International Journal of Computer Applications (0975 - 8887)
Volume 146 - No., July 2016*

Identifying Conflicting Routes in Control Table of Indian Railways Interlocking System Using NuSMV

K Sriram
SSN College of Engineering
Rajiv Gandhi Road, Kalavakkam
Chennai
sriram1433@cse.ssn.edu.in

S Sheerazuddin
SSN College of Engineering
Rajiv Gandhi Road, Kalavakkam
Chennai
sheerazuddins@ssn.edu.in

ABSTRACT

A "control table" is a functional specification of the signalling system of a railway section. It specifies the routes on which passage of trains is allowed. Control tables for interlocking system in Indian Railways is done by a vendor. These control tables are verified for correctness by another vendor. The process followed to generate these control tables remains a black box, i.e., unknown. Through this paper, we propose a working system to explore this unknown process and come up with correct control tables for a given layout of railway section.

General Terms

Railway Interlocking, Formal Methods

Keywords

Railway Interlocking, Formal Methods, Model Checking, NuSMV

1. INTRODUCTION

A railway station or a railway section is represented as a layout diagram. This layout diagram consists of track segments connected with one another, along with signals, points and level crossings. Signals convey the information regarding operation of train or the track on which the train is set to move, to the driver. Points are the intersection of two track segments. It is used to turn a moving train from one track line to another track line. Level Crossings are the control gates that co-ordinate the movement between road transport and railway without collision.

A "control table" is a functional specification of the signalling system of a railway section. It specifies the routes on which passage of trains is allowed. An interlocking system implements this control table making sure that the running of trains is safe, i.e., without collision and derailment.

Control tables for interlocking system in Indian Railways is done by a vendor. These control tables are verified for correctness by another vendor. The process followed to generate these control tables remains a black box, i.e., unknown.

Through this paper, we propose a formal methods based working system to explore this unknown process and come up with correct control tables for a given layout of railway section. We use NuSMV, a model checking tool developed by CMU, to generate entries in the control tables thereby ensuring their correctness. This paper reports the initial results of our research. We propose to follow it up with the verification of interlocking system being used in Indian Railways.

The paper is strucured as follows. The next section describes the related work done in the field of application of formal methods to generation and verification control tables for various railway systems in the world. In the third section, we give a very brief introduction to railways and its interlocking system. Section 4 describes our research with a case study in great detail. We round off with a concluding section at the end.

2. RELATED WORK

There have been many attempts to apply formal methods to railways and their associated interlocking systems. Indeed, this is the subject of the **TRain** *Grand Challenge* [1] proposed by Dines Bjorner.

In [8], the authors use SAT-based technique to verify interlocking system of Westinghouse Rail Systems, UK. They implement a prototype that takes as input: the interlocking system (defined using ladder logic), a model of the railway section, and a a signalling principle; if a counter-example is identified, the system provides a LaTeX document detailing the state of the interlocking system when the counter-example appears. They have also defined a formal language in which one can precisely represent signalling principles. The safety conditions over which ladder logic-based interlocking system is to be verified are derived from these signalling principles.

In [12], the author uses CSP [3] to model the Queensland railway interlocking system. This formal model is checked against the safety requirements, derived from the signalling principles, using the corresponding model checker FDR [6].

In [13], Winter et. al., describe an automated tool for design and verification of control tables. The semantics of control tables is modelled by means of the formal notation ASM (Abstract State Machines) [2], developed by Yuri Gurevich, which is easier to read and understand. The formal model is automatically transformed into NuSMV code using the tool support developed by Del Castillo and Winter [4]. The safety requirements to be checked are modelled in CTL [5], the temporal logic supported by NuSMV. They also propose many optimizations to the ASM model to improve efficiency of the model checking process. Further, they provide a

