DESCRIPTION OF ROUTE DEPENDENCIES FOR COMPUTER-BASED RAILWAY SIGNALLING SYSTEMS

Michał GRZYBOWSKI[1], Jakub MŁYŃCZAK[2], Lucyna SOKOŁOWSKA[3]

[1, 2] *Silesian University of Technology, Faculty of Transport and Aviation Engineering, Katowice, Poland*
[3] *Railway Research Institute IK, Warsaw, Poland*

Abstract:

Efficient and safe movement of trains on railway lines is assured by railway signalling systems. These systems assure safety of railway transport by preservation of dependencies. A significant fraction of dependencies is related to route setting, i.e., preparation of train travel through the specified running paths. The use of computer technology in modern systems allows for greater functionality and smaller physical devices dimension than older types of systems. This results in increased complexity of track layout and increased area of a single interlocking. Systems with considerable number of routes, over a thousand are becoming increasingly common. The research is concerned with the problem of describing route dependencies in computer-based interlocking configuration. During the research existing solutions have been analyzed and a method of describing route dependencies based on requirements of PKP PLK has been proposed. The proposed solution has been verified by functional testing of prototype interlocking. During the research, a formalism for describing the route has been devised, which allows for dependency realization by track layout independent computer program. In addition, during research algorithms for automatic verification of conflicting route exclusion correctness have been designed, which allows for reduction of effort required for verification of conflicting route exclusion function. The proposed method of describing route dependencies allows for simplification of interlocking design and for automation of application data preparation. The method of implementing dependencies also allows for automation of conflicting route exclusion function verification, which decreased the effort required for verification activities while maintaining the high-quality standards.

Keywords: railway signalling, route dependencies, interlocking

Contact:
1) michal.grzybowski@polsl.pl [https://orcid.org/0000-0002-4841-147X] – corresponding author; 2) jakub.mlynczak@polsl.pl [https://orcid.org/0000-0003-2947-7980]; 3) lsokolowska@ikolej.pl [https://orcid.org/0000-0002-0699-4312]

1. Introduction

1.1. Background

Efficient and safe movement of trains on railway lines is assured by railway signalling systems. These systems assure safety of railway transport by preservation of dependencies. Dependency is a fundamental term of railway signalling and is defined by (PKP PLK, 2019) as any technical mean assuring that a given operation may be carried out only when specified conditions are satisfied.

In station systems a significant part of dependencies is related to route locking, i.e., with assuring proper conditions for train movement via designated running path (figure 1). During signalling system design, the designer specifies all possible train movements (routes), which will be supervised by the signalling system. Afterwards, during system construction, all dependencies specified in the design are implemented in the system.

Nowadays computer-based signalling systems are widely used, in which dependencies are carried out by the software. This implementation technology allows for greater system functionality while minimizing the required space for the system equipment. The consequence of this are tendencies to introduce more complex track layouts and to implement greater numbers of routes in the system. Complex installations appear more often, which manage even thousands of routes. Greater number of routes increases the effort to design and implement the interlocking part of the system, which is responsible for dependency checking. Thus, it provides the reason to look for route dependencies implementation methods, which reduce the effort needed to implement the system.

The work presents a proposal of route dependencies description for computer-based railway signalling systems and presents a case study of the application of the methods proposed by the work. Authors present a method of describing route dependencies in computer-based railway signalling system, describe how the proposed description may be used to implement interlocking functions by a generic interlocking program and analyze how improved route dependencies description method may allow for reducing the effort needed to design and verify correctness of interlocking of a railway signalling system.

The proposed description of route dependencies simplifies system configuration compared to existing boolean equation designs. The simplification reduces the effort needed to configure and verify the correctness of the system. Thus, the proposal answers directly the issue of interlocking design automation and verification raised by Zabłocki in (Zabłocki, 2014). The route dependencies description method proposed by the authors also improves on the existing methods by facilitating sectional route release, which is a commonly requested functionality in modern railway signalling systems.

1.2. Routes

As presented in section 1.1, station railway signalling systems protect train movement carried out in routes. According to (Dąbrowa-Bajon, 2014), route consists of following parts:

– Running path – part of track, which the train will run through,
– Safety distance (overlap) – part of track in advance of running path, where a train may enter due to difficulty with braking before running path end point,
– Protection devices (flank protection) – devices, which protect train running in the running path or safety distance from collision with another train,
– Approach distance – part of track in rear of running path, which when occupied should prohibit route cancellation due to close vicinity of train to running path start point.

Route elements are presented in figure 2.

Railway signalling system facilitates route setting by checking the possibility to set route and by route setting automation. The system performs the following actions after receiving route setting request:

1. Checks if all route setting dependencies are satisfied,
2. (in case of systems with automatic point switching) Sets route elements to required position (points / turnouts in particular),
3. Locks route (immobilize route elements and prohibit setting of conflicting routes) and displays proceed signal aspect on route start point signal.

The systems should also automatically release locked routes after detection of train passage. In simpler systems the route was released in its entirety after the train passage. In modern signalling systems, a sectional route release is employed for greater traffic capacity. In such arrangement route fragments (sub-routes / route sections) are released by the

system after detection of train passage through some specific track fragment. This leads to greater traffic capacity, because in order to allow second train movement (set another route) it is not needed to wait for complete passage of the first train through entire route.

The systems should also automatically release locked routes after detection of train passage. In simpler systems the route was released in its entirety after the train passage. In modern signalling systems, a sectional route release is employed for greater traffic capacity. In such arrangement route fragments (sub-routes / route sections) are released by the system after detection of train passage through some specific track fragment. This leads to greater traffic capacity, because in order to allow second train movement (set another route) it is not needed to wait for complete passage of the first train through entire route.

