Metrics and approximations in Scheduling Theory

Alexander Lazarev

V. A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences, Moscow, Russia
Lomonosov Moscow State University, Moscow, Russia
Institute of Physics and Technology State University, Moscow, Russia
National Research University Higher School of Economics, Moscow, Russia

orsot.ru

jobmath@mail.ru

V.A. TRAPEZNIKOV
INSTITUTE
OF CONTROL
SCIENCES
OF RUSSIAN ACADEMY
OF SCIENCES

May 21, 2019

Outline

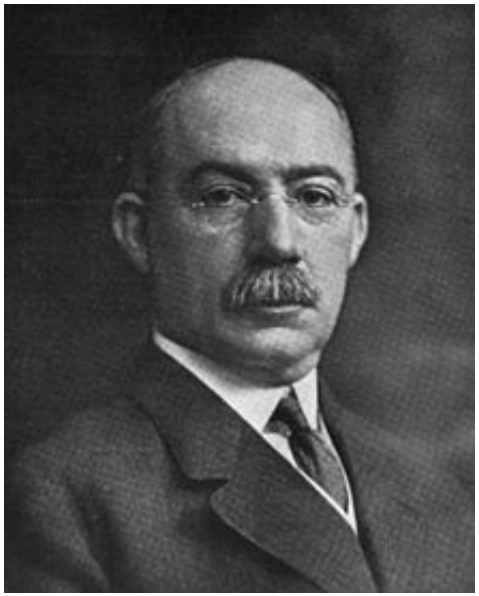
Section 1

History and challenges of Scheduling Theory

History and challenges of Scheduling Theory

Gantt chart and assembly line

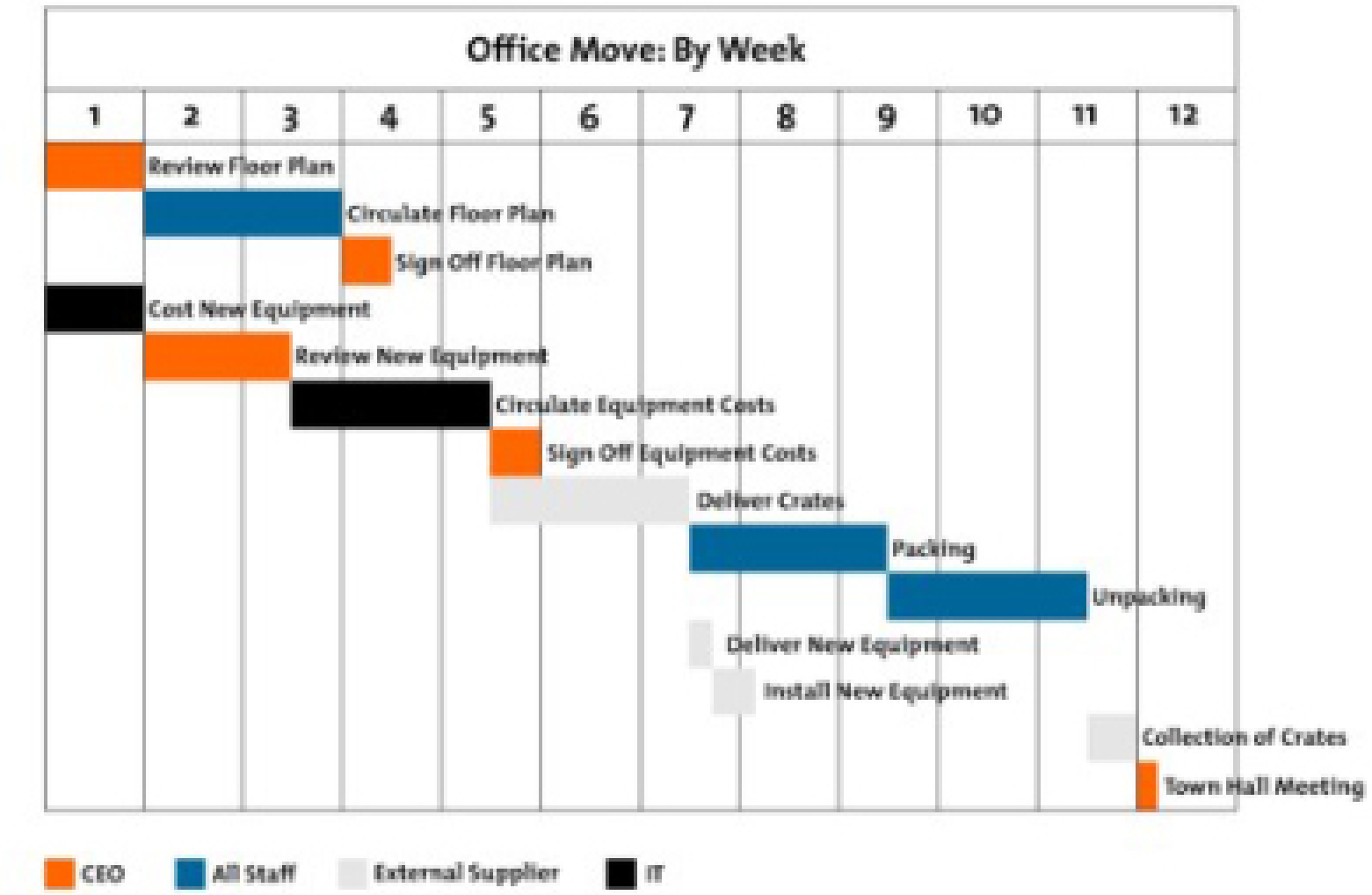
Gantt chart



Henry Laurence Gantt (1861-1919), American mechanical engineer and management consultant who is best known for his work in the development of scientific management. In the 1903 he introduced a graphical method of project schedule representation known as the Gantt chart (Gantt diagram).

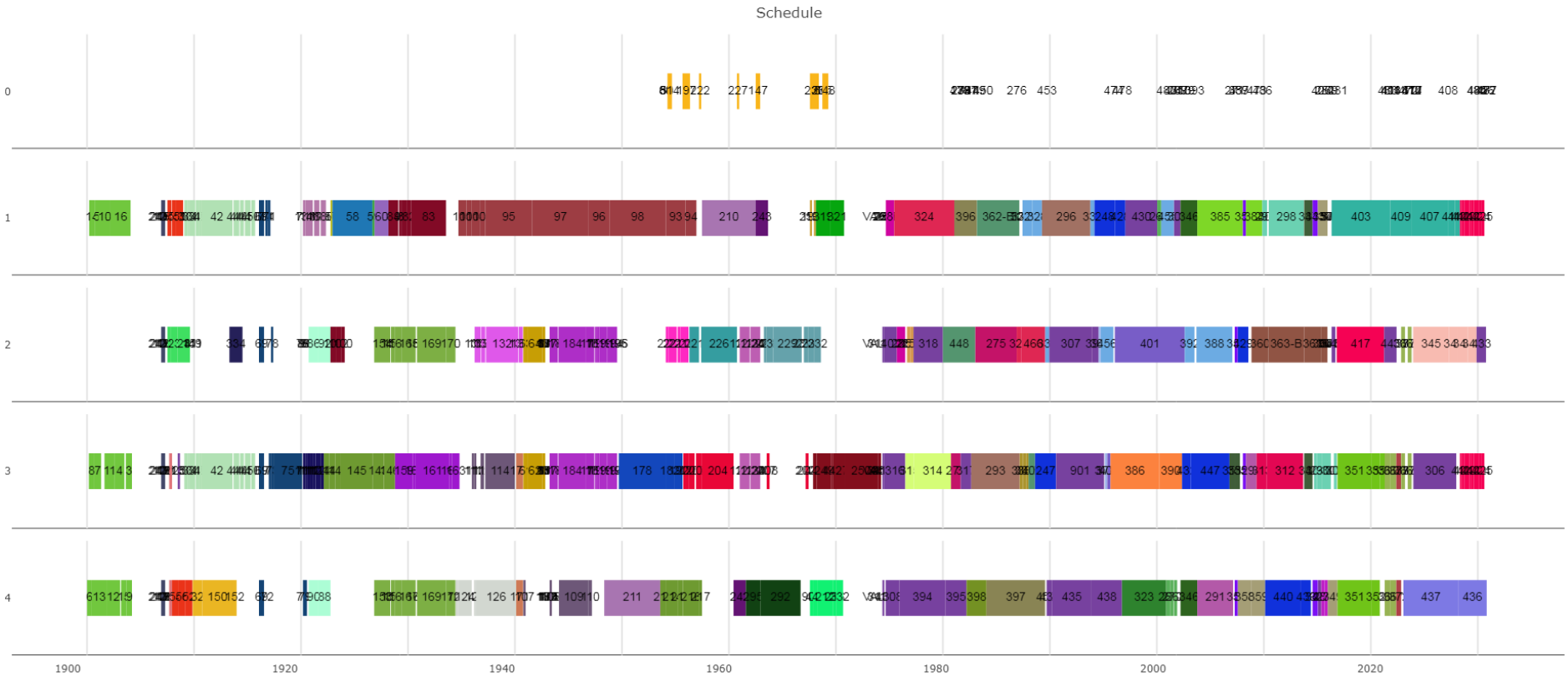
"A graphical daily balance in manufacture" (1903)
"Organizing for Work" (1919)

Gantt chart



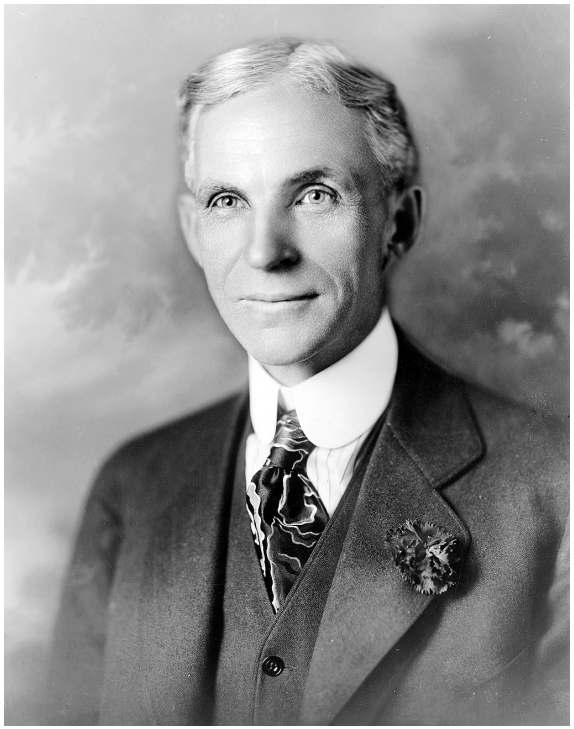
An example of Gantt chart

Gantt chart



Modern Gantt chart for production lines

The history of assembly line - The Ford Company



Henry Ford (July 30, 1863 – April 7, 1947) – a business magnate, the founder of the Ford Motor Company, and the sponsor of the development of the assembly line technique of mass production.

Ford assembly line

Ford magneto assembly line, 1913

Ford Model T assembly line

Scheduling theory term and pioneers

Scheduling theory term

Richard Ernest Bellman (1920–1984), American applied mathematician, famous for his work on dynamic programming and numerous important contributions in other fields of mathematics. In the 1954 he introduced the term "scheduling theory".

"Mathematical Aspects of Scheduling Theory" (1955)

Pioneers of scheduling theory. First results

J. R. Jackson. Scheduling a production to minimize maximum tardiness. Research Report 43, Management Science Research Project, University of California at Los Angeles, 1955

Pioneers of scheduling theory. First results

J. R. Jackson. Scheduling a production to minimize maximum tardiness. Research Report 43, Management Science Research Project, University of California at Los Angeles, 1955

W. E. Smith. Various optimizers for single-stage production. Naval Research Logistic Quarterly, 3:59-66, 1956

Pioneers of scheduling theory. First results

J. R. Jackson. Scheduling a production to minimize maximum tardiness. Research Report 43, Management Science Research Project, University of California at Los Angeles, 1955

W. E. Smith. Various optimizers for single-stage production. Naval Research Logistic Quarterly, 3:59-66, 1956

S. M. Johnson. Optimal two-and-three-stage production schedules with set-up times included. Naval Research Logistics Quarterly, 1:61-68, 1954

Pioneers of scheduling theory. First results

J. R. Jackson. Scheduling a production to minimize maximum tardiness. Research Report 43, Management Science Research Project, University of California at Los Angeles, 1955

W. E. Smith. Various optimizers for single-stage production. Naval Research Logistic Quarterly, 3:59-66, 1956

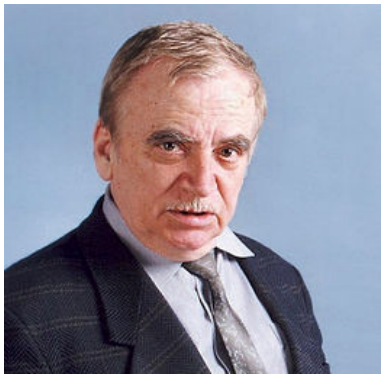
S. M. Johnson. Optimal two-and-three-stage production schedules with set-up times included. Naval Research Logistics Quarterly, 1:61-68, 1954

First monograph on Scheduling Theory

R. W. Conway, W. L. Maxwell, L. W. Miller. *Theory of Scheduling*, 1967 (Russian edition in 1975)

Pioneers of scheduling theory in USSR

Tanaev, V.S. and Shkurba, V.V.
Vvedenie v teoriyu raspisanii (Introduction to Scheduling Theory), Moscow: Nauka, 1975



Computational complexity in Scheduling Theory

Computational complexity

- If computational complexity of the algorithm that solves the problem is $O(n^k)$ operations, where k is some constant number independent from n, then this problem is called solvable in polynomial time. Algorithms for the problems mentioned before (Jackson's, Smith's, Johnson's problems) are polynomial. $O(\ n\log n)$
- All problems that are solvable within polynomial time formulate a class of problems denoted as P. Algorithms with corresponding computational complexity are called *polynomial*.
- If complexity of the algorithm depends on the values of numerical parameters of an example, for example, $O(nA)$, then this algorithm is called *pseudo-polynomial*.
- If complexity of the algorithm has the form of $O(n^x y^n)$, where x and y are some constants, then this algorithm is called *exponential*.

Class NP

- Suppose that we have a computer that includes a special "guessing" component (oracle).
- The oracle, given correct input data (i.e. a solution to the given instance exists), provides some (possibly correct) output data.
- The output data provided by oracle needs to be verified, i. e. we should construct an algorithm that checks if the output data contains a correct solution that is in accordance with provided input data. The problem of verifying data provided by oracle could also be formulated as an instance of recognition problem.

Class NP

- Class NP includes all the problems to which the solution (if such exists) can be guessed by an oracle, and:
- The amount of data in solution provided by oracle is polynomially bounded;
- The solution provided by oracle can be verified in polynomial time.

