Improved Local Freight Train Classification

Jens Maue[1]

Joint Work with Peter Márton[2] and Marc Nunkesser[1]

[1]Institute of Theoretical Computer Science, ETH Zürich, Switzerland

[2]Department of Transportation Networks, University of Žilina, Slovakia

RailZurich 2009 - 12 February 2009

[1]Partially supported by the Future and Emerging Technologies Unit of EC (IST priority - 6th FP) under contract no. FP6-021235-2 (project ARRIVAL).

[2]Partially supported by the Slovak grant foundation under grant no. 1-4057-07 (project "Agent Oriented Models of Service Systems").

Local Freight Train Classification

Local freight train

- multi-destination freight train
- cars ordered by destinations

Train classification

- special sorting problem
- classification yard

Outline

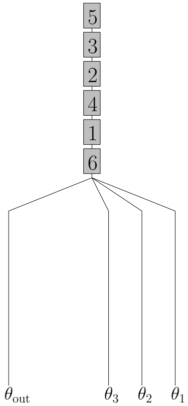
Train Classification in General

Classification Schedules

IP Formulation
- Basic Model
- Real-World Instance
- Real-World Restrictions

Concluding Remarks

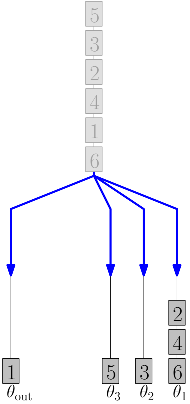
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

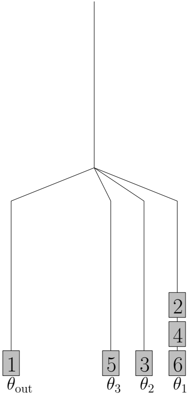
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

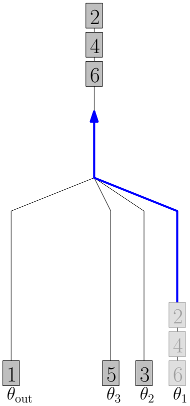
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

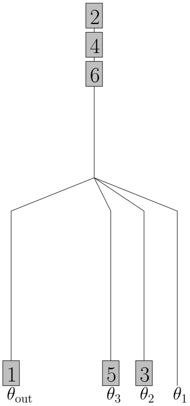
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

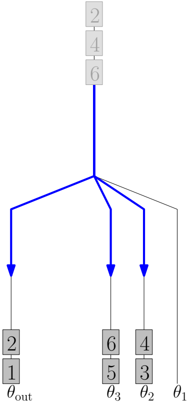
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

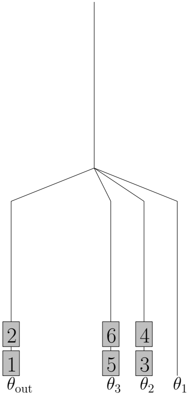
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

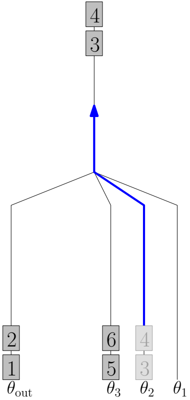
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

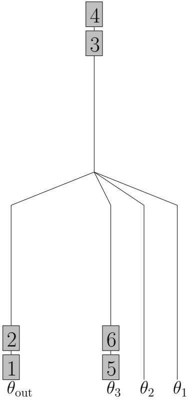
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

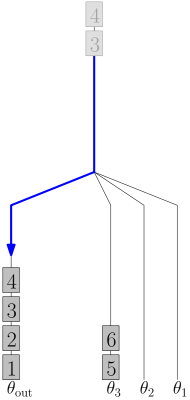
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

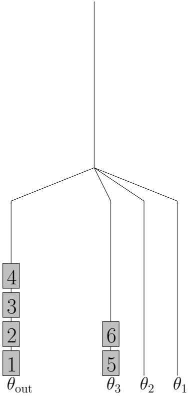
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

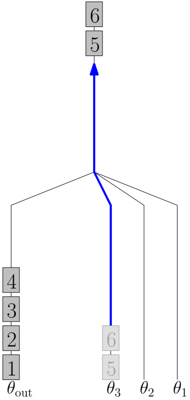
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