Route is a fundamental concept of railway signalling. Due to the role of route concept in train movement protection, railway practice has developed standard methods of route description in railway signalling system design documentation. The standard method of route description in Polish railways is interlocking control table (figure 3).
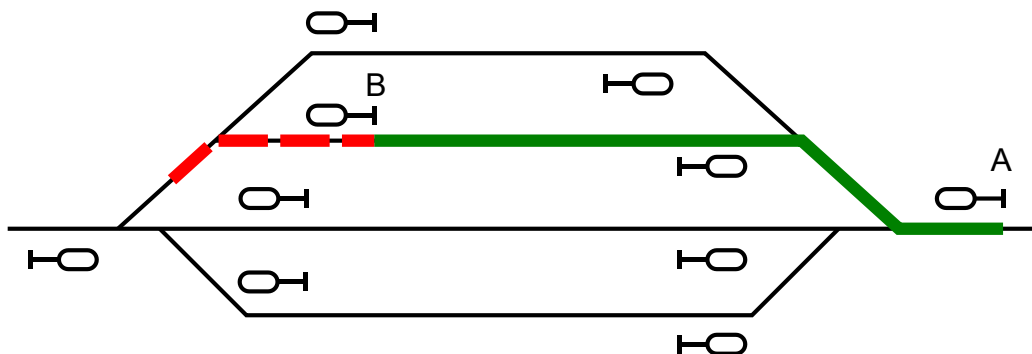

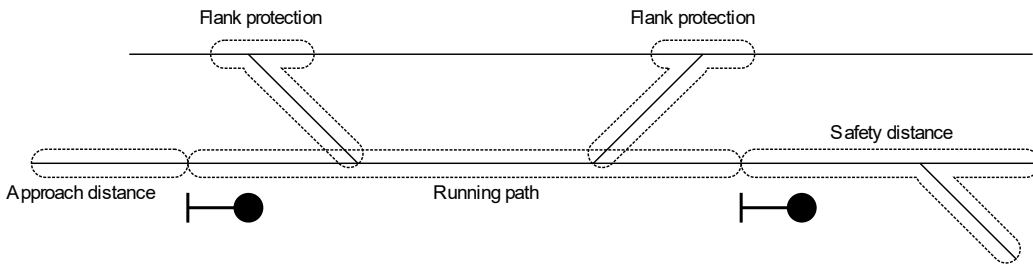Fig. 1. Example route (source: original work)

Fig. 2. Route structure (source: original work)

Fig. 3. Fragment of interlocking control table for Poznań PoD (source: (Stránský, 2020))

The table specifies for each route implemented in the system:

− routes, which cannot be set simultaneously (incompatible routes),
− required points' positions,
− track sections used in the route (track sections, which need to be clear to allow for route setting and for opening the signal),
− other elements used in the route (e.g., level crossings in case of control table in figure 3).

More detailed description of interlocking control table may be found in (Jurczak, 2011).

The interlocking control table does not fully specify implementation of route dependencies in the system. The table does not describe route partitioning into sub-routes (route sections) and does not specify rules of train passage control. In practice these details are settled during the design of interlocking circuits (in case of relay systems) or during development of computer-based interlocking configuration based on interlocking control tables and track layout.

1.3. Paper structure

The work is structured as follows:

− Section 2 contains the literature review. It discusses the historical interlocking designs, the main types of interlocking designs and the connections between the legacy relay interlocking and modern computer-based interlocking. The section also reviews a different approaches to describe routes in the computer-based interlocking and main issues related to route description in computer-based interlocking.
− Section 3 contains the route model for computer-based interlocking proposed by the authors and describes how routes can be described in computer-based interlocking software configuration and sketches how the route data can facilitate the interlocking functions.
− Section 4 presents algorithms proposed by the authors for automation of route description preparation and for verification of route compatibility.
− Section 5 contains a case study for the application of the route description model proposed by the authors. The section also contains experimental results related to the

reduction of the effort needed to design and verify the interlocking.
− Section 6 contains the conclusions of the work.

2. Literature review

Railway signalling systems have a long history. The first were mechanical interlockings, in which all dependencies were checked mechanically, in interlocking bed (Huang, 2020). Interlocking bed construction guaranteed that control levers could be operated only in appropriate state of other levers and devices. This effect could be obtained by various methods. In British interlocking bed traffic commands were issued by tappet operation, while interlocking was done be sets of locks prohibiting certain combinations of tappet positions (figure 4).
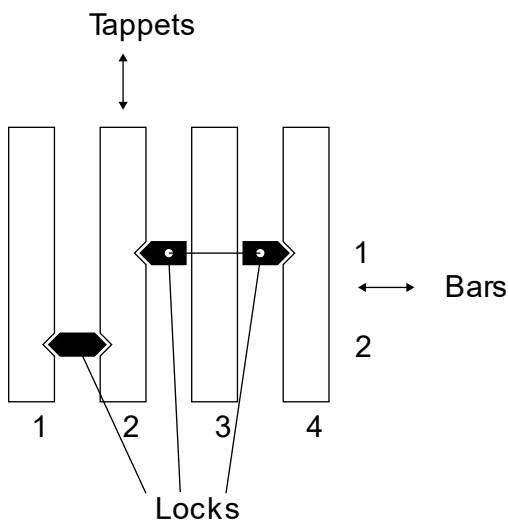
Fig. 4. Interlocking in British interlocking bed (source: original work based on (Such, 1956))

Mechanical systems exhibited simple construction, but were limited in terms of the number of dependencies, which could be implemented. In practice, interlocking bed allowed for implementing several dozen routes. Bigger interlocking beds were problematic due to their size and weight.