procedure for decomposing large interlocking systems into smaller ones, which along with model optimizations makes model checking feasible for realistic case studies.
[9] describes a preliminary toolset for automatic generation and verification of control tables for Iranian Railways. In this paper, the semantics of control table is formally modelled as state machines and the safety properties are verified, as usual, using NuSMV tool.
In [7], the authors describe experiments in model checking safety properties of interlocking system of Czech Railways. They use three different tools, NuSMV, Lesar and Mocha. Also, they automatically extract test sequences for the interlocking system using the counter example generation mechanism of these model checking tools.
In [11], the control tables for State Railway of Thailand are modelled using Colored Petri Nets (CPN). The CPN model comprises of two components namely *Signalling Layout Model* and *Interlocking Model*. ML functions are used on arc inscription in the Interlocking Model. These ML functions are generated directly from the content of control table using Extensible Stylesheet Language Transformations (XSLT). CPN Tools are used to check safety properties via a state space search.
[10] describes the use of formal methods for modelling interlocking systems in Turkish Railways. They discuss how automata and petri nets can be used to model the design of safety critical interlocking software which is a part of Turkish National Railway Signalization Project.

3. RAILWAYS AND INTERLOCKING

Railways are split up into railway sections i.e., train stations and depots and open lines connecting the sections. An example railway section is presented in Figure 1. This paper focuses on generating correct interlocking systems controlling railway sections.
A railway section is made up of the following components:
Track Segments. Train lines are split up into segments, and each segment is associated with a track circuit which can detect if a train is on the segment.
Signals. Signals are placed between track segments, and a signal is only visible from one direction. Signals show different aspects; these aspects inform the train driver about the state of the line ahead.
Points. Points are a special type of track segment used to merge two lines into one line. A train can drive over a set of points if it has been locked, i.e. reached a definite position, and has been so locked into position physically and virtually by software. The two possible positions of a set of points, when it is locked, are called normal and reverse. The normal position is when the points allow trains to travel straight over the points and reverse is when the points allow trains to branch off of, or on to, the line. Each set of points in a railway section is given a unique identifier in addition to the unique track segment identifier.
Routes. Routes consist of a sequence of sequentially-connected track segments that begin and end at signals, possibly through a set of points. Routes are defined by control tables which are created when a railway section is designed. Routes can be set to indicate that a train is using or about to use the route.
Track plans, such as presented in Figure 1, describe how these components are topologically configured. The operation of the various components in a railway section is defined using control tables. These contain information about when a route can be set, positions of the points, and the aspect a signal should display. Control tables are responsible for enforcing the signalling principles.
Railway interlocking systems are designed to implement the constraints in the control tables. A partial control table for the given layout is presented in the table 1. We shall discuss in detail the above railway section layout and its control table in Section 4.

4. IMPLEMENTATION AND CASE STUDY

In implementing a solution to this problem, understanding the signalling system of Indian Railways requires much importance. The railway section or railway station is represented on a layout. The layout consists of track segments, signals and points. The track segments are segments of track line. A train moves on a track by occupying and releasing a track segment. Signals provide the visually encoded information for the train drivers. They specify information like the speed at which the train must move, track on which the train has been allocated to move. Points are the intersection points of two track segments that help a train to change between track lines. A point can be set as *normal* or *reverse*.
Every track segment has an unique ID in a given layout. They may also have one or more labels to be used for specifying route through the railway section. Signals have ID to denote them uniquely. Points also have unique ID to be specified. A route is a sequence of track segments starting from a signal. It also contains details about points concerned with the route, to be whether normal or reverse.
An example railway section layout has been given in Figure 1. In this example, there are two track lines with 17 track segments, 4 points, 14 signals and 9 labels. To give this layout as input, it must be specified using a formal model such as graph. All the labels and ID are specified as attributes of vertices and edges in the graph.
This layout consists of 5 different types of signals namely, Calling on home, Home, Shunt, Starter and Advanced starter. The 14 signals can be classified into these types. 1B and 32B are calling on home. 1A and 32A are home. 9 and 17 are shunt. 3, 2, 6, 30, 31 and 27 are starter. 8 and 25 are advanced starter. The starter signals can be further classified into main line starter(2 and 31) and loop line starter(3, 6, 30 and 27). A train can move either towards up or towards down through a railway section. Hence it is important to provide signalling for both the directions.
The points 50, 52, 63 and 65 allow a train to move from main line to loop line or vice versa. A main line is the sequence of track segments that can allow a train to move from one railway section to another. A loop line is a sequence of track segments that branches from main line. When calling on home and shunt signal are given to the train driver, he need not stop the train at the specified track segment. The difference between calling on home and shunt signal is that the train moving on calling on home signal will have higher speed than shunt signal. The train following shunt signal will be moving at a very low speed of 10-20 kmph. Also, for train following shunt signal, there will be a shunter person to monitor the train movement which is not in calling on home.
The above layout is represented as a bidirectional graph to make it easier for a program to parse and identify the various aspects of a railway section. The track segments are represented as vertices using circular nodes. An edge connects two vertices if a train can move from one track segment to another. Labels for track segments are represented using elliptical nodes. Signals at a track segment are represented using rectangular nodes. Point IDs for specific track segments are represented using triangular nodes. The graph for above layout is given in Figure 2.
The graph as shown in Figure 2 is represented as text in a file and given as input to the program. The program parses the input file and recognizes the graph as adjacency list. Other information such as signals, point IDs and track labels are also parsed and recognized.

