Safety Criteria for Hoorn–Kersenboogerd Railway Station

Wan Fokkink
Utrecht University, Department of Philosophy
Heidelberglaan 8, 3584 CS Utrecht, The Netherlands
fokkink@phil.ruu.nl

Abstract

We formulate several classes of safety criteria for railway yards in terms of observable behaviour. These criteria are meant to protect trains from collisions and from derailments. We identify a number of safety criteria, and present instances of these classes for the case of the railway yard at station Hoorn–Kersenboogerd. These criteria have all been checked by means of the Stålmarck theorem prover, using a methodology from Groote, Koorn and Van Vlijmen.

1 Introduction

At a growing number of Dutch railway stations, including Hoorn–Kersenboogerd, computer equipment based on a Vital Processor Interlocking[1] (VPI) is used in order to ensure safe movement of trains. Apart from a number of hardware checks, a VPI essentially executes a program that consists of a large number of assignments of the form $v = \phi$ with v a variable and ϕ a Boolean formula, which expresses dependencies between objects such as points, signals and level crossings, taking into account detailed information such as delays of electrical devices. Each railway station is supplied with its own set of assignments, based on the particularities of the road map of the railway tracks. Properties of such lists of assignments is the focus of this paper.

Security in a railway yard is ensured using its Boolean formula as follows. Each second, the VPI executes a control cycle, which starts with reading new values for the input variables, which are determined by the environment. Next, these values are latched by the VPI, and they are used to compute the new values for the internal variables and for the output variables. Finally, the output values are transmitted to the outside world, where they are used in managing the signals, points and level crossings. After some idle time, in order to fill up the second, the VPI executes the next control cycle. A specification of VPI, together with the verification of several desirable properties of VPI, is presented in [4]

The production of a set of assignments for a specific railway yard is an involved human business, and even for a small railway station, the resulting set of assignments

[1] ® VPI and Vital Processor Interlocking are registered trademarks of the General Railway Signal Company.

is large and complicated. So a bottleneck in the strive for a hundred percent security of train movements guided by VPI, is the possible existence of flaws in the assignments. Hence, it is worthwhile to try and find a mechanized way to examine the assignments on the presence of defects.

In order to build a sound testing environment for this purpose, it is desirable to have a classification of safety requirements for railway yards, together with a description how to obtain the requirements for each class from a particular road map of railway tracks. This paper constitutes a first attempt to formulate such a classification of safety criteria, in the case of the comparatively simple road map at railway station Hoorn–Kersenboogerd.

We have discovered several classes of safety requirements. These safety criteria are all meant to protect trains from collisions and from derailments. For each class, we describe in detail which requirements are generated in the case of the road map at Hoorn–Kersenboogerd. Each requirement is supplied with ample informal comments, so that it should not be necessary to have a thorough knowledge of the notations and concepts that are used in the assignments, in order to grasp the safety requirements.

Our classification does not cover the full range of desirable safety criteria (see Section 7 on future research), that is, validity of these criteria does not ensure that trains will never collide nor derail. However, the safety criteria that are described in this paper do make a satisfactory collection for a first testing system to check the safety of a set of assignments. Experience learns that often, flaws in the assignments upset the validity of one of the safety requirements that are formulated in this paper. In particular, by automatic checking of our safety criteria, we may have spotted mistakes in the draft set of assignments for railway station Heerhugowaard.

Assume a set of assignments S, related to some railway yard. By taking the conjunction of all the assignments, S can be turned into a large Boolean formula Φ. Certain requirements involve time, such as 'if a section has been unoccupied for ten seconds, then ...'. Hence, we want to describe the dependencies between the objects in the railway yard in a period of time. Therefore we make copies $\Phi_0, \Phi_1, ..., \Phi_n$ of Φ, where Φ_i describes the dependencies at the railway yard i seconds ago. Finally, the conjunction $\Phi_0 \wedge \cdots \Phi_n$ is the desired Boolean formula, which describes the dependencies between the objects in the railway yard in the last n seconds.