Technical development brought improved railway signalling systems. Relay systems were important step forward. In relay system, dependencies were implemented by appropriate relay circuits. Example in figure 5 presents dependency between locking

route from signal A (energizing of SA relay) and vacancy of track section itA (checked normally open contact of itA relay) and of point 1 being in plus position (checked normally open contact of Kn1+ relay).

In Polish railways widely used was system E (described by album of diagrams (Szeniawski et al., 1989)), which used dedicated circuits for each route. Systems of this kind are called free-wired interlockings. Free-wired interlocking was a good fit for simple track layout but was problematic for larger installations – system installation required design and construction of large number of relay circuits.

Designers tried to address the large effort required to design and construct the system. One of solutions was geographic interlocking, such as IZH-111 system. Geographic interlocking employed standard modules dedicated to controlling specific wayside devices, such as home signal, exit signal, point etc. Modules had uniform interface, which allowed them to freely connect them with one another. Such

arrangement simplified system design and construction. The designer needed to reflect station track layout in modules, connection between modules (figure 6) and modules' parameters. The system manufacturer could manufacture and verify proper functioning of modules in factory. Site activities could be limited to connecting the modules with each other and to connecting modules to wayside devices and mimic panel.

The name geographic interlocking stems from the manner of implementing dependencies. The dependencies were established based on module connections, which were themselves based on station geography. Design of geographic interlocking installation was simplified compared to free-wired interlocking, but the devices were actually more complex. Modules were in principle universal, so they needed to handle all possible traffic situations. Handling of traffic situations was configured by appropriate parameters, which resulted in modules having redundant / unused relays.
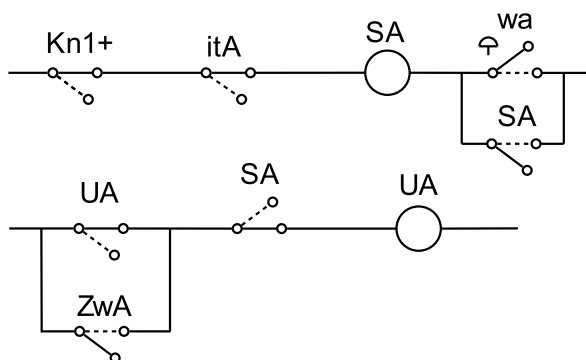

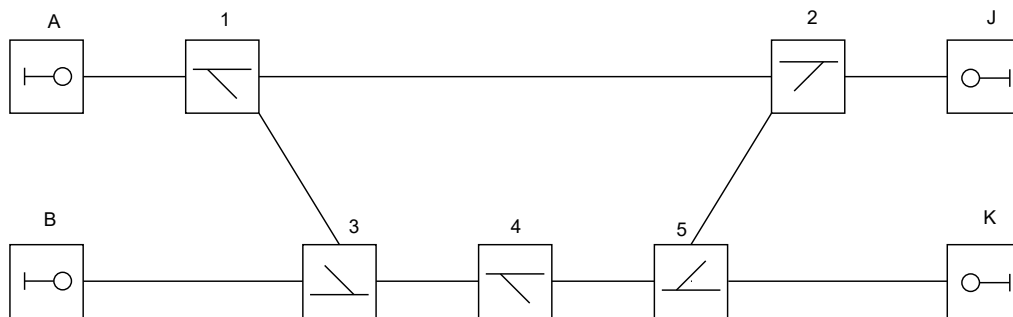Fig. 5. Fragment of interlocking relay circuits of System E (source: original work)

Fig. 6. Module connections in geographic interlocking (source: original work)

Due to long time of active use and development of relay systems in the second half of XX century, these systems are still present in large number. As has been noted in (Bădău, 2022), relay interlockings are still dominant in Europe as of 2022. In addition, infrastructure managers have developed standard relay interlocking solutions, such as system E (Szeniawski et al., 1989) on PKP PLK railway network. Availability of standard relay designs has influenced the shape of computer-based systems, as observed by (Maciejewski, 2009). A popular design is boolean equation interlocking, which is adaptation of ladder diagram-based languages used in PLCs to railway signalling system specifics. This design is used for example by EBILock (Hagelin, 1998) and VPI (van Vlijmen et al., 1999). In boolean equation-based interlocking, system state is represented by sequence of state variables. The behavior of a state variable is specified by boolean expressions (boolean equations), which specify under which condition the variable assumes particular values. Theoretical underpinnings boolean equation-based interlocking are presented in (Maciejewski et al., 2010). Additional details on structure and number of required boolean equations for an interlocking is presented in (Zabłocki, 2009).

Boolean equation-based systems replace relays with state variables and relay circuits with boolean equations. This symmetry lets computer-based system manufacturer employ specialists experienced with relay systems to design compute-based systems. It also allowed to use standard albums of diagrams, which some infrastructure managers have developed. This symmetry also allows computer-based systems to classify into free-wired interlocking or geographic interlocking, similarly to relay interlocking. Some infrastructure managers, such as RFI (Italian railway network company) sponsor work on digitizing and reverse engineering existing relay circuits, in order to be able to replace them with computer-based ones (Amendola et al., 2022). Discussion of computer-based free-wired and geographic interlockings is provided by (Wontorski et al., 2020) based on EBILock (geographic) and MOR-3 (free-wired) interlockings, which are widely used in Poland. Wontorski acknowledges the symmetry between the structure of geographic relay interlocking and geographic computer-based interlocking by describing elements of EBILock interlocking corresponding to the elements of geographic relay interlocking.