The input file consists of 6 sections, each separated by the keyword 'END'.
The first section consists of neighbouring vertices where the first vertex on left denotes a track segment on the up side of the railway section with respect to the other track segment. The second section consists of the impossible path that a train can take at a point. Also, the point ID of a point is given with this info. The third section consists of signal information and the track segment at which it is positioned. The fourth section consists of labels given for track segments. If a label corresponds to platform at which a train must stop, it is given with an extra information '-P'. The fifth section consists of track info of track segments to which they belong to. The sixth section consists of signal and labels for which the control table needs to be generated. These denote the route end points as the control table will be generated. This input format is given below.

```
C1T 25T
25T 1T
1T 50BT
50BT 50AT
50BT 52BT
52BT 52AT
52BT 02T
02T 63BT
50AT 01T
01T 63AT
63AT 63BT
63BT 65BT
52AT 03T
03T 65AT
65AT 65BT
65BT 32T
32T 8T
8T C32T
END
52BT 50BT 50AT 50
02T 52BT 52AT 52
02T 63BT 63AT 63
63BT 65BT 65AT 65
END
C1T (1B down calling_on_home)
C1T (1A down home)
1T (25 up advanced_starter)
1T (9 down shunt)
01T (30 up starter)
02T (31 up starter)
03T (27 up starter)
01T (3 down starter)
02T (2 down starter)
03T (6 down starter)
32T (17 up shunt)
32T (8 down advanced_starter)
C32T (32A up home)
C32T (32B up calling_on_home)
END
C1T J
1T H
01T A A1-P
02T B
03T C C1-P
32T K
C32T L
END
```

```
T1 C1T 25T 1T 50BT 52BT 02T 63BT 65BT 32T 8T C32T
T2 50AT 01T 63AT
T3 52AT 03T 65AT
END
1A A A1 B C C1
1B A B C
2 K
3 K
6 K
8 L
9 A B C
17 A B C
25 J
27 H
30 H
31 H
32A A A1 B C C1
32B A B C
```

The program which generates the control table for a given layout is built using Ruby v1.9.3 programming language on Ubuntu 14.04 LTS OS. The program follows the architecture as shown in Figure 3. The Routes Generator module, makes use of a slightly modified version of Depth First Search Algorithm to come up with routes. Control Table Generator module, analyses through the generated routes to find out what must be the status of points for a train to follow the route. The consistent routes combination verifier, makes use of NuSMV model checker to verify if two trains following two different routes, will have a collision or not. A basic check must be made to ensure that no two routes should have different status of point which is common between them. For example, routes 1A-A and 1A-A1 cannot be enabled at same time, as the point 63 is reverse for 1A-A and normal for 1A-A1.