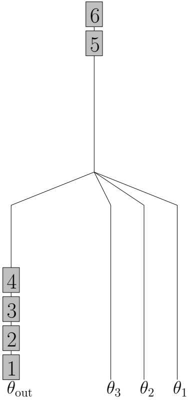
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

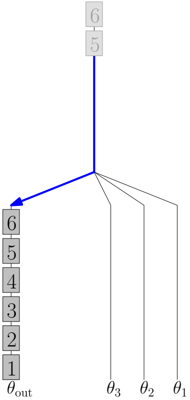
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

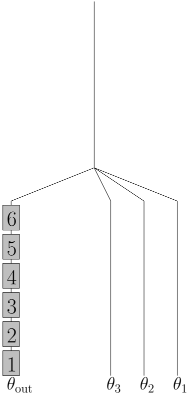
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

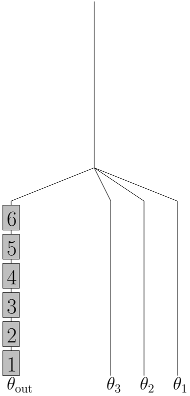
Example Classification Process



Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

Example Classification Process



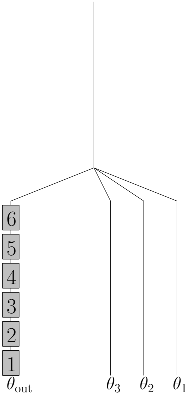
Train Classification

- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

Classification Process

1. initially roll-in input train
2. alternately pull out and roll in
3. finish with ordered train

Example Classification Process



Train Classification

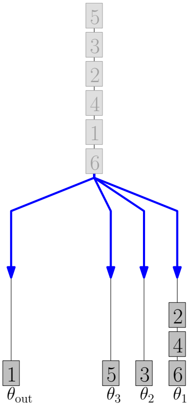
- goal: ascendingly ordered train on θ_{out}
- use available tracks θ_1, θ_2, and θ_3

Classification Process

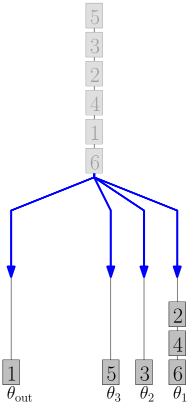
1. initially roll-in input train
2. alternately pull out and roll in
3. finish with ordered train

Objective: number h of pull-out steps

Schedule Encoding [JMMN07]



Schedule Encoding [JMMN07]

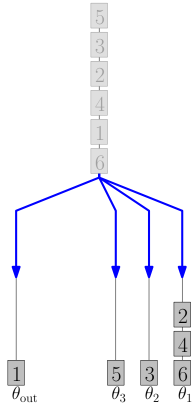


5	1	0	0
3	0	1	0
2	0	0	1
4	0	1	1
1	0	0	0
6	1	0	1

Schedule representation

- assignment of cars to bitstrings of length h
- rows: bitstring b^j encodes journey of jth car
- columns: bits encode sequence of pull-out steps
- bit $b^j_i = 1$ iff jth car visits track pulled in ith step

Schedule Encoding [JMMN07]



5	1	0	0
3	0	1	0
2	0	0	1
4	0	1	1
1	0	0	0
6	1	0	1

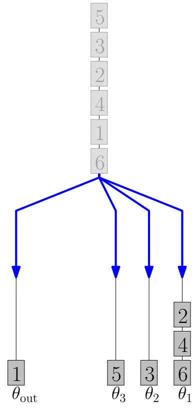
Schedule representation

- assignment of cars to bitstrings of length h
- rows: bitstring b^j encodes journey of jth car
- columns: bits encode sequence of pull-out steps
- bit $b^j_i = 1$ iff jth car visits track pulled in ith step

Schedule derivation: two consecutive cars τ and $\tau + 1$

- correct order: assign same bitstring
- reversed order: assign bigger bitstring to $\tau + 1$

Schedule Encoding [JMMN07]



	θ_3	θ_2	θ_1
5	1	0	0
3	0	1	0
2	0	0	1
4	0	1	1
1	0	0	0
6	1	0	1

Schedule representation

- assignment of cars to bitstrings of length h
- rows: bitstring b^j encodes journey of jth car
- columns: bits encode sequence of pull-out steps
- bit $b^j_i = 1$ iff jth car visits track pulled in ith step

