in normal (straight) position and tracks AB, AC, AD and AE all are clear.

The interlocking behaviour also allocates *locks* on points to particular route requests to keep them locked in position, and releases such locks when trains have passed. For example, the setting of Route 3B obtains locks on points 101 and 102, and sets them to reverse. The locks are released only after the train has passed. The precise point at which the locks are released is safety-critical: if released too early a collision can occur (e.g. see Scenario 4 of Figure 7).

3 CSP||B Architecture

The architecture of our model[1] is depicted in Figure 2. CSP is used to describe the driving rules of trains in order to control their movement and enable a controller to issue route requests. The B part models the impact of the current movement of trains on the track equipment and focuses on interlocking.

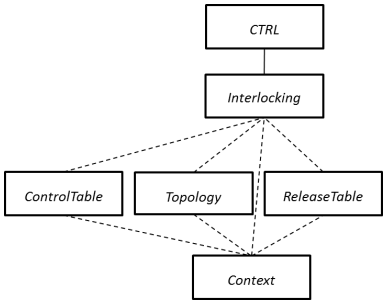


Fig. 2: Architecture.

- The *CTRL* component is a CSP description of the driving rules that control the movement of trains. In particular, this is where we include the rule that a train driver should not pass through a stop signal.
- The *Interlocking* component is a B-machine that describes the general interlocking rules: that route requests can

[1] CSP||B model download: http://www.csp-b.org/b2012-double-junction.zip

be granted or refused, depending on the location of the trains and status of locks with respect to the control table. It also models the positions of the trains, and the state of the signals and points. This component is generic and does not depend on the particular track topology.

– The *ControlTable*, *Topology*, *ReleaseTable*, and *Context* components describe the particular track under consideration. These describe respectively: the control table: the track topology (i.e. how the tracks are connected); the release table which gives the rules for releasing the locks on the points; and the datatypes to model of the tracks and points.

This approach was first presented in [4], for the *Mini-Alvey* track circuit. Applying it to the Double Junction presents some new modelling challenges, and requires some extensions to the CSP‖B modelling approach to allow for entry and exit locations, and the track crossing.

Modelling entry and exit: The entry and exit locations are incorporated as additional track segments, as shown in Figure 3 and connected to the appropriate tracks within the track topology (see Figure 4), such that no track precedes an *entry* track, and no track follows an *exit* track. Trains not currently in the model can be introduced to an entry track by means of an *enter* operation as shown in Figure 5, and exit via an *exit* operation. The CSP component reflects this treatment: trains simply appear at *entry* tracks, and disappear from *exit* tracks. This behaviour is described in Figure 6.

$$ENTRY = \{Entry1, Entry2, Entry3\} \ \wedge \ EXIT = \{Exit1, Exit2, Exit3\}$$

Fig. 3: Datatype definitions for special tracks defined in *Context* machine.

Modelling the track crossing: The double junction also introduces the track crossing BW, a track segment that can be crossed in two directions. In modelling the track topology using the relation *next* (see Figure 4), we model the crossing as two track segments, $BW1$ (between BV and BX), and $BW2$ (between AC and CM). Their co-location is modelled by the inclusion of a further set, $collision = \{BW1, BW2\}$, used to represent the occurrence of a collision when both $BW1$ and $BW2$ are occupied simultaneously.

$$\begin{array}{l} next \in TRACK \leftrightarrow TRACK \ \wedge \\ next = \{(AA \mapsto AB), (AB \mapsto AC), (AC \mapsto AD), (AC \mapsto BW2), (AD \mapsto AE), (AE \mapsto Exit1), \\ \quad (BZ \mapsto Exit3), (BY \mapsto BZ), (BX \mapsto BY), (BW1 \mapsto BX), (BV \mapsto BW1), \\ \quad (BU \mapsto BV), (BW2 \mapsto CM), (CM \mapsto CL), (CL \mapsto Exit2), (DP \mapsto DR), (DR \mapsto BX), \\ \quad (Entry1 \mapsto AA), (Entry2 \mapsto BU), (Entry3 \mapsto DP)\} \end{array}$$