Both relay interlocking and boolean equation computer-based interlockings have disadvantages. Free-wired interlockings require designing many relay circuits / boolean equations and their design rules are often quite complex. In addition, practical matters (circuits complexity and tendency to minimize number of relays) may lead to reusing some circuits for a number of routes, which complicates system evolution during its lifetime.

Geographic interlockings exhibit high module complexity, which in turn increases the effort needed to understand and analyze system behavior. The dependencies are specified implicitly, in module connections and module parameters, which makes analysis more challenging.

A common disadvantage of both free-wired and geographic interlockings is the difficulty of design process automation. For free-wired interlocking, automation requires encoding rules of designing circuits and rules of grouping them. For geographic interlocking, it requires encoding rules of selecting module parameters.

Different take on the problem has been used in the SSI system. According to (Cribbens, 1987), SSI instead of boolean equations represent route dependencies as route setting rules. Each rule describes how system shall react to external stimuli – which sub-routes shall be checked if they are not locked, which points should be checked for proper position etc. Each rule also contains commands, which specify how route should be set and locked – commands to set points in proper position and commands to lock appropriate sub-routes. The SSI system design allows for more natural description of the route and for description of distinct routes to be independent, but the route description language is highly specialized. SSI route descriptions need to explicitly specify which sub-routes need to be not locked (normal). It means that design of SSI configuration needs to explicitly account specifying conflicting sub-routes.

An overview of route representation has been provided in (Maciejewski, 2009). Maciejewski has observed that computer-based systems employ structures similar to those of relay devices and suggested that modern computer-based signalling system should employ design based on digital form of interlocking control table or route file. Current work agrees with that statement and describes in greater

detail how routes can be represented as route file and how route file can be prepared with aid of appropriate automation.

Ikram notes in (Ikram et al., 2022) the need to unify interlocking structure for increased interoperability on the railway network. Ikram proposes using Service-Oriented Architecture for specifying system border interfaces and formalizes them in SoaML in (Ikram et al., 2023). The present work can be considered as complimentary – it focuses on the description of principal interlocking functionality, which needs to be considered to achieve interlocking harmonization on functional level.

Lutovac presents in (Lutovac et al., 1998) a conceptually simple approach of representing interlocking control table (figure 3) as data structure, on which generic interlocking program operates. The design is thus similar to the author's proposal. The difference between current work and (Lutovac et al., 1998) is in the complexity of the route structure. Lutovac settled on control tables, while current work proposes uses of route file. The difference may seem technical, but it has a practical consequence – proposal of Lutovac cannot describe system with sectional route release functionality, which is easily accommodated by the model presented in current work. On the other hand, Lutovac model allows for easy derivation of route description from standard interlocking control table, while current work needs additional tooling, described in section 4.

Research also considers methods of automating the construction of interlocking control tables. An algorithm to derive interlocking control table from the scheme plan was presented in (Mirabadi et al., 2009). Compared to that work, current work describes the derivation of route description in form of route file, which facilitates implementation of sectional route release in the interlocking. Authors of (Jurczak et al., 2021) proposed interlocking control table generation algorithms based on CAD drawings designed with aid of AutoCAD and dedicated tools. The paper also provides a formalization of route description. Compared to that work, the current paper proposes a more light-weight attempt of automation, exploiting the structure of proposed route descriptions. An automation of interlocking control tables generation based on CAD drawings was also pursued in (Ristic, 2023). Ristic proposed derivation of a graph representing the track layout from the drawing of scheme plan and generation of control tables

with aid of Mathematica. Algorithms proposed by Ristic are extensible, but in the form described in (Ristic, 2023) do not generate sufficient data to provide input for signalling system with sectional route release, in contrast to proposal presented in current work. Sedykh presents in (Sedykh et al., 2018) the interlocking control table preparation activity as part of interlocking design process. Sedykh proposes an integrated process, in which the control table may be automatically generated based on the track layout in form of CAD drawing.

Wang presents analyzes track layout and describes route finding algorithms in (Wang et al. 2013). The algorithms employ graph-theoretic matrices to not only determine the possible routes in the interlocking but also to determine route compatibility and required point position in routes. The current work provides an argument, that simpler, less conceptually sophisticated algorithms may also be used for this task.

Das proposes in (Das et al., 2021) use of dedicated CAD tools for building a track layout, which then can be an input for interlocking control table generation algorithm. The paper describes in detail not only determination of route running path, but also consideration of route safety distance, flank protection and analysis of incompatible routes. Authors of that work also point out that the control table can be used to obtain LTL (Linear Temporal Logic) formulas for verifying interlocking consistency with the control tables. Compared to that work, the algorithms proposed in current work are more limited. Authors' experience with modern railway signalling systems indicate that complex track layout greatly complicate the flank protection and safety distance design rules. In authors' opinion, the input data for automatic interlocking control table generation tools would need large number of parameters, which would make the room for mistakes in specifying track layout parameters. Authors opted to a simpler solution – to specify the necessary flank protections and safety distances manually, but to do it per track layout element instead of per route in order to eliminate redundant work.

