2018 57th Annual Conference of the
Society of Instrument and Control Engineers of Japan (SICE).
September 11-14, 2018, Nara, Japan

Model Checking-based Safety Verification of a Petri Net Representation of Train Interlocking Systems

B. Aristyo[1], K. Pradityo[1], T.A. Tamba[2,3,†], Y.Y. Nazaruddin[1,3] and A. Widyotriatmo[1]

[1]Instrumentation and Control Research Group, Institut Teknologi Bandung, Bandung, Indonesia
(Tel: +62-22-250-4424; E-mail: yul@tf.itb.ac.id)
[2]Dept. of Electrical Engineering (Mechatronics), Parahyangan Catholic University, Bandung, Indonesia
(Tel: +62-22-203-2700; E-mail: ttamba@unpar.ac.id)
[3]National Center for Sustainable Transportation Technology, Bandung, Indonesia

Abstract: This paper describes an implementation of formal methods for safety verification of a railway interlocking system model based on model checking techniques. In the proposed method, a model for the interlocking system is constructed using timed-arc petri net and the safety specifications are expressed as computation tree logic formulas. These model and specification are then used in TAPAAL model checker to perform the model checking of safety verification task. Simulation results are presented to illustrate the advantages of the proposed verification method.

Keywords: Train interlocking system, safety verification, model checking, timed-arc petri net, computation tree logic.

1. INTRODUCTION

In railway systems domain, the notion of interlocking refers to an integration of several subsystems that work together and simultaneously to control the movements of trains in a station and/or between different stations. Based on the monitored information about the status or movements of objects (e.g. track topology, points, switches, signals and circuits) in the railway yard (i.e. the application data), the interlocking system uses computerized signaling programs to decide whether the routing of a particular train towards a particular track segment is allowed or denied. The resulting rule or decision system of such signaling principles is known as *control table* and serves as the functional specification by which a train interlocking system operates. It is then clear that the overal objective of an interlocking system is to ensure the safety and avoid the conflicting movements and routings of different trains at a particular location of the railway track.

Train interlocking system is categorized as a *safety-critical* system and as such the correctness of its design and implementation processes is of paramount importance [1, 2]. For instance, the EN50126 [3] and IEC61508 [4] standards set a safety integrity level 4 (SIL-4) requirement for the railway interlocking systems and require the use of formal methods for verifying the correctness of their functional specifications before being deployed in stations and tracks. Despite these requirements, current practices in verifying the railway interlocking have mainly relied on conventional methods such as exhaustive simulations/testings by experienced designers [1]. These current practices thus not only fail to meet the standards but are also costly, time consuming and error-prone.

Recently, various studies and implementations of formal verification techniques [2, 5] to railway interlocking systems have been reported. For overview regarding trends and challenges in formal design, specification and verification of train interlocking systems, refer to [6-14].

† T.A. Tamba is the corresponding author and presenter of this paper.

This paper reports some results of our current works in developing formal methods for safety verification of a railway interlocking system based on model checking techniques. In our work, the model of the interlocking system is constructed using *timed-arc petri net* (TAPN) [15] whereas the safety specifications are expressed using *computation tree logic* (CTL) formulas [5]. These choices of model and specification formula allow us to use off the shelf TAPAAL [16] model checker to perform the model checking of safety verification task. Simulation results are presented to illustrate the use of the proposed method in verifying railway interlocking systems.

2. PRELIMINARIES

2.1. Railway interlocking systems

2.1.1. Main components

The train interlocking consists of several components that are located in either in the main control station or rail yard. The components in the main control station include interlocking panel/interface, train blocks information, data logger and power source. The components in the yard include railway signals, track switches (or points or turnouts), train detection sensors, data transmission/communication systems and protection systems. Among these components, the switch, the signaling and the train detection systems play important roles and have direct impacts on the interlocking operation. These components are shown in Fig. 1 and briefly described below.

- A *switch* guides the transition of trains from one track to the other and may operates in either *normal* (allows the train through the main track) or *diverging* (allows the train diverge to another track) state. It is equipped with a *point machine* which moves the switch states from one to another (by moving the so-called *switch blade* laterally on the track), locks the switch while passed through by the train and performs the validation of the switch's state.
- A *signal* is used to communicate/exchange interlocking commands/rules between the on-board train driver and

the signaling operator at a control station. In modern railway infrastructures, a set of colour light signals (green, yellow, red) informs the on-board train driver about the occupancy status of adjacent track segments and guide the train movements along the prespecified routes (green means safe to proceed, yellow means proceed with caution and red means stop).