For each safety requirement, we want to be sure that it holds in all circumstances. Thus, in the case of a particular requirement R, we desire that the Boolean expression $(\Phi_0 \wedge \cdots \Phi_n) \Rightarrow R$ holds for all possible values of variables in this expression. Experience learns that $\Phi_0 \wedge \cdots \Phi_n$ is in general too large to allow manipulations of such a Boolean expression on a computer with 'reasonable' capacity. Hence, as a first step, subformulas of $\Phi_0 \wedge \cdots \Phi_n$ that do not contribute to the value of R are removed by means of a slice algorithm [9], producing a, usually considerably smaller, formula Ψ. Finally, satisfiability of $\neg(\Psi \Rightarrow R)$ is checked by means of some theorem prover.

All the requirements that are presented in this paper, for the set of assignments for station Hoorn–Kersenboogerd, have been checked by means of the Stålmarck theorem prover. This tool can handle large Boolean formulas, by the application of smart algorithms for computations in classical logic. For information on innovative constructions that have been implemented in this theorem prover, see [6, 7]. In order to apply the Stålmarck theorem prover, we have used the Prolog interface NP Module [1]. We have also tried to check the safety requirements in an improved BDD based theorem prover [3], but requirements which involve time, such as 'if a signal shows red for one second', turned out to be too hard to handle for this tool.

Surprisingly, the small set of assignments for station Hoorn–Kersenboogerd that has been placed at the disposal of Utrecht University, seems to contain a bug. That is, some of the safety requirements which should hold in all circumstances, could not be proved to be correct for this set of assignments by the Stålmarck theorem prover.

The Stålmarck theorem prover has been applied before to verify interlocking equations in a computer controlled interlocking system used by the Swedish state railways [8]. A more general framework for the requirements analysis of safety critical systems has been proposed in [2].

Acknowledgements. Gea Kolk, Peter Musters and Robert Straatman from 'Holland Railconsult' provided valuable information on the technical subtleties of the assignments for Hoorn–Kersenboogerd. Jan Friso Groote and Bas van Vlijmen guided me into the world of VPI. Paul Berghege and Kiavash Shams helped to verify the safety requirements.

2 Notations and Basic Concepts

2.1 The road map at Hoorn–Kersenboogerd

Basically, a railway yard consists of a collection of linked railway tracks, supplied with features such as signals, points and level crossings. Figure 1 depicts a schematic view of the road map of the railway yard at Hoorn–Kersenboogerd. In this figure, the objects 52D, 62A-C, 66A-C, 69A-B, 70A-C, 73A-B, 74A-B denote tracks, the objects 60, 62, 64, 66, 68, 70, 72, 74 denote signals, the objects 69, 73 denote points, and object 35.0 denotes a level crossing.

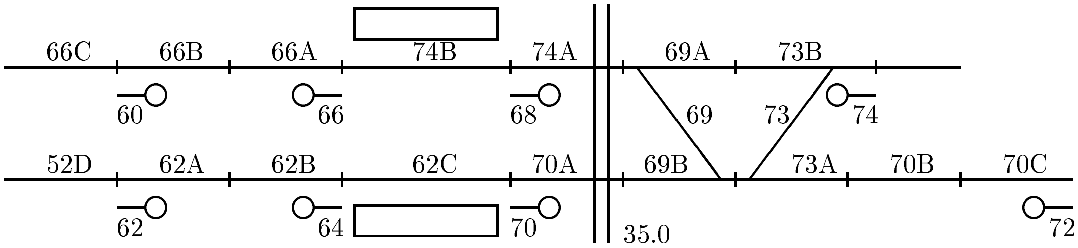

Figure 1: The road map at Hoorn–Kersenboogerd

2.2 Vital logic code

Each of the objects in a railway yard can attain a certain number of states:

- a railway track is either occupied or unoccupied,
- a signal shows either red or flashing yellow or yellow or green, possibly together with a number in order to impose a speed limit.
- a point is either in reverse or in normal position, or in neither of both,
- a level crossing is either open or closed, or neither of both.

In order to ensure absolute safety on the railway yard, certain combinations of states are to be avoided. For example, if a level crossing is open, then the track where it is situated must be unoccupied, in order to avoid a collision of a train with passing street traffic. In Vital Logic Code (VLC), such dependencies are expressed by means of Boolean expressions. First, each possible state of an object in the railway yard is represented in the form of a Boolean variable, which is true if and only if this state is attained. For example, the state 'track 70A is unoccupied' is represented by the variable 70A_TR_DI, and the state 'level crossing 35.0 is open' is represented by the variable 350_XR_DBO. Then dependencies between states can be expressed by means of Boolean expressions. For example, 'if level crossing 35.0 is open, then track 70A is unoccupied' becomes 350_XR_DBO $\Rightarrow$ 70A_TR_DI.