Scippacercola describes in (Scippacercola et al., 2022) use of Model-Driven Engineering. Scippacercola argues for introducing models to interlocking development for increasing quality and decreasing the effort needed. The present work does not call for application of full Model-Driven

Engineering, but it shows how routes can be modelled in a computer-based interlocking and that appropriate model construction may allow to reduce verification effort by application of simple methods. Chandaluri argues in (Chandaluri et al., 2024) for creating interlockings digital twin, which allows for analyzing interlocking behavior in simulated environment. Chandaluri proposes design of digital twin as FPGA system. The present work focuses on high-level functionality. The route description design does not allow for detailed system simulation, but it facilitates functional verification of the interlocking. Research considers various ways of verifying interlocking correctness. Kuang describes in (Kuang et al., 2022) the use of automated tests and discusses the problem of test ordering to reduce test time. Haxthausen describes in (Haxthausen et al., 2023) application of formal methods to verify interlocking correctness. Haxthausen describes the use of model checking and techniques to partition the formal model into smaller parts to facilitate model checking of large interlockings. Iliasov describes in (Iliasov et al., 2023) application of automated theorem proving to verification of SSI interlocking. Iliasov presents analysis of model checking applicability to verification of SSI interlocking and deems it not applicable due to complexity of large SSI interlockings. The current work concentrates on defining structure of route description and shows that proper definition lends itself to automatic verification of route compatibility, which can be considered as middle ground between testing and formal verification – it provides higher quality results than testing while sidestepping the performance issues related with formal verification.

3. Route dependencies description

Computer technology allows for making the route dependencies description more similar to the design. A route can be described as a structure with attributes:

− route start point (signal),
− route end point (signal or track),
− approach section (sequence of track sections),
− sequence of sub-routes (route sections), each of which consists of:
  • running path elements (sequence of track sections and point with specified position),

  • flank protection elements (sequence of track sections, points with specified position and signals),
− safety distance (sequence of track sections, points with specified position and signals)

The structure is simple enough to be easily formalized in form of formal grammar, which afterwards can be used to specify input data for computer program. Authors have constructed a prototype of interlocking component of computer-based railway signalling systems as part of the research. The prototype is configured by text files described by LL(1) grammar, which follow the presented structure:

```
route <route identifier>
      start_signal <start signal>
      end_signal <end signal>
      berth <approach section>
      part (sub-route)
      route_elements (running path
elements)
      flank_elements (flank protec-
tion elements)
      …
overlap_elements (safety-distance ele-
ments)
end
```

A sample route description looks as follows:

```
route POC_SEM_A_FICT_2PW_ppp
      start_signal SEM_A
      end_signal FICT_2PW
      berth IT2PG
      part
      route_elements
        track  ITA
      part
      route_elements
        plus   ZW2
        plus   ZW3
        track  IZ23
      flank_elements
        plus   ZW1
        minus  ZW4
overlap_elements
      track  IT2PW
end
```

The example describes route POC_SEM_A_FICT_2PW_ppp from signal SEM_A to endpoint FICT_2PW (representing line track 2PW). The approach section is track section IT2PG. The route consists of two sub-routes:

1. Sub-route consisting of track section ITA,
2. Sub-route consisting of point section IZ23, points 2 and 3 in position plus and points 1 and 4 as flank protection elements in position plus and minus respectively.

The route also defines safety distance (overlap) consisting of track section IT2PW. As the example shows, the selected description method is highly readable and natural. The description is also easy to handle by software:

1. A route can be set, if:
    a. No route element is part of another set route (possible implementation of this functionality is presented in (Grzybowski et al., 2021)),
    b. Running path, flank protection and safety distance elements are in state suitable for route setting (according to rules implemented in interlocking software),
2. Route start signal may be opened if all running path, flank protection and safety distance elements are in state suitable for signal opening (according to rules implemented in interlocking software),
3. The description defines the order of track sections in the route for passage detection, the software controls track section occupation and release sequence according to its own rules and the specified track section order,
4. The description defines the number of sub-routes and their elements. The software needs to track the track section occupation and release sequence to detect train passage over the sub-route and to release the running path and flank protection elements associated with the sub-route.

The above description is the sketch of software operation, but it shows that route handling algorithms are simple and natural.

4. Automation of route description generation and verification

The presented formalism is more labor-intensive than geographic interlocking design (which consists of list of modules, their parameter and connections between modules). To reduce the effort needed to design interlocking in the presented manner, authors have developed tools for automatic generation of route descriptions. As it is easily seen, the route description consists of two main parts – list of track sections, ordered in the passage order and protection elements. The protection elements are associated with:

– points (turnouts) in running path (flank protection elements are determined by the running path and point position in the running path),
– route endpoint (safety distance).

Route decomposition into sub-routes is the choice of designer, but the natural choice is to decompose the routes by track sections in the running path – one sub-route for all elements located in a track section and associated flank protection elements.

These observations lead to simple algorithm for generation of route descriptions. Algorithm input data are:

– station track layout, with assignment of each wayside device (signals, points / turnouts, track sections) to track layout elements,
– description of safety distance for each signal / for each route endpoint,
– description of flank protection of each point, which can be part of route's running path.

The algorithm is as follows:

1. For each signal X:
a. Determine track section AS in rear of signal X,
b. Find all sequences S of consecutive track sections leading to signals in the same orientation as signal X or route endpoints,
c. For each sequence S create route description:
    I. Assume X as route start-point,
    II. Assume AS as approach section,
    III. Assume Y, to which sequence S leads, as route endpoint,
    IV. For each track section TS in sequence S create sub-route:
        1. Add TS as sub-route running path element,
        2. Add all wayside devices (point, level crossings etc.) located in track section TS to sub-route as running path elements,
        3. Add all flank protection elements related to points in sub-route's running path to sub-route flank protection,
    V. Assume safety distance associated with Y as route safety distance.