Reduction of one problem to another

It is said that problem A can be reduced to problem B in polynomial time ($A \propto B$), if a modification algorithm exists, such that:

Reduction of one problem to another

It is said that problem A can be reduced to problem B in polynomial time ($A \propto B$), if a modification algorithm exists, such that:

- The algorithm transforms any given instance I_A of problem A into a corresponding instance I_B of problem B in polynomial time

Reduction of one problem to another

It is said that problem A can be reduced to problem B in polynomial time ($A \propto B$), if a modification algorithm exists, such that:

- The algorithm transforms any given instance I_A of problem A into a corresponding instance I_B of problem B in polynomial time
- The answer to received instance I_B of problem B is **"YES"** if and only if the answer to the corresponding instance I_A of problem A is **"YES"**, too. (Or, less strictly, the solutions of corresponding instances I_A, I_B of problems A, B always match)

NP-complete and NP-hard problems

Problem B is called *NP*-hard, if any other problem $A \in NP$ can be reduced to problem B in polynomial time.

NP-complete and NP-hard problems

Problem B is called *NP*-hard, if any other problem $A \in NP$ can be reduced to problem B in polynomial time.

Problem B is called *NP*-complete, if:

- B is *NP*-hard;
- B belongs to class *NP*.

If any *NP*-complete problem is solvable in polynomial time, then all of the *NP*-complete are solvable in polynomial time ($P = NP$).

NP-complete and NP-hard problems

Problem B is called *NP*-hard, if any other problem $A \in NP$ can be reduced to problem B in polynomial time.

Problem B is called *NP*-complete, if:

- B is *NP*-hard;
- B belongs to class *NP*.

If any *NP*-complete problem is solvable in polynomial time, then all of the *NP*-complete are solvable in polynomial time ($P = NP$).

NP-hard problem B is called *NP*-hard in the strong sense if there is no pseudo-polynomial algorithm of solving this problem (supposed that $P \neq NP$).

Classification of problems in Machine scheduling

Each problem is denoted as $\alpha|\beta|\gamma$, where

- α describes characteristics of the problem that are related to machines
- β describes constraints and conditions of processing of requests.
- γ describes objective function.

Classification of problems in Machine scheduling

α describes characteristics of the problem related to machines. Possible values of α:

- 1 — single machine
- *Pm* — parallel machines
- *Qm* — parallel machines (non-equivalent)
- *Fm* — Flow-shop problem
- *Om* — Open-shop problem
- *Jm* — Job-shop problem
- ...

Classification of problems in Machine scheduling

β describes constraints and conditions of processing of requests. Possible contents of field β:

- r_j — release dates are specified
- d_j — due dates are specified
- D_j — deadlines are specified
- *prec* — precedence relations are specified
- *pmnt* — preemption is allowed
- *batch* — batching problem: groups of requests (*batches*) can be processed simultaneously.
- Other conditions: $p_j = p$, ...

γ describes objective function (e.g., C_{max}).

Denotations in Scheduling Theory

Objective functions:

- C_j — completion time
- $L_j = C_j - d_j$ — lateness
- $T_j = \max\{0, C_j - d_j\}$ — tardiness
- $E_j = \max\{0, d_j - C_j\}$ — earliness
- U_j — unit penalty: equals 1 if job j is late ($C_j > d_j$) and 0 in the opposite case

If request weights w_j are provided, all of the previous objective functions are called *weighed*, and are multiplied by the value of request weight (ex., weighed tardiness $w_j T_j$ is calculated as $w_j \max\{0, C_j - d_j\}$)

Denotations in Scheduling Theory

Optimization criteria:

1. Min-max criteria
   - $C_{max} \to min$ — minimizing maximum completion time (makespan), $C_{max} = \max\limits_{j \in N} C_j$. These problems are also called *performance problems*.
   - $L_{max} \to min$ — minimizing maximum lateness $L_{max} = \max\limits_{j \in N} L_j$
2. Summary criteria
   - $\sum\limits_{j \in N} C_j \to min$ — minimizing total completion time
   - $\sum\limits_{j \in N} T_j \to min$ — minimizing total tardiness
   - $\sum\limits_{j \in N} U_j \to min$ — minimizing total number of late jobs

Also, problems of maximizing these objective functions are considered (ex., $\sum\limits_{j \in N} T_j \to max$).

Problem complexity classification

NP-hardness in strong sense is a qualitative property!

Satisfability problem (SAT)

Boolean formula $f(x_1, x_2, ..., x_n)$, operations: AND, OR, NOT, (,)

$$\exists x_i = \{FALSE, TRUE\}, i \in \{1, ..., n\} : \ f(x_1, x_2, ..., x_n) = TRUE?$$

Cook, S. (1971). The complexity of theorem proving procedures. Proceedings of the Third Annual ACM Symposium on Theory of Computing. pp. 151–158. doi:10.1145/800157.805047.

Garey, M. R.; Johnson, D. S. (1979). Victor Klee (ed.). Computers and Intractability: A Guide to the Theory of NP-Completeness. A Series of Books in the Mathematical Sciences. San Francisco, Calif.: W. H. Freeman and Co. pp. x+338. ISBN 0-7167-1045-5. MR 0519066.

Classification of problems in Machine scheduling

Thus, record $F2|r_j|C_{max}$ denotes problem of minimizing makespan in Flow-shop system with two machines in case of non-simultaneous admission of requests. Other examples: $1|p_j = p,\ r_j|\sum w_j T_j$, $Pm|r_j,\ pmtn|\sum C_j,\ \ldots$

Classification of problems in Machine scheduling

Thus, record $F2|r_j|C_{max}$ denotes problem of minimizing makespan in Flow-shop system with two machines in case of non-simultaneous admission of requests. Other examples: $1|p_j = p, r_j|\sum w_j T_j$, $Pm|r_j, pmtn|\sum C_j$, ...

Some of previously considered problems in terms of machine scheduling:

- $1|r_j|L_{max}$ (Jackson's problem with non-zero release times) is NP-hard in the strong sense
- $1|r_j|\sum C_j$ (Smith's problem with non-zero release times) is NP-hard
- $F3||C_{max}$ (Johnson's problem with more than 2 machines) is NP-hard in the strong sense

Challenges in Scheduling Theory

Complexity challenges in Scheduling Theory

- The majority of formulations are NP-hard in the strong sense.
- In this case for real-life scaled problems it is impossible to find proven optimal solution (if $P \neq NP$).
- It leads to the demand for fast algorithms with «good» solutions?

Complexity challenges in Scheduling Theory

A set of «inspired by nature» heuristic methods

- Tabu search
- Simulated Annealing
- Ant Colony Optimization
- Particle Swarm Optimization

\+ speed and simple structure

– no estimations of accuracy (optimization criteria value delta)

Complexity challenges in Scheduling Theory

A set of «inspired by nature» heuristic methods

- Tabu search
- Simulated Annealing
- Ant Colony Optimization
- Particle Swarm Optimization

\+ speed and simple structure
– no estimations of accuracy (optimization criteria value delta)

Polynomial-Time Approximation Scheme (PTAS)

\+ guaranteed polynomial and accuracy estimations
– accuracy forms the complexity, e.g. $O(n^{\epsilon})$

Proposed alternative solution method

Metric approach

- Guaranteed accuracy provided by error upper bound estimations.
- Polynomial complexity does not depend on the accuracy.
- Method gives quantitative complexity estimations for the problem in addition to the qualitative property of NP-hardness.

Method is based on:

- a metric function for problem input data instance space;
- metric-based estimations of accuracy;
- polynomially-solvable subclasses of problem input data instances.

Practical challenges in Scheduling Theory

- In industrial cases objective functions are often unknown or are not clearly defined (e.g. RZD schedules, Gagarin Cosmonaut Training Center plans).
- Plans and schedules do not significantly change their structure for years.
- New solutions are formed based on a set of previous schedule structure.

Practical challenges in Scheduling Theory

- In industrial cases objective functions are often unknown or are not clearly defined (e.g. RZD schedules, Gagarin Cosmonaut Training Center plans).
- Plans and schedules do not significantly change their structure for years.
- New solutions are formed based on a set of previous schedule structure.

A new proposition for these cases

Objective function approximation

- There exists a set of previous problem input data instances and solutions.
- Objective function is unknown but linear to the completion time of the job.
- The first goal: to find the form and coefficients of the objective function;
- The second goal: to provide the solution for the next instance.

Section 2

Theoretical results in Scheduling

2 Theoretical results in Scheduling

- Metrics approach in scheduling theory
  - The problem $1|r_j|L_{\max}$
  - $1|r_j|L_{\max}$ solvable cases
  - Pareto-optimal cases
  - Instance metric
  - The closest solvable instance construction LP-problem
  - Metrics for $1|r_j|\sum T_j$
  - Measure of polynomial unsolvability
  - Example: Metrics for the railway scheduling problem
- Objective function approximation
  - Motivation and basic idea
  - The problem $1||\sum \omega_j C_j$
  - Solvability
  - Approximation problem
- Dual complexity reduction
- Graphical approach

Metrics approach in scheduling theory

The problem $1|r_j|L_{\max}$ - minimizing maximum lateness

$1|r_j|L_{\max}$

Single machine, n jobs

r_j – release time;

$p_j > 0$ – processing time;

d_j – due date.

$j \in N = \{1, 2, \ldots, n\}$

The problem $1|r_j|L_{\max}$ - minimizing maximum lateness

$1|r_j|L_{\max}$

Single machine, n jobs
r_j – release time;
$p_j > 0$ – processing time;
d_j – due date.
$j \in N = \{1, 2, \ldots, n\}$

Preemptions of a job are not allowed. The machine can process at most one job at any time.

The problem $1|r_j|L_{\max}$ - minimizing maximum lateness

$1|r_j|L_{\max}$

Single machine, n jobs
r_j – release time;
$p_j > 0$ – processing time;
d_j – due date.
$j \in N = \{1, 2, \ldots, n\}$

Preemptions of a job are not allowed. The machine can process at most one job at any time.

A schedule describes order of processing the jobs: a permutation(sequence) $\pi = (j_1, j_2, ..., j_n)$.

The problem $1|r_j|L_{\max}$ - minimizing maximum lateness

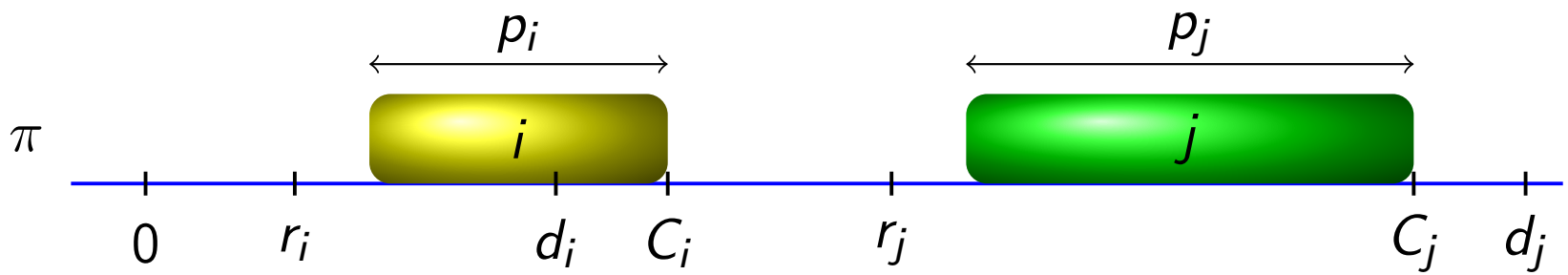
$1|r_j|L_{\max}$

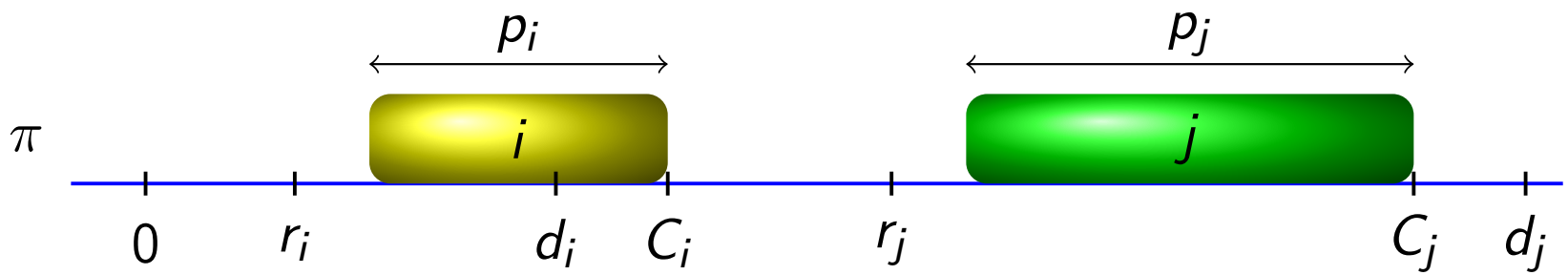
Single machine, n jobs
r_j – release time;
$p_j > 0$ – processing time;
d_j – due date.
$j \in N = \{1, 2, \ldots, n\}$

Preemptions of a job are not allowed. The machine can process at most one job at any time.

A schedule describes order of processing the jobs: a permutation(sequence) $\pi = (j_1, j_2, ..., j_n)$.