VLC incorporates three kinds of variables:

- input variables, whose values are determined by the environment,
- output variables, which determine the states at the signals, points and level crossings in the railway yard,
- internal variables, which together with the input variables are used to determine the values of the output variables.

A set of assignments is constructed from variables, the conjunction denoted by &, the disjunction denoted by #, the negation denoted by ~, and the implication denoted by $\Rightarrow$. (These notations are based on the notational conventions that are used in the Stålmarck theorem prover.) As a binding convention, negation binds stronger than conjunction and disjunction, which in turn bind stronger than implication.

2.3 Safety criteria

In order to build a sound testing environment for VLC, it is desirable to have a classification of safety requirements for railway yards. In this paper, we formulate several classes of safety criteria:

1. Dependency relations between signals; the aspect of a signal may yield restrictions on the range of aspects of a previous signal.
2. If a signal does not show red, then there is a *route* connected to this signal. That is, if a train passes this signal, the points behind this signal are positioned such that the train will reach the next signal without the possibility of being derailed at a point.

3. If a signal does not show red for one second, then the counter-signals in its route are to show red.

4. If a signal does not show red nor flashing yellow for one second, and if the route connected with this signal overlaps with the route connected with some other signal, then that other signal must show red.

5. If a signal does not show red nor flashing yellow for one second, then there is no train present on its route.

6. A point can only be reversed if it is unoccupied.

7. If a level crossing is open, then tracks near the level crossing must be either unoccupied, or separated from the level crossing by a red signal.

Given a railway yard, we can produce a specific set of safety requirements by taking instantiations of the classes above. As an example, we shall describe in detail which safety requirements are generated in the case of the road map at Hoorn–Kersenboogerd.

2.4 Description of variables

In order to obtain classes of safety requirements that are independent of the specific VPI, we will make sure to use input and output variables only in the safety requirements. In the safety requirements, we shall encounter the following kinds of input variables, which all end on DI (Direct Input).

- TR_DI and TRPR_DI.

  Variables of this form denote that a track is unoccupied. For example, 62C_TR_DI is true if there is no train present on track 62C, and 66A_TRPR_DI is true if there is no train present on track 66A. (The distinction in notation expresses a distinction in the number of available contacts at the relay which detects whether the track is occupied. This distinction is of no importance in VLC.)

- RWP_DI and NWP_DI

  Variables of this form express whether a specific point is in reverse or in normal position respectively. For example, 69_RWP_DI means that point 69 is in reverse position.

In the safety requirements, we shall encounter the following kinds of output variables, which end either on ACO (Alternating Current Output) or on DBO (Double Break Output).

- R_ACO and GLFL_ACO and GL_ACO and GR_ACO.

  Variables of this form denote the aspect at a specific signal. For example, 74_R_ACO or 74_GLFL_ACO or 74_GL_ACO or 74_GR_ACO is true if signal 74 shows red or flashing yellow or yellow or green respectively.

- XR_DBO.

  Variables of this form express that a certain level crossing is open.

- RWR_DBO and NWR_DBO.

  Variables of this form indicate that a certain point is being put into reverse or normal position respectively. (In theory, a point can be instructed to go into reverse and normal position at the same time. In practice, this will never be the case; it would cause the point to break down.)

In the safety requirements, it is sometimes necessary to know the value of a variable in the past. For example, this is the case in 'if a signal shows red for at least one second'. Such values are expressed by variables with the suffix J_*n*, yielding the value of the variable *n* seconds ago. For example, the variable 74_R_ACO_J_1 is true if and only if the signal 74 showed red one second ago.

3 Dependency Relations for Signals

In the next sections, we shall describe which are the safety requirements that are generated by the classes that were formulated in Section 2.3 in the specific case of Hoorn–Kersenboogerd railway station.

The aspect of a signal may cause restrictions on the range of aspects at previous signals. For example, if a signal shows red or flashing yellow, then in general the most liberal aspect for a previous signal is yellow.