The algorithm reduces the effort needed to prepare description of all routes to preparing description of station track layout, flank protection elements and safety distance elements related to particular points of the track layout.

Another disadvantage of the proposed formalism (shared with geographic systems) is lack of explicit specification of incompatible routes. This problem theoretically does not affect free-wired interlocking systems, in which the designer constructs dedicated relay circuits / boolean equations controlling lack of incompatible routes set. It needs to be pointed out that free-wired systems can get difficult to handle for large installations, which can implement even few thousand routes.

The proposed formalism has the advantage of complete and independent route descriptions. It allows (in the case of properly structured interlocking software) to automatically verify route compatibility. The algorithm is simple in the case of interlocking software offering the functionality of locking the route without checking correct state of wayside devices:

1. For each route R_1:
a. Lock route R1 without checking correct state of wayside devices,
b. For each route R2:
   i. Check if interlocking software allows for locking route R_2 without checking correct state of wayside devices – if yes, then mark routes R_1 and R2 as compatible, otherwise as incompatible,
c. Release route R1.

In order to verify correctness of proposed solution, authors have developed the prototype of interlocking software and prototypes of tools implementing the presented algorithms. Route description generation tool has been implemented in 1116 lines of Haskell source code and route compatibility verifier has been implemented in 124 lines of C source code.

5. Case study

During research, route descriptions have been prepared and analyzed for two interlockings: a model station on double track line with 4 stabling tracks (figure 7) and a real-world example of Poznań PoD (Poznań Jeżyce) junction (figure 8).

Complexity of considered interlockings is summarized in table 1. As can be seen, the model station is a typical small to medium complexity station. The Poznań PoD junction is a lower complexity junction but exhibits a similar number of routes to the model interlocking.

For both interlockings, route descriptions have been created for all possible train routes. Route descriptions have been generated automatically with the aid of tool described in section 4. Table 2 presents the size of input data files (track layout description and protection description) and of the output data file (route descriptions) in number of lines of the text file. As can be seen from table 2, automatic route generation reduces the effort needed to create route description for an interlocking by 4-7 times. The obtained result can also be compared with the complexity of traditional free-wired boolean-equation based interlocking. As presented in (Zabłocki, 2009), a station of similar complexity to the model station would require around 7800 boolean equations. The comparison with the size of route description for model station or the required input data provides a strong argument for interlocking systems based on the proposed formalism.
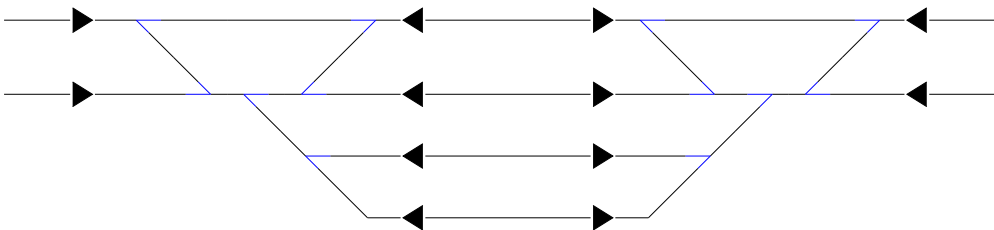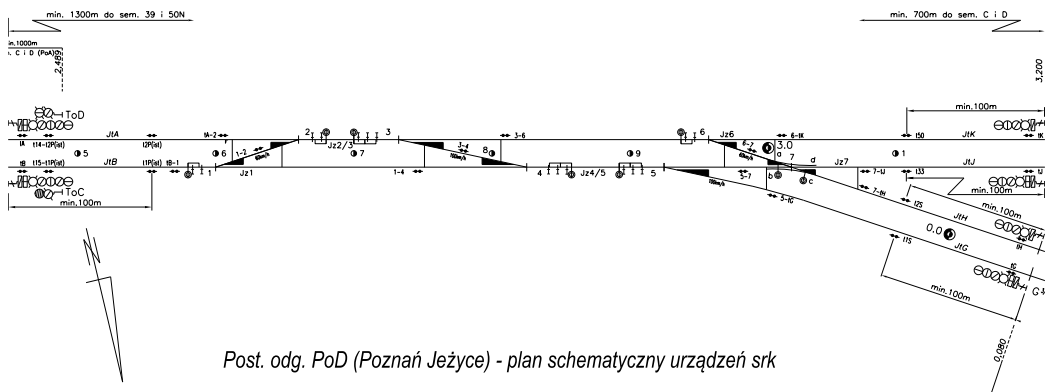

Fig. 7. Track layout of model station (source: original work)

Post. odg. PoD (Poznań Jeżyce) - plan schematyczny urządzeń srk

Fig. 8. Poznań PoD junction scheme plan (source: Stránský, 2020)

Table 1 Interlocking characteristics (source: original work)

Interlocking	Signals	Points	Track sections	Routes
Model station	12	16	32	36
Poznań PoD	6	8	17	30

Table 2 Size of route description data

Interlocking	Track layout description [lines]	Protection description [lines]	Route description [lines]	Output to input size ratio
Model station	160	165	1548	4.76
Poznań PoD	104	75	1203	6.72

Lastly, a measurement of tool processing time has been collected. Both tools have been run on a PC with AMD Ryzen 8600 CPU and 32 GB RAM. As table 3 shows, the running time for both tools can be considered negligible for small complexity interlockings. While route description generation time is satisfactory, route compatibility verification time warrants a more detailed look. The tool checks compatibility of each pair of routes during its execution. To obtain manually the same information, one would need to manually test system behavior for each pair of routes (with exception of checking route with itself). In case of model station, it would result in 1260 test cases, while in case of Poznań PoD it would result in 870 test cases. A manual test execution time can be estimated from below to be at least 6 seconds (2 seconds to lock first route, 2 seconds to attempt locking the second route, 2 seconds to release the first route). Thus, equivalent manual testing of model station would take at least 2 hours and 6 minutes, and manual testing of Poznań PoD junction would take at least 1 hour and 27 minutes. In the authors' opinion tool running time is very satisfactory and should be even more compelling in case of larger stations.