• Train *detection systems* refer to sensor instruments on the track that are used to detect the presence of the train in a particular track segment. Modern railway infrastructures mainly use *axle counter* for this purpose.

2.1.2. SIL requirement

Due to their safety-critical requirements, the safety of railway operation, control and protection systems has to be fulfilled in accordance to suitable documents and standards. One such standard is the EN50128 [3] which was issued by the European Committee for Electrotechnical Standardization (CENELEC). The EN50128 defines the interlocking system as a safety-critical for which the SIL-4 requirements need to be fulfilled. Furthermore, it also recommends the use of formal methods for the design, specification and verification of software and hardware components of railway interlocking systems.

2.2. Safety verification using model checking

One aspect of formal methods is formal verification which concerns with the verification of whether a system model satisfies certain properties (e.g. safety, nonblocking, etc.). Model checking is an algorithmic verification method for checking whether a system model satisfies certain properties [5, 17]. The model to be verified is typically described as transition systems whereas the properties to be checked usually contain dynamical aspects (i.e. depend on time) that are expressed as temporal logics.

There currently exist various computational tools to automatically perform the model checking process. These tools, known as *model checker*, exhaustively explore all possible evolutions (also called *runs* or *trajectories*) of the system model in an automatic manner to check the satisfaction of the specified properties in any of such runs. If the model satisfies the queried properties, the model checker returns a confirmation message. Otherwise, the model checker returns *counter examples* which contain part of the model's runs where such properties are violated (cf. Fig. 2). The advantages of using model checking thus lie not only on the fact that it performs the verification automatically (hence minimizes human errors) but also returns counter examples regarding the failure of the verification which later may be used to guide the model correction process or other tasks.

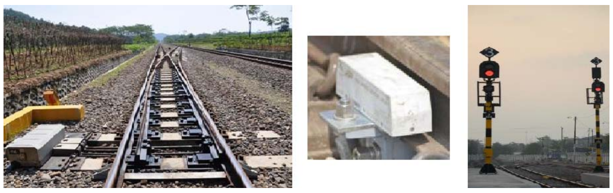
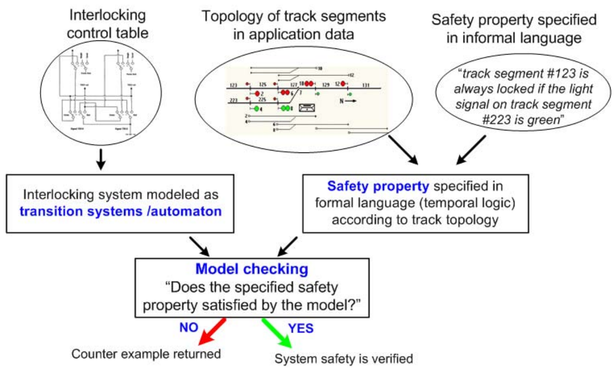


Fig. 1 Interlocking field components: *switch* (left), train detection sensor (center) and signals (right).



Interlocking control table
Topology of track segments in application data
Safety property specified in informal language
"track segment #123 is always locked if the light signal on track segment #223 is green"
Interlocking system modeled as transition systems /automaton
Safety property specified in formal language (temporal logic) according to track topology
Model checking
"Does the specified safety property satisfied by the model?"
NO
YES
Counter example returned
System safety is verified

Fig. 2 Model checking schematic for safety verification of train interlocking system.

This paper models the considered interlocking system by TAPN and expresses the (safety) properties to be verified as CTL. TAPAAL [16] is used for model checking.

2.2.1. Timed-arc petri nets (TAPN)

TAPN is class of *petri nets* model which includes temporal aspects. Formally, a (*labelled*) TAPN is a 6-tuple

$$\mathcal{N} = \{\mathcal{P}, \mathcal{T}, \mathcal{F}, c, \mathcal{A}, \lambda\} \tag{1}$$

where $\mathcal{P}$ is a finite set of *places*, $\mathcal{T}$ is a finite set of *transitions* such that $\mathcal{P} \cap \mathcal{T} = \emptyset$, $\mathcal{F} \subseteq (\mathcal{P} \times \mathcal{T}) \cup (\mathcal{T} \times \mathcal{P})$ is a *flow relation*, $c : \mathcal{F}|_{\mathcal{P}\times\mathcal{T}} \to \mathcal{I}$ is a *time constraint* which assigns a time interval to every *arc* from a place to transition, $\mathcal{A}$ is a set of *labels* (or *actions*) and $\lambda : \mathcal{T} \to \mathcal{A}$ is a *labelling function*. For the TAPN model in Eq. (1), we also associate the following two notations:

$$^{\bullet}t := \{p \,|\, (p,t) \in \mathcal{F}\}, \qquad t^{\bullet} := \{p \,|\, (t,p) \in \mathcal{F}\} \tag{2}$$