The dependencies between signals are described on the 'OBJ page'. The requirements that are imposed by dependencies between signals can be copied from this OBJ page without any complications. In the case of Hoorn–Kersenboogerd, an example of such a requirement is: 'if signal 70 shows red or flashing yellow, then signal 68 must show either red or flashing yellow or yellow'. So in terms of Boolean notation, we obtain the following requirement:

```
70_R_ACO # 70_GLFL_ACO => 62_R_ACO # 62_GLFL_ACO # 62_GL_ACO
```

Copying dependencies between signals from an OBJ page is a routine matter. Hence, we omit the safety requirements that are yielded by this procedure.

4 Roadways

A *route*, connected to a signal, is the route that a train follows after it has passed the signal, until it reaches a next signal in the same direction. A route is determined by the position of the points on the track. If a signal does not show red, then there has to be a route from this signal to a next signal. That is, if a train passes this signal, the points behind this signal are positioned such that the train will reach the next signal without the possibility of being derailed at a point.

If a signal does not show red, then there must always be a route connected to this signal.

First, the requirements for signals with respect to routes are illustrated for signal 74. The requirements with respect to routes for the other signals are presented with less comments in the subsequent sections.

4.1 Signal 74

If signal 74 does not show red, then there must be some route from signal 74 to a following signal. That is, the points 73 and 69 are to determine a route from signal 74 onwards. There are three possible routes:

1. point 73 is reverse and point 69 is reverse,
2. point 73 is reverse and point 69 is normal,
3. point 73 is normal and point 69 is normal.

Note that the situation 'points 73 and 69 are both reverse' is not possible, because then a train passing signal 74 would ride open point 69. Thus, we obtain the following requirement.

```
~74_R_ACO => (73_RWP_DI & ~73_NWP_DI & ((69_RWP_DI & ~69_NWP_DI)
                                      # (69_NWP_DI & ~69_RWP_DI)))
          # (73_NWP_DI & ~73_RWP_DI & 69_NWP_DI & ~69_RWP_DI)
```

If signal 74 does not show red for at least one second, and if some counter-signal in the route of signal 74 does not show red, then that counter-signal must show red. Hence, we find the following requirements.

- Each possible route from signal 68 would cross each possible route from signal 74, so if signal 74 does not show red for at least one second, then signal 68 must show red.

  ```
  ~(74_R_ACO_J_1 # 74_R_ACO) => 68_R_ACO
  ```

- If both point 73 and point 69 are normal, then the routes from signal 74 and from signal 70 do not cross. However, if point 73 is reverse, then the routes from signal 74 and from signal 70 do cross. So if signal 74 does not show red for at least one second, and if point 73 is reverse, the signal 70 must show red.

  ```
  ~(74_R_ACO_J_1 # 74_R_ACO) & 73_RWP_DI => 70_R_ACO
  ```

If signal 74 does not show red nor flashing yellow for at leat one second, then routes which overlap with the route of signal 74 must all be guarded by a red signal. This leads to a second category of requirements for signals in relation with routes.

- Suppose that signal 74 does not show red nor flashing yellow for at least one second. Furthermore, let the route from signal 74 lead to signal 66, that is, let either point 73 be normal, or let both point 73 and 69 be reverse. Then the signal 60 must show red.

  ```
  ~(74_R_ACO_J_1 # 74_R_ACO # 74_GLFL_ACO_J_1 # 74_GLFL_ACO) &
   (73_NWP_DI # (73_RWP_DI & 69_RWP_DI)) => 60_R_ACO
  ```

- Suppose that signal 74 does not show red nor flashing yellow for at least one second. Furthermore, let the route from signal 74 lead to signal 64, that is, let point 73 be reverse and point 69 be normal. Then the signal 62 must show red.

  ```
  ~(74_R_ACO_J_1 # 74_R_ACO # 74_GLFL_ACO_J_1 # 74_GLFL_ACO)
   & 73_RWP_DI & 69_NWP_DI => 62_R_ACO
  ```

4.2 Signal 72

From signal 72 onwards there are two possible routes. Namely, if signal 72 does not show red, then point 73 must be normal (otherwise a train could ride open this point) and point 69 can be either reverse or normal.

```
~72_R_ACO => 73_NWP_DI & ~73_RWP_DI & ((69_RWP_DI & ~69_NWP_DI)
                                     # (69_NWP_DI & ~69_RWP_DI))
```

The first category of safety criteria for signal 72 with respect to routes consists of two requirements.

