Recoverable-Robust Timetables for Trains on Single-Line Corridors

Work partially supported by the Future and Emerging Technologies Unit of EC (IST priority - 6th FP), under contract no. FP6-021235-2 (project ARRIVAL)

Gianlorenzo D'Angelo[1] Gabriele Di Stefano[1]
Alfredo Navarra[2]

[1]Dept. of Electrical and Information Engineering, University of L'Aquila, Italy
{gianlorenzo.dangelo,gabriele.distefano}@univaq.it

[2]Dept. of Mathematics and Informatics, University of Perugia, Italy
navarra@dipmat.unipg.it

Timetabling

Schedule the departure and arrival time of trains in order to reduce the traveling time for passengers

Delay Management

Modify the timetable when unpredictable events cause delays

Recoverable Robustness

Design the timetable in order to easily recover when delays occur [Liebchen et al. 2007, Cicerone et al. 2007]

This work

This work studies the Recoverable Robustness approach for timetabling in restricted topologies (Tree) and applies it to Italian single-line corridors

Previous Works

- Defined the recoverable robust timetabling ($\mathcal{RTT}$) [Cicerone et al. 2008]
- $\mathcal{RTT}$ is NP-hard [Cicerone et al. 2008]
- Linear time approximation algorithm [Cicerone et al. 2008]
- $\mathcal{RTT}$ remains NP-hard when the topology is restricted to out-trees [D. et al. 2008]
- pseudo-polynomial time optimal algorithm [D. et al. 2008]

Results of the paper

- We modelled single-line corridors as out-trees
- We implemented the algorithm in [D. 2008] which optimally solves $\mathcal{RTT}$ and applied it to Italian single-line corridors
- We experimentally showed that the algorithm is effective and efficient in practical cases

Outline

Outline

The timetabling problem [Schöbel, 2007]

Scheduling the departure and arrival time of trains

Instances

- A *event activity network* $\mathcal{N} = (\mathcal{E}, \mathcal{A})$ made of departure and arrival events $\mathcal{E}$ and activities $\mathcal{A}$
- The *minimum time* $L(a)$ needed for each activity $a \in \mathcal{A}$
- The *number of passengers* $w(v)$ involved in each event $v \in \mathcal{E}$

Solutions

A scheduled time $\Pi(v)$ for each event $v \in \mathcal{E}$ in the network, such that Π satisfies the minimum duration time of each activity

Objective

Minimizing the overall traveling time for passengers
$\min f = \sum_{v \in \mathcal{E}} w(v)\Pi(v)$

Timetabling problem TT

$$\min f = \sum_{v \in \mathcal{E}} w(v)\Pi(v)$$

subject to

$$\begin{aligned} \Pi(v) - \Pi(u) &\geq L(a), && \text{for each } a = (u, v) \in \mathcal{A} \\ \Pi(v) &\in \mathbb{N}, && \text{for each } v \in \mathcal{E} \end{aligned}$$

- I: set of instances $i = (\mathcal{N}, L, w)$
- $F(i)$: set of feasible solutions Π for $i \in I$

Modification function M

We allow only one delay on an activity a of at most α time
We can model it as an increase of the minimal duration time a
Given $i = (\mathcal{N}, L, w)$,

$$M(i) = \{(\mathcal{N}, L', w) : L' \text{ differs from } L \text{ by at most one activity } \}$$

Recovery capabilities $\mathbb{A}$

A recovery algorithm can change the time of at most Δ *affected events*

The Recoverable Robust Timetabling Problem $\mathcal{RTT}$

$\mathcal{RTT} = (TT, M, \mathbb{A})$ is the problem of finding a timetable that can be recovered by changing the time of at most Δ events when a delay of at most α time occurs

Robust Algorithm

A *robust algorithm* for TT is any algorithm A_r which solves $\mathcal{RTT}$.