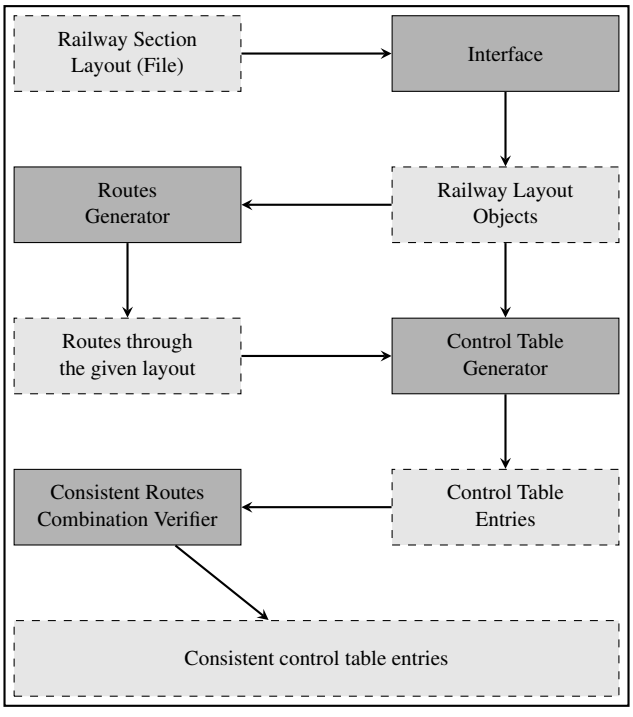

Fig. 3. Project Architecture

As the trains following calling on home and shunt signals differ from the trains following other signals, there is a need to come up with two different types of models. As an example, lets build models for 1A-A and 9-A, as shown below in Figures 4 and 5. The point 63 is set to reverse by 1A-A and so the train following 9-A can move till the end. If another route, for example 32A-C1 is considered, it requires the points 52 and 65 to be in normal and reverse respectively, making the train following 9-A route, to stop at 63AT.
This model is represented in NuSMV program. The routes are in the form of modules. They are allowed to move as synchronous non-deterministic transition model from the main module.
The condition to be checked for non collision to ensure safety is given by the below LTL formula, where *train_1* is the train following a route and *train_2* is the train following a different route.

G !(train_1.track_id = train_2.track_id)

In this manner, all the routes are verified for safety and the conflicting routes are identified from the given railway section layout.

5. CONCLUSION AND FUTURE WORK

In this paper, we have shown that a NuSMV program can effectively detect collisions between two routes that can practically be set at a given time. This allows to identify conflicting routes easily for a given railway section layout of Indian Railways as a graph represented in a file. Thus, we are able to correctly generate control entries for Indian Railway interlocking systems.
As future work, approach lock and back lock entries, as shown in Table 1 can also be generated for the control table. This can be used to simulate for actual behaviour of trains through the given railway section layout verifying for non-collision as safety measure.
The signalling principles that guide the generation of control table entries are currently hardwired in the implementation. Usually, these principles vary with different Railway systems. In fact, they differ even within the zones of Indian Railways. We propose to develop a formal language to describe these signalling principles.
Acknowledgement: We profusely thank Dr. V. Poornachandra Rao, Chief Engineer (Retd.), Indian Railways Institute of Signal Engineering and Telecommunications, for teaching us the intricacies of Indian Railway signalling system and also guiding this research.

6. REFERENCES

[1] Dines Bjørner. Train: The railway domain - A "grand challenge" for computing science & transportation engineering. In *Building the Information Society, IFIP 18th World Computer Congress, Topical Sessions, 22-27 August 2004, Toulouse, France*, pages 607–611, 2004.

[2] Egon Börger and Robert F. Stärk. *Abstract State Machines. A Method for High-Level System Design and Analysis*. Springer, 2003.

[3] Stephen D. Brookes, C. A. R. Hoare, and A. W. Roscoe. A theory of communicating sequential processes. *J. ACM*, 31(3):560–599, 1984.

[4] Giuseppe Del Castillo and Kirsten Winter. Model checking support for the ASM high-level language. In *Tools and Algorithms for Construction and Analysis of Systems, 6th International Conference, TACAS 2000, Held as Part of the European Joint Conferences on the Theory and Practice of Software, ETAPS 2000, Berlin, Germany, March 25 - April 2, 2000, Proceedings*, pages 331–346, 2000.

[5] E. Allen Emerson. Temporal and modal logic. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 995–1072. 1990.

[6] Thomas Gibson-Robinson, Philip J. Armstrong, Alexandre Boulgakov, and A. W. Roscoe. FDR3 - A modern refinement checker for CSP. In *Tools and Algorithms for the Construction and Analysis of Systems - 20th International Conference, TACAS 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014. Proceedings*, pages 187–201, 2014.