Let $\mathcal{B}(\mathbb{R}^+)$ denotes the set of finite multisets on nonnegative real numbers $\mathbb{R}^+$. Consider a function $\mathcal{M} : \mathcal{P} \to \mathcal{B}(\mathcal{R}^+)$ which we denote as the *marking* of the TAPN in Eq. (1). A marking allows us to assign a finite number of *tokens* to each place $p \in \mathcal{P}$ where each of such tokens is also annoted with a real number called *age*. We define initial markings $\mathcal{M}_0$ as markings with all tokens of age 0. We define the pair $(\mathcal{N}, \mathcal{M}_0)$ as a *marked TAPN*.

Consider a TAPN $\mathcal{N}$, a marking $\mathcal{M}$ and any $t \in \mathcal{T}$. The dynamics of $\mathcal{N}$ are defined by two types of transition rules, namely *firing* and *time-elapsing* transitions. Each of such transitions occurs if t is enabled. Note that t is said to be *enabled* by $\mathcal{M}$ if and only if for all $p \in^{\bullet} t$, there exists $x \in \mathcal{M}(p)$ such that $x \in c(p,t)$. A firing transition t on $\mathcal{N}$, denoted as $\mathcal{M}[t\rangle\mathcal{M}'$, occurs when the enabling of t by $\mathcal{M}$ produces a new marking $\mathcal{M}'$. A time-elapsing transition $\epsilon(r)$ on $\mathcal{N}$, denoted as $\mathcal{M}[\epsilon(r)\rangle\mathcal{M}'$, is induced by a time-delayed action $\epsilon(r)$ of length $r \in \mathbb{R}^+$ and occurs if for all $p \in \mathcal{P}$ and a new marking $\mathcal{M}'$ it holds that $\mathcal{M}'(p) = \mathcal{M}(p) + r$.

The dynamics of a marked TAPN $(\mathcal{N}, \mathcal{M}_0)$ induce a *labelled transition system* (LTS) $\mathbb{T}$ of the form

$$\mathbb{T}(\mathcal{N}, \mathcal{M}_0) \triangleq (S,\ s_0,\ \mathcal{A},\ \longrightarrow,\ \mathcal{O},\ \phi) \tag{3}$$

where $S := (\mathcal{P} \to \mathcal{B}(\mathbb{R}^+))$ is a finite set of states (defined as the markings of the TAPN $\mathcal{N}$), $s_0 := \mathcal{M}_0$ is an initial state (defined as the initial markings of $\mathcal{N}$), $\mathcal{A}$ is a set of actions with $\mathcal{A} \cap \mathbb{R}^+ = \emptyset$, and $\longrightarrow \subseteq S \times (\mathcal{A} \cup \mathbb{R}^+) \times S$ denotes a transition relation defined as follows

$$\mathcal{M} \xrightarrow{t} \mathcal{M}', \qquad \text{if } \mathcal{M}[t\rangle\mathcal{M}' \text{ for some } t \in \mathcal{T},$$

$$\mathcal{M} \xrightarrow{\epsilon(r)} \mathcal{M}', \qquad \text{if } \mathcal{M}[\epsilon(r)\rangle\mathcal{M}' \text{ for some } r \in \mathbb{R}^+.$$

The symbol $\mathcal{O}$ denotes a finite set of observations of $\mathbb{T}$ which is obtained under the observation map $\phi : S \to \mathcal{O}$.

2.2.2. Model checking of TAPN over CTL specifications

For the LTS in Eq. (3), we define a *trajectory* of $\mathbb{T}$ as an infinite sequence $s_0, s_1, s_2, \ldots$ which satisfies $s_i \in S$ and $(s_i, s_{i+1}) \in \longrightarrow$ for all $i \geq 0$. Each of such trajectories defines a *word* $w_0, w_1, w_2, \ldots$ through the observation map ϕ (i.e. $w_i = \phi(s_i)$). The set of all words that are generated by the set of all trajectories with initial state s_0 defines the *language* $\mathcal{L}_{\mathbb{T}}(s_0)$ of $\mathbb{T}$.

