Constraint programming based iterative heuristic for scheduling trains and maintenance tasks

Team 2.7.2 solution

S. Carpov L. Cudennec F. Galea

CEA, LIST

2016 RAS problem solving competition

Our team

- Researchers at CEA (LIST Institute)
- CEA – French Alternative Energies and Atomic Energy Commission
  - 10 research centers across France
  - >15.000 people
  - Research areas:
    - nuclear energy (fission and fusion)
    - **technological research for industry**
    - defense and security
    - fundamental research in physical and life sciences.

Outline

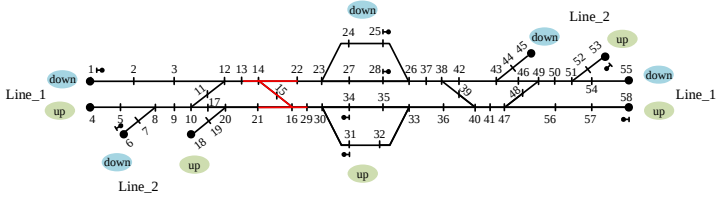
Outline

Routing Trains through a Railway Network

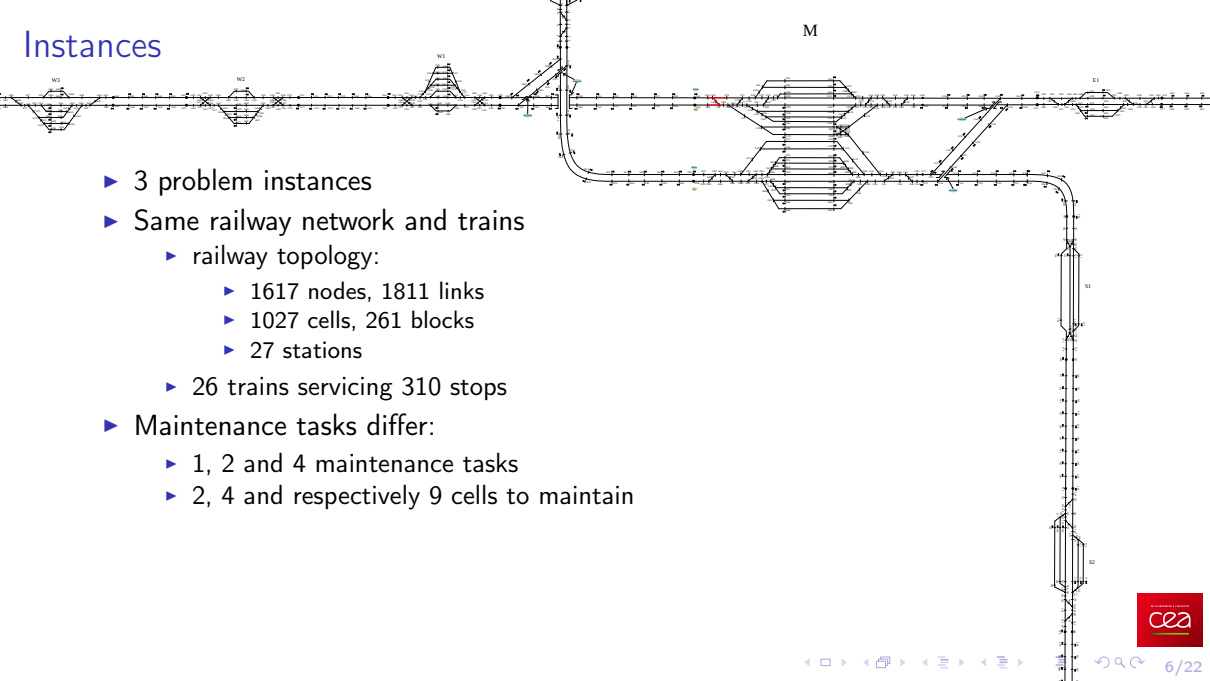
Joint optimization on train timetabling and maintenance task scheduling

- Problem input:
  - railway network description
    - nodes, links, cells, blocks, stations
  - trains to route and timetable
    - origin, destination, time window, stops
  - maintenance tasks
    - duration, cells to maintain, time window

- Goal:
  - train timetabling
    - railway network routes
    - node arrival times
  - maintenance task scheduling
- Objective: minimize total travel time of all trains



Instances



- 3 problem instances
- Same railway network and trains
  - railway topology:
    - 1617 nodes, 1811 links
    - 1027 cells, 261 blocks
    - 27 stations
  - 26 trains servicing 310 stops
- Maintenance tasks differ:
  - 1, 2 and 4 maintenance tasks
  - 2, 4 and respectively 9 cells to maintain

