Scheduling single track train traversals
A new model of scheduling with setups

Per Kreuger
Joint work with M. Aronsson, M. Bohlin and M. Forsgren

Swedish Institute of Computer science (SICS)

2006-03-15

Outline

1. Background
   - Track resource scheduling
   - Headways and sequence dependence
   - Drawbacks of sequence dependent model

2. A new model
   - Some observations
   - A two resource model
   - Refinements
   - Implementation
   - Testing, evaluation and future work

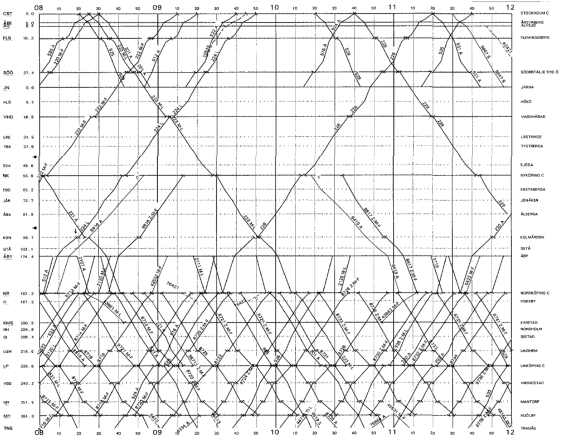
Outline

1. Background
   - Track resource scheduling
   - Headways and sequence dependence
   - Drawbacks of sequence dependent model
2. A new model
   - Some observations
   - A two resource model
   - Refinements
   - Implementation
   - Testing, evaluation and future work

Railway track resource scheduling

- Railway tracks currently consist of individual blocks which can accommodate at most one train at any given time
- Track streaches between meeting points in the network are broken into smaller blocks to allow several trains to enter such stretches before the first one has left it
  - The number of blocks in a typical track networks is very large
  - The number of tracks between meeting points in the Swedish network is $\lesssim$ 2200, each consisting of several blocks
  - Data on individual blocks not readily available since it is not used in the current (manual) process
- For single track stretches ($\approx$ 70% of Swedish track network) the a track between two meeting points is completely allocated w.r.t. traffic in the other direction

Production graph (sequence diagram)



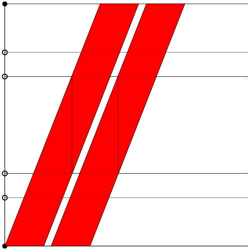
Outline

1. Background
   - Track resource scheduling
   - Headways and sequence dependence
   - Drawbacks of sequence dependent model

2. A new model
   - Some observations
   - A two resource model
   - Refinements
   - Implementation
   - Testing, evaluation and future work

Headway abstraction

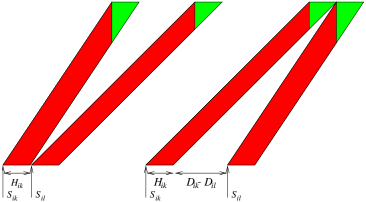
- One way of reducing the size of the problem is to use a headway abstraction (KCSÅ01)



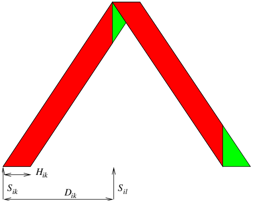
- The longest time taken to traverse any block on a track connecting two meeting points is taken as the "duration" of the task
- Since speeds are different this "duration" is sequence dependent

Sequence dependence

- Speed differences



- Single track / two way traffic



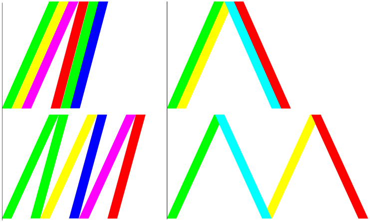
- *Unique* (depending on relative speed) setup times between start times of each pair of tasks!
- (KCSÅ01) implements model with variant of SICStus' `serialized` constraint (with setup time parameter matrix)
- Scales well for regions with mainly double tracks and reasonably homogeneous traffic

Outline

1. Background
   - Track resource scheduling
   - Headways and sequence dependence
   - Drawbacks of sequence dependent model
2. A new model
   - Some observations
   - A two resource model
   - Refinements
   - Implementation
   - Testing, evaluation and future work

Drawbacks of sequence dependent model

- Strongest reasoning of constraint applied *only to "duration"* of each task which is equivalent to the *minimum* headway
- But there is in general a very *big* span of setups for single tracks and for heterogeneous traffic
  - Leads to insufficient propagation for single tracks and where speed differs significantly between trains



- Does not lend itself well to further abstraction of longer stretches, with embedded meeting places, e.g between junctions

Outline

Observations

- Each track between meeting points can accommodate several trains simultaneously
    - How many, depends on the relation between traversal times and headway's for that track
- If the headway allows up to n trains to enter the track in the same direction before the first one has exited, we can regard the track as a resource with capacity of n with respect to the tasks corresponding to these trains
- For a single track with two way traffic, each train in one direction allocates the track resource completely with respect to all trains in the opposite direction