This paper uses CTL formulas to specify the properties of the LTS representation of the TAPN in Eq. (3). CTL formulas are defined over a set of *atomic propositions* Φ by using standard Boolean operators and a set of temporal operators (note that an atomic proposition is a statement or declarative sentence that must be true or false). Some of the standard Boolean operators used in CTL are: $\neg$ (negation), $\vee$ (disjunction), $\wedge$ (conjunction), $\Rightarrow$ (implication) and $\Leftrightarrow$ (equivalence). The temporal operators (combined with quantification over runs) in CTL generally take the form "QT" in which T $\in$ {X, F, G, U} specifies the temporal aspect (e.g. X (*next*), F (*finally*), G (*globally*), U (*until*)) and Q $\in$ {E,A} describes the quantification aspect of the formula (e.g. E (*there exists an execution*) and A (*for all executions*)). For instance, the formula "AG($\neg\,\Phi$)" means "the statement/proposition Φ never holds" (*invariance* property), whereas the formula "AG($\Phi_1 \Rightarrow$ AFΦ_2)" means "whenever proposition Φ_1 holds then eventually proposition Φ_2 will also hold" (*reactivity* property).

Given the LTS in Eq. (3) and a CTL specification Φ, the task of checking whether the Language $\mathcal{L}_{\mathbb{T}}(s_0)$ of $\mathbb{T}$ (originating at s_0) satisfies Φ is called *model checking* and often denoted as $\mathbb{T} \models \Phi$. There currently exist various off the shell model checking tools (called *model checker*) that can be used to automatically test if "$\mathbb{T} \models \Phi$" holds.

3. SYSTEM DESCRIPTION & MODEL

3.1. System description

The considered track layout with its corresponding interlocking components are depicted in Fig. 3. The layout consists of eight track segments which can generally be classified into the following two types of tracks:

- *through segments* which include segments with no intersections (segments [OSA], [OSB], [OSC] and [OSD]).
- *junction segments* which include segments with intersection (segments [OSJ1], [OSJ2], [OSJ3] and [OSJ4]).

Table 1 Allowed routes for the layout in Fig. 3.

Route	Seg.	P1	P2	P3	P4	Controlled segments
R1	**A-C**	N	N	D	N	[OSA],[OSJ1],[OSJ2], [OSJ3],[OSJ4],[OSC]
R2	**A-D**	N	D	N	N	[OSA],[OSJ1],[OSJ2], [OSJ3],[OSJ4],[OSD]
R3	**C-A**	N	N	N	N	[OSC],[OSJ1],[OSJ2], [OSJ3],[OSJ4],[OSA]
R4	**C-B**	N	N	D	D	[OSC],[OSJ1],[OSJ2], [OSJ3],[OSJ4],[OSB]
R5	**B-C**	N	N	D	D	[OSB],[OSJ1],[OSJ2], [OSJ3],[OSJ4],[OSC]
R6	**B-D**	N	N	N	N	[OSB],[OSJ1],[OSJ2], [OSJ3],[OSJ4],[OSD]
R7	**D-A**	N	D	N	N	[OSD],[OSJ1],[OSJ2], [OSJ3],[OSJ4],[OSA]
R8	**D-B**	N	N	N	N	[OSD],[OSJ1],[OSJ2], [OSJ3],[OSJ4],[OSB]

A train may enters or exits at edge points **A, B, C** and **D**. The transfer of a train between different segments is controlled signals s_i and switches W_j for $i, j = 1, \ldots, 4$.

The movement of a train through the layout should follow one of the eight allowed routes $\mathbf{R}\ell$ ($\ell = 1, \ldots, 8$) as specified in Table 1. In this table, the second column (Seg.) lists the initial-end segments, the third to sixth columns (P1–P4) specify the required position of the related switches (N = normal, D = diverging) and the last column lists all controlled segments in each route. In addition to this set of routes, the following assumptions are also used to constrain each train's operational scenario.

Assumption 1: Each train can only move forward

Assumption 2: Each segment can be occupied by train movement which starts from either left or right hand side.

Assumption 3: Each colour light signal only indicates the accessability of the following segment

This paper does not examines the train detection system and assumes the availability of reliable position data.

3.2. System modeling using TAPN

This section presents a TAPN model for the interlocking system described in Section 3.1. The model is constructed modularly whereby the TAPN of each interlocking component is first built and then combined together.