Table 3 Processing time

Interlocking	Route description generation time (ms)	Route compatibility verification time (ms)	Optimistic manual route compatibility testing time (min)
Model station	30	40.2	126
Poznań PoD	30	25.2	87

The case study compares the effort needed to design route description only with manual work, for:

1. The surveyed methods of route description automation were aimed at derivation of interlocking control tables and not configuration data which could be directly used by computer-based interlocking system,

2. The general principle (describe the track layout and try to derive automatically route description) is similar between the current work and surveyed methods, thus authors assume similar effort needed to use them. The differences could stem from the quality of the tooling to enter the data (entering data manually versus drawing the track layout in CAD-like fashion), which are not of interest to this work.

Authors have not compared the performance of route compatibility verification tool designed as part of this research with automatic methods, for the authors have not found any verification algorithm which is not a variation of simple automation of manual tests, with all the complexities it entails. For example, in order to test compatibility of two routes, any automatic tool needs to be able to lock the route in the first place. While manual tester will be generally able to lock any route by familiarizing the signalling rules obeyed by the system under test, the automatic tool needs to be taught any rules which need to be considered to lock a route. For example, the route dependencies may specify that a route can be locked only if successive route is locked (in case of a very short routes) or that some interfaces must be simulated in appropriate fashion in order to allow route locking.

The aim of the performance analysis was rather to demonstrate a very satisfactory performance obtained from an essentially trivial tool (the implementation of the tool is just 124 lines of C source code), which is facilitated by proposed route description method and route compatibility evaluation method proposed in (Grzybowski et al., 2021).

6. Conclusion

Route dependency description problem has been introduced in the paper. A survey of existing solutions and traditional classification has been presented. Route description for railway signalling systems is often overlooked issue, but it is of great importance for the effort needed to design and verify interlocking. Authors have presented a novel route description method for computer-based railway signalling systems, which:

1. is natural and concise,
2. describes route dependencies in full,
3. allows for easy processing in computer-based systems.

The proposed model allows for easier interlocking design than existing boolean equation-based interlockings, for it allows the designer to specify the description of the route directly in system configuration instead of forcing the designer to derive boolean equations consistent with the interlocking control table. Compared to earlier route models for computer-based systems, it allows for easy implementation of sectional route release, which is often a mandatory requirement for computer-based railway signalling systems. The earlier route models considered neither decomposition of route into sub-routes nor the ordering of track sections in the route. This omission limited the earlier models to only releasing routes in their entirety. The route model presented in this work takes both aspects into consideration. The route model allows for easy partitioning of a route into sub-routes. The explicit listing of the track sections in the order of train passage is simple and sufficient information for configuration of the sectional route release functionality in the generic interlocking program.

The paper also presented practical method advantages – ease of automation of route description generation and ease of route compatibility verification. Experiments conducted by the authors suggest that route description preparation effort may be decreased by factor of 4-7 by use of proper tooling. The experiments on route compatibility verification show that this problem is very amenable to automation and that proper tooling may reduce the verification time in case of simple interlockings from around two hours to less than 100 ms.

References

1. Amendola, A., Becchi, A., Cavada, R., Cimatti, A., Ferrando, A., Pilati, L., Scaglione, G., Tacchella, A., Zamboni, M. (2022). NORMA: a tool for the analysis of Relay-based Railway Interlocking Systems. In:

Tools and Algorithms for the Construction and Analysis of Systems, Publisher: Springer; 125–142. https://doi.org/10.1007/978-3-030-99524-9_7

2. Bădău, F. (2022). Railway Interlockings – A Review of the Current State of Railway Safety Technology in Europe. *Promet – Traffic&Transportation* 34, 443–454. https://doi.org/10.7307/ptt.v34i3.3992

3. Chandaluri, R., Nelakuditi, U.R. (2024). Performance evaluation of electro-mechanical railway interlocking system for digital twin application. *Computers and Electrical Engineering* 116, 109225. https://doi.org/10.1016/j.compeleceng.2024.109225

4. Cribbens, A.H. (1987). Microprocessors in railway signalling: the Solid-State Interlocking. *Microprocessors and Microsystems* 11, 264–272.

5. Dąbrowa-Bajon, M. (2014). *Podstawy sterowania ruchem kolejowym*, 3rd ed.; Publisher: Warsaw, Poland.

6. Das, A., Gangwar, M.J., Ghosh, D., Mandal, C., Sengupta, A., Waris, M.M. (2021). Automatic Generation of Route Control Chart From Validated Signal Interlocking Plan. *IEEE Transactions on Intelligent Transportation Systems* 22(10), 6516–6525. https://doi.org/10.1109/TITS.2020.2993794

7. Grzybowski, M., Młyńczak, J. (2021). Reprezentacja wykluczeń w komputerowych systemach srk. In *Zagadnienia Inżynierii Transportu i Mechaniki*; Perzyński, T., Eds.; 47–54; Publisher: Uniwersytet Technologiczno-Humanistyczny im. Kazimierza Pułaskiego w Radomiu, Radom.