Graham R.L., Lawler E.L., Lenstra J.K., Rinnooy Kan A.H.G. **1979**


p_i
p_j
π
i
j
0
r_i
d_i
C_i
r_j
C_j
d_j



$$F(\pi) = \max_{j \in N}\{C_j - d_j\} \to \min_{\pi}$$

NP-hard in strong sense

Lenstra J.K., Rinnooy Kan A.H.G., Brucker, P. **1977**

$1|r_j|L_{\max}$ solvable cases

1) $r_j = 0, \forall\, j \in N$. $O(n \log n)$
Jackson J.R. **1955**

1') $d_j = const, \forall\, j \in N$. $O(n \log n)$

1'') $p_j = const, \forall\, j \in N$.
Simons B. **1983**. $O(n^2 \log n)$

$1|r_j|L_{\max}$ solvable cases

2) $O(n^3 \log n)$

$$\begin{cases} d_1 \leq d_2 \leq \cdots \leq d_n; \\ d_1 - r_1 - p_1 \geq d_2 - r_2 - p_2 \geq \cdots \geq d_n - r_n - p_n. \end{cases} \quad (1)$$

2') $d_j = r_j + p_j + const, \forall\ j \in N.$ $O(n^3 \log n)$

$\{1, P, Q, R\}|r_j|\{L_{\max}, C_{\max}\}$ $O(n^3 \log n)$

Lazarev A.A., Sadykov R.R., Sevastyanov S.V. **1988-2007**

$1|r_j|L_{\max}$ solvable cases

2) $O(n^3 \log n)$

$$\begin{cases} d_1 \le d_2 \le \cdots \le d_n; \\ d_1 - r_1 - p_1 \ge d_2 - r_2 - p_2 \ge \cdots \ge d_n - r_n - p_n. \end{cases} \quad (1)$$

2') $d_j = r_j + p_j + const, \forall\ j \in N.$ $O(n^3 \log n)$

$\{1, P, Q, R\}|r_j|\{L_{\max}, C_{\max}\}$ $O(n^3 \log n)$

Lazarev A.A., Sadykov R.R., Sevastyanov S.V. **1988-2007**

3) $\max\limits_{k \in N}\{d_k - r_k - p_k\} \le d_j - r_j, \forall\ j \in N.$ $O(n^2 \log n)$

Hoogeveen J. A. **1996**

$1|r_j|L_{\max}$ solvable cases

2) $O(n^3 \log n)$

$$\begin{cases} d_1 \le d_2 \le \cdots \le d_n; \\ d_1 - r_1 - p_1 \ge d_2 - r_2 - p_2 \ge \cdots \ge d_n - r_n - p_n. \end{cases} \quad (1)$$

2') $d_j = r_j + p_j + const, \forall\ j \in N.$ $O(n^3 \log n)$

$\{1, P, Q, R\}|r_j|\{L_{\max}, C_{\max}\}$ $O(n^3 \log n)$

Lazarev A.A., Sadykov R.R., Sevastyanov S.V. **1988-2007**

3) $\max\limits_{k \in N}\{d_k - r_k - p_k\} \le d_j - r_j, \forall\ j \in N.$ $O(n^2 \log n)$

Hoogeveen J. A. **1996**

$1|r_j|L_{\max}$ solvable cases

4) *NP*-hard in ordinary sense $O(n^2P + np_{\max}P)$

$$\begin{cases} d_1 \leq d_2 \leq \cdots \leq d_n; \\ r_1 \geq r_2 \geq \cdots \geq r_n; \\ r_j, p_j, d_j \in \mathbb{Z}^+, \forall\ j \in N. \end{cases} \tag{2}$$

Lazarev A.A., Schulgina O.N. **1998**

$P = r_{\max} + \sum_{j=1}^{n} p_j - r_{\min},\ r_{\max} = \max_{j \in N} r_j,\ r_{\min} = \min_{j \in N} r_j,\ p_{\max} = \max_{j \in N} p_j$

$1|r_j|L_{\max}$ solvable cases

5)

$$\begin{cases} d_1 \leq d_2 \leq \cdots \leq d_n; \\ d_1 - \alpha r_1 - \beta p_1 \geq d_2 - \alpha r_2 - \beta p_2 \geq \cdots \geq d_n - \alpha r_n - \beta p_n; \\ \alpha \in [1, \infty), \beta \in [0, 1]. \end{cases} \tag{3}$$

$1|r_j|L_{\max}$ solvable cases

5)

$$\begin{cases} d_1 \le d_2 \le \cdots \le d_n; \\ d_1 - \alpha r_1 - \beta p_1 \ge d_2 - \alpha r_2 - \beta p_2 \ge \cdots \ge d_n - \alpha r_n - \beta p_n; \\ \alpha \in [1, \infty), \beta \in [0, 1]. \end{cases} \quad (3)$$

5')

$d_j = \alpha r_j + \beta p_j + const, \ \forall \ j \in N, \alpha \in [1, \infty), \beta \in [0, 1].$ **2009**

$1|r_j|L_{\max}$ solvable cases

5)

$$\begin{cases} d_1 \le d_2 \le \cdots \le d_n; \\ d_1 - \alpha r_1 - \beta p_1 \ge d_2 - \alpha r_2 - \beta p_2 \ge \cdots \ge d_n - \alpha r_n - \beta p_n; \\ \alpha \in [1, \infty), \beta \in [0, 1]. \end{cases} \quad (3)$$

5')

$d_j = \alpha r_j + \beta p_j + const, \ \forall \ j \in N, \alpha \in [1, \infty), \beta \in [0, 1].$ **2009**

$O(n^3 \log n)$

Pareto-optimal cases

$1 \mid d_i \leq d_j, d_i - \alpha r_i - \beta p_i \geq d_j - \alpha r_j - \beta p_j \mid L_{\max}, C_{\max}$

$$\begin{cases} d_1 \leq d_2 \leq \cdots \leq d_n; \\ d_1 - \alpha r_1 - \beta p_1 \geq d_2 - \alpha r_2 - \beta p_2 \geq \cdots \geq d_n - \alpha r_n - \beta p_n; \\ \alpha \in [1, \infty), \beta \in [0, 1]. \end{cases} \tag{4}$$

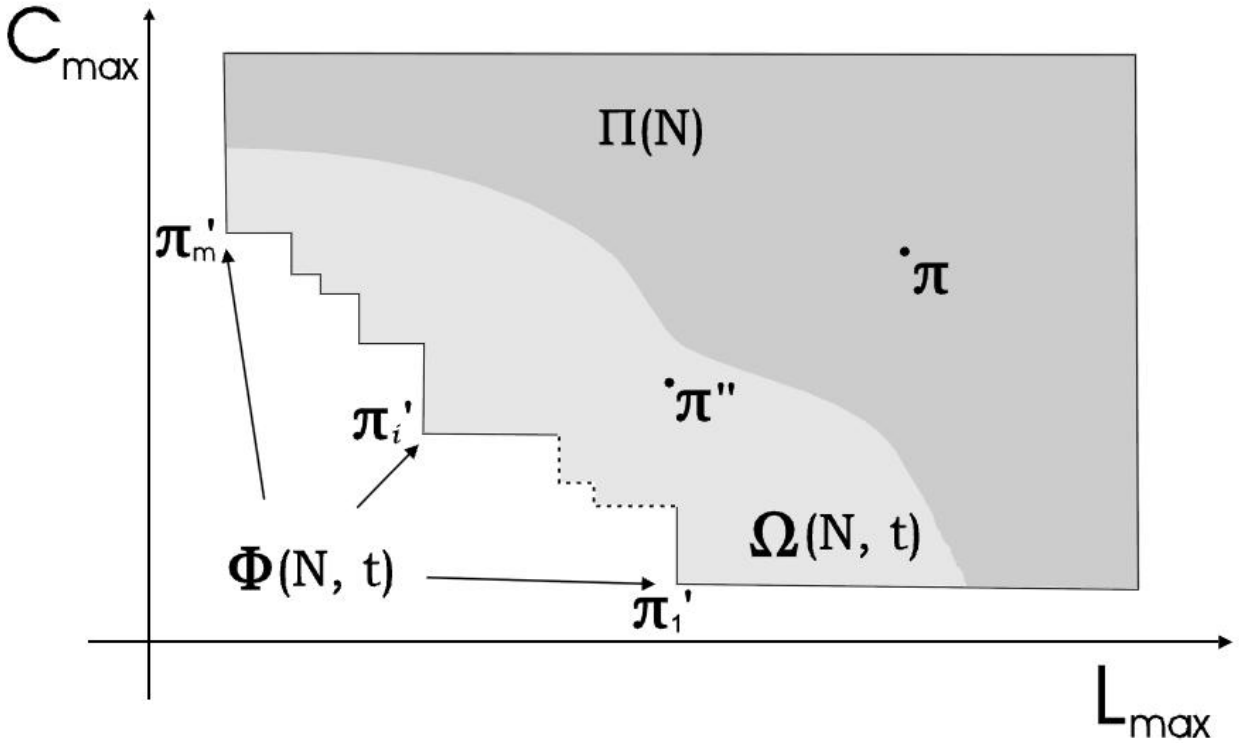
$1 \mid d_i \leq d_j, d_i - \alpha r_i - \beta p_i \geq d_j - \alpha r_j - \beta p_j \mid L_{\max}, C_{\max}$

$$1 \leq || \Phi(N, t) || \leq n$$

$O(n^3 \log n)$

Pareto-optimal cases

$1 \mid d_i \leq d_j, d_i - \alpha r_i - \beta p_i \geq d_j - \alpha r_j - \beta p_j \mid L_{max}, C_{max}$



Instance metric

The approach

- Set of parameters $\Omega = \{r_1, ..., r_n, p_1, ..., p_n, d_1, ..., d_n\}$ characterizes an instance.
- An instance can be considered as a vector in 3n-dimensional space of parameters.

Instance metric

The approach

- Set of parameters $\Omega = \{r_1, ..., r_n, p_1, ..., p_n, d_1, ..., d_n\}$ characterizes an instance.
- An instance can be considered as a vector in 3n-dimensional space of parameters.

Definitions

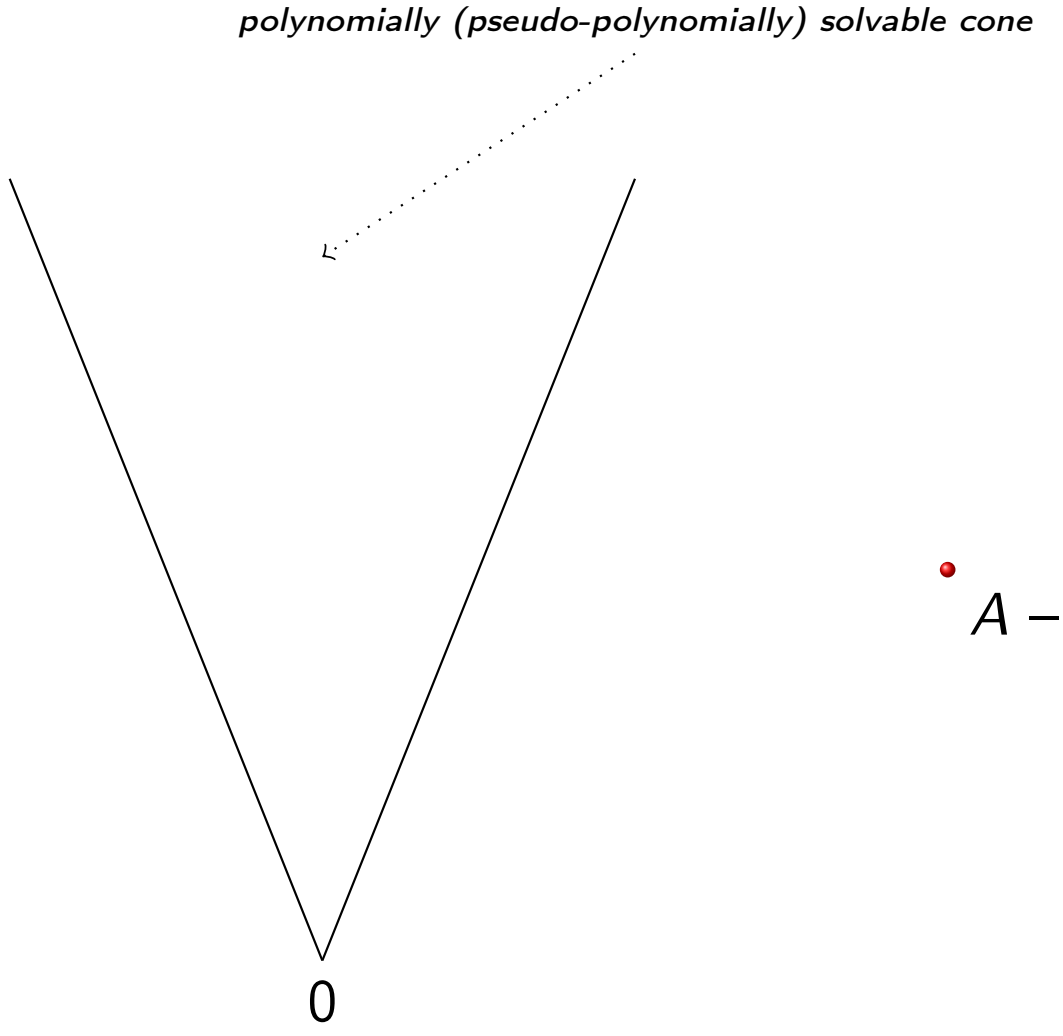
- For a particular value of parameter $\omega \in \Omega$ in the instance A we will use upper index : ω^A.
- The value of the objective function F in the instance A under the schedule π will be denoted as $F^A(\pi)$.
- We denote the optimal schedule for the instance A as π^A.

Instance metric

Any instance is point in $m = 3n$-dimension space.

Instance metric

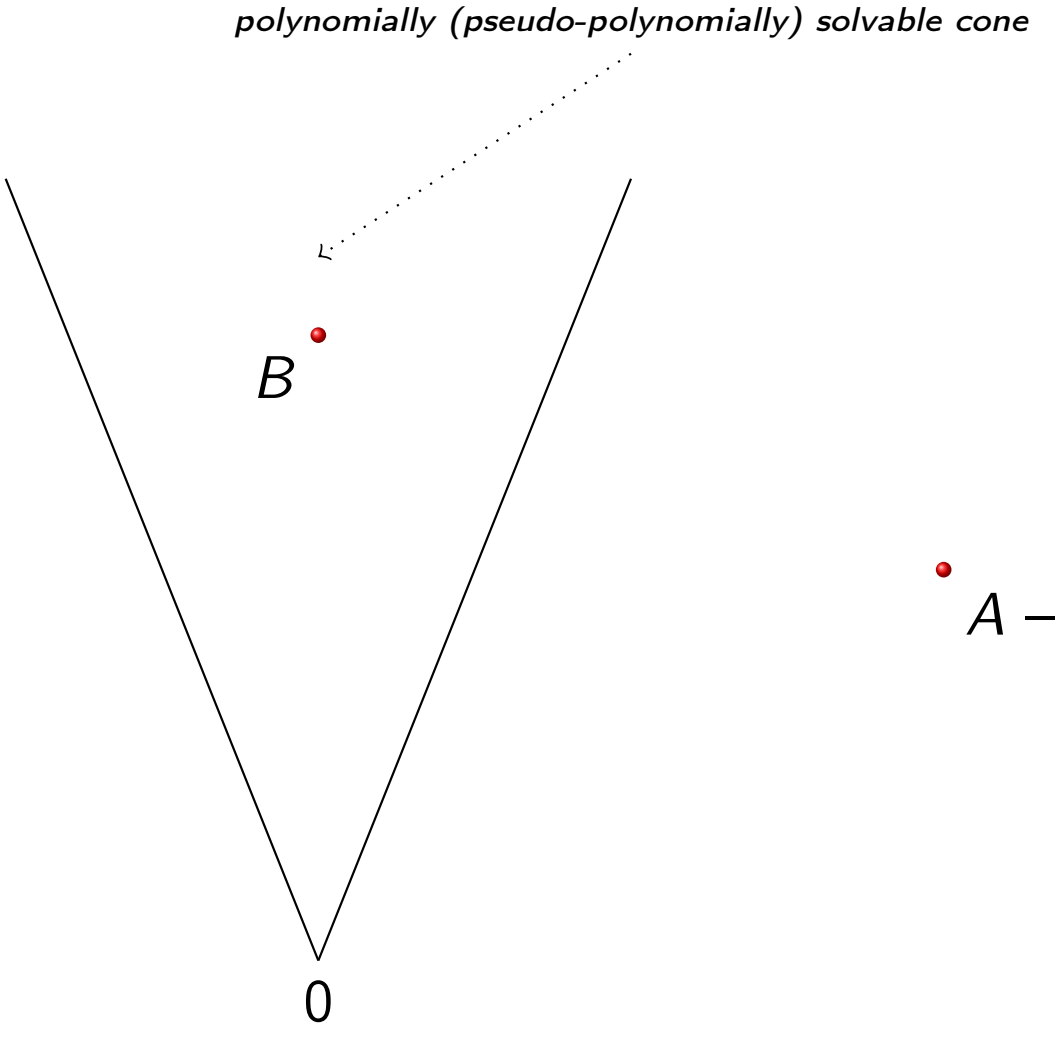
Any instance is point in $m = 3n$-dimension space.