[7] Tomas Hlavaty, Libor Preucil, and Petr Stepan. Case study: Formal specification and verification of railway interlocking system. In *27th EUROMICRO Conference 2001: A Net Odyssey, 4-6 September 2001, Warsaw, Poland*, pages 258–263, 2001.

[8] Karim Kanso, Faron Moller, and Anton Setzer. Automated verification of signalling principles in railway interlocking systems. *Electr. Notes Theor. Comput. Sci.*, 250(2):19–31, 2009.

[9] Ahmad Mirabadi and Mohammad Bemani Yazdi. Automatic generation and verification of railway interlocking control tables using fsm and nusmv. In *Engineering Modelling 21 (2008) 1-4*, pages 57–63, 2008.

[10] Mehmet T. Sylemez, Mustafa S. Durmu, Uur Yldrm, Serhat Trk, and Arcan Sonat. The application of automation theory to railway signalization systems: The case of turkish national railway signalization project. In *18th IFAC World Congress, 2011, Italy*, pages 10752–10757, 2011.

[11] Somsak Vanit-Anunchai. Experience using coloured petri nets to model railway interlocking tables. In *Proceedings 2nd French Singaporean Workshop on Formal Methods and Applications, FSFMA 2014, Singapore, 13th May 2014.*, pages 17–28, 2014.

[12] K. Winter. Model checking railway interlocking systems. In *Computer Science 2002, Twenty-Fifth Australasian Computer Science Conference (ACSC2002), Monash University, Melbourne, Victoria, January/February 2002*, pages 303–310, 2002.

[13] Kirsten Winter and Neil J. Robinson. Modelling large railway interlockings and model checking small ones. In *Computer Science 2003, Twenty-Sixth Australasian Computer Science Conference (ACSC2003), Adelaide, South Australia, February 2003*, pages 309–316, 2003.