- Each possible route from signal 70 crosses each possible route from signal 72, so if signal 72 does not show red for at least one second, then signal 70 has to be red.

  ```
  ~(72_R_ACO_J_1 # 72_R_ACO) => 70_R_ACO
  ```

- If the points 73 and 69 are both normal, then the routes from the signals 68 and 72 do not cross. However, if point 69 is reverse, then the routes from the signals 68 and 72 do cross. So, if signal 72 does not show red for at least one second, and if point 69 is reverse, then signal 68 has to show red.

  ```
  ~(72_R_ACO_J_1 # 72_R_ACO) & 69_RWP_DI => 68_R_ACO
  ```

The second category of safety criteria for signal 72 with respect to routes also consists of two requirements.

- Suppose that signal 72 does not show red nor flashing yellow for at least one second. Furthermore, let the route from signal 72 lead to signal 66, that is, let point 69 be reverse. Then the signal 60 must show red.

```
~(72_R_ACO_J_1 # 72_R_ACO # 72_GLFL_ACO_J_1 # 72_GLFL_ACO)
 & 69_RWP_DI => 60_R_ACO
```

Suppose that signal 72 does not show red nor flashing yellow for at least one second. Furthermore, let the route from signal 72 lead to signal 64, that is, let point 69 be normal. Then the signal 62 must show red.

```
~(72_R_ACO_J_1 # 72_R_ACO # 72_GLFL_ACO_J_1 # 72_GLFL_ACO)
 & 69_NWP_DI => 62_R_ACO
```

4.3 Signal 70

From signal 70 onwards there are two possible routes. Namely, if signal 70 does not show red, then point 69 must be normal, and point 73 can be either reverse or normal.

```
~70_R_ACO => 69_NWP_DI & ~69_RWP_DI & ((73_RWP_DI & ~73_NWP_DI)
                                      # (73_NWP_DI & ~73_RWP_DI))
```

The first category of safety criteria for signal 70 with respect to routes consists of two requirements.

- A route from signal 74 crosses a route from signal 70 if point 73 is reverse. Hence, if signal 70 does not show red for at least one second, and if point 73 is reverse, then signal 74 has to be red.

  ```
  ~(70_R_ACO_J_1 # 70_R_ACO) & 73_RWP_DI => 74_R_ACO
  ```

- A route from signal 72 always crosses a route from signal 70. Hence, if signal 70 does not show red for at least one second, then signal 72 has to be red.

  ```
  ~(70_R_ACO_J_1 # 70_R_ACO) => 72_R_ACO
  ```

The second category of safety criteria for signals with respect to routes does not lead to any requirements for signal 70. Namely, the signals just behind the route of signal 70 are all outside the range of Hoorn–Kersenboogerd.

4.4 Signal 68

From signal 70 onwards there are three possible routes. Namely, if point 69 is reverse, then point 73 can be either reverse or normal, and if point 69 is normal, then point 73 has to be normal too.

```
~68_R_ACO => (69_RWP_DI & ~69_NWP_DI & ((73_RWP_DI & ~73_NWP_DI)
                                       # (73_NWP_DI & ~73_RWP_DI)))
           # (69_NWP_DI & ~69_RWP_DI & 73_NWP_DI & ~73_RWP_DI)
```

The first category of safety criteria for signal 68 with respect to route consists of two requirements.

- A route from signal 74 always crosses a route from signal 68. Hence, if signal 68 does not show red for at least one second, then signal 74 has to be red.

  ```
  ~(68_R_ACO_J_1 # 68_R_ACO) => 74_R_ACO
  ```

- A route from signal 72 crosses a route from signal 70 if point 69 is reverse. Hence, if signal 70 does not show red for at least one second, and if point 69 is reverse, then signal 72 has to be red.

  ```
  ~(68_R_ACO_J_1 # 68_R_ACO) & 69_RWP_DI => 72_R_ACO
  ```

The second category of safety criteria for signals with respect to routes does not lead to any requirements for signal 68. Again, the signals just behind the route of signal 68 are all outside the range of Hoorn–Kersenboogerd.