- A – "hard" instance

Instance metric

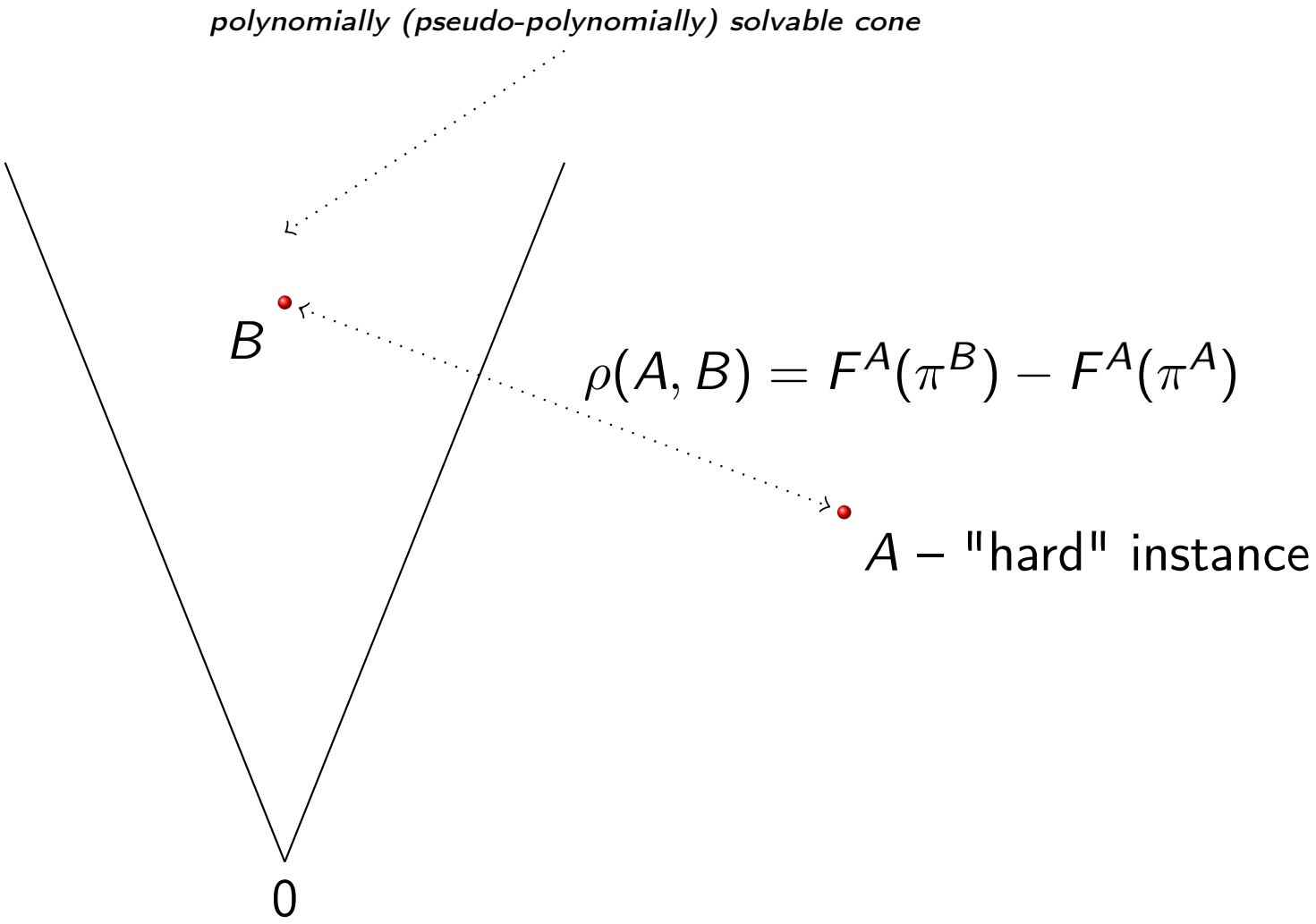
Any instance is point in $m = 3n$-dimension space.



A – "hard" instance

Instance metric

Any instance is point in $m = 3n$-dimension space.



Instance metric

- An absolute error of the approximation scheme is bounded by the metric function $\rho(A, B)$.
- The problem $1|r_j|L_{max}$ is reduced to the minimization of the function $\rho(A, B)$ – from arbitrary instance A to the closest polynomially solvable instance B.

$1|r_j|L_{max}$

$$0 \leq \rho(A,B) = F^A(\pi^B) \quad - \quad F^A(\pi^A) \leq$$
$$(\max\{r_j^A - r_j^B\} - \min\{r_j^A - r_j^B\}) +$$
$$(\sum |p_j^A - p_j^B|) +$$
$$(\max\{d_j^A - d_j^B\} - \min\{d_j^A - d_j^B\})$$

Instance metric

Metric properties

$$\varphi(A) = \max_{j \in N}(r_j^A) - \min_{j \in N}(r_j^A) + \max_{j \in N}(d_j^A) - \min_{j \in N}(d_j^A) + \sum_{j \in N} |p_j^A| \geq 0.$$

$$\begin{cases} \varphi(A) = 0 \Longleftrightarrow A \equiv 0; \\ \varphi(\alpha A) = \alpha \varphi(A); \\ \varphi(A + B) \leq \varphi(A) + \varphi(B). \end{cases} \tag{5}$$

$||A|| = \varphi(A)$ $\rho(A, B) = ||A - B||$.

The closest solvable instance construction LP-problem

$||A|| = \varphi(A)$ $\rho(A, B) = ||A - B||$

The closest solvable instance construction LP-problem

$||A|| = \varphi(A)$ $\rho(A, B) = ||A - B||$

Polynomially (pseudo-polynomially) solvable case

$$\mathcal{A}R + \mathcal{B}P + \mathcal{C}D \leq \mathcal{H}$$

$\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$ – matrixes, $R, P, D, \mathcal{H}$ – vectors.

The closest solvable instance construction LP-problem

Projection of an instance A to a polynomially (pseudo-polynomially) solvable case

The minimum absolute error among all instances from solvable area,– instance B.

The closest solvable instance construction LP-problem

Projection of an instance A to a polynomially (pseudo-polynomially) solvable case

The minimum absolute error among all instances from solvable area,– instance B.

$O(n \log n)$

$$\begin{cases} \rho(A,B) = (x_r - y_r) + \sum(x_p - y_p) + (x_d - y_d) \to min \\ y_r \le r_j^A - r_j^B \le x_r, \forall\ j; \\ -x_p^j \le p_j^A - p_j^B \le x_p^j, \forall\ j, x_p^j \ge 0; \\ y_d \le d_j^A - d_j^B \le x_d, \forall\ j; \\ \mathcal{A}R^B + \mathcal{B}P^B + \mathcal{C}D^B \le \mathcal{H}. \end{cases}$$

The closest solvable instance construction LP-problem

$$
\begin{cases}
\rho(A,B) = (x_r - y_r) + \sum\limits_j (x_p^j - y_p^j) + (x_d - y_d) \to \min\limits_{\substack{x_r, y_r, x_p^j, x_d, y_d, \\ r_j^B, p_j^B, d_j^B, \forall\, j}} \\
y_r \le r_j^A - r_j^B \le x_r, \forall\, j; \\
-x_p^j \le p_j^A - p_j^B \le x_p^j, \forall\, j, x_p^j \ge 0; \\
y_d \le d_j^A - d_j^B \le x_d, \forall\, j; \\
d_1^B \le d_2^B \le \cdots \le d_n^B; \\
d_1^B - \alpha r_1^B - \beta p_1^B \ge d_2^B - \alpha r_2^B - \beta p_2^B \ge \cdots \ge d_n^B - \alpha r_n^B - \beta p_n^B; \\
\alpha \in [1, \infty), \beta \in [0, 1].
\end{cases}
$$

$4 + 4n$ variables, $8n - 2$ inequalities $O(n \log n)$

The closest solvable instance construction LP-problem

Example of $\mathcal{A}R^B + \mathcal{B}P^B + \mathcal{C}D^B \leq \mathcal{H}$:

Inequalities for the subclass $1|d_i \leq d_j,\ d_i - r_i - p_i \geq d_j - r_j - p_j|L_{max}$

Instance $I = \{(r_j^I, p_j^I, d_j^I)|j \in N\}$ belongs to this subclass, if there exists the numbering $\{1, 2, ..., n\}$, which satisfies the following inequalities

$$d_1^I \leq ... \leq d_n^I;\ \Delta_1^I \geq ... \geq \Delta_n^I,$$

where $\Delta_j^I = d_j^I - r_j^I - p_j^I$. For this subclass $\mathcal{A}^{(n-1)\times n}$ is

$$\mathcal{A}R^B + \mathcal{A}P^B - \mathcal{A}D^B \leq 0,\ \mathcal{A}D^B \leq 0.$$

$$and\ \mathcal{A}^{(n-1)\times n} = \begin{bmatrix} 1 & -1 & 0 & 0 & ... & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & ... & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & ... & 0 & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... & ... \\ 0 & 0 & 0 & 0 & ... & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & ... & 0 & 1 & -1 \end{bmatrix};$$

Metrics for $1|r_j|L_{\max}$ and in general case

$1|r_j|L_{\max}$

Lazarev A.A. *Estimation of Absolute Error in Scheduling Problems of Minimizing the Maximum Lateness*, Dokl. Math., Vol. 76, 2007, P. 572–574.

General case

$$F(\pi) = \sum_{j \in N} \phi_j(\pi, r_1, ..., r_n, p_1, ..., p_n, d_j)$$

$$\rho(A, B) = \sum_{j \in N} \sum_{i \in N} (R_{ji}|r_j^A - r_j^B| + P_{ji}|p_j^A - p_j^B|) + \sum_{j \in N} D_j|d_j^A - d_j^B|,$$

where $R_{ji} \geq |\frac{\partial \phi_j}{\partial r_i}|$, $P_{ji} \geq |\frac{\partial \phi_j}{\partial p_i}|$, $D_{ji} \geq |\frac{\partial \phi_j}{\partial d_i}|$.

Metrics for $1|r_j|\sum T_j$

Problem formulation

Set $N = \{1, 2, ..., n\}$ of n independent jobs must be processed on a single machine.

- The machine can handle only one job at a time.
- Preemptions are not allowed.
- The machine is ready to start processing at time 0.

For each job j, $j \in N$, a processing time $p_j \geq 0$, release date $r_j \geq 0$ and due date d_j are given.

In early schedule π: $S_{j_1} = r_{j_1}$ and $S_{j_k} = \max\{r_{j_k}, C_{j_{k-1}}\}$ for $k = 2, ..., n$,

Metrics for $1|r_j|\sum T_j$

Objective function

- $T_j(\pi) = \max\{0, C_j(\pi) - d_j\}$ is the tardiness of the job j in the schedule π.
- $\sum_{j \in N} T_j(\pi)$ is the total tardiness in the schedule π.

Metrics for $1|r_j|\sum T_j$

Objective function

- $T_j(\pi) = \max\{0, C_j(\pi) - d_j\}$ is the tardiness of the job j in the schedule π.
- $\sum\limits_{j \in N} T_j(\pi)$ is the total tardiness in the schedule π.

The total tardiness minimization problem is denoted as $1|r_j|\sum T_j$.

Metrics for $1|r_j|\sum T_j$

Objective function

- $T_j(\pi) = \max\{0, C_j(\pi) - d_j\}$ is the tardiness of the job j in the schedule π.
- $\sum\limits_{j \in N} T_j(\pi)$ is the total tardiness in the schedule π.

The total tardiness minimization problem is denoted as $1|r_j|\sum T_j$.

Du J., Leung J.Y.T. *Minimizing total tardiness on one machine is NP-hard* Mathematics of Operations Research, Vol. 15. 1990, N. 3, P. 483 – 495. Problem $1|r_j|\sum T_j$ is NP-hard in the ordinary sense.

Metrics for $1|r_j|\sum T_j$

Theorem

Function

$$\rho(A,B) = n \cdot \max_{j \in N} |r_j^A - r_j^B| + n \cdot \sum_{j \in N} |p_j^A - p_j^B| + \sum_{j \in N} |d_j^A - d_j^B|$$

satisfies the axioms of metric function and is applicable as parameters space metric.

Lemma

For any instances A, B and schedule π

$$|\sum_{j \in N} T_j^A(\pi) - \sum_{j \in N} T_j^B(\pi)| \leq \rho(A,B)$$

Metrics for $1|r_j|\sum T_j$

Lemma

For any instances A, B and schedule π

$$|\sum_{j\in N} T_j^A(\pi) - \sum_{j\in N} T_j^B(\pi)| \leq \rho(A,B)$$

Theorem

For any instances A and B

$$\sum_{j\in N} T_j^A(\pi^B) - \sum_{j\in N} T_j^A(\pi^A) \leq 2\rho(A,B)$$

There π^A and π^B are optimal schedules for instances A and B, respectively.

Metrics for $1|r_j|\sum T_j$

LP approximation model

$$min\ f = n \cdot (y^r - x^r) + n \cdot \sum_{j=1}^{n}(y_j^p - x_j^p) + \cdot \sum_{j=1}^{n}(y_j^d - x_j^d)$$

s.t.

$$x^r \le r_j^A - r_j^B \le y^r,$$

$$x_j^p \le p_j^A - p_j^B \le y_j^p,$$

$$x_j^d \le d_j^A - d_j^B \le y_d,$$

$$r_j^B \ge 0,\ p_j^B \ge 0,\ j \in N,$$

$$\mathcal{A} \cdot R^B + \mathcal{B} \cdot P^B + \mathcal{C} \cdot D^B \le \mathcal{H}$$

Solvable case class constraints

LP with 7n+2 variables : r_j^B, p_j^B, d_j^B, x_j^p, y_j^p, x_j^d, y_j^d, x^r, y^r, $j = 1, ..., n$.

