Verifying Real-Time Aspects of the European Train Control System

A survey on AVACS R1

Johannes Faber
j.faber@uni-oldenburg.de
Overview on the Approach

- **Aim:** automatic verification of complex high-level specifications with infinite state space

- **Combined specification formalism:** CSP-OZ-DC
  
  - integrating data, processes, time

- **Real-time requirements:** DC
Overview on the Approach (2)

- Automatic verification
  - OL
  - CSP-OZ-DC
  - Phase Event Automata
  - Transition Constraint Systems
  - SB
  - Abstraction Refinement Model Checker

- Demonstrated by an application from the ETCS
European Train Control System

ETCS

- Future train control system
- Financed by the European Commission
- Developed by a group of railway companies

Purposes

- Ensures cross-border interoperability
- Improves railway safety
- Increases traffic density
European Train Control System (2)

Features

- Replacement of national trackside systems
- Cooperation of trains and radio block centres (RBCs)
- Communication using a GSM-R radio connection
- Negotiation of train locations, speed and integrity
- Application of the *moving block principle*
Case Study: Emergency Messages

- Considers treatment of emergency messages in ETCS
- Comprises real-time requirements
- Guarantees reliability properties
  - Reaction time not exceeded
  - Trains never collide
The case study comprises
- Communication aspects
- Data aspects
- Real-time aspects

Basic data types (booleans, constants, ...)
- Potentially infinite ($\mathbb{R}$, $\mathbb{N}$)

Events with parameters (infinite range)
The specification language **CSP-OZ-DC** combines

