Railway Infrastructure Verification and RDFox

Bjørnar Luteberget / Christian Johansen

July 4, 2016

UiO : University of Oslo

RailCOMPLETE

Railway verification and formal methods

- Railway systems: large-scale, safety-critical infrastructure
- High safety requirements: SIL 4 for passenger transport
- Increasingly computerized components
- Typical use of formal methods in railways: model checking of control systems

Objective

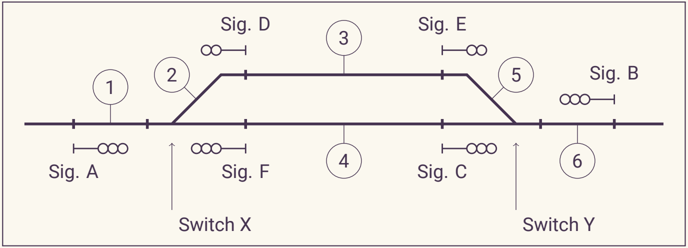
Given a railway signalling and interlocking design,
verify that it complies with regulations.

Secondary objectives:

- Integrate with engineering/design tools
  - On-the-fly verification (“lightweight”)
  - Usable for engineers who are not formal methods experts
- Find suitable language for expressing regulations

> “Formal methods will never have a significant impact until they can be used by people that don’t understand them.”
>
> — (attributed to) Tom Melham

Railway designs for signalling and interlocking



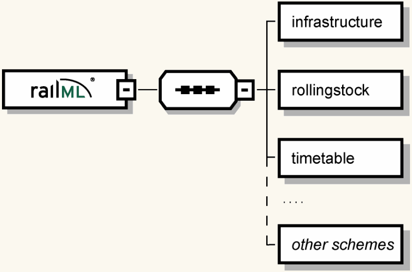
(a) Track and signalling component layout

Route	Start	End	Sw. pos	Detection sections	Conflicts
AC	A	C	X right	1, 2, 4	AE, BF
AE	A	E	X left	1, 2, 3	AC, BD
BF	B	F	Y left	4, 5, 6	AC, BD
BD	B	D	Y right	3, 5, 6	AE, BF

(b) Tabular interlocking specification

The railML XML standard data exchange format

- Thoroughly modelled infrastructure schema
- XML schema development by international standard committee



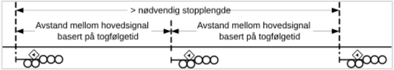

```xml
<tracks>
  <track id="tr0" name="01">
    <trackTopology>
      <trackBegin id="x399" pos="0.000000" absPos="34
        <connection id="co399" ref="co397"/>
      </trackBegin>
      <trackEnd id="y151" pos="80.000000" absPos="346
        <connection id="co151_2" ref="co151_1"/>
      </trackEnd>
    </trackTopology>
    <trackElements>
      <speedChanges>
        <speedChange id="spu399" pos="0.000000" absPo
        <speedChange id="spd403" pos="30.000000" absP
        <speedChange id="spu405" pos="30.000000" absP
        <speedChange id="spd151" pos="80.000000" absP
      </speedChanges>
      <gradientChanges>
        <gradientChange id="gr399" pos="0.000000" abs
      </gradientChanges>
      <radiusChanges>
        <radiusChange id="ra399" pos="0.000000" absPo
      </radiusChanges>
      <platformEdges>
        <platformEdge id="pe399" pos="0.000000" absPo
      </platformEdges>
    </trackElements>
    <ocsElements>
      <signals>
        <signal id="si399" pos="0.000000" absPos="346
" code="6"/>
```

Technical regulations

- In our case study: Norwegian regulations from infrastructure manager Jernbaneverket
- Static kind of properties, often related to object properties, topology and geometry (examples later)

https://trv.jbv.no/wiki/Signal/Prosjektering/Lyssignal

e) Dersom nødvendig stopplengde er lengre enn avstanden mellom to etterfølgende hovedsignal, skal det benyttes gjennomsignalering ved hjelp av ATC (Signal/Prosjektering/ATC), se Figur 7.