Metrics for $1|r_j|\sum T_j$

Solvable classes

- $\{\mathcal{PR} - case : p_j = p,\ r_j = r,\ j \in N\}$;
- $\{\mathcal{PD} - case : p_j = p,\ d_j = d,\ j \in N\}$;
- $\{\mathcal{RD} - case : r_j = r,\ d_j = d,\ j \in N\}$;

Lemma

For each class the minimum of the function $f(p, d, r)$ could be constructed in $O(n)$ operations. For example, for $\mathcal{PR} - case$ it has the minimum at the point with $p \in \{p_1^A, ..., p_n^A\}$ and $r = \frac{r_{max}^A - r_{min}^A}{2}$, where $r_{max}^A = \max\limits_{j \in N} r_j^A$, $r_{min}^A = \min\limits_{j \in N} r_j^A$.

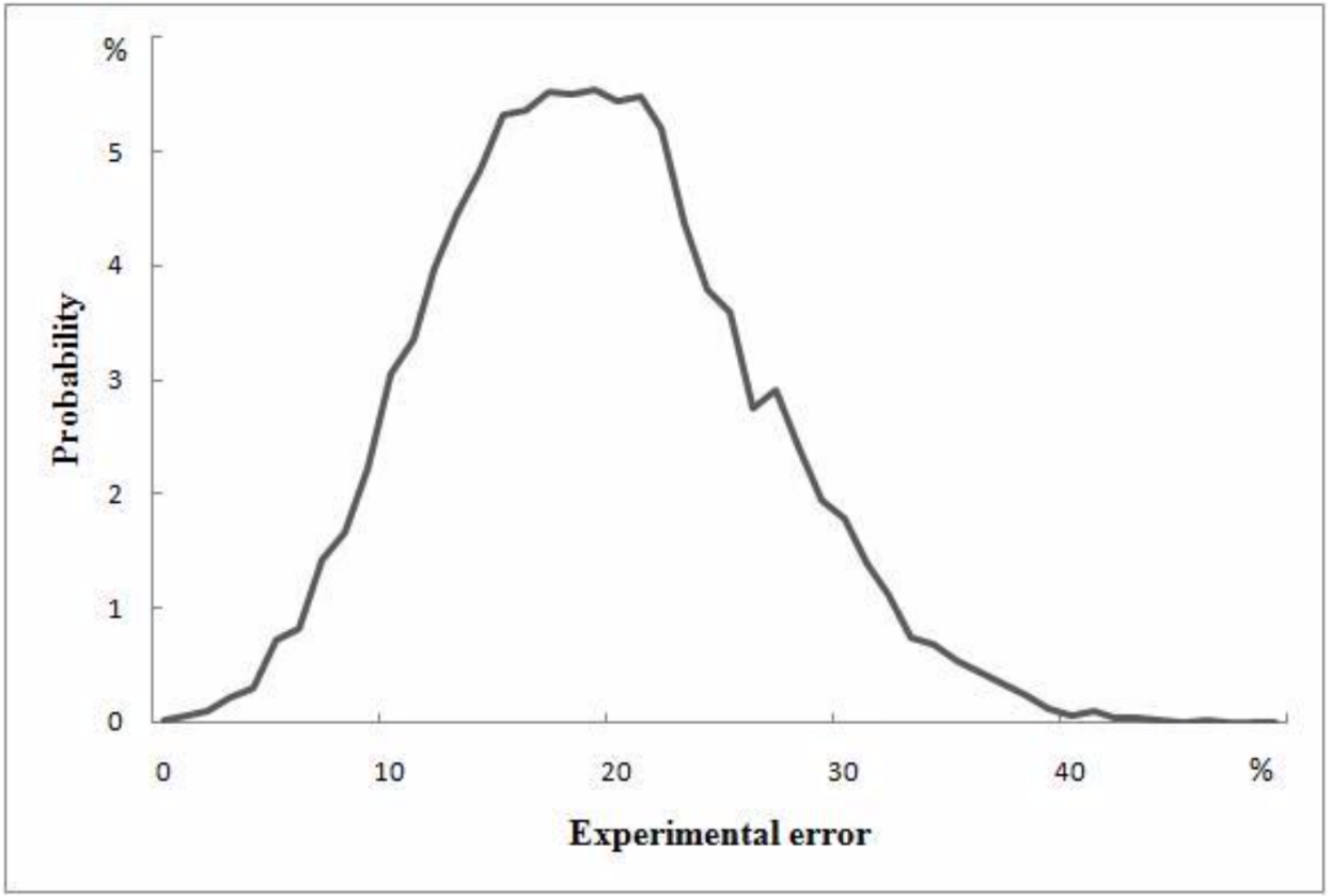
Metrics for $1|r_j|\sum T_j$

Computational experiments

- $n = 4, 5, ..., 10$
- 10000 instances were generated for each value of n
- $p_j \in [1, 100]$
- $d_j \in [-100, 100]$
- $r_j \in [0, 100]$

- F_a denotes an approximate objective value of an instance
- F^* denotes an optimal objective value of an instance
- $\delta = F_a - F^*$ is exeperimental error
- $\Delta = \frac{F_a - F^*}{2\rho(A,B)}$ is the ratio of experimental error and it's upper bound

Metrics for $1|r_j|\sum T_j$



The typical distribution of experimental error.

Metrics for $1|r_j|\sum T_j$

Table: Average experimental error in percentage of the theoretical error

n	$\mathcal{PR}$-case	$\mathcal{PD}$-case	$\mathcal{RD}$-case
4	19%	4,5%	15%
5	19,5%	6,2%	17,2%
6	19,2%	7,3%	18,4%
7	19,6%	8,5%	19,4%
8	19,3%	9,2%	20,7%
9	19,4%	10%	21,7%
10	19%	10,5%	22,5%

Measure of polynomial unsolvability

Problem $1|r_j|L_{max}$ polynomial solvable classes

R). $r_j = const$ (Jackson 1955);

D). $d_j = const$ (Lawler 1973);

P). $p_j = const$ (Simons 1978);

H). $d_j - p_j - A \leq r_j \leq d_j - A, A = const$ (Hoogeveen 1991);

RD). $r_1 \leq \cdots \leq r_n, d_1 \leq \cdots \leq d_n$ (Hoogeveen 1991);

L). $d_1 \leq \cdots \leq d_n, d_1 - p_1 - r_1 \geq \cdots \geq d_n - p_n - r_n$ (Lazarev 2008);

LA). $d_1 \leq \cdots \leq d_n, d_1 - \alpha p_1 - \beta r_1 \geq \cdots \geq d_n - \alpha p_n - \beta r_n,$
$\alpha = const, \beta = const, \alpha \in [0, 1], \beta \in [0, +\infty]$ (Lazarev, Arkhipov 2010).

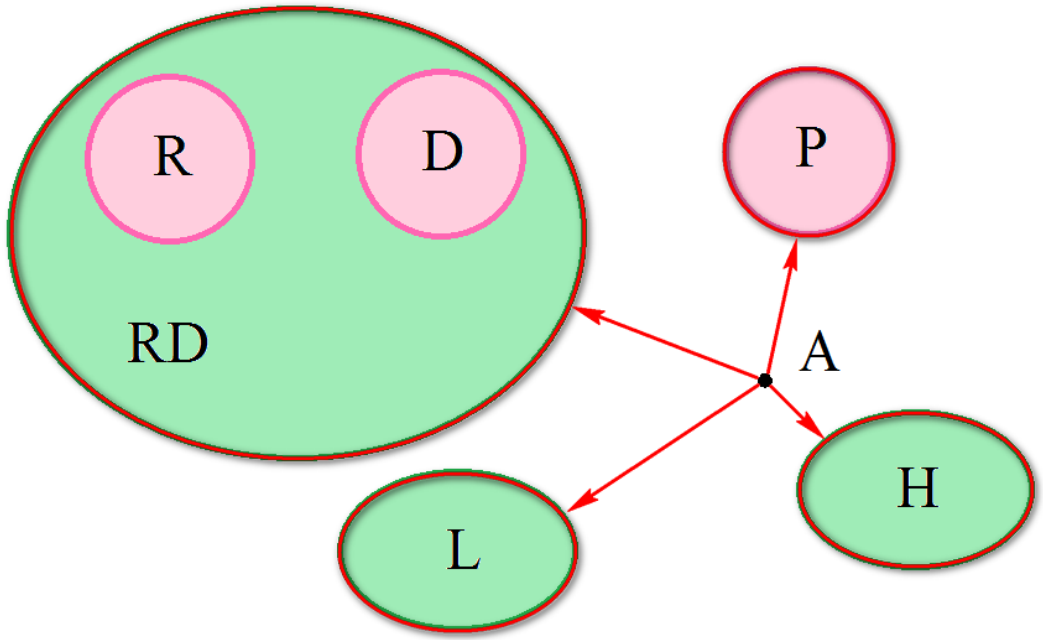
Measure of insolvability

Measure of insolvability of the instance A relative to the area X:

$$\rho^X(A) = \min_{B \in X} \rho(A, B).$$

Complex measure

$E(A) = \min\{\rho^L(A), \rho^H(A), \rho^P(A), \rho^{RD}(A)\}$.



Scalable parameters

Problem A

$r_1, r_2, \ldots, r_n;\ p_1, p_2, \ldots, p_n;\ d_1, d_2, \ldots, d_n.$

$$L^*_{max}(A) = L^A.$$

Problem kA

$kr_1, kr_2, \ldots, kr_n;\ kp_1, kp_2, \ldots, kp_n;\ kd_1, kd_2, \ldots, kd_n.$

$$L^*_{max}(A) = L^{kA} = kL^A.$$

Scalable parameters

Problem A

$r_1, r_2, \ldots, r_n;\ p_1, p_2, \ldots, p_n;\ d_1, d_2, \ldots, d_n.$

$$L^*_{max}(A) = L^A.$$

Problem kA

$kr_1, kr_2, \ldots, kr_n;\ kp_1, kp_2, \ldots, kp_n;\ kd_1, kd_2, \ldots, kd_n.$

$$L^*_{max}(A) = L^{kA} = kL^A.$$

Problems kA & kB

$$\rho(kA, kB) = k\rho(A, B).$$

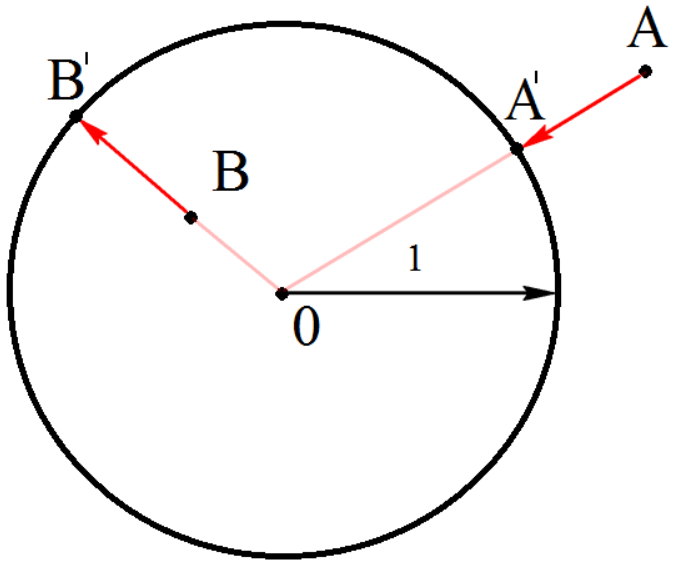
Normalization

Normalization factor

$$NF(A) = \sqrt{\sum_{j=1}^{n} r_j + \sum_{j=1}^{n} p_j + \sum_{j=1}^{n} d_j}$$

Normalized parameters

$$r_j^{A'} = \frac{r_j^A}{NF(A)};\ p_j^{A'} = \frac{p_j^A}{NF(A)};\ d_j^{A'} = \frac{d_j^A}{NF(A)}.$$



Upper bound estimation

Theorem

For each instance A' which belongs to the 3n-dimensional unit sphere following inequalities holds:

$$E(A') < 1.$$

And if $\forall j \in N$ parameters $r_j, p_j, d_j \geq 0$, then:

$$E(A') < \frac{1}{\sqrt{2}}$$

holds.

Proof

$$NF(A') = \sum_{j=1}^{n} r_j^{A'} + \sum_{j=1}^{n} p_j^{A'} + \sum_{j=1}^{n} d_j^{A'} = 1,$$

$\rho^{RD}(A') = \min\limits_{R,D \geq 0} \{R + D\}$, $\forall i, j \in N$, which holds $(d_j^{A'} - d_i^{A'})(r_j^{A'} - r_i^{A'}) < 0$:

$$\left[\begin{array}{l} |r_i^{A'} - r_j^{A'}| \leq R; \\ |d_i^{A'} - d_j^{A'}| \leq D. \end{array} \right.$$

Proof

Hence, $\exists i_1, j_1, i_2, j_2 \in N$:

$$\begin{cases} r_{i_1}^{A'} - r_{j_1}^{A'} \geq E(A'); \\ d_{i_2}^{A'} - d_{j_2}^{A'} \geq E(A'). \end{cases}$$

And, due to $E(A') \leq \rho^P(A')$, $\exists j_3$:

$$p_{j_3} > 0.$$

$$NF(A') = 1 \geq (r_{i_1}^{A'})^2 + (r_{j_1}^{A'})^2 + (d_{i_2}^{A'})^2 + (d_{j_2}^{A'})^2 + (p_{j_3}^{A'})^2.$$

Proof

$$(r_{i_1}^{A'})^2 + (r_{j_1}^{A'})^2 + (d_{i_2}^{A'})^2 + (d_{j_2}^{A'})^2 + (p_{j_3}^{A'})^2 > E(A')^2,$$

and

$$(r_{i_1}^{A'})^2 + (r_{j_1}^{A'})^2 + (d_{i_2}^{A'})^2 + (d_{j_2}^{A'})^2 + (p_{j_3}^{A'})^2 > 2E(A')^2$$

if $r_{i_1}^{A'}, r_{j_1}^{A'}, d_{i_2}^{A'}, d_{j_2}^{A'}, r_{j_3}^{A'}$ are non-negative. Hence,

$$E(A') < 1,$$

and

$$E(A') < \frac{1}{\sqrt{2}},$$

if $r_{i_1}^{A'}, r_{j_1}^{A'}, d_{i_2}^{A'}, d_{j_2}^{A'}, r_{j_3}^{A'}$ are non-negative. QED.

Strengthening theorem

Algorithm Shrage: for every instance A with non-negative parameters of jobs it is possible to construct the solution in $O(n \log n)$ operations with guaranteed accuracy $e^{ED} = \max\limits_{j \in N} p_j$

Strengthen theorem

For each instance A which belongs to the 3n-dimensional unit sphere following inequalities holds that if $\forall j \in N$ parameters $r_j, p_j, d_j \geq 0$, then

$$\min\{e^{ED}, E(A)\} < \frac{1}{\sqrt{3}}.$$

Schrage L. Obtaining Optimal Solutions to Resource Constrained Network Scheduling Problems. Unpublished manuscript 1971.

Metrics approach and Insolvability measure : conclusion

- Metrics allow to construct solutions with guaranteed accuracy in polynomial time.