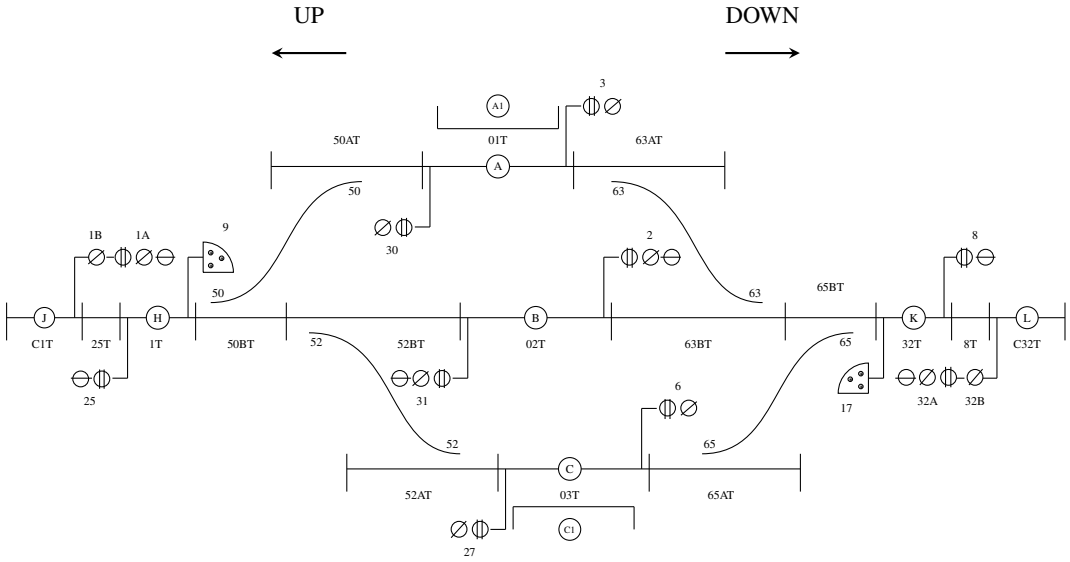

Fig. 1. A Railway Section

Table 1. Partial Control Table for Railway Section Layout in Figure 1

Route No.	**Locks & Detect Points**	**Lock Routes**	**Controlled by Tracks**	**App. Tracks**	**Back Lock Tracks**
$1A-A$	$50R, 63R, 65N$	$1B - A, 9 - A, 17 - A, 25 - J, 30 - H, 32A - A, 32B - A$	$25T, 1T, 50BT, 50AT, 01T, 63AT, 63BT, 65BT, 32T$	120 sec	$25T, 1T, 50BT, 50AT$
$1A-A_1$	$50R, 63N$	$1B - A, 9 - A, 17 - B, 17 - C, 25 - J, 30 - H, 32B - B, 32B - C$	$25T, 1T, 50BT, 50AT, 01T, 63AT$	120 sec	$25T, 1T, 50BT, 50AT$
$1A-B$	$50N, 52N, 63N, 65N$	$1B - B, 9 - B, 17 - B, 25 - J, 31 - H, 32A - B, 32B - B$	$25T, 1T, 50BT, 52BT, 02T, 63BT, 65BT, 32T$	120 sec	$25T, 1T, 50BT, 52BT$

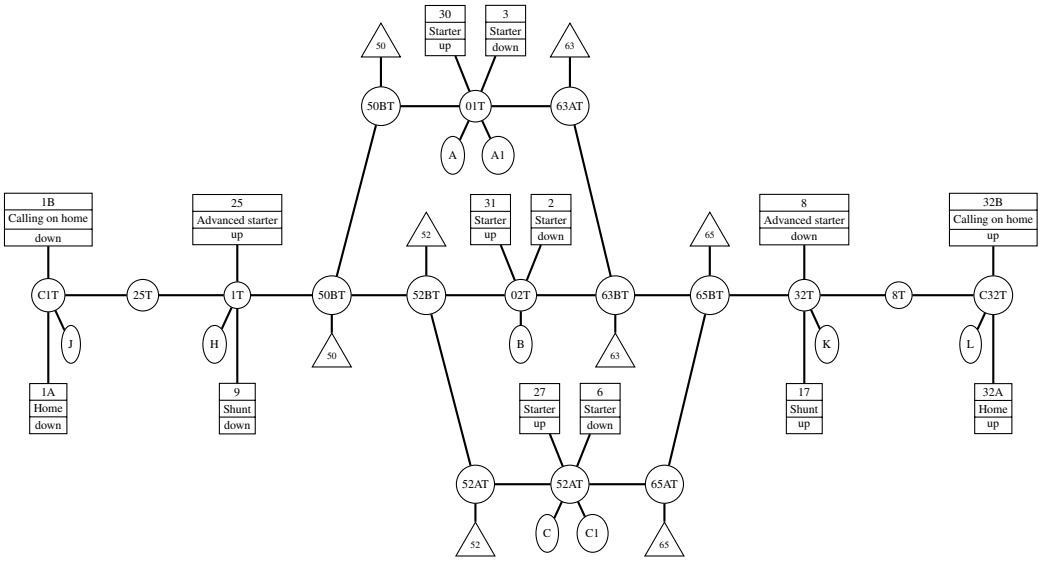

Fig. 2. Graph representation of railway section layout in Figure 1

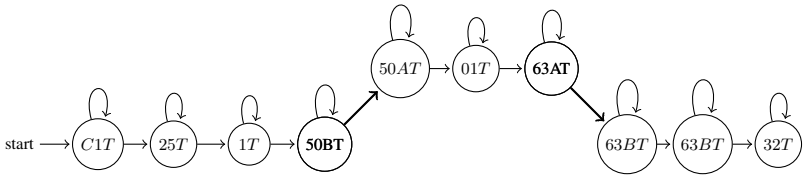

Fig. 4. Model of route 1A-A through the Railway Section

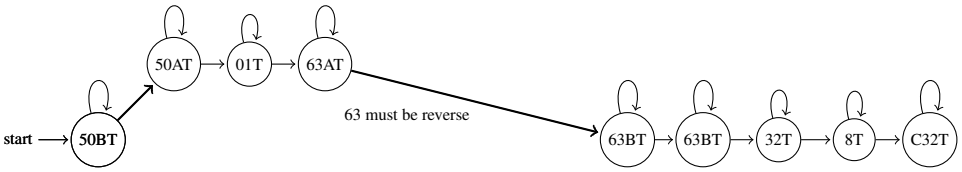

Fig. 5. Model of route 9-A through the Railway Section