Fig. 4: Datatype definitions for special tracks defined in *Topology* machine.

$$
\begin{array}{l}
enter(t,p) = \ \mathbf{PRE}\ t \in TRAIN \wedge t \notin dom(pos) \wedge p \in ENTRY \wedge p \notin ran(pos) \\
\quad\quad\quad\ \mathbf{THEN}\quad pos(t) := p \quad || \quad occupiedTracks := occupiedTracks \cup \{p\} \\
\quad\quad\quad\quad\quad\quad || \ emptyTracks := emptyTracks - \{p\} \\
\quad\quad\quad\ \mathbf{END}
\end{array}
$$

Fig. 5: *enter* operation from *Interlocking* machine.

$$
\begin{array}{l}
TRAIN\_OFF(t) = enter!t?newp \rightarrow TRAIN\_CTRL(t, newp) \\
TRAIN\_CTRL(t, currp) = nextSignal!t?s \rightarrow \\
\quad \mathbf{if}\ (s == none\ or\ s == green) \\
\quad \mathbf{then}\ (move.t.currp?newp \rightarrow \\
\quad\quad\quad (\mathbf{if}\ (member(newp, EXIT)) \\
\quad\quad\quad \mathbf{then}\ (exit!t!newp \rightarrow TRAIN\_OFF(t)) \\
\quad\quad\quad \mathbf{else}\ TRAIN\_CTRL(t, newp)) \\
\quad\quad \Box\ stay.t.currp \rightarrow TRAIN\_CTRL(t, currp)) \\
\quad \mathbf{else}\ stay.t.currp \rightarrow TRAIN\_CTRL(t, currp)
\end{array}
$$

Fig. 6: Fragment of *CTRL* CSP process.

4 Safety and Model Checking Scenarios

The CSP‖B model of the Double Junction can be analysed using the ProB model-checker. We have analysed variations of the model for collision-freeness.

The requirement that two trains should never be on the same track is captured by the following clause in the invariant, stating that every train should be on a different track (i.e. the mapping from trains to tracks is injective). This clause was introduced in [4]:

$$pos : TRAIN \rightarrowtail TRACK \tag{1}$$

Additionally, in this model, the requirement that two trains should not both be on the crossing at the same time is captured by the following clause of the invariant:

$$\neg(collision \subseteq occupiedTracks) \tag{2}$$

We considered 6 scenarios, and in each case used ProB to check for invariant violations in the model. Since collision-freedom is captured in the invariant, possible collisions would be detected as invariant violations. As well as checking the intended model, we also explored unsafe variations in the control table (not obtaining all the required locks, shown in Figure 8) and the release table (releasing a lock too early), in Scenarios 3 and 4. Scenario 5 considered the full model, however the analysis took too long, so Scenario 6 introduced a restriction on the CSP controller to reduce the state space by removing events that did not progress the state (i.e. red signals, and trains not moving).

Scenarios 1–2 only included clause (1) above in the invariant, so only checked whether one train could run into another. Scenarios 3–6 also included clause (2)

in the invariant to also check for collisions on the crossing, and adjusted the the control table for $3B$ and $4A$ so that only the relevant track across the crossing, rather than both tracks, needed to be clear to grant the route. This resulted in a larger state space. The analysis results for these six scenarios are given in Figure 7.

	Description	**States Checked**	**Result**
1	B machines alone. No CSP: no driving rules, so trains can always move	x	Simple violation: train moves through red light and collides
2	B machines and CTRL	80,921	No violation
3	Some locks on points dropped (shown in Fig. 8), so that routes only obtain locks on the points that they will travel over, not on the adjacent tracks	13,341	Violation trace found, comprising 36 events leading to a collision (illustrated in Fig. 9)
4	Alteration to Release Table so that a lock on P102 is released one segment too early	171,320	Violation trace with 23 events found, leading to a collision
5	B machines, CTRL and crossings considered	$> 1M$, 22 hours	Terminated
6	CSP CTRL constrained to reduce state space	288,828	No violation

Fig. 7: Scenarios checked for the Double Junction.