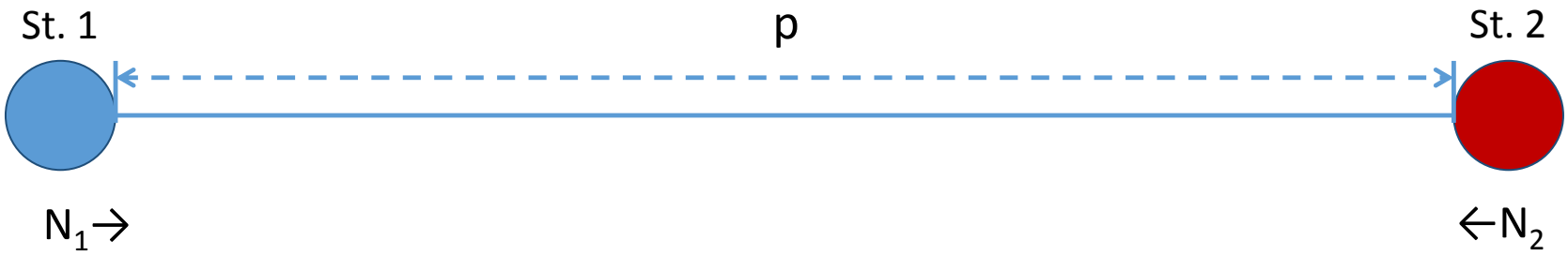
Metrics approach and Insolvability measure : conclusion

- Metrics allow to construct solutions with guaranteed accuracy in polynomial time.
- Measure of polynomial insolvability forms the quantitative property in addition to the NP-hardness qualitative property!
- Theoretically and practically significant result.

Metrics approach and Insolvability measure : conclusion

- Metrics allow to construct solutions with guaranteed accuracy in polynomial time.
- Measure of polynomial insolvability forms the quantitative property in addition to the NP-hardness qualitative property!
- Theoretically and practically significant result.
- Example: Metrics for the railway scheduling problem.

Example: Metrics for the railway scheduling problem



Initial data

- $|N_1| = n$, $|N_2| = n'$, $N = N_1 \cup N_2$, $|N| = n + n'$.
- All trains have equal speed, track traversing time – p.
- Minimal time between the departure of two trains from one station – β.
- The transportation starts at time $t = 0$.

Objective function

- We consider a family of objective functions. In schedule σ, for each train $i \in N$ $S_i(\sigma)$ – it's departure time; $C_i(\sigma)$ – arrival time, $C_i(\sigma) = S_i(\sigma) + p$.
- The approach is demonstrated on the maximum lateness objective function $L_{max}(\sigma)$, $L_{max}(\sigma) = \max\limits_{i \in N} L_i = \max\limits_{i \in N}\{C_i(\sigma) - d_i\}$.

Example: Metrics for the railway scheduling problem

Instances

- Denote the problem as $STR2$ (Single Track Railway Scheduling Problem).
- The $STR2|r_j|L_{max}$ (with release times r_j) problem instance: $2n+2$ parameters, d_j and r_j for each train $j \in N$ are given plus two general parameters β and p.
- We consider the problem instances as points in the $2n$-dimensional space of parameters, denoted as $\Omega = \{r_1, ..., r_n, d_1, ..., d_n\}$.

Metric function

$$\rho(A, B) = \max_{j \in N} |r_j^A - r_j^B| + \max_{j \in N} |d_j^A - d_j^B|$$

satisfies the axioms of metric function. For any instances A, B and schedule π

$$|L_{max}^A(\pi) - L_{max}^B(\pi)| \leq \rho(A, B)$$

Example: Metrics for the railway scheduling problem

Optimal schedules π^A and π^B for instances A and B, respectively

For any instances A and B: $L^A_{max}(\pi^B) - L^A_{max}(\pi^A) \leq 2\rho(A, B)$.

LP approximation model (find solvable instance B for A)

$$\min \; y + x$$

subject to

$$-y \leq d_j^A - d_j^B \leq y, \; \forall j \in N,$$

$$-x \leq r_j^A - r_j^B \leq x, \; \forall j \in N,$$

$$0 \leq r_j^B, \; \forall j \in N,$$

$$\mathcal{A} \cdot R^B + \mathcal{B} \cdot D^B \leq H \; (\textit{solvable instance class constraints})^{\,*}.$$

- r_j^A and d_j^A are given, and x, y, and r_j^B, d_j^B are unknown for all $j \in N$;
- $2n + 2$ variables and $5n + m$ constraints, $n = |N|$, m – the number of inequalities in *.

Example: Metrics for the railway scheduling problem

Polynomially solvable cases

- For $\{PR : r_j = r,\ \forall j \in N\}$, which is the problem $STR2||L_{max}$, we have $\rho(A,B)_{PR} = \max\limits_{j\in N} |r_j^A - r|$.
- For $\{PD : d_j = d,\ \forall j \in N\}$, which is the problem $STR2|r_j|C_{max}$ that has the same complexity as $STR2||L_{max}$, we have $\rho(A,B)_{PD} = \max\limits_{j\in N} |d_j^A - d|$.
- For $\{PDR : r_i \le r_j \Rightarrow d_i \le d_j,\ \forall i,j \in N,\ i < j\}$ when i and j are from the same station, the case with agreeable due dates and arrival dates for each station, we have $\rho(A,B)_{PDR} = \max\limits_{j\in N} |r_j^A - r_j^B| + \max\limits_{j\in N} |d_j^A - d_j^B|$.

Thus, for an arbitrary instance A, the nearest instance

- in class PR is $\{B_{PR} :\ r_j^B = \frac{r_{max}^A + r_{min}^A}{2},\ d_j^B = d_j^A,\ \forall j \in N\}$;
- in class PD is $\{B_{PD} :\ d_j^B = \frac{d_{max}^A + d_{min}^A}{2},\ r_j^B = r_j^A,\ \forall j \in N\}$;
- in class PDR the nearest instance B is constructed by solving the LP with the special form of the inequality (*).

Objective function approximation

Motivation

- Basically, a person (company, organization) often constructs schedules and plans for a day, week, month, etc...
- What is your personal planning goal?

Motivation

- Basically, a person (company, organization) often constructs schedules and plans for a day, week, month, etc...
- What is your personal planning goal?
- To catch all the deadlines or due dates?

Motivation

- Basically, a person (company, organization) often constructs schedules and plans for a day, week, month, etc...
- What is your personal planning goal?
- To catch all the deadlines or due dates?
- Maximize the number of completed tasks or the income?

Motivation

- Basically, a person (company, organization) often constructs schedules and plans for a day, week, month, etc...
- What is your personal planning goal?
- To catch all the deadlines or due dates?
- Maximize the number of completed tasks or the income?
- For many cases the criterion is not clearly formalized.
- However, the schedule structure in general is the same from one period to another.

Motivation

- Basically, a person (company, organization) often constructs schedules and plans for a day, week, month, etc...
- What is your personal planning goal?
- To catch all the deadlines or due dates?
- Maximize the number of completed tasks or the income?
- For many cases the criterion is not clearly formalized.
- However, the schedule structure in general is the same from one period to another.
- How can we use the previous schedules to construct the next one if we have no clear objective function?

Motivation

- Basically, a person (company, organization) often constructs schedules and plans for a day, week, month, etc...
- What is your personal planning goal?
- To catch all the deadlines or due dates?
- Maximize the number of completed tasks or the income?
- For many cases the criterion is not clearly formalized.
- However, the schedule structure in general is the same from one period to another.
- How can we use the previous schedules to construct the next one if we have no clear objective function?
- And if we have the schedules: π_{-N},

Motivation

- Basically, a person (company, organization) often constructs schedules and plans for a day, week, month, etc...
- What is your personal planning goal?
- To catch all the deadlines or due dates?
- Maximize the number of completed tasks or the income?
- For many cases the criterion is not clearly formalized.
- However, the schedule structure in general is the same from one period to another.
- How can we use the previous schedules to construct the next one if we have no clear objective function?
- And if we have the schedules: π_{-N}, $\pi_{-(N-1)}$,

Motivation

- Basically, a person (company, organization) often constructs schedules and plans for a day, week, month, etc...
- What is your personal planning goal?
- To catch all the deadlines or due dates?
- Maximize the number of completed tasks or the income?
- For many cases the criterion is not clearly formalized.
- However, the schedule structure in general is the same from one period to another.
- How can we use the previous schedules to construct the next one if we have no clear objective function?
- And if we have the schedules: π_{-N}, $\pi_{-(N-1)}$, ...,

Motivation

- Basically, a person (company, organization) often constructs schedules and plans for a day, week, month, etc...
- What is your personal planning goal?
- To catch all the deadlines or due dates?
- Maximize the number of completed tasks or the income?
- For many cases the criterion is not clearly formalized.
- However, the schedule structure in general is the same from one period to another.
- How can we use the previous schedules to construct the next one if we have no clear objective function?
- And if we have the schedules: π_{-N}, $\pi_{-(N-1)}$, ..., π_0 and we must construct the next schedule π_1?

The basic idea

- We consider the «inverted» scheduling problem.
- There is a set K of given pairs of instances I_k and schedules π_k^0, $|K| = N$,
- Schedule π_k^0 is the optimal solution for the corresponding instance I_k.
- The problem is to find the form and coefficients of the objective function.
- The objective function is linear to the completion time of the job.

The problem $1||\sum \omega_j C_j$

$1||\sum \omega_j C_j$

- Single machine, n jobs;
- $p_j > 0$ – processing time;
- $j \in N = \{1, 2, \ldots, n\}$;
- no precedence relations between jobs.

Preemptions of a job are not allowed. The machine can process at most one job at any time.

A schedule describes order of processing the jobs: a permutation(sequence) $\pi = (j_1, j_2, ..., j_n)$.

Solvability

$1|r_j|\sum \omega_j C_j$ is NP-hard in the strong sence

Лазарев А.А., Гафаров Е.Р. Теория расписаний. Задачи и алгоритмы // Москва, МГУ, 2011, 222 С.

$1||\sum \omega_j C_j$ is solvable – generalized Smith theorem

There exists an optimal schedule $\pi^* = (j_1, ..., j_n)$, such that

$$\frac{\omega_{j_1}}{p_{j_1}} \geq \frac{\omega_{j_2}}{p_{j_2}} \geq ... \geq \frac{\omega_{j_n}}{p_{j_n}}.$$

Smith W.E. Various optimizers for single-stage production // Naval Res. Logist. Quart. 1956. No. 3. P. 59–66.

Approximation problem

- We consider the problem $1||\sum \omega_j C_j$.
- N given pairs of instances $I_k = \{p_1, ..., p_n\}$ and schedules π_k^0.
- Schedule π_k^0 is the optimal solution for the corresponding instance I_k.
- The problem is to find the coefficients ω_j of the objective function.

The property of an optimal schedule

$$\sum_{j=1}^{n} C_j^k(\pi)\omega_j \geq \sum_{j=1}^{n} C_j^k(\pi_k^0)\omega_j, \ \forall \pi \neq \pi_k^0, \ k \in \{1, ..., N\}$$

- In general case ω_j are defined by the set of $N(n! - 1)$ inequalities.
- Is it possible to allocate the subset of M (polynomial number) of independent inequalities, which forms the equal system?.

Approximation problem

Basic system of inequalities for $1|r_j|\sum \omega_j C_j$

$$\frac{\omega_{j_1^k}}{p_{j_1^k}} \geq \frac{\omega_{j_2^k}}{p_{j_2^k}} \geq ... \geq \frac{\omega_{j_n^k}}{p_{j_n^k}},\ k \in \{1, ..., N\}.$$

Transformations

- Consider arbitrary pair of jobs $\forall i, j \in \{1, ..., n\},\ i \neq j$.
- Separate the set K into two subsets $K_{i,j}$ and $K_{j,i}$, depending on the positions of i and j in π_k^0.

$$K_{i,j} = \{k \in K :\ \pi_k^0 = (..., i, ..., j, ...)\};$$

$$K_{j,i} = \{k \in K :\ \pi_k^0 = (..., j, ..., i, ...)\}.$$

- From the basic system:
$\frac{\omega_{jk}}{\omega_{ik}} \leq \frac{p_{jk}}{p_{ik}},\ k \in K_{i,j}$ and $\frac{\omega_{jk}}{\omega_{ik}} \geq \frac{p_{jk}}{p_{ik}},\ k \in K_{j,i}$.

Approximation problem

Effective system of inequalities

$$\forall i,j \in \{1,...,n\},\ i \neq j;$$

$$K_{i,j} = \{k \in K :\ \pi^0_k = (...,i,...,j,...)\},\ K_{j,i} = \{k \in K :\ \pi^0_k = (...,j,...,i,...)\};$$

$$X(i,j) = \max_{k \in K_{j,i}} (\frac{p^k_j}{p^k_i}),\ Y(i,j) = \min_{k \in K_{i,j}} (\frac{p^k_j}{p^k_i});$$

$$X(i,j) \leq \frac{\omega_j}{\omega_i} \leq Y(i,j).$$

Lemmas

- The basic and effective systems of inequalities are equal.
- The set of solutions of both systems is the convex polyhedral cone.

Approximation problem

The strengthening of inequalities

Multiplication property of inequaltites forms the strengthening procedure:

$$X(i,j) := \max\{X(i,j), \max_{l=\{1,\dots,n\},\ l\neq i,\ l\neq j}\{X(i,l)X(l,j)\}\},\ i,j \in \{1,\dots,n\},\ i \neq j.$$

It can be repeated till some final $\tilde{X}$ and $\tilde{Y}$.

If $\omega = \{\omega_1, \dots, \omega_n\}$ is the solution of approximation problem $1||\sum \omega_j C_j$, then $\gamma\omega = \{\gamma\omega_1, \dots, \gamma\omega_n\}$ is also the soltuion of this problem, i.e. it can be scaled. Therefore, we can always assume that $\omega_l = 1$ for some arbitrary one index l.