4.5 Signal 66

From signal 66 onwards, there are no points involved. Only the first category of safety criteria leads to a a requirement for signal 66. Namely, if signal 66 does not show red for at least one second, then signal 60 has to show red.

```
~(66_R_ACO_J_1 # 66_R_ACO) => 60_R_ACO
```

4.6 Signal 64

From signal 64 onwards, there are no points involved. Only the first category of safety criteria leads to a a requirement for signal 64. Namely, if signal 64 does not show red for at least one second, then signal 62 has to show red.

```
~(64_R_ACO_J_1 # 64_R_ACO) => 62_R_ACO
```

4.7 Signal 62

Between signals 62 and signal 70, there are no points present.

The first category of safety criteria leads to a one requirement for signal 62. Namely, if signal 62 does not show red for at least one second, then signal 64 has to show red.

```
~(62_R_ACO_J_1 # 62_R_ACO) => 64_R_ACO
```

The second category of safety criteria for signal 62 with respect to routes consists of two requirements. *These requirements are both invalid, according to the Stålmarck theorem prover.*[2]

[2] We have found that the two requirements become valid if we add the possibility that signal 72 or signal 74 shows flashing yellow respectively.

- Suppose that signal 62 does not show red nor flashing yellow for at least one second. Furthermore, let the route from signal 72 lead to signal 64, that is, let both point 73 and point 69 be normal. Then signal 72 has to show red.

  ```
  ~(62_R_ACO_J_1 # 62_R_ACO # 62_GLFL_ACO_J_1 # 62_GLFL_ACO)
   & 73_NWP_DI & 69_NWP_DI => 72_R_ACO
  ```

- Suppose that signal 62 does not show red nor flashing yellow for at least one second. Furthermore, let the route from signal 74 lead to signal 64, that is, let point 73 be reverse and let point 69 be normal. Then signal 74 has to show red.

  ```
  ~(62_R_ACO_J_1 # 62_R_ACO # 62_GLFL_ACO_J_1 # 62_GLFL_ACO)
   & 73_RWP_DI & 69_NWP_DI => 74_R_ACO
  ```

4.8 Signal 60

Between signals 60 and signal 68, there are no points present.

The first category of safety criteria leads to a one requirement for signal 60. Namely, if signal 60 does not show red for at least one second, then signal 66 has to show red.

```
~(60_R_ACO_J_1 # 60_R_ACO) => 66_R_ACO
```

The second category of safety criteria for signal 60 with respect to route consists of two requirements. *These requirements are both invalid, according to the Stålmarck theorem prover.*[3]

- Suppose that signal 60 does not show red nor flashing yellow for at least one second. Furthermore, let the route from signal 74 lead to signal 66, that is, let point 73 and point 69 both be normal or both be reverse. Then signal 74 has to show red.

  ```
  ~(60_R_ACO_J_1 # 60_R_ACO # 60_GLFL_ACO_J_1 # 60_GLFL_ACO)
   & ((73_NWP_DI & 69_NWP_DI) # (73_RWP_DI & 69_RWP_DI))
  => 74_R_ACO
  ```

- Suppose that signal 60 does not show red nor flashing yellow for at least one second. Furthermore, let the route from signal 72 lead to signal 66, that is, let point 73 be normal and let point 69 be reverse. Then signal 72 has to show red.

  ```
  ~(60_R_ACO_J_1 # 60_R_ACO # 60_GLFL_ACO_J_1 # 60_GLFL_ACO)
   & 73_NWP_DI & 69_RWP_DI => 72_R_ACO
  ```

[3] We have found that the two requirements become valid if we add the possibility that signal 74 or signal 72 shows flashing yellow respectively.

5 Occupation of Tracks

If a signal does not show red or flashing yellow, then there must be no trains present on its route.

5.1 Signal 74

In this section, we assume that signal 74 does not show red or flashing yellow.