Price of robustness

The worst case ratio between the cost of the solution computed by A_r and the optimal one is called *price of robustness of* A_r.

$$P_{rob}(\mathcal{RTT}, A_r) = \max_{i \in I} \left\{ \frac{f(A_r(i))}{\min\{f(x) : x \in F(i)\}} \right\}$$

The minimum price of robustness among all the robust algorithms is called *price of robustness of problem*

Outline

Recoverable Robust Timetabling problem

Data description

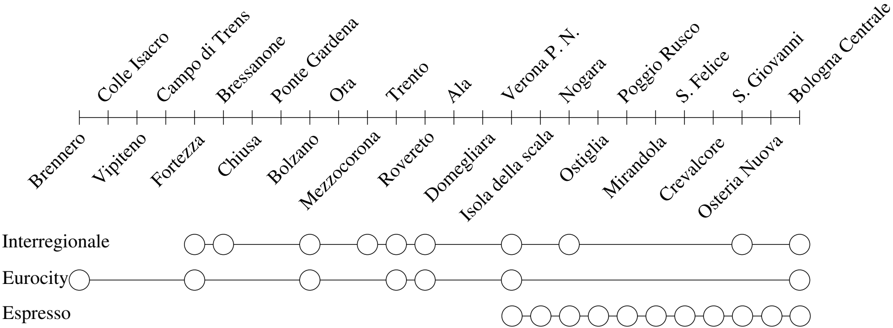
Algorithm

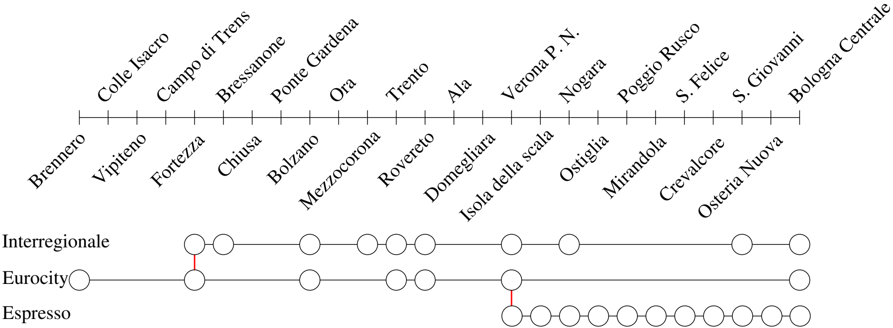
Experimental results

Conclusions

Single line corridors

- A corridor is a set of subsequent stations served by many trains of different type
- In practice (and intuitively) slow trains wait for faster trains to allow passengers to change from one train to another
- We require that changes between trains are only those connecting a fast train to the starting event of a slow train


Colle Isacro
Campo di Trens
Bressanone
Ponte Gardena
Ora
Trento
Ala
Verona P. N.
Nogara
Poggio Rusco
S. Felice
S. Giovanni
Bologna Centrale
Brennero
Vipiteno
Fortezza
Chiusa
Bolzano
Mezzocorona
Rovereto
Domegliara
Isola della scala
Ostiglia
Mirandola
Crevalcore
Osteria Nuova
Interregionale
Eurocity
Espresso


Brennero
Colle Isacro
Vipiteno
Campo di Trens
Fortezza
Bressanone
Chiusa
Ponte Gardena
Bolzano
Ora
Mezzocorona
Trento
Rovereto
Ala
Domegliara
Verona P. N.
Isola della scala
Nogara
Ostiglia
Poggio Rusco
Mirandola
S. Felice
Crevalcore
S. Giovanni
Osteria Nuova
Bologna Centrale
Interregionale
Eurocity
Espresso

Outline

Slack times

According to a timetable Π, the slack time $s(a)$ of an activity $a = (u, v)$ is the amount of time assigned to an activity in addiction to its minimum time needed $L(a)$

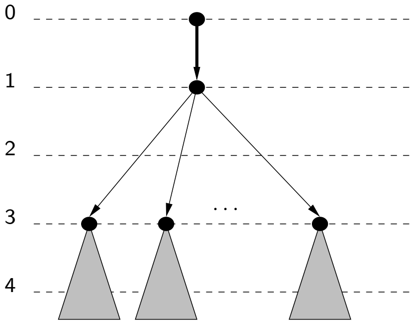
$$s(a) = \Pi(v) - \Pi(u) - L(a)$$

A robust timetable assigns at one least slack time of α every Δ subsequent events

Idea

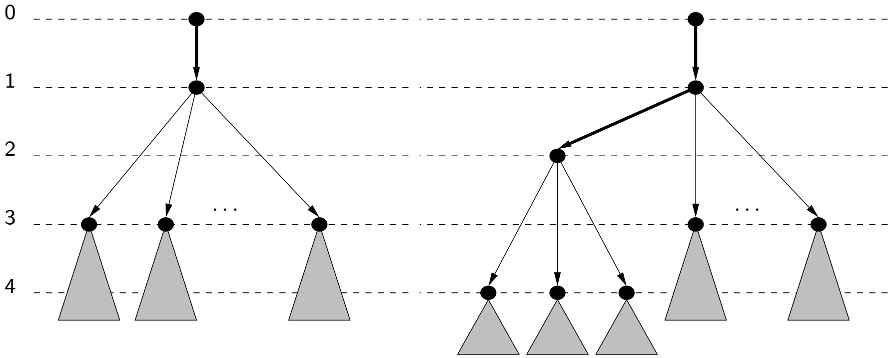
Assign the slack times as late as possible