8. Hagelin, G. (1998). ERICSSON Safety System for Railway Control. In *Software Diversity in Computerized Control Systems*, Springer.

9. Haxthausen, A.E., Fantechi, A. (2023). Compositional Verification of Railway Interlocking Systems. *Formal Aspects of Computing* 35(1), 4:1–4:46. https://doi.org/10.1145/3549736

10. Huang, L. (2020). The Past, Present and Future of Railway Interlocking System. In *IEEE 5th International Conference on Intelligent Transportation Engineering*, 170–174, Beijing, China.

11. Ikram, A., Amghar, M., Eleuldj, M. (2022). Distributed Architecture for Interoperable Signaling Interlocking. In: Networking, Intelligent Systems and Security, Publisher: Springer, Singapore; , 215–230. https://doi.org/10.1007/978-981-16-3637-0_15

12. Ikram, A., Eleuldj, M., Amghar, M. (2023). Services interfaces for interoperability of signaling computer interlocking on border. *International Journal of Electrical and Computer Engineering* 13(3), 2640–2651. https://doi.org/10.11591/ijece.v13i3.pp2640-2651

13. Iliasov, A., Taylor, D., Laibinis, L., Romanovsky, A. (2023). Practical Verification of Railway Signalling Programs. *IEEE Transactions on Dependable and Secure Computing* 20(1), 695–707.

14. Jurczak, M. (2011). Structure of interlocking table. *Archives of Transport System Telematics* 4(2), 18–24.

15. Jurczak, M., Młyńczak, J. (2021). Method for automation of generation interlocking tables for station traffic control devices. *WUT Journal of Transportation Engineering* 131; 45–58. https://doi.org/10.5604/01.3001.0014.8100

16. Kuang, J., He, T. (2022). Research on automatic test sequence generation method of computer interlocking test. *Journal of Physics: Conference Series* 2246, 012072. https://doi.org/10.1088/1742-6596/2246/1/012072

17. Lutovac, D., Lutovac, T. (1998). Towards an universal computer interlocking system. *FACTA UNIVERSITATIS (NIŠ)* 11(1), 25–49.

18. Maciejewski, M. (2009). Zróżnicowanie struktur komputerowych urządzeń zależnościowych. *Zeszyty naukowo-techniczne SITK RP, oddział w Krakowie* 149, 249–261.

19. Maciejewski, M., Zabłocki, W. (2010). Metoda tworzenia funkcji równań zależnościowych w systemach SRK. *Logistyka* 4.

20. Mirabadi, A., Yazdi, M. (2009). Automatic generation and verification of railway interlocking control tables using FSM and NuSMV. *Transport Problems* 4(3), 103–110.

21. PKP PLK (2019). *Wytyczne techniczne budowy urządzeń sterowania ruchem kolejowym* Ie-4, Publisher: Warsaw, Poland.

22. Ristic, I. (2023). Automated development of railway signalling control tables: A case study from Serbia. *Mechatronics and Intelligent Transportation Systems* 2(4); 227–235. https://doi.org/10.56578/mits020404

23. Scippacercola, F., Zentai, A., Russo, S. (2022). Experiencing Model-Driven Engineering for Railway Interlocking Systems. In: Certifications of Critical Systems – The CECRIS Experience, Publisher: River Publishers, New York; 31–63. https://doi.org/10.1201/9781003337485-2

24. Sedykh, D., Gordon, M., Efanov, D. (2018). Computer-Aided Design of Railway Signalling Systems in Russian Federation. In *2018 International Conference on Industrial Engineering, Applications, and Manufacturing (ICIEAM)*, 1–7, Moscow, Russia. https://doi.org/10.1109/ICIEAM.2018.8728662

25. Stránský, M. (2020). Scheme Plan of Poznań PoD (Poznań Jeżyce) junction, P244 PW SRK 02, AŽD Praha

26. Such, W.H. (1956). *Mechanical and Electrical Interlocking (British Practice)*; Publisher: The Institution of Railway Signal Engineers, England.

27. Szeniawski, Z., Makała, J. (1989). *Album schematów przekaźnikowych urządzeń zabezpieczenia ruchu kolejowego typu E*; Publisher: Centralne Biuro Projektowo-Badawcze Budownictwa Kolejowego.

28. van Vlijmen, SFM., Groote, JF., Koorn, JWC. (1999). The Vital Processor Interlocking. *Electronic Notes in Theoretical Computer Science* 21, 1–56.

29. Wang, D., Chen, X., Huang, H. (2013). A graph theory-based approach to route location in railway interlocking. *Computers & Industrial Engineering* 66(4), 791–799.

30. Wontorski, P., Kochan, A. (2020). *Komputerowe systemy kierowania i sterowania ruchem kolejowym. Część 1: funkcje, elementy i układy*, 1st ed.; Publisher: Warsaw, Poland.

31. Zabłocki, W. (2009). Synteza funkcji zależnościowych stacyjnego systemu srk. *Zeszyty naukowo-techniczne SITK RP, oddział w Krakowie* 149, 523–261.

32. Zabłocki, W. (2014). Zagadnienie sprzeczności i wykluczeń specjalnych w technice srk. *Zeszyty naukowo-techniczne SITK RP, oddział w Krakowie* 2(104), 399–406.