Figur 7: Forsignalplassering og gjennomsignalering

f) Et forsignal skal plasseres på foregående hovedsignals mast dersom avstanden mellom det tilhørende hovedsignalet og det foregående hovedsignalet er ≤ 2200 meter.

g) Mellom et forsignal og det tilhørende hovedsignalet skal det ikke plasseres andre hoved- eller forsignal.

h) Et forsignal skal plasseres slik at siktavstanden oppfyller kravene til enten "brutt sikt" eller til "ubrutt sikt" i Tabell 4:

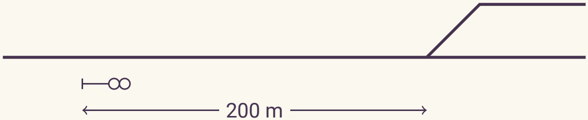
Tabell 4: Siktkrav til forsignal

Sikt	Strekningens høyeste tillatte kjørehastighet [km/h]																	
	40	45	50	55	60	65	70	75	80	85	90	95	100	105	110	115	120	125	≥130
	Siktavstand [m]																		
Ubrutt	78	88	97	107	117	126	136	146	156	165	175	185	194	204	214	224	233	243	250

Technical regulations

Example from regulations:

- A *home main signal* shall be placed at least 200 m in front of the first controlled, facing switch in the entry train path.



- Can be classified as follows:
  - Object properties
  - Topological layout properties
  - Geometrical layout properties
  - Interlocking properties

Formalization of rule checking

- Formalize the following information
  - The CAD design (extensional information, or facts)
  - The regulations (intensional information, or rules)
- Use a solver which:
  - Is capable of verifying the rules
  - Runs fast enough for on-the-fly verification

Datalog

- Basic Datalog: conjunctive queries with fixed-point operators ("SQL with recursion")
  - Guaranteed termination
  - Polynomial running time (in the number of facts)
- Expressed as logic programs in a Prolog-like syntax:

$$a(X, Y) \text{ :- } b(X, Z), c(Z, Y)$$

$$\Updownarrow$$

$$\forall x, y : ((\exists z : (b(x, z) \land c(z, y))) \to a(x, y))$$

- We also use:
  - Stratified negation (negation-as-failure semantics)
  - Arithmetic (which is "unsafe")

Encoding facts and rules in Datalog

- The process of formalizing the railway data and rules to Datalog format is divided into three stages:
  1. Railway designs (station data) – facts
  2. Derived concepts (used in several rules) – rules
  3. Technical regulations to be verified – rules
- Now, more details about each stage...

Input documents representation

- Translate the railML XML format into Datalog facts using the ID attribute as key:

$$
\begin{aligned}
\textit{track}(a) &\leftarrow \text{element}_a \text{ is of type } \texttt{track}, \\
\textit{signal}(a) &\leftarrow \text{element}_a \text{ is of type } \texttt{signal}, \\
&\vdots \\
\textit{pos}(a,p) &\leftarrow (\text{element}_a.\texttt{pos} = p), \quad a \in \text{Atoms}, p \in \mathbb{R}, \\
&\vdots \\
\textit{signalType}(a,t) &\leftarrow (\text{element}_a.\texttt{type} = t), \\
& t \in \{\text{main, distant, shunting, combined}\}\,.
\end{aligned}
$$

Input documents representation

- To encode the hierarchical structure of the railML document, a separate predicate encoding the parent/child relationship is added:

$$\mathit{belongsTo}(a, b) \leftarrow b \text{ is the closest XML ancestor of } a \text{ whose element type inherits from } \texttt{tElementWithIDAndName}.$$

Derived concepts

- Derived concepts are defined through intermediate rules
- Railway concepts defined independently of the design
- Example:

$$\textit{directlyConnected}(a,b) \leftarrow \exists t : \textit{track}(t) \wedge \textit{belongsTo}(a,t) \wedge \textit{belongsTo}(b,t),$$

$$\begin{aligned}\textit{connected}(a,b) \leftarrow\ & \textit{directlyConnected}(a,b) \vee (\exists c_1, c_2 : \textit{connection}(c_1,c_2) \wedge \\ & \textit{directlyConnected}(a,c_1) \wedge \textit{connected}(c_2,b)).\end{aligned}$$

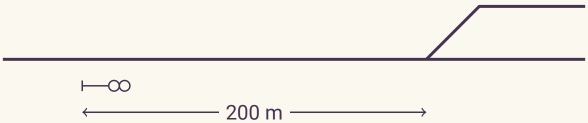
- A library of concepts allows concise expression of technical regulations