Example for $\Delta = 2$ and $L(a) = 1$ for each $a \in \mathcal{A}$

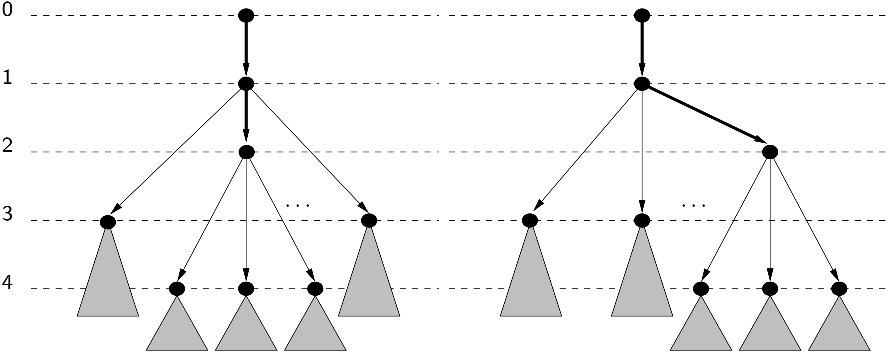


. . .

Example for $\Delta = 2$ and $L(a) = 1$ for each $a \in \mathcal{A}$



Example for $\Delta = 2$ and $L(a) = 1$ for each $a \in \mathcal{A}$



Outline

Corridor	Line	N. of Stations	N. of Trains
BrBo	Brennero–Bologna	48	68
MdMi	Modane–Milano	54	291
BzVr	Bolzano–Verona	27	65
PzBo	Piacenza–Bologna	17	25

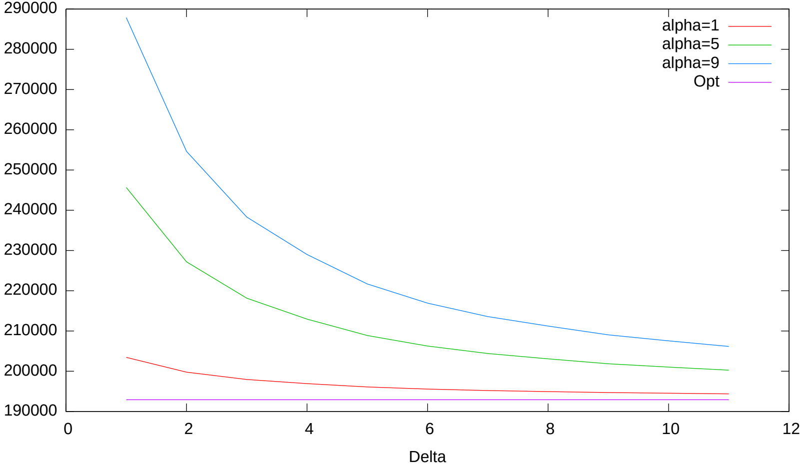
Table: Data used in the experiments.

Corridor	N. of Nodes	Max Traveling Time	Avg Activity Time	Max N. of Hops
BrBo	1103	516	9	66
MdMi	4358	318	8	27
BzVr	648	197	5	37
PzBo	163	187	10	14

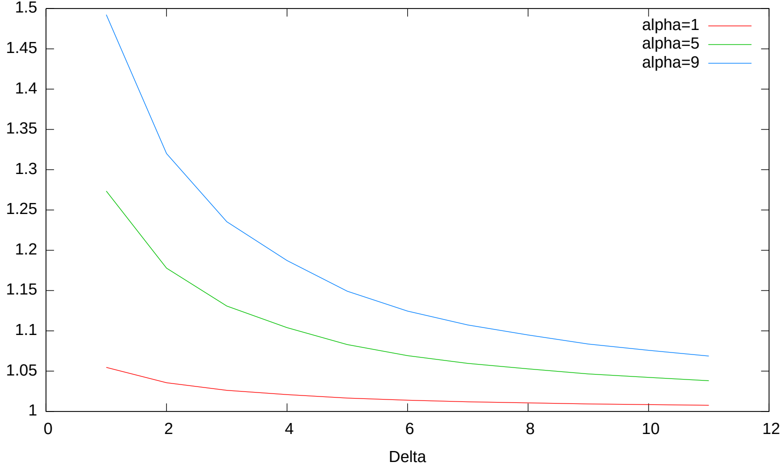
Table: Sizes of the trees.