- Communicating Sequential Processes (CSP)
- Object-Z (OZ)
- Duration Calculus (DC)

```
Train
  chan send : [id : TrainID]
  ...
  main = Running || HandleEM || ...
  HandleEM = receive.EmergencyWarning?id → ...
  position : ℝ ...
  brakingMode = None ...
  com_applyBrakes ...
  com_computeSBI ...
  ¬(true; ↓indicationToDriver; □driverAck ∧ 5 < ℓ)
  ...
```

```
CSP-OZ-DC is given an operational semantics in Phase Event Automata (PEA) [Hoenicke 2006]

- Timed automata variant
- Synchronise on data, events and time
- Compositional

\[ A(CSP-OZ-DC) = A(CSP) \parallel A(OZ) \parallel A(DC) \]

\[ A_1 \models \phi \quad \Rightarrow \quad A_1 \parallel A_2 \models \phi \]
Verification of CSP-OZ-DC (2)

Aim: verify whether $\mathcal{A} \models \phi$

- Translation of DC *Test Formulae* into PEA [Meyer, Faber, Rybalchenko 2006]
- Check for reachability of "bad" states in $\mathcal{A} \parallel \mathcal{A}(\phi)$
- Representation of PEA in Transition Constraint Systems (TCS)
- Abstraction refinement model checking: ARMC [Rybalchenko 2002]
Verification of CSP-OZ-DC (3)

- Specification in DC

  - “After emergency detection the follower brakes or the driver acknowledges within 8 seconds”
    \[
    \neg \Diamond (\uparrow \text{detectEmergency}; \\
    \Box \text{applyEB}.2 \land \Box \text{driverAck}.2 \land 8 < \ell)
    \]

  - “Position of follower is never beyond the rear end of the leader”
    \[
    \neg \Diamond ([\text{position}.1 > \text{position}.0 - \text{Length}.0])
    \]
PEA with a “bad” state for property

\[ \neg \Diamond (\uparrow \text{send.EmergencyWarning}.1 \land \square \text{applyEB}.1 \land 13 < \ell) \]
## Results

<table>
<thead>
<tr>
<th>Task</th>
<th>Locs</th>
<th>Trans</th>
<th>Vars</th>
<th>Preds</th>
<th>Abstr</th>
<th>Refs</th>
<th>TA</th>
<th>ARMC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>178</td>
<td>6.1T</td>
<td>31</td>
<td>46</td>
<td>347</td>
<td>22</td>
<td>25s</td>
<td>26m</td>
</tr>
<tr>
<td>Running (decomp. 1)</td>
<td>8</td>
<td>150</td>
<td>20</td>
<td>11</td>
<td>11</td>
<td>8</td>
<td>2.5s</td>
<td>7.5s</td>
</tr>
<tr>
<td>Running (decomp. 2)</td>
<td>20</td>
<td>899</td>
<td>22</td>
<td>8</td>
<td>32</td>
<td>8</td>
<td>4.0s</td>
<td>21.5</td>
</tr>
<tr>
<td>Running (decomp. 3)</td>
<td>48</td>
<td>1.2T</td>
<td>27</td>
<td>13</td>
<td>93</td>
<td>10</td>
<td>5.9s</td>
<td>45s</td>
</tr>
<tr>
<td>Running (decomp. 4)</td>
<td>48</td>
<td>1.7T</td>
<td>27</td>
<td>11</td>
<td>70</td>
<td>7</td>
<td>6.3s</td>
<td>47.5s</td>
</tr>
<tr>
<td>Delivery</td>
<td>122</td>
<td>18T</td>
<td>20</td>
<td>41</td>
<td>2.2T</td>
<td>32</td>
<td>50s</td>
<td>86m</td>
</tr>
<tr>
<td>Delivery (decomp. 1)</td>
<td>14</td>
<td>366</td>
<td>14</td>
<td>9</td>
<td>29</td>
<td>8</td>
<td>2.7s</td>
<td>13.9s</td>
</tr>
<tr>
<td>Delivery (decomp. 2)</td>
<td>17</td>
<td>173</td>
<td>10</td>
<td>25</td>
<td>17</td>
<td>17</td>
<td>2.2s</td>
<td>1.9s</td>
</tr>
<tr>
<td>Delivery (decomp. 3)</td>
<td>12</td>
<td>71</td>
<td>9</td>
<td>12</td>
<td>9</td>
<td>9</td>
<td>1.9s</td>
<td>0.7s</td>
</tr>
<tr>
<td>Delivery (decomp. 4)</td>
<td>17</td>
<td>156</td>
<td>12</td>
<td>25</td>
<td>19</td>
<td>17</td>
<td>2.2s</td>
<td>2.6s</td>
</tr>
<tr>
<td>Delivery (decomp. 5)</td>
<td>7</td>
<td>28</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>1.6s</td>
<td>0.1s</td>
</tr>
<tr>
<td>Braking 1</td>
<td>44</td>
<td>240</td>
<td>17</td>
<td>45</td>
<td>44</td>
<td>3</td>
<td>3s</td>
<td>5.1s</td>
</tr>
<tr>
<td>Braking 2</td>
<td>172</td>
<td>1.6T</td>
<td>33</td>
<td>63</td>
<td>88</td>
<td>59</td>
<td>9s</td>
<td>35.3s</td>
</tr>
</tbody>
</table>

Verifying Real-Time Aspects of the European Train Control System – p.13
First large-scale application example for the declarative, high-level language CSP-OZ-DC involving

- Infinite data
- Real-time
- Communication aspects (infinite parameter types)
- Object-orientation

- Focusing message passing and infinite data types
- State explosion problem

\[ \Rightarrow \text{Decomposition for CSP-OZ-DC/PEA/DC needed} \]
Ongoing/Future Work

Several variants

CSP-OZ-DC

PEA

TCS

Slicing
Task analysis
FTA
Decomposition
Data types
ARMC

Verifying Real-Time Aspects of the European Train Control System – p. 15
Ongoing/Future Work (2)

- More complex data types (sequences, arrays) [Jacobs, Sofronie-Stokkermans 2006]
- Decomposition approaches to cope with larger specifications
  - Slicing [Brückner, Wehrheim 2005]
  - Fault tree guided decomposition
- Combination with scheduling analysis
- Completion of tool support
References

- J. Hoenicke
  *Combination of Processes, Data, and Time*

- J. Hoenicke and P. Maier
  *Model-checking of Specifications Integrating Processes, Data and Time*

- S. Jacobs and V. Sofronie-Stokkermans
  *Applications of Hierarchical Reasoning in the Verification of Complex Systems*
J. Faber and R. Meyer

*Model Checking Data-Dependent Real-Time Properties of the European Train Control System*


R. Meyer, J. Faber and A. Rybalchenko

*Model Checking Duration Calculus: a Practical Approach*


I. Brückner and H. Wehrheim

*Slicing an Integrated Formal Method for Verification*

B. Cook, A. Podelski and A. Rybalchenko

*Terminator: Beyond safety (tool description)*


A. Rybalchenko

*A Model Checker Based on Abstraction Refinement*


ARMC


Verification tools for PEA

[http://csd.informatik.uni-oldenburg.de/projects/epea.html](http://csd.informatik.uni-oldenburg.de/projects/epea.html)

[http://csd.informatik.uni-oldenburg.de/~moby/](http://csd.informatik.uni-oldenburg.de/~moby/)