3.2.1. Switch module

In the TAPN model construction, a switch is assumed to be in a normal (N) initial condition. The switch is modeled to have the following six possible states (places):

- W#_Aktif: switch # is activated

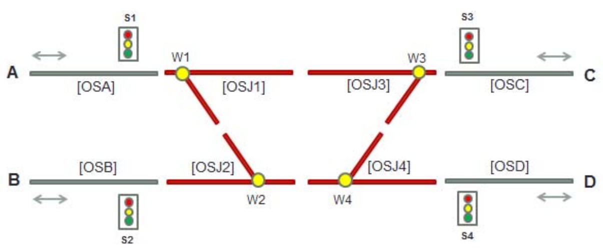


Fig. 3 Track layout with its interlocking components.

- W#_L: switch # is in N condition
- W#_B: switch # is in D condition
- W#_LkeB: switch # changes from N to D conditions
- W#_BkeL: switch # changes from D to N conditions
- OSJ#: occupancy status of junction segments

with # denotes an arbitrary switch tag number. Four possible transitions (T0, T1, T2, T3) between these states are also identified as illustrated on the TAPN graph of a switch W1 in Fig. 4. This particular graph illustrates that the switch W1 is initially in the N condition. The transition T0 of the switch to the W1_LkeB will be triggered if W1 has been activated and at the same time the segment [OSJ1] is not occupied. The W1_LkeB will then triggers transition T1 to W1_B, concluding the transition of switch W1 from N to D states. The reverse transition (D to N) may be defined in a similar manner.

3.2.2. Color light signal module

Our model considers a three-colors light (red, yellow, green) for the signal module of the interlocking system with the following six possible states to be used as places (# denotes an arbitrary light signal tag number).

- S#_Aktif: light signal # is activated to green
- S#_AktifKL: light signal # is activated to yellow
- S#_Merah: light signal # is red
- S#_Kuning: light signal # is yellow
- S#_Hijau: light signal # is green
- MSJ#: train enters first segment in the requested route

In our implementation, a green signal indicates that a train is allow to execute a route which only consists of through segments (e.g. **R1** or **R6**), whereas a yellow signal indicates that a train is allow to execute a route which consists of junction segments (e.g. **R2** or **R4**)

Fig. 5 shows the TAPN graph of the signal modul for a particular color light signal S1. The initial condition in this graph sets the light signal to be red. If a train request a route with through segments, then place S1_Aktif will be activated and the signal state change to S1_Hijau (green light). Similarly, If a train request a route with junction segments, then place S1_AktifK will be activated and the signal state change to S1_Kuning (yellow light). In both cases, the signal state will change to S1_Merah (red light) when the first segment in the requested route is currently being occupied by another train (as indicated by the place MSJ1 in the graph).

3.2.3. Route module

To model all possible executions of the interlocking system, a TAPN model for each route in Table 1 was developed. Fig. 6 shows the TAPN graph for a particular route **R1** (i.e. **A – C**). This graph illustrates the process of route request and the following activation of all necessary signals. For instance, the graph shows that this route can only be activated when its first and last controlled segments are not being occupied by other trains. Furthermore, the initial transition T0 in this graph utilizes several inhibitor arcs to ensure that this route cannot be activated whenever other routes having intersecting controlled segments are currently active. If these transition requirements are satisfied, then T0 is activated and the system state moves to place R1_Aktif which indicates that the route is activated and the corresponding switches and signals modules should be adjusted accordingly.

3.2.4. Track module

The track module essentially combines all the previously mentioned TAPN modules and is used to simulate the train movements from the initial to the end segments of a chosen route. In total, four track modules were constructed to cover all possible movements of the train with starting point from **A, B, C, D** on the layout in Fig. 3.

Fig. 7 shows the track module for routes starting at **A**. The initial place of this TAPN has two tokens to indicate that the layout may consists of at most two trains operating simultaneously. Examples instances where such simultaneous movements may occur are given by the route pairs (**R1**,**R6**), (**R1**,**R8**), (**R3**, **R6**) and (**R3**, **R8**). Whenever a train is in one of the initial segments ([OSA],[OSB], [OSC], [OSD]), it will request a specific route and then checks the occupancy status, the switch positions, and the signals states along those segments. In this regard, such

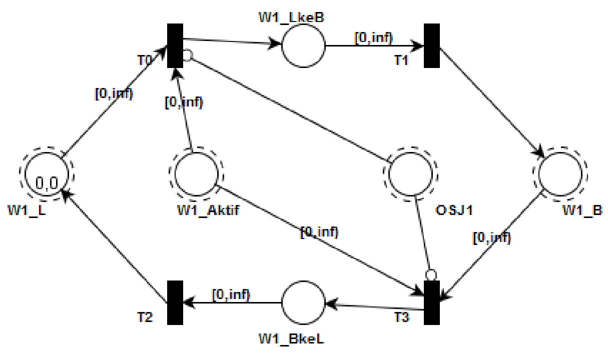


Fig. 4 TAPN model of the switch module.

