Constraint programming for real-time train circulation management in railway nodes

Joaquín Rodríguez

INRETS (Institut National de Recherche sur les Transports et leur Sécurité)
Outlines

- Introduction
- Real scenario
- Models
  - Simulation model
  - Constraint programming model
- Solution method
- Results
- Conclusions/issues
Introduction

Train circulation management
Paris-Nord Control Center
Train circulation management
Case study: Pierrefitte-Gonesse node

Paris
Gare du Nord

1 LN
P
a
r
i
s
G
a
r
du
N
or
d

2 LN

2 LN

VSD
DC
RT

RC

1 LN

2 LN

2 LN

1 LN

2 LN

2 LN

1 M

2 M

1 M

1 M

1 LN

2 LC

2 LC

1 LN

2 LC

2 LC

Hight–Speed
Line

Freight Line

15..18 km
duration = 5..7 mn

Chantilly

3rd AMORE Research Seminar 02-04 October 2002 – p.4/34
Train circulation management
Case study: Pierrefitte-Gonesse node

Mixed traffic:
- High Speed (North TVG, Eurostar, Thalys)
- Intercity
- Freight
Train circulation management
Case study: Pierrefitte-Gonesse node

Different speed limits

- 200 km/h
- 160 km/h
- 120 km/h
- 60 km/h

High-Speed Line
Freight Line
Disruptions:

◆ train delays
◆ unplaned trains
◆ unavailable infrastructure devices (points, signals, . . .)
◆ breakdown (faulty) traction vehicle
◆ . . .

⇒ The forecasted plan can no longer be used
Operator has to:
- Anticipate the conflicts,
- Use alternative routes, train orders.

Disrupted situations:
- A lot of information,
- Little time to make a decision.

☞ Tool to generate and evaluate a set of solutions
- find a good solution (i.e. not necessary optimal)
- criterion: minimise the sum of the delays caused to the trains

☞ Model based on simulation and constraint programming (CP)
Case study: Pierrefitte-Gonesse node
6 trains
Case study: Pierrefitte-Gonesse node

Disruptions:

<table>
<thead>
<tr>
<th>Trains</th>
<th>Delays(seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>North TVG 7177</td>
<td>+7</td>
</tr>
<tr>
<td>North TVG 9349</td>
<td>+210</td>
</tr>
<tr>
<td>Intercity 76731</td>
<td>+280</td>
</tr>
<tr>
<td>Freight 42439</td>
<td>+1958</td>
</tr>
<tr>
<td>Intercity 156310</td>
<td>+1610</td>
</tr>
<tr>
<td>Eurostar 9032</td>
<td>-215</td>
</tr>
</tbody>
</table>
Real scenario

Simulation of the 6 train scenario

Comparison Scenario reel / Voie libre absolue

Time-distance graph of:
- train runs with maximum speed
- train runs of the simulated scenario

sum of delays = 1210s (≈ 20 mn)
Models

Functional diagram

Train movements planning

Operator

Signalling system

Driver

Train

Models:

Contraint programming

Simulation
Train movements
Time-Distance graph of head
Simulation model

Train movements
Time-Distance graph of head+tail
Train movements
Time-Distance graph of head+tail
Train movements are detected by track circuits
Time-Distance graph of detection
Train movements are detected by track circuits
Time-Distance graph of detection
Simulation model

Train movements are detected by track circuits

Time-Distance graph of detection → input data of the CP model
for each (train, route)

sequence of events:
- track circuit start of detection
- track circuit end of detection

Use of Multi-Train railway simulator (SNCF) : SISYFE
Constraint programming model

Train circulation management $\equiv$ Scheduling Problem

i.e. *allocating scarce resources to activities over time*

**Model:**
- activities
- resources
- resource constraints
- temporal constraints
Constraint programming model

Time-Distance graph

Track circuit
Constraint programming model

Time-Distance graph → Gantt chart of activities

track circuit

run

clearing

activity duration
A train is a *sequence of activities*

Activity $\rightarrow$ ?
Ressource $\rightarrow$ ?

track circuit

activity $i$  $\rightarrow$  activity $i+1$  $\rightarrow$  activity $i+2$
A train is a *sequence of activities*

- **Activity** $\rightarrow$ elementary run
- **Ressource** $\rightarrow$ track circuit

![Diagram showing a train as a sequence of activities with track circuit and activity labels](image-url)
Resource constraints

✓ capacity(track circuit) = 1
Resource constraints

✓ capacity(track circuit) = 1

✓ requires(\text{activity}_i, t_{c_i}) \quad t_{c_i} : \text{track circuit variable of the activity}_i
Constraint programming model

Resource constraints

✓ capacity(track circuit) = 1

✓ requires(activity_i,tc_i)  \( tc_i \) : track circuit variable of the activity_i

<table>
<thead>
<tr>
<th>Assignment constraint</th>
<th>r</th>
<th>tc_1</th>
<th>tc_2</th>
<th>tc_3</th>
<th>tc_4</th>
<th>tc_5</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>r</td>
<td>z1</td>
<td>z2</td>
<td>z3</td>
<td>z*</td>
<td>z5</td>
</tr>
<tr>
<td>P2</td>
<td>r</td>
<td>z1</td>
<td>z2</td>
<td>z4</td>
<td>z6</td>
<td>z5</td>
</tr>
</tbody>
</table>