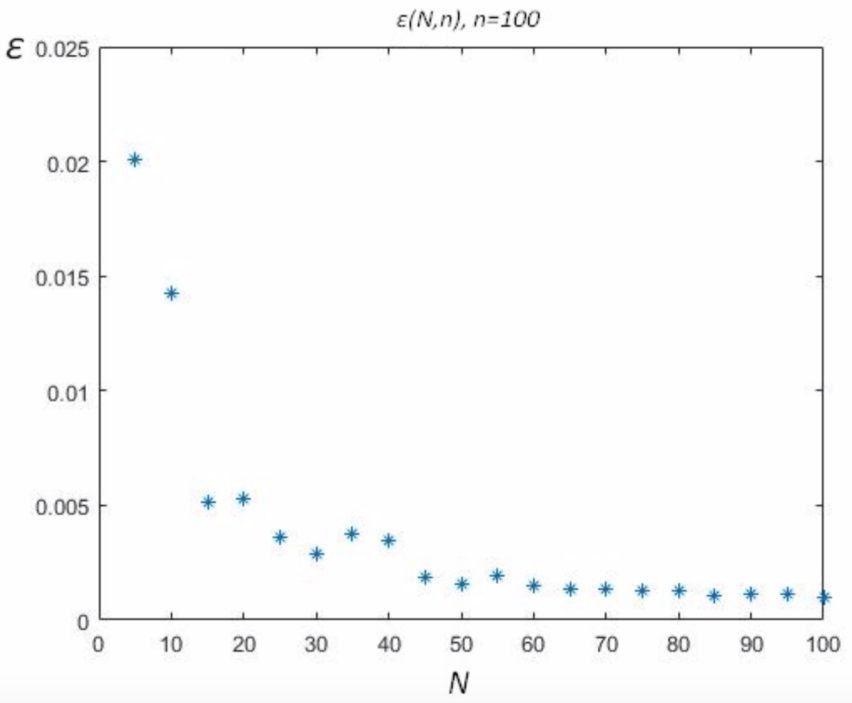
Theorem

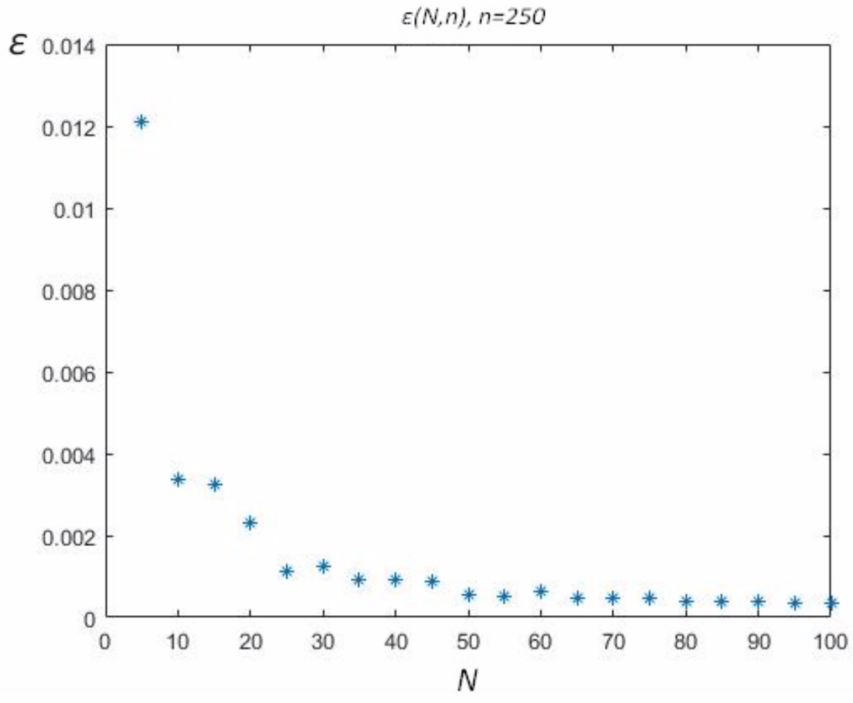
Vector $\omega = \{\omega_1, \dots, \omega_n\}$ is the solution of the effective system, if

$$\omega_j = \begin{cases} 1, & \text{if } j = l; \\ (\tilde{X}(l,j) + \tilde{Y}(l,j)/2, & j \neq l. \end{cases} \quad (6)$$

Computations

- Random sets with N instances $I_k = \{p_1^k, ..., p_n^k\}$, n jobs and ω_j^0 were generated (distributed in $[0, 1]$).
- All the valued were rationed $\omega_j := \frac{\omega_j}{||\omega||}$, $\omega_j^0 := \frac{\omega_j^0}{||\omega^0||}$.
- $\epsilon(N, n) = \frac{1}{n} \sum_{j=1}^{n} \frac{|\omega_j - \omega_j^0|}{\omega_j^0}$, the error decreases (converges to 0) with growing N!





Dual complexity reduction

Duality for non-decreasing penalty functions

Initial problem

$$\mu^* = \min_{\pi \in \Pi(N)} \max_{k=\overline{1,n}} \varphi_{j_k}(C_{j_k}(\pi)), \tag{7}$$

Non-decreasing functions $\varphi_j(C_j(\pi))$

Duality for non-decreasing penalty functions

Initial problem

$$\mu^* = \min_{\pi \in \Pi(N)} \max_{k=\overline{1,n}} \varphi_{j_k}(C_{j_k}(\pi)), \tag{7}$$

Non-decreasing functions $\varphi_j(C_j(\pi))$

Dual problem

$$\nu^* = \max_{k=\overline{1,n}} \min_{\pi \in \Pi(N)} \varphi_{j_k}(C_{j_k}(\pi)). \tag{8}$$

$r_j = 0, \forall \quad j \in N$

Conway R.W., Maxwell W.L., Miller L.W. Theory of Scheduling // Addison-Wesley, Reading, MA. 1967.

Duality for non-decreasing penalty functions

$$\nu^* = \max_{k=\overline{1,n}} \min_{\pi \in \Pi(N)} \varphi_{j_k}(C_{j_k}(\pi))$$

Duality for non-decreasing penalty functions

$$\nu^* = \max_{k=\overline{1,n}} \min_{\pi \in \Pi(N)} \varphi_{j_k}(C_{j_k}(\pi))$$

$$\nu_k = \min_{\pi \in \Pi(N)} \varphi_{j_k}(C_{j_k}(\pi)), k = 1, 2, \ldots, n. \quad (9)$$

Duality for non-decreasing penalty functions

$$\nu^* = \max_{k=\overline{1,n}} \min_{\pi \in \Pi(N)} \varphi_{j_k}(C_{j_k}(\pi))$$

$$\nu_k = \min_{\pi \in \Pi(N)} \varphi_{j_k}(C_{j_k}(\pi)), k = 1, 2, \ldots, n. \tag{9}$$

$$\nu^* = \max_{k=\overline{1,n}} \nu_k. \tag{10}$$

Duality for non-decreasing penalty functions

Lemma

$\varphi_j(t), j = 1, 2, \ldots, n$, *any not decreasing functions* $1 \mid r_j \mid \varphi_{\max}$,
$\forall \quad k = 1, 2, \ldots, n, \qquad \nu_n \geq \nu_k, \qquad \nu^* = \nu_n.$

Duality for non-decreasing penalty functions

Lemma

$\varphi_j(t), j = 1, 2, \ldots, n$, *any not decreasing functions* $1 \mid r_j \mid \varphi_{\max}$,
$\forall \quad k = 1, 2, \ldots, n, \qquad \nu_n \geq \nu_k, \qquad \nu^* = \nu_n.$

Algorithm

$\pi^r = (i_1, i_2, \ldots, i_n), \qquad r_{i_1} \leq r_{i_2} \leq \cdots \leq r_{i_n};$
$\pi_k = (\pi^r \setminus i_k, i_k), k = 1, 2, \ldots, n, \qquad \varphi_{i_k}(C_{i_k}(\pi_k));$
$\nu^* = \max\limits_{k=\overline{1,n}} \varphi_{i_k}(C_{i_k}(\pi_k)).$

Duality for non-decreasing penalty functions

Lemma

$\varphi_j(t), j = 1, 2, \ldots, n$, *any not decreasing functions* $1 \mid r_j \mid \varphi_{\max}$,
$\forall \quad k = 1, 2, \ldots, n, \qquad \nu_n \geq \nu_k, \qquad \nu^* = \nu_n.$

Algorithm

$\pi^r = (i_1, i_2, \ldots, i_n), \qquad r_{i_1} \leq r_{i_2} \leq \cdots \leq r_{i_n};$
$\pi_k = (\pi^r \setminus i_k, i_k), k = 1, 2, \ldots, n, \qquad \varphi_{i_k}(C_{i_k}(\pi_k));$
$\nu^* = \max\limits_{k=\overline{1,n}} \varphi_{i_k}(C_{i_k}(\pi_k)).$

$O(n^2)$

Duality for non-decreasing penalty functions

Initial problem

$$\mu^* = \min_{\pi \in \Pi(N)} \max_{k=\overline{1,n}} \varphi_{j_k}(C_{j_k}(\pi)), \tag{11}$$

Non-decreasing function $\varphi_j(C_j(\pi))$

Duality for non-decreasing penalty functions

Initial problem

$$\mu^* = \min_{\pi \in \Pi(N)} \max_{k=\overline{1,n}} \varphi_{j_k}(C_{j_k}(\pi)), \tag{11}$$

Non-decreasing function $\varphi_j(C_j(\pi))$

Dual problem

$$\nu^* = \max_{k=\overline{1,n}} \min_{\pi \in \Pi(N)} \varphi_{j_k}(C_{j_k}(\pi)). \tag{12}$$

Duality for non-decreasing penalty functions

Initial problem

$$\mu^* = \min_{\pi \in \Pi(N)} \max_{k=\overline{1,n}} \varphi_{j_k}(C_{j_k}(\pi)), \tag{11}$$

Non-decreasing function $\varphi_j(C_j(\pi))$

Dual problem

$$\nu^* = \max_{k=\overline{1,n}} \min_{\pi \in \Pi(N)} \varphi_{j_k}(C_{j_k}(\pi)). \tag{12}$$

Theorem

$\varphi_j(t), j = 1, 2, \ldots, n$, *any non-decreasing functions* $1 \mid r_j \mid \varphi_{\max}$,
$\forall \quad k = 1, 2, \ldots, n,$ $\mu^* \geq \nu^*$.

Duality for non-decreasing penalty functions

Initial problem

$$\mu^* = \min_{\pi \in \Pi(N)} \max_{k=\overline{1,n}} \varphi_{j_k}(C_{j_k}(\pi)), \tag{11}$$

Non-decreasing function $\varphi_j(C_j(\pi))$

Dual problem

$$\nu^* = \max_{k=\overline{1,n}} \min_{\pi \in \Pi(N)} \varphi_{j_k}(C_{j_k}(\pi)). \tag{12}$$

Theorem

$\varphi_j(t), j = 1, 2, \ldots, n$, *any non-decreasing functions* $1 \mid r_j \mid \varphi_{\max}$,
$\forall \quad k = 1, 2, \ldots, n,$ $\mu^* \geq \nu^*$.

Branch and bound

Duality for non-decreasing penalty functions

Initial problem is NP-hard in the strong sense!

Preceding, Dual problem

G : single machine $O(n^2)$

G : many machines NP-hard in the ordinary sense

Non-decreasing penalty functions $\varphi_j(C_j(\pi))$

Graphical approach

Graphical approach

Partition problem

Consider a sorted set of n positive integer numbers $B = \{b_1, b_2, \ldots, b_n\}$, $b_1 \geq b_2 \geq \cdots \geq b_n$. Divide the set B into two subsets B_1, B_2, so that

$$|\sum_{i \in B_1} b_i - \sum_{i \in B_2} b_i| \to \min$$

Graphical approach

Partition problem

Consider a sorted set of n positive integer numbers $B = \{b_1, b_2, \ldots, b_n\}$, $b_1 \geq b_2 \geq \cdots \geq b_n$. Divide the set B into two subsets B_1, B_2, so that

$$|\sum_{i \in B_1} b_i - \sum_{i \in B_2} b_i| \to \min$$

One-dimensional Knapsack problem

This problem can be viewed as an integer programming problem:

$$\begin{cases} f(x) = \sum_{i=1}^{n} c_i x_i \to \max \\ \sum_{i=1}^{n} w_i x_i \leq W \\ x_i \in \{0, 1\},\ i = 1, \ldots, n \end{cases}$$

Graphical approach

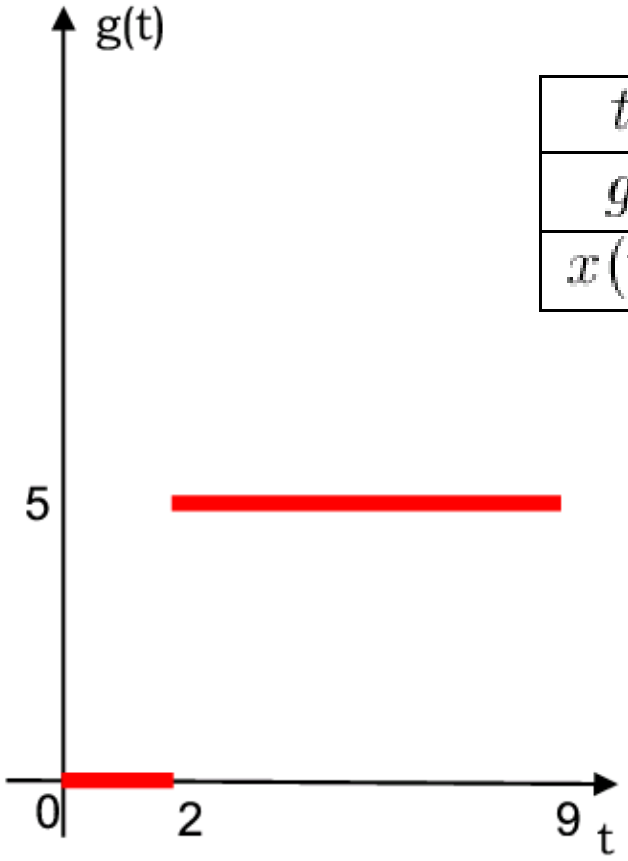
If $c_i = a_i = b_i, i = 1, \ldots, n$ and $W = \frac{1}{2} \sum_{i=1}^{n} b_i$, then Partition problem and One-dimensional Knapsack problem are equivalent.

$$f(x) = 5x_1 + 7x_2 + 6x_3 + 3x_4 \longrightarrow \max$$

$$2x_1 + 3x_2 + 5x_3 + 7x_4 \leq 9$$

$$x_i \in \{0, 1\},\ i = 1, \ldots, 4.$$

Step 1



t	0	2
g	0	5
$x(t)$	$(0,,,)$	$(1,,,)$

t	$g_1(t)$	$x(t)$
0	0	$(0,,,)$
1	0	$(0,,,)$
2	5	$(1,,,)$
3	5	$(1,,,)$
4	5	$(1,,,)$
5	5	$(1,,,)$
6	5	$(1,,,)$
7	5	$(1,,,)$
8	5	$(1,,,)$
9	5	$(1,,,)$

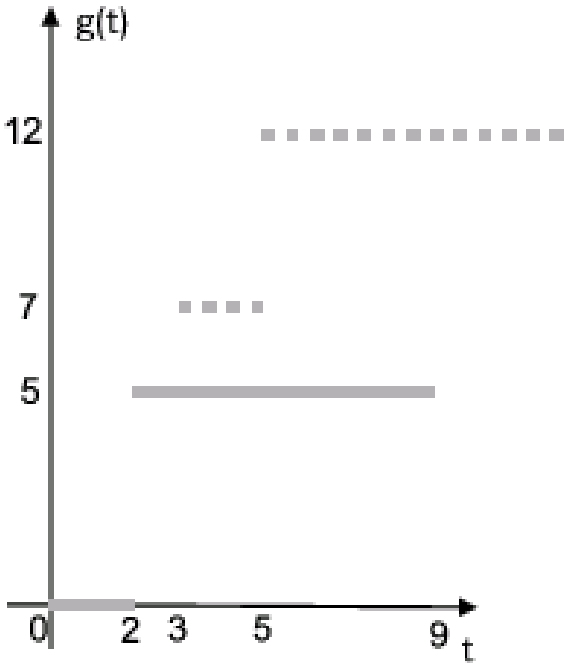
$$f(x) = 5x_1 + 7x_2 + 6x_3 + 3x_4 \longrightarrow \max$$
$$2x_1 + 3x_2 + 5x_3 + 7x_4 \leq 9$$
$$x_i \in \{0, 1\},\ i = 1, \ldots, 4.$$