$lockTable = \{A3 \mapsto P101, B3 \mapsto P101, B3 \mapsto P102, A4 \mapsto P101, A4 \mapsto P102, A16 \mapsto P10\}$
replaced by the following: ($B3 \mapsto P102, A4 \mapsto P101$ missing)
$lockTable = \{A3 \mapsto P101, B3 \mapsto P101, A4 \mapsto P102, A16 \mapsto P102\}$

Fig. 8: Error in obtaining of locks modelled in *ControlTable*.

5 Discussion

We have seen that the CSP||B architecture has been resilient to more complex interlocking requirements, and has supported the model-checking of collision-freeness under these extensions. The relatively small state spaces indicate that our modelling approach is a good fit to the problem. Future work will extend the analysis to handle emergency stops (trains passing red signals and stopping in the following track segment) by extending the driving rules. We will also include points moving under trains more explicitly in the model. As well as safety, the model is suitable for analysing the capacity of the track plan: the maximum number of trains it can hold without compromising safety.

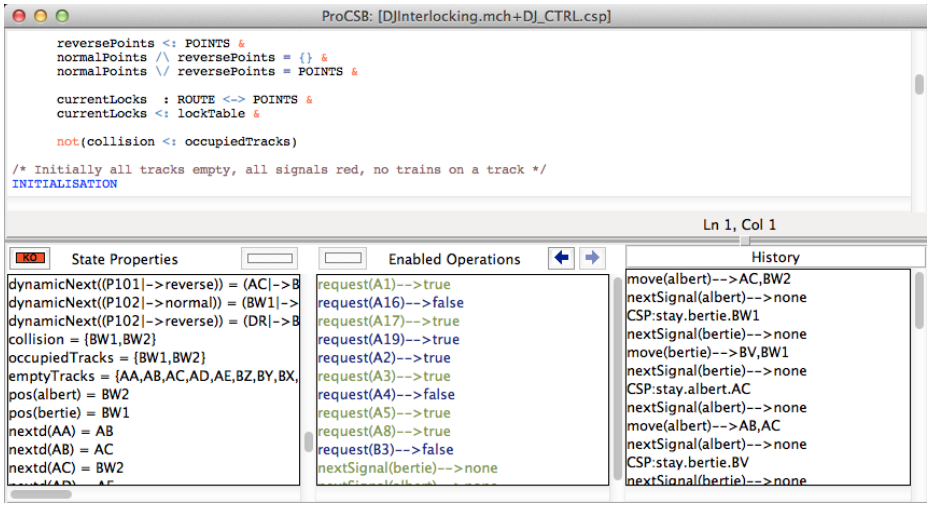


Fig. 9: Screen Shot of Invariant Violation for Scenario 3: Error in Obtaining Locks.

References

1. J-R. Abrial, *Modelling in Event B*, CUP, (2010).
2. Y. Isobe, H. Nguyen, M. Roggenbach, F. Moller, *Safety and line capacity in railways - An approach in Timed CSP,* IFM (2012) (to appear).
3. M. Leuschel, M. Butler, *ProB: an automated analysis toolset for the B method,* Int. J. Softw. Tools Technol. Transf. **(10)**2, 1433–2779 (2008).
4. F. Moller, H. Nguyen, M. Roggenbach, S. Schneider, H. Treharne, *Combining event-based and state-based modelling for railway verification,* Tech. Rep. CS-12-02, University of Surrey (2012).
5. A. Russo, L. Ladenberger, *A Formal Approach to Safety Verification of Railway Signaling Systems*, RAMS (2012).
6. D. Sabatier, *Formal proofs for the NYCT line 7 (Flushing) modernization project*, ABZ (2012) (to appear).
7. S. Schneider, H. Treharne, *CSP theorems for communicating B machines*, Formal Asp. Comput. **(17)**4,390–422 (2005).