The route from signal 74 always includes track 73B, so this track has to be unoccupied.

```
~(74_R_ACO # 74_GLFL_ACO) => 73B_TR_DI
```

If point 73 is normal, or if the points 73 and 69 are both reverse, then the route from signal 74 includes the tracks 69A and 74A and 74B. So these tracks have to be unoccupied.

```
~(74_R_ACO # 74_GLFL_ACO) & (73_NWP_DI # (73_RWP_DI & 69_RWP_DI))
 => 69A_TR_DI & 74A_TR_DI & 74B_TR_DI
```

If point 73 is reverse, then the route from signal 74 includes the tracks 73A and 69B. So these tracks have to be unoccupied.

```
~(74_R_ACO # 74_GLFL_ACO) & 73_RWP_DI => 73A_TR_DI & 69B_TR_DI
```

If point 73 is reverse and point 69 is normal, then the route from signal 74 includes the tracks 70A and 62C. So these tracks have to be unoccupied.

```
~(74_R_ACO # 74_GLFL_ACO) & 73_RWP_DI & 69_NWP_DI
=> 70A_TR_DI & 62C_TR_DI
```

5.2 Signal 72

In this section, we assume that signal 72 does not show red or flashing yellow.

The route from signal 72 always includes the tracks 70C and 70B and 73A and 69B, so these tracks have to be unoccupied.

```
~(72_R_ACO # 72_GLFL_ACO)
=> 70C_TRPR_DI & 70B_TR_DI & 73A_TR_DI & 69B_TR_DI
```

If point 69 is reverse, then the route from signal 72 includes the tracks 69A and 74A and 74B. So these tracks have to be unoccupied.

```
~(72_R_ACO # 72_GLFL_ACO) & 69_RWP_DI
=> 69A_TR_DI & 74A_TR_DI & 74B_TR_DI
```

If point 69 is normal, then the route from signal 72 includes the tracks 70A and 62C. So these tracks have to be unoccupied.

```
~(72_R_ACO # 72_GLFL_ACO) & 69_NWP_DI => 70A_TR_DI & 62C_TR_DI
```

5.3 Signal 70

Assume that signal 70 does not show red or flashing yellow.

Since the track behind signal 74 is a dead end, each route leading to this track has to be guarded by a signal that shows either red or flashing yellow. Hence, since signal 70 does not satisfy its restriction, its route must not lead to the tracks behind signal 74. In other words, point 73 must be normal.

```
~(70_R_ACO # 70_GLFL_ACO) => 73_NWP_DI
```

Hence, the route from signal 70 consists of the tracks 70A and 69B and 73A and 70B and 70C, so these tracks all have to be unoccupied.

```
~(70_R_ACO # 70_GLFL_ACO)
=> 70A_TR_DI & 69B_TR_DI & 73A_TR_DI & 70B_TR_DI & 70C_TRPR_DI
```

5.4 Signal 68

Assume that signal 68 does not show red or flashing yellow.

Again, under this circumstance the route from signal 68 is not allowed to lead to the track behind signal 74. This means that point 69 is reverse and point 73 is normal.

```
~(68_R_ACO # 68_GLFL_ACO) => 69_RWP_DI & 73_NWP_DI
```

Hence, the route from signal 68 consists of the tracks 74A and 69A and 69B and 73A and 70B and 70C, so these tracks all have to be unoccupied.

```
~(68_R_ACO # 68_GLFL_ACO) => 74A_TR_DI & 69A_TR_DI & 69B_TR_DI
 & 73A_TR_DI & 70B_TR_DI & 70C_TRPR_DI
```

5.5 Signal 66

Assume that signal 66 does not show red or flashing yellow. The route from signal 66, as far as Hoorn–Kersenboogerd is concerned, consists of the tracks 66A and 66B and 66C. So these tracks all have to be unoccupied.

```
~(66_R_ACO # 66_GLFL_ACO) => 66A_TRPR_DI & 66B_TRPR_DI & 66C_TR_DI
```

5.6 Signal 64

Assume that signal 64 does not show red or flashing yellow. The route from signal 64, as far as Hoorn–Kersenboogerd is concerned, consists of the tracks 62B and 62A and 52D. So these tracks all have to be unoccupied.

```
~(64_R_ACO # 64_GLFL_ACO) => 62B_TRPR_DI & 62A_TRPR_DI & 52D_TR_DI
```

5.7 Signal 62

Assume that signal 62 does not show red or flashing yellow. The route from signal 62 consists of the tracks 62A and 62B and 62C, so these tracks all have to be unoccupied.

```
~(62_R_ACO # 62_GLFL_ACO) => 62A_TRPR_DI & 62B_TRPR_DI & 62C_TR_DI
```