\( r \) : the variable of the alternatives routes
Constraint programming model

Gantt chart of activities: Block signalling system?

activity duration

run

clearing

track circuit
Constraint programming model

Gantt chart of activities: Block signalling system?

activity duration
Constraint programming model

Gantt chart of activities: 2 aspect block signalling system

- track circuit
- signal
- block

run clearing

activity duration
Constraint programming model

Gantt chart of activities: 3 aspect block signalling system
Example of routes P1, P2
### Temporal constraints

<table>
<thead>
<tr>
<th>$r$</th>
<th>$est_1$</th>
<th>$est_2$</th>
<th>$est_3$</th>
<th>$est_4$</th>
<th>$est_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>P2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>16</td>
<td>16</td>
</tr>
</tbody>
</table>

$est_i$: earliest start time of the activity $i$
**Temporal constraints**

<table>
<thead>
<tr>
<th></th>
<th>$est_1$</th>
<th>$est_2$</th>
<th>$est_3$</th>
<th>$est_4$</th>
<th>$est_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>P2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>16</td>
<td>16</td>
</tr>
</tbody>
</table>

$est_i$ : earliest start time of the activity$_i$

<table>
<thead>
<tr>
<th></th>
<th>$eft_1$</th>
<th>$eft_2$</th>
<th>$eft_3$</th>
<th>$eft_4$</th>
<th>$eft_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>8</td>
<td>12</td>
<td>17</td>
<td>17</td>
<td>19</td>
</tr>
<tr>
<td>P2</td>
<td>8</td>
<td>14</td>
<td>18</td>
<td>20</td>
<td>22</td>
</tr>
</tbody>
</table>

$eft_i$ : earliest finish time of the activity$_i$
Temporal constraints

\[ ft_i = eft_i + \sum_{j=1}^{i} d_j \]

- \( ft_i \): finish time of activity \( i \)
- \( d_i \): delay of activity \( i \)
**Temporal constraints**

\[ ft_i = eft_i + \sum_{j=1}^{i} d_j \]

- \( ft_i \): finish time of activity\(_i\)
- \( d_i \): delay of activity\(_i\)

**Block synchronization constraint**

<table>
<thead>
<tr>
<th>( r )</th>
<th>( s_1 )</th>
<th>( s_2 )</th>
<th>( s_3 )</th>
<th>( s_4 )</th>
<th>( s_5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>P2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

- \( s_i \): index of the activity to synchronize the activity\(_i\)
Constraint programming model

Temporal constraints

\[ \checkmark ft_i = eft_i + \sum_{j=1}^{i} d_j \]

\( ft_i \) : finish time of activity\(_i\)
\( d_i \) : delay of activity\(_i\)

<table>
<thead>
<tr>
<th>Block synchronization constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>( r )</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>P1</td>
</tr>
<tr>
<td>P2</td>
</tr>
</tbody>
</table>

\( s_i \) : index of the activity to synchronize the activity\(_i\)

\[ \checkmark st_i = est_i + \sum_{j=1}^{s_i-1} d_j \]

\( st_i \) : start time of activity\(_i\)
Solution method

Tree search

1. Each node:
   (a) labelling procedure
   (b) consistency (propagation) procedure

2. Optimisation: each time a solution is found, post a constraint on the criterion variable

Variable choice order:

1. Resource assignment variables of activities \( (tc_i) \)
   (a) Minimum size domain

2. Time variables of activities \( (st_i, ft_i) \)
   (a) Minimum earliest value
Solution method

Implementation:

C++ Library Ilog Solver, Scheduler, Views

Validation of solutions:

Start, end of detection

CP model

Routes, Train ranks, Entrance times

Simulator

Simulator
Real scenario:

Size of the instance problem:
- Number of activities per train: [20, 36]
- Number of variables: 3019
- Number of constraints: 1737
- Number of alternative routes per train: [4, 8]
Assignment problem (6 trains): 25920 combinations
Simulation of the 6 train scenario

Comparison Scenario reel / Voie libre absolue

Time-distance graph of:
- train runs with maximum speed
- train runs of the simulated scenario

sum of delays = 1210s
Results

Best solution (Temps CPU = 32s)

Time-distance graph of:
- train runs with maximum speed
- train runs of the simulated scenario
  sum of delays = 20s
Simulation of the best solution

Comparaison Voie libre absolue / solution simulee CSP criteres = 20

- train runs with maximum speed
- train runs of the simulated scenario

sum of delays = 300s
Best solution after relaxation of entrance constraint

Comparison Voie libre absolue / solution simulee CSP critere = 28 + fluidification (1mn retard injection)

- train runs with maximum speed
- train runs of the simulated scenario

sum of delays = 80s
Results

Graphic tools
Conclusions:

- CP model:
  - Input data: infrastructures, start/finish time detection
  - Signalling system (number of aspects)
  - No description of all possible conflicts
- Scenarios 6-10 trains: solutions with good ratio criterion value/CPU

Issues:

- Pre-processing to simplify the model
- Model acceleration/deceleration of trains
- Integrated CP-MIP approach
- Application to capacity analysis