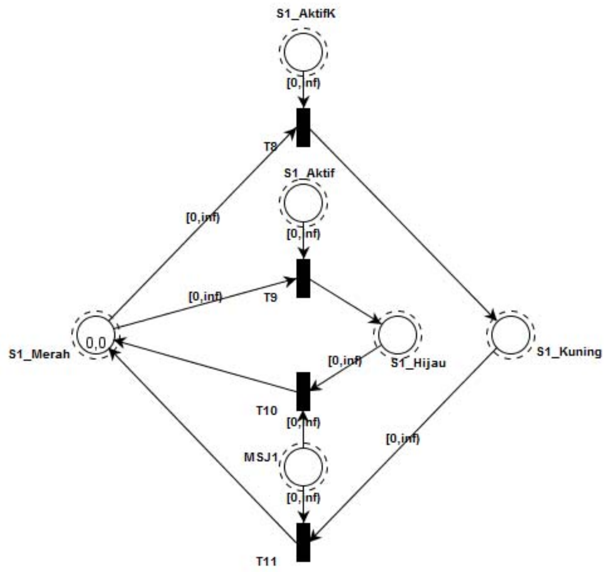


Fig. 5 TAPN model of the signal module.

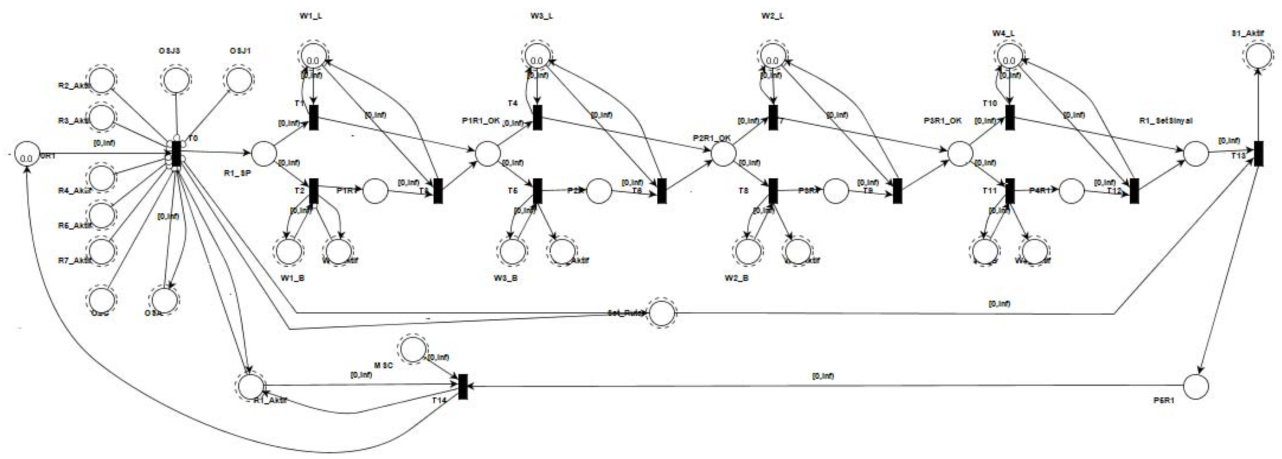


Fig. 6 TAPN graph for route R1.

checks can be performed using the logics in the TAPN model of the previously mentioned TAPN modules. Once the train completes the route, it will be deactivated to allow other trains to request it.

4. MODEL CHECKING AND SAFETY VERIFICATION RESULTS

4.1. Safety specifications

The safety specifications to be verified were adopted from an Indonesian government regulation document regarding the safety requirements for railway traffic and transport. Since the requirements in this document were written in standard language, they need to be translated into formal language in the form of CTL formula. Below, we list five examples of CTL specifications obtained after such a translation and used in model checking.