$\Delta \in \{1, 2, \ldots, 11\}$
$\alpha \in \{1, 5, 9, 13, 17\}$

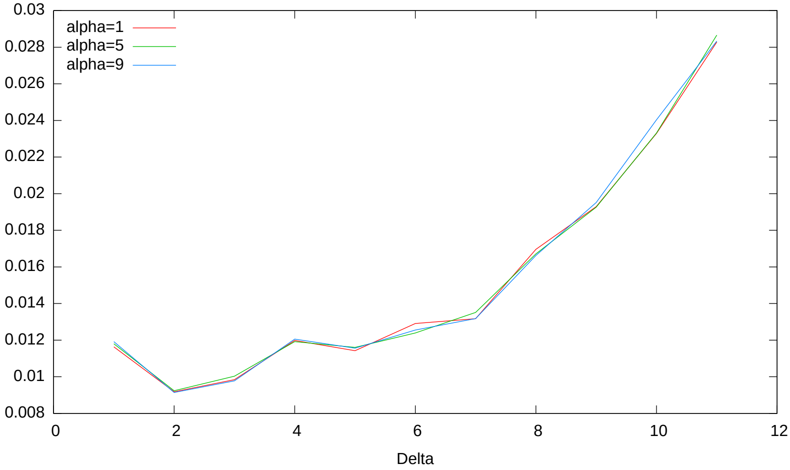
BrBo Objective function



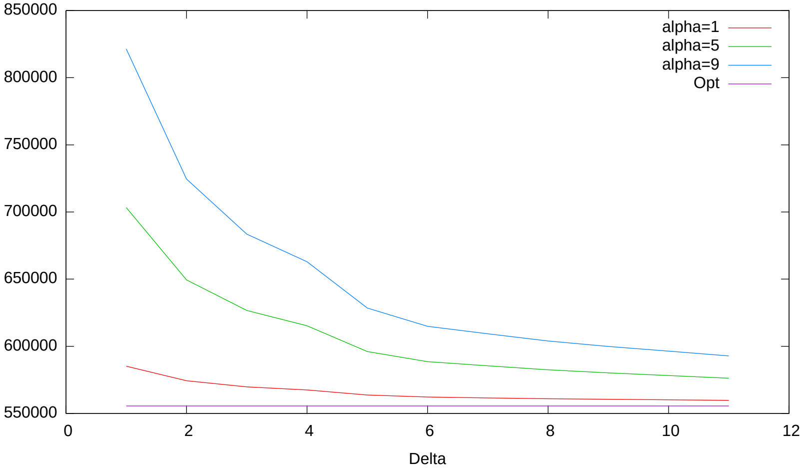
BrBo Price of robustness



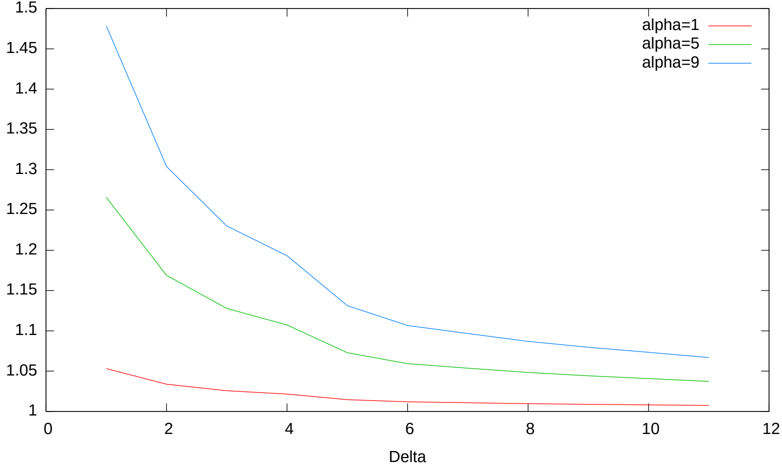
BrBo Computational time



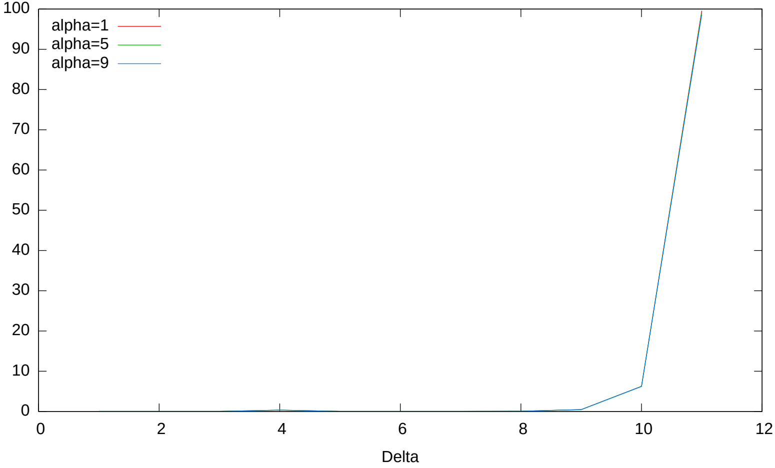
MdMi Objective function



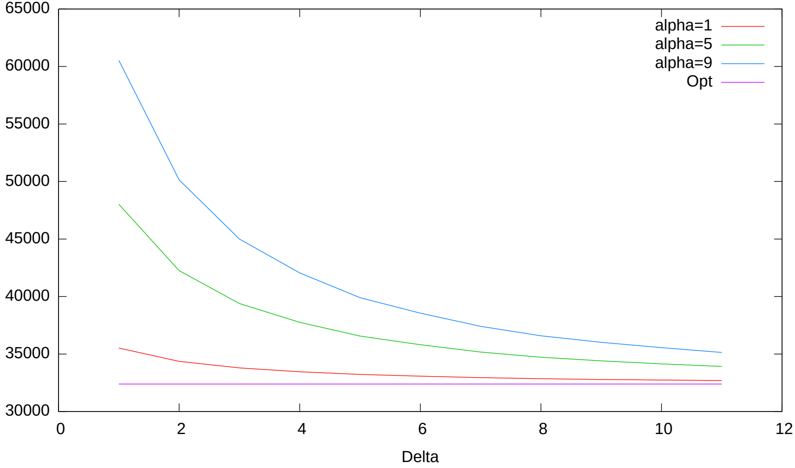