Outline

Railway network graph

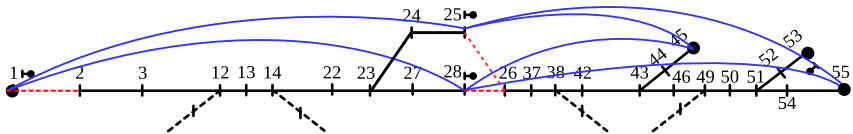
Definition

- A graph $G = (N, L, W)$:
  - N set of railway nodes
  - L set of railway links and *block traversal links* (introduced on next slide)
  - W set of link weighting functions
    - $W = \{c_t : t \in T\}$ and $c_t : L \to \mathbb{N}$
- Weighting function $c_t(l)$ gives link $l \in L$ traversal time:
  - depends on train speed multiplier and link speed limit
  - includes minimum dwelling time for station siding tracks
  - infinite value for station main tracks where train t should stop

Railway network graph

Block traversal links

- Railway network block sections give authorized ways to traverse the network
- Block traversal links:
  - artificial links corresponding to the traversal of blocks
  - block traversal link weight is the sum of weights of corresponding links
  - block first link removed from G



Railway network graph

Paths

- Valid railway network traversal routes with respect to constraints:
  - link minimum running time
  - train required stops
  - minimum dwelling time
  - block section selection
- Correspond to train routes with no maintenance tasks and independent train traversals
  - i.e. G path lengths are lower bounds to actual train travel times

Global lower bound

- Global lower bound on total travel times for all the trains:

$$GLB = \sum_{t \in T} short\_path\_len(G, orig(t), dest(t), c_t)$$

  - $short\_path\_len(G, s, e, c_t)$ – shortest path length from node s to node e using weighting function c_t
- Gives total travel time for a solution not considering train collision and maintenance tasks
- We are looking for solution values as close as possible to GLB

Simple Heuristic

General idea

Decompose problem solving in two steps:

1. Route trains through the railway network
2. Schedule trains and maintenance tasks using a constraint programming (CP) model

Heuristic first-step

Railway network train routing

1. Build an auxiliary railway network graph G' (built from G)
   - Restrict "pessimistically" speed limits of links impacted by maintenance tasks
     - I.e. use the worst case link traversal time
     - Simplified problem model where all trains always traverse maintained links at low speed
   - Restrict middle station traversal by train origin and destination
     - Reduces congestion and balances train traversal of middle station
2. Find for each train the shortest path in graph G'

CP model

Interval variables (IV)

Interval variable definition

- decision variable whose value is an interval of integers
- modeled using 2 values: start and end of the interval

Employed IVs

- Train link traversal
  - path links found in the first step
- Train block[1] traversal
  - block interval variables "span" over respective link interval variables
- Maintenance task
  - maintained cells are grouped => one interval variable per maintenance task

[1] To ease the exposition all cells are considered blocks.

CP model

Constraints and objective

- Link traversal and maintenance tasks minimal duration
- Order of link traversal (respect train paths)
- Starting time windows of trains and maintenance tasks
- No overlap
  - Same link for different trains IVs
  - Same block for different trains IVs [2]
  - Block and maintenance tasks IVs with common cells
- Objective: minimize the sum of link IV durations

[2] Special case for arrival blocks.

Iterative heuristic

- Second step of simple heuristic is a cumbersome process
  - Large CP model for off–the-shelf solvers
  - Poor search space exploration
- Main idea of iterative heuristic:
  - fix the schedule for some trains and solve the reduced CP model
  - do this iteratively
- Instead of solving a large CP problem solve several easier sub-problems

Iterative heuristic

Iterative heuristic steps

1. Find a starting solution (simple heuristic, small time limit)
2. Repeat several times:
   - 2.1 Choose a set of trains to schedule
   - 2.2 Fix other trains schedule and solve the reduced CP model

Step 2.1 – train to schedule choice

- Use gap between train travel time and its lower bound to make decision:
  - A choose trains having the largest gap
  - B privilege trains with common traversal segments besides lower bound gap

Outline

Experimentation

- `python` for heuristics first-step and *GLB*
- ILOG CP solver for constraint programming models
- Execute on a single core of an AMD Opteron 6172 processor (2.1GHz)
- Simple and iterative heuristics time limit – 2 hours
- *GLB* used to assert solution quality
- Beer & pizzas

Results

Total travel time of all trains

Heuristic	Solution value in seconds (gap to *GLB*)		
	Case 1 1 MOT	Case 2 2 MOTs	Case 3 4 MOTs
Simple	157459 (1.83%)	158339 (2.40%)	161341 (4.34%)
Iterative A	156096 (0.95%)	**156865 (1.45%)**	160608 (3.87%)
Iterative B	**155294 (0.43%)**	157176 (1.65%)	**158985 (2.82%)**
GLB	154625		

- *GLB* proves to be rather powerful on given instances
- Less maintenance tasks => smaller solution gap to *GLB*

CP model generalization ideas

- Consider several traversal ways of stations (e.g. stations with >1 siding track)
  - allows to switch heading train
- Several possible paths for train traversal (generalization of previous case)
  - e.g. dynamic choice of middle station traversal path
- Use one IV for each cell of maintenance tasks (do not group maintained cells)
  - adds flexibility to the model

Thanks!