- Requirement 1: *at all time, each switch is never simultaneously in the normal (N) and diverge (D) positions*

This requirement can be expressed as the following specification Φ_1 on switch `W#` in TAPAAL:

$\Phi_1 =$ `AG!(W#_B=>1 and W#_L=>1)`

- Requirement 2: *at all time, each light signal never produce more than one color at the same time*

This requirement can be expressed as the following specification Φ_2 on switch `S#` in TAPAAL:

```
Φ2 = AG!((S#_Hijau=>1 and S#_Kuning=>1)
     or (S#_Kuning=>1 and S#_Merah=>1)
     or (S#_Hijau=>1 and S#_Merah=>1))
```

- Requirement 3: *when a route is activated, all related switches should be in the correct positions*

This requirement can be expressed as the following specification Φ_3 on route `R1` in TAPAAL:

```
Φ3 = EF(R1_Aktif=>1 and W1_L=>1 and
        (W3_L=>1)
```

- Requirement 4: *a yellow color signal indicates the route to acces has junction segment*

This requirement can be expressed as the following specification Φ_4 in TAPAAL:

$\Phi_4 =$ `EF(S1_Kuning=>1&W1_B>=1&W2_B>=1)`

- Requirement 5: *at all time, a segment should not be occupied by more than one train*

Table 2 Results of model checking in TAPAAL.

Spec. Φ_i	Verified?	Est. time	Est. memory
Φ_1	Yes	0.13 sec	<1 MB
Φ_2	Yes	0.09 sec	<1 MB
Φ_3	Yes	0.14 sec	59 MB
Φ_4	Yes	0.36 sec	5 MB
Φ_5	Yes	0.15 sec	<1 MB

This requirement can be expressed as the following specification Φ_5 in TAPAAL:

```
Φ5 = AG!(OSJ1=>2 or OSJ2=>2 or OSJ3=>2
         or OSJ4=>2 or OSA>=2 or OSB>=2
          or OSC=>2 or OSD>=2)
```

4.2. Model checking results

Table 2 summarizes the results obtained by model checker TAPAAL version 3.2.1. It can be concluded from the second column of this table that the constructed TAPN interlocking system model in Fig. 3 satisfies all of the queried safety specifications. The table also shows that the estimated time (Est. time) and memory (Est. memory) to verify each specification are relativeley small (i.e. the required time is less than 1 second and the maximum used memory is 59 MB). These results thus show the advantages of using model checking techniques for safety verification of railway interlocking models.

5. CONCLUSION

This paper has presented an approach to safety verification of railway interlocking systems based on model checking techniques. Simulation results were presented to illustrate the implementation of the proposed method.

ACKNOWLEDGMENT

This work was supported by the Indonesia's Ministry of Research, Technology & Higher Education (Kemenristekdikti) under 2018 Competency-Based Research scheme and also supported in part by USAID through the SHERA program under grant no. IIE00000078-ITB-1.