Schedule derivation: two consecutive cars τ and $\tau + 1$

- correct order: assign same bitstring
- reversed order: assign bigger bitstring to $\tau + 1$
- objective: length h of schedule

Basic IP Model [MN09]

$$\min \sum_{\substack{1 \leq i \leq n \\ 0 \leq j < h}} b_i^j$$

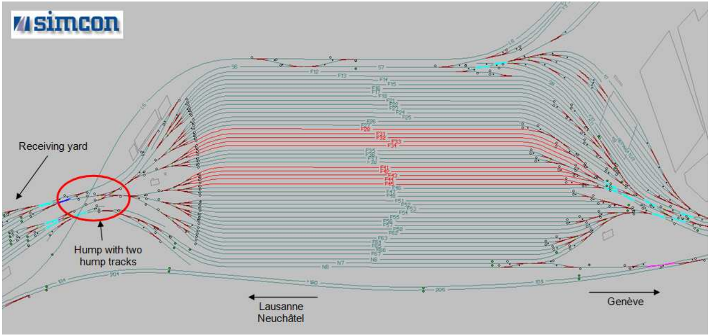
$$\text{s.t.} \quad \sum_{0 \leq j < h} 2^i b_i^j \geq \text{rev}(j-1, j) + \sum_{0 \leq j < h} 2^i b_i^{j-1} \quad \forall j \in \{1, \ldots, n\} \setminus F \quad (1)$$

$$\sum_{1 \leq i \leq n} b_i^j \leq C \quad \forall i \in \{0, \ldots, h-1\} \quad (2)$$

$$b_i^j \in \{0, 1\} \quad \forall j \in \{1, \ldots, n\}, \; \forall i \in \{0, \ldots, h-1\} \quad (3)$$

- $\text{rev}(j-1, j) = 1$ iff cars $j-1$ and j in reversed order in incoming train
- F subset of cars that are first in their respective outgoing train
- classification tracks have capacity C

Real-World Instance: Lausanne-Triage



Traffic data

- single day in 2005
- volume 328 cars
- 23 outgoing trains

Infrastructure and operation

- two parallel humps
- local freight trains: collect on ten tracks
- time window for pull-out steps
- further tracks for outgoing train formation

Extended IP Model [MN09]

Additional constraints for Lausanne-Triage

- initial roll-in restricted to ten tracks
- assignment of outgoing trains to either hump
- respect departure times

Extended IP Model [MN09]

Additional constraints for Lausanne-Triage

- initial roll-in restricted to ten tracks
- assignment of outgoing trains to either hump
- respect departure times

Resulting schedule

- one step shorter
- one track less required
- verification by computer simulation in progress (not finished yet)

Concluding Remarks

Conclusion

- encoding yields flexible IP model
- adapts to various real-world restrictions
- Lausanne-Triage: save one step and track

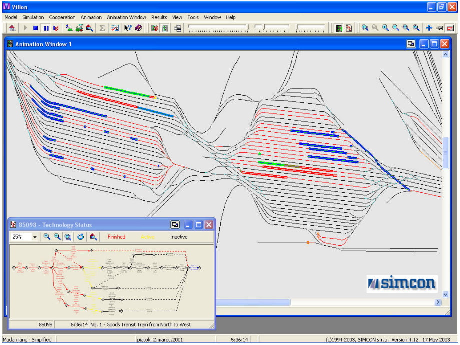
Concluding Remarks

Conclusion

- encoding yields flexible IP model
- adapts to various real-world restrictions
- Lausanne-Triage: save one step and track

Ongoing work

- computer simulation for Lausanne-Triage (Villon)
- evaluation of 2-approximation
- time-dependent input
- robustness questions



References

Riko Jacob, Peter Márton, Jens Maue, and Marc Nunkesser.
Multistage methods for freight train classification.
In *Proc. of the 7th Workshop on Algorithmic Methods and Models for Optimization of Railways (ATMOS-07)*, pages 158–174, Wadern, Germany, 2007. IBFI Schloss Dagstuhl.

Jens Maue and Marc Nunkesser.
Evaluation of computational methods for freight train classification schedules.
Technical Report TR-0184, ARRIVAL, 2009.