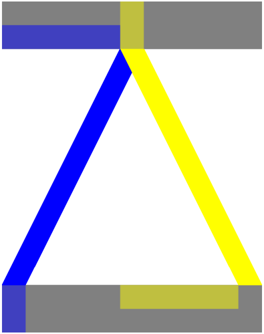
Outline

Single track model

- Each (single) track can be modelled as *two* (cumulative) *resources* of capacity n, one for each end of the track
  - These resources can be thought of as the exit and entry signals of the two involved stations
- Each track traversal is modelled as *two tasks*, one for each of the two resources
  - The resource closest to the start of the traversal should be allocated *completely* (resource consumption n), starting at the beginning of the traversal and for the *duration of the headway*
  - The resource closest to the the end of the traversal should be allocated with a resource consumption of 1, starting at the beginning of the traversal and for the *duration of the traversal*

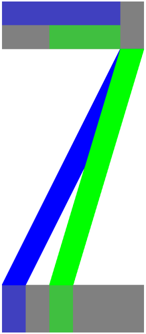
Sequence diagram for single track

- Nicely handles change of directions

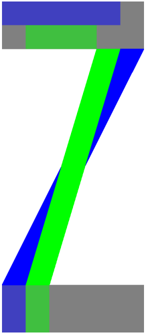


but:

- Maintains hdw only at start



- Allows overtakings



Outline

Sorting

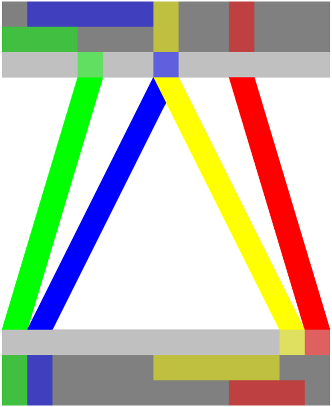
- Overtakings can be excluded by ensuring that a sorting the departure times for all traversals in one direction for a given track yields the same permutation of some arbitrary inital order as that of sorting the corresponding arrival times
- This can be done easily with `sorting/3` constraints

```
prmtn(UpDeps,UpPrmtn),
prmtn(UpArrs,UpPrmtn),
prmtn(DwnDeps,DwnPrmtn),
prmtn(DwnArrs,DwnPrmtn)

prmtn(Tms,Prmtn) :-
    length(Tms,L),
    length(Prmtn,L),
    length(TmS,L),
    sorting(Tms,Prmtn,TmS).
```

Four resource model

- Maintaining the headway at the end of the traversal can be achieved by introducing two additional unitary resources
- Each traversal allocates a task starting at the *time of arrival* and of *headway duration* to the resource closest to the end of the traversal



Outline

1. Background
   - Track resource scheduling
   - Headways and sequence dependence
   - Drawbacks of sequence dependent model
2. A new model
   - Some observations
   - A two resource model
   - Refinements
   - Implementation
   - Testing, evaluation and future work

Implementation

- Implementation uses a 4 machine `cumulatives/3` and four `sorted/3` (as above) constraints for each track resource

```
cumulatives([task(DpTm,Hdw,_,Hght,1),task(ArTm,Hdw,_,1,3),
             task(DpTm,TrvTm,ArTm,1,4)|Tsks],
            [machine(1,Hght),machine(2,1),
             machine(3,1),machine(4,Hght)],
            [bound(upper)])
```

- Variants with classical `cumulative` has also been tried but `cumulatives/3` scales better (+ allows late track choice)
- One big cumulatives for all tracks performs slightly worse
- Using the redundant *task intervals* algorithm yields significantly worse performance (very large but sparse problem?)

Outline

Scaling evaluation

- The current implementation schedules a problem consisting of 1205 trains, 7543 track traversals and 34 compound track resources in a few minutes
  - This includes scheduling of 8748 stops at 25 stations modelled as classical cumulative resources
  - Of the 34 track resources , 16 are single tracks, i.e. traversed in both directions
  - The problem was created by relaxing an existing time table for the traffic between Järna and Linköping in Sweden
  - How relaxation is done influences performance significantly, here departures and arrivals were relaxed ± 1 hrs and stop times at stations ± 20 min
  - Note that even though the number of tasks is very large, the resulting schedules are very sparse

Future work

- Comparison with the (KCSÅ01) model not yet done but should be easy
- Comparison with with block level model would be interesting but data is missing
- Variants with relaxed sorting conditions and reduced task heights *may* be used to model longer stretches of (aggregated) track with limited meeting and/or overtaking capacity, e.g.
  - Lower the height of the task allocated to the departure resource to allow limited no. of meetings on an aggregated track resource
  - Allow for limited no. of differences in sorting permutations to allow fixed no. of overtakings on aggregated resource
- Characterise class of sequence dependent setup time scheduling problems that can be encoded using this type of encoding

P. Kreuger, M. Carlsson, T. Sjöland, and E. Åström.
Sequence dependent task extensions for trip scheduling.
Technical Report T2001:14, SICS, 2001.