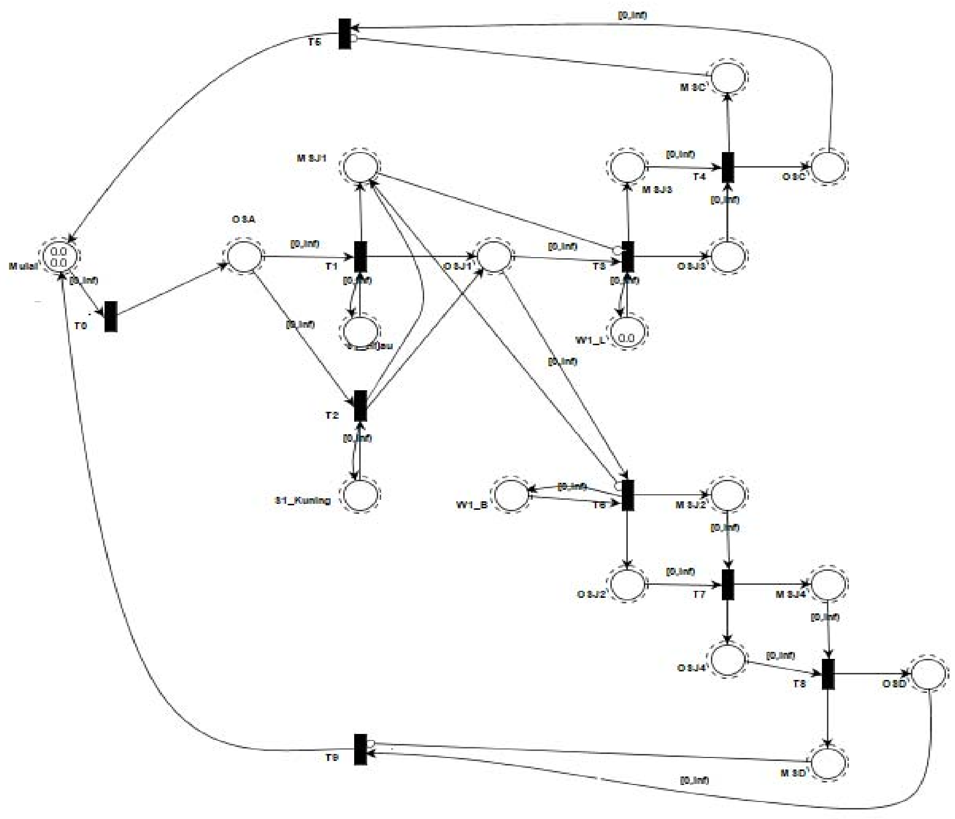


Fig. 7 TAPN graph for the track module.

REFERENCES

[1] S. Busard et al., "Verification of Railway Interlocking Systems," *Proc. Intl. Wksh. Engineering Safety and Security Systems*, pp. 19–31, 2015.

[2] A. Cimatti et al., "Formal Verification of a Railway Interlocking using Model Checking," *Formal Aspects of Computing*, Vol. 10, No. 4, pp. 361–380, 1998.

[3] https://standards.globalspec.com/std/1272146/cenelec-en-50126-1

[4] http://www.iec.ch/functionalsafety/standards/

[5] C. Baier, J.-P. Katoen, and K. G. Larsen, "Principles of model checking," *MIT Press*, pp. 492–497, 2008.

[6] W. J. Fokkink & P. R. Hollingshead, "Verification of Interlockings: from Control Tables to Ladder Logic Diagrams," *Proc. Wksh. Formal Methods for Industrial Critical Systems*, pp. 171–185, 1998.

[7] K. Winter et al., "Tool Support for Checking Railway Interlocking Designs," *Proc. Australian Wksh. Safety Related Programmable Systems*, pp. 101–107, 2005.

[8] K. Kanso, F. Moller, and A. Setzer, "Automated Verification of Signalling Principles in Railway Interlocking Systems," *Electronic Notes in Theoretical Computer Science*, Vol. 250, pp. 19–31, 2009.

[9] A. E. Haxthausen & J. Peleska, "Formal Development and Verification of a Distributed Railway Control System," *IEEE Transactions on Software Engineering*, Vol. 26, No. 8, pp. 687–701, 2000

[10] M. Durmus & M. Soylemez, "Railway Signalization & Interlocking Design via Automation Petri Nets," *Proc. Asian Control Conf.*, pp. 1558–1563, 2009.

[11] A. M. Hagalisletto, J. Bjrk, I. C. Yu, & P. Enger, "Constructing and Refining Large-Scale Railway Models Represented by Petri Nets," *IEEE Transactions on Systems, Man, and Cybernetics: Part C*, Vol. 37, No. 4, pp. 444–460, 2007.

[12] W.M.P van der Aalst & M. A. Odijk, "Analysis of Railway Stations by Means of Interval Timed Coloured Petri Nets," *Real-Time Systems*, Vol. 9, No. 3, pp. 1–23, 1995.

[13] A. Fantechi, W. Fokkink & A. Morzenti, "Some Trends in Formal Methods Applications to Railway Signaling," *Formal Methods for Industrial Critical Systems: Survey of Applications*, pp. 167–183, 2013.

[14] A. Fantechi, "Distributing the Challenge of Model Checking Interlocking Control Tables", *Proc. Symp. Leveraging Applications of Formal Methods, Verification and Validation*, pp. 276–289, 2012.

[15] M. Nielsen, V. Sassone & J. Srba, "Towards a Notion of Distributed Time for Petri Nets," *Proc. Conf. Application & Theory of Petri Nets*, pp. 23–31, 2001.

[16] A. David et al., "TAPAAL 2.0: Integrated Development Environment for Timed-Arc Petri Nets," *Proc. Conf. Tools & Algorithms for the Construction & Analysis of Systems*, pp. 492–497, 2012.

[17] A. Turnip & T.A. Tamba, "Algorithmic modeling and analysis of nonlinear biological systems," *Proc. Asian Control Conf.*, pp. 1–6, 2015.