Technical regulations as Datalog rules

- Detecting errors in the design corresponds to finding objects involved in a regulation violation
- To *validate* the rules in a given design, we show that there are no satisfiable instances of the *negation* of the rule
- Some examples:
  - Example 1, home signal placement: topological and geometrical layout property for placement of a home signal
  - Example 2, train detector conditions: relates interlocking to topology
- These are Jernbaneverket regulations which are relevant for automatic verification

Rule: example 1

- A *home main signal* shall be placed at least 200 m in front of the first controlled, facing switch in the entry train path.
- Uses arithmetic and negation

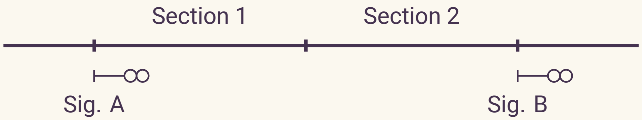


$$\textit{isFirstFacingSwitch}(b, s) \leftarrow \textit{stationBoundary}(b) \land \textit{facingSwitch}(s) \land \neg(\exists x : \textit{facingSwitch}(x) \land \textit{between}(b, x, s)),$$

$$\textit{ruleViolation}(b, s) \leftarrow \textit{isFirstFacingSwitch}(b, s) \land (\neg(\exists x : \textit{signalFunction}(x, \text{home}) \land \textit{between}(b, x, s)) \lor (\exists x, d, l : \textit{signalFunction}(x, \text{home}) \land \land \textit{distance}(x, s, d, l) \land l < 200).$$

Rule: example 2

- Each pair of adjacent train detectors defines a track detection section. For any track detection sections overlapping the route path, there shall exist a corresponding condition on the activation of the route.



Tabular interlocking:

Route	Start	End	Sections must be clear
AB	A	B	1, 2

Rule: example 2

$$\textit{adjacentDetectors}(a, b) \leftarrow \textit{trainDetector}(a) \wedge \textit{trainDetector}(b) \wedge \neg \textit{existsPathWithDetector}(a, b),$$

$$\textit{detectionSectionOverlapsRoute}(r, d_a, d_b) \leftarrow \textit{trainRoute}(r) \wedge \textit{start}(r, s_a) \wedge \textit{end}(r, s_b) \wedge \textit{adjacentDetectors}(d_a, d_b) \wedge \textit{overlap}(s_a, s_b, d_a, d_b),$$

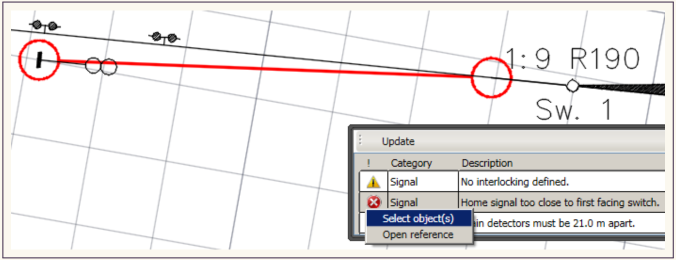
$$\textit{detectionSectionCondition}(r, d_a, d_b) \leftarrow \textit{detectionSectionCondition}(c) \wedge \textit{belongsTo}(c, r) \wedge \textit{belongsTo}(d_a, c) \wedge \textit{belongsTo}(d_b, c).$$

$$\textit{ruleViolation}(r, d_a, d_b) \leftarrow \textit{detectionSectionOverlapsRoute}(r, d_a, d_b) \wedge \neg \textit{detectionSectionCondition}(r, d_a, d_b).$$

Prototype tool implementation

- Prototype using XSB Prolog tabled predicates, front-end is the RailCOMPLETE tool based on Autodesk AutoCAD
- Rule base in Prolog syntax with structured comments giving information about rules

```
%| rule: Home signal too close to first facing switch.
%| type: technical
%| severity: error
homeSignalBeforeFacingSwitchError(S,SW) :-
    firstFacingSwitch(B,SW,DIR),
    homeSignalBetween(S,B,SW),
    distance(S,SW,DIR,L), L < 200.
```



Current work

- Incremental updates (view maintenance)
  - Changes in the CAD design causes the whole verification to start over
  - More efficient: recompute only the parts that are affected by the changes
- B/F algorithm and RDFox might be suitable
- Semantic web standards and railway ontology
  - Translate railML XSD into OWL?
  - Translate Datalog rules into OWL/SWRL?
  - Closed-world assumption
  - Higher-arity predicates ($\mathit{distance}(X, Y, L, D)$)