MdMi Price of robustness



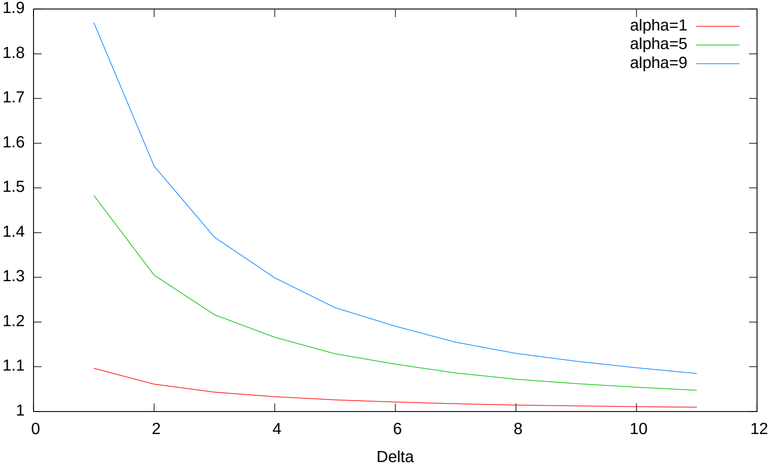
MdMi Computational time



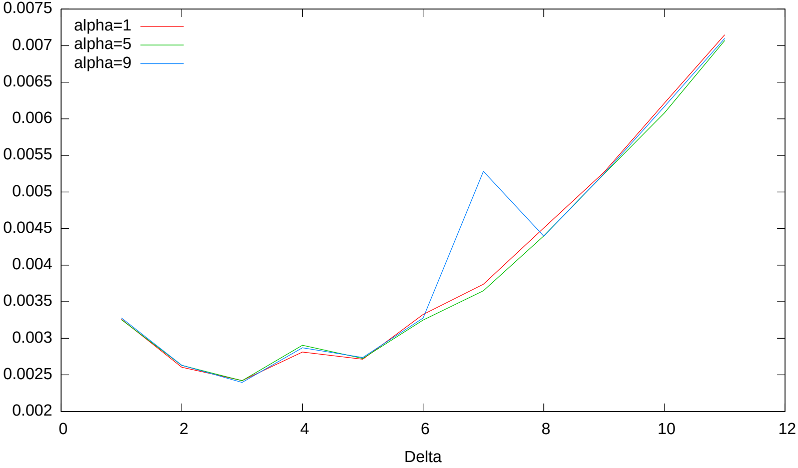
BzVr Objective function



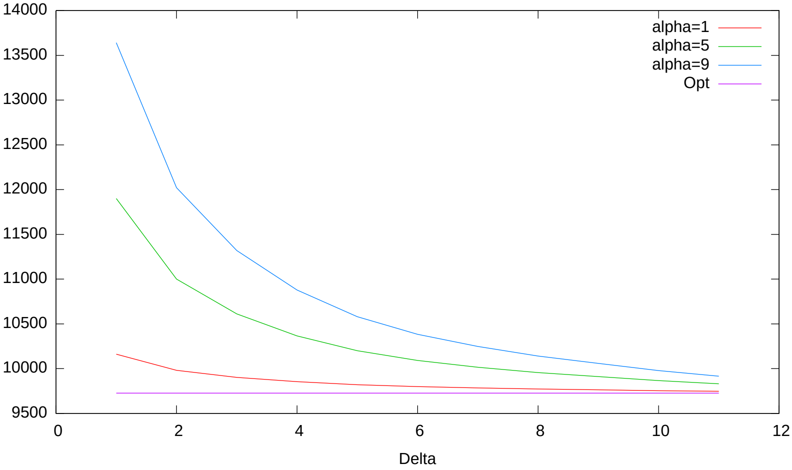
BzVr Price of robustness



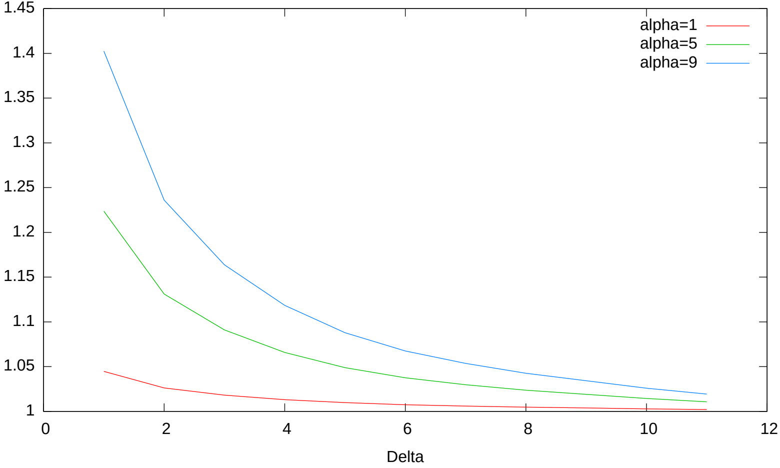
BzVr Computational time



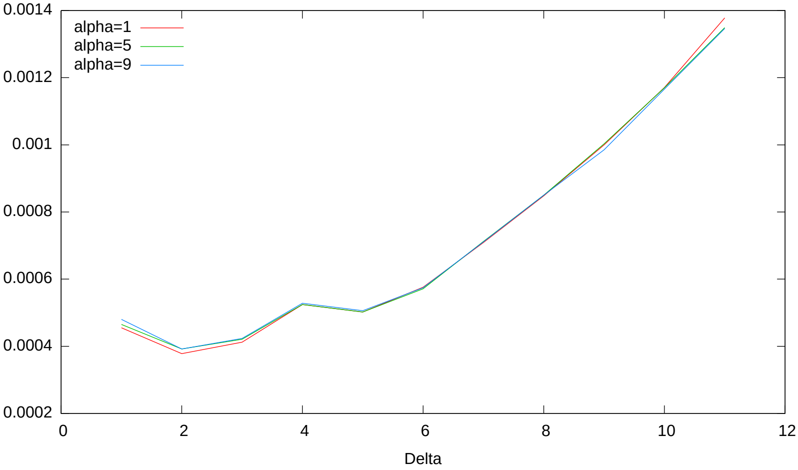
PzBo Objective function



PzBo Price of robustness



PzBo Computational time



Outline

- We modelled the timetabling problem for single-line corridors as out-trees
- We used the algorithm in [D. et al. 2008] in order to cope with one single delay
- We experimentally shown the performances of the algorithm applied on real data
- Although the problem is proved to be NP-hard, the obtained results show the applicability of the algorithm