5.8 Signal 60

Assume that signal 60 does not show red or flashing yellow. The route from signal 60 consists of the tracks 66B and 66A and 74B, so these tracks all have to be unoccupied.

```
~(60_R_ACO # 60_GLFL_ACO) => 66B_TRPR_DI & 66A_TRPR_DI & 74B_TR_DI
```

5.9 Points 69 and 73

While a point is being put in reverse or in normal position, the point has to be unoccupied. For the points 69 and 73, this leads to the following requirement.

```
69_RWR_DBO # 69_NWR_DBO => 69A_TR_DI & 69B_TR_DI

73_RWR_DBO # 73_NWR_DBO => 73A_TR_DI & 73B_TR_DI
```

6 Crossings

If level crossing 35.0 is open, then track 74A has to be unoccupied, and either signal 68 shows red, or track 74B is unoccupied.

```
350_XR_DBO => 74A_TR_DI & (68_R_ACO # 74B_TR_DI)
```

Similarly, if level crossing 35.0 is open, then track 70A has to be unoccupied, and either signal 70 shows red, or track 62C is unoccupied.

```
350_XR_DBO => 70A_TR_DI & (70_R_ACO # 62C_TR_DI)
```

7 Summary and Future Research

We have presented several categories of safety criteria, in VLC, and we have worked out what specific criteria are generated from these categories in the comparatively simple case of Hoorn–Kersenboogerd railway station. These criteria have been verified using the Stålmarck theorem prover.

Thus, we have obtained the routine to produce safety criteria for railway stations in VLC. As a next step, this routine is to be applied to other railway stations, with a considerably larger set of assignments than the one for Hoorn–Kersenboogerd railway station. For example, at the moment we are occupied with producing the safety criteria for the much more complicated situation at Heerhugowaard railway station.

Ideally, the routine to produce safety criteria should result in an algorithm which can be implemented, so that the safety criteria for a railway station can be produced automatically. Namely, in the case of Hoorn–Kersenboogerd there are few railway tracks, so that the routine yields few requirements, which can be produced by hand. But the VPI has also been installed at stations where there are many more distinct routes, which leads to an inevitable explosion in the number of safety criteria.

An important question is whether the categories of safety criteria that have been developed in this paper are sufficient to ensure absolute safety on railway tracks, meaning that trains will never collide nor derail. In fact, we can already give a negative answer to this question, because we are aware of an important class of safety criteria that we could not express properly in terms of VLC.

Example 7.1 *If a train travelling from signal 72 to point 70 has passed point 69, then it is allowed, after a short time delay, to change point 69 into reverse, and to enable a route from signal 68 to signal 72.*

This situation cannot be described in VLC, due to the fact that there is no input variable which expresses whether a train present on track 69B has just past point 69 or is travelling towards point 69.

A solution to this problem may well be to use modal logic. It seems that safety criteria such as the one in Example 7.1 can be expressed quite easily by means of modal logic, such as described in [5].

References

[1] S. Andersson. *NP Module*, 1994.

[2] R. de Lemos, A. Saeed, and T. Anderson. A train set as a case study for the requirements analysis of safety-critical systems. *The Computer Journal*, 35(1):30–40, 1992.

[3] J.F. Groote. Hiding propositional constants in BDDs. Logic Group Preprint Series 120, Utrecht University, 1994.

[4] J.F. Groote, J.W.C. Koorn, and S.F.M. van Vlijmen. The safety guaranteeing system at station Hoorn–Kersenboogerd. Logic Group Preprint Series 121, Utrecht University, 1994. To appear in *Proceedings 10th IEEE Conference on Computer Assurance (COMPASS'95)*, Maryland, June 1995.

[5] J.F. Groote and S.F.M. van Vlijmen. A modal logic for μCRL. Logic Group Preprint Series 114, Utrecht University, 1994.

[6] G. Stålmarck. A note on the computational complexity of the pure classical implication calculus. *Information Processing Letters*, 31(6):277–278, 1989.

[7] G. Stålmarck. Normalization theorems for full first order classical natural deduction. *Journal of Symbolic Logic*, 56(1):129–149, 1991.

[8] G. Stålmarck and M. Säflund. Modelling and verifying systems and software in propositional logic. In *Proceedings SAFECOMP'90*, pages 31–36. Pergamon Press, 1990.

[9] F. Tip. A survey of program slicing techniques. Report CS-R9438, CWI, Amsterdam, 1991.