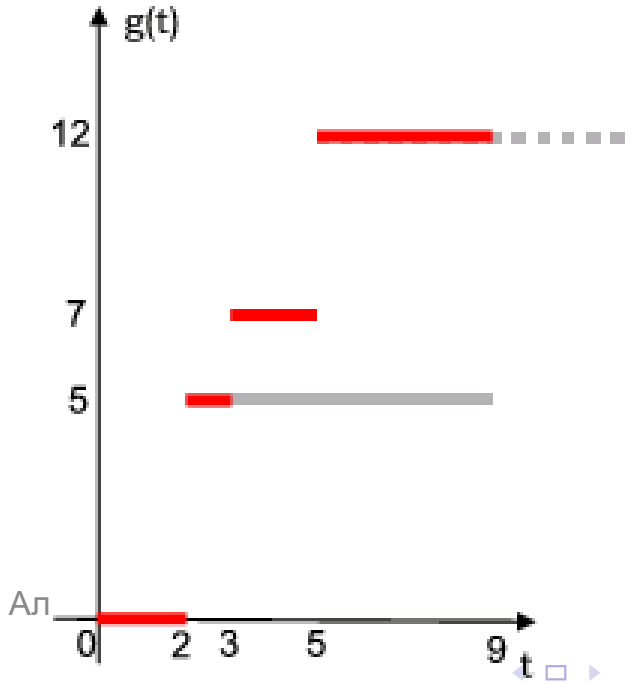
Step 2

t	0	2
g	0	5
$x(t)$	$(0,,,)$	$(1,,,)$

t	0	2	3	5
g	0	5	7	12
$x(t)$	$(0,0,,)$	$(1,0,,)$	$(0,1,,)$	$(1,1,,)$

Let's consider 4 points:
0, 2, 0+3, 2+3





$$f(x) = 5x_1 + 7x_2 + 6x_3 + 3x_4 \longrightarrow \max$$

$$2x_1 + 3x_2 + 5x_3 + 7x_4 \leq 9$$

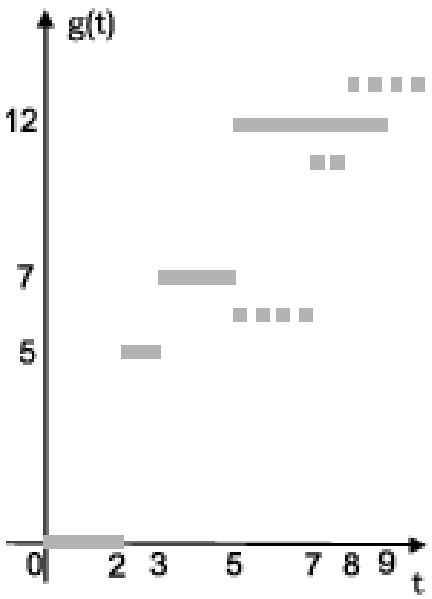
$$x_i \in \{0, 1\},\ i = 1, \ldots, 4.$$

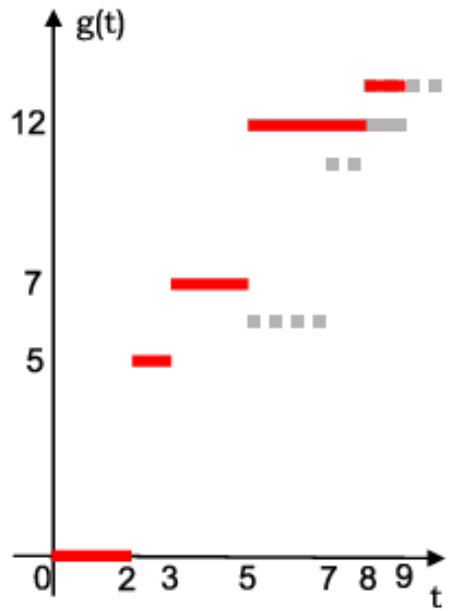
Step 3

t	0	2	3	5
g	0	5	7	12
$x(t)$	$(0, 0, ,)$	$(1, 0, ,)$	$(0, 1, ,)$	$(1, 1, ,)$

Let's consider 7 points:
0, 2, 3, 5, 0+5, 2+5, 3+5.
Point 5+5 > 9 is not considered

t	0	2	3	5	8
g	0	5	7	12	13
$x(t)$	$(0, 0, 0,)$	$(1, 0, 0,)$	$(0, 1, 0,)$	$(1, 1, 0,)$	$(0,1,1,)$

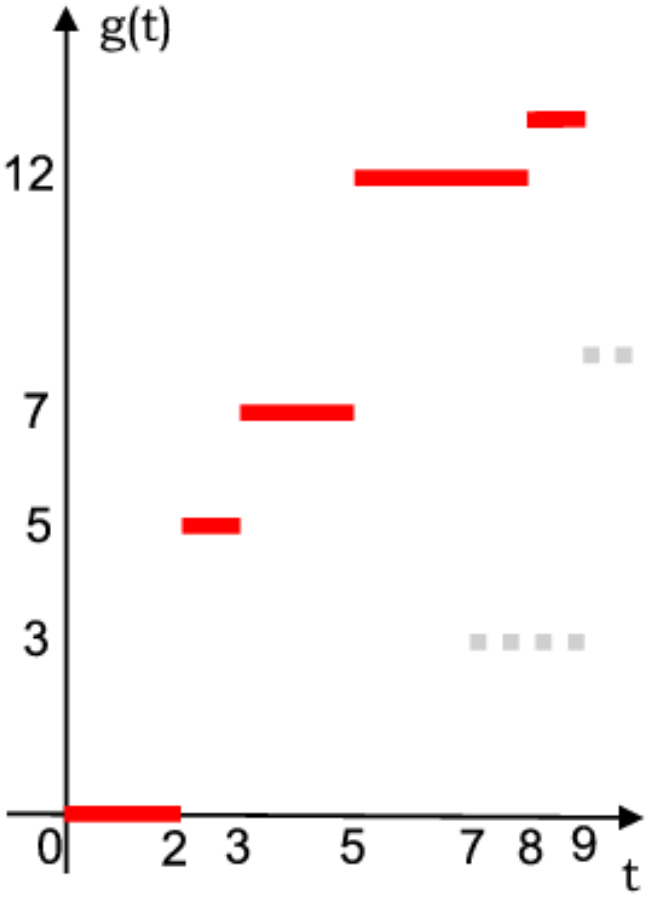




$$f(x) = 5x_1 + 7x_2 + 6x_3 + 3x_4 \longrightarrow \max$$
$$2x_1 + 3x_2 + 5x_3 + 7x_4 \leq 9$$
$$x_i \in \{0, 1\},\ i = 1, \ldots, 4.$$

Step 4

t	$g_1(t)$	$x(t)$	$g_2(t)$	$x(t)$	$g_3(t)$	$x(t)$	$g_4(t)$	$x(t)$
0	0	(0,,,)	0	(0,0,,)	0	(0,0,0,)	0	(0,0,0,0)
1	0	(0,,,)	0	(0,0,,)	0	(0,0,0,)	0	(0,0,0,0)
2	5	(1,,,)	5	(1,0,,)	5	(1,0,0,)	5	(1,0,0,0)
3	5	(1,,,)	7	(0,1,,)	7	(0,1,0,)	7	(0,1,0,0)
4	5	(1,,,)	7	(0,1,,)	7	(0,1,0,)	7	(0,1,0,0)
5	5	(1,,,)	12	(1,1,,)	12	(1,1,0,)	12	(1,1,0,0)
6	5	(1,,,)	12	(1,1,,)	12	(1,1,0,)	12	(1,1,0,0)
7	5	(1,,,)	12	(1,1,,)	12	(1,1,0,)	12	(1,1,0,0)
8	5	(1,,,)	12	(1,1,,)	13	(0,1,1,)	13	(0,1,1,0)
9	5	(1,,,)	12	(1,1,,)	13	(0,1,1,)	13	(0,1,1,0)

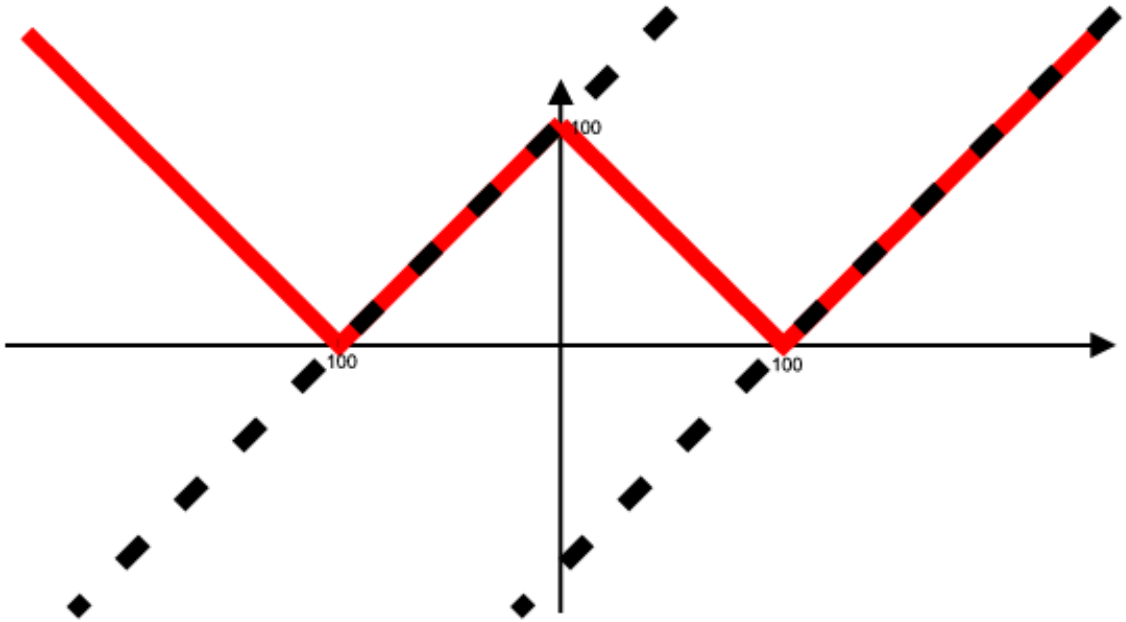


t	0	2	3	5	8
g	0	5	7	12	13
$x(t)$	(0, 0, 0, 0)	(1, 0, 0, 0)	(0, 1, 0, 0)	(1, 1, 0, 0)	(0,1,1,0)

$$\frac{c_1}{a_1} \geq \frac{c_2}{a_2} \geq \ldots \geq \frac{c_n}{a_n}.$$

$B = \{100, 70, 50, 20\}$

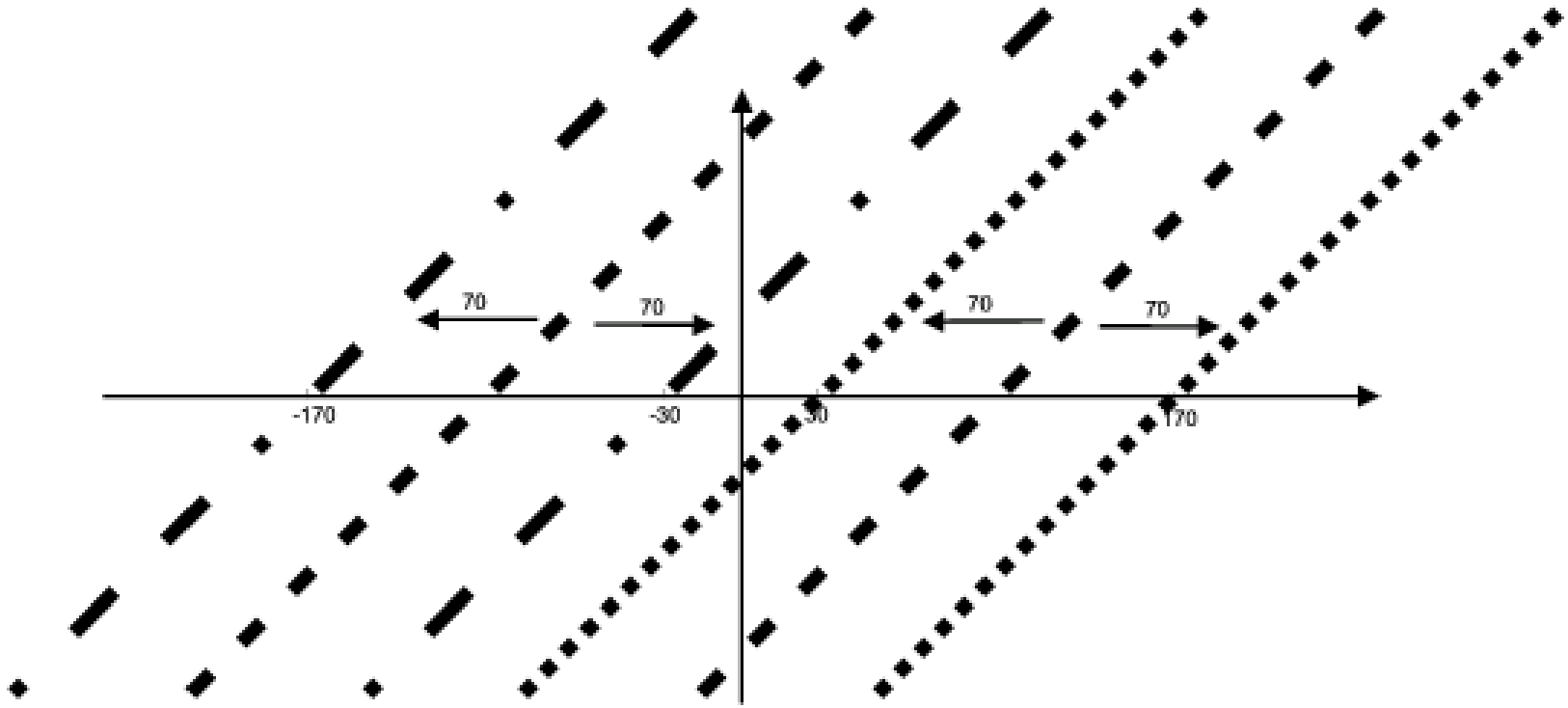
Step 1



-100	100
$(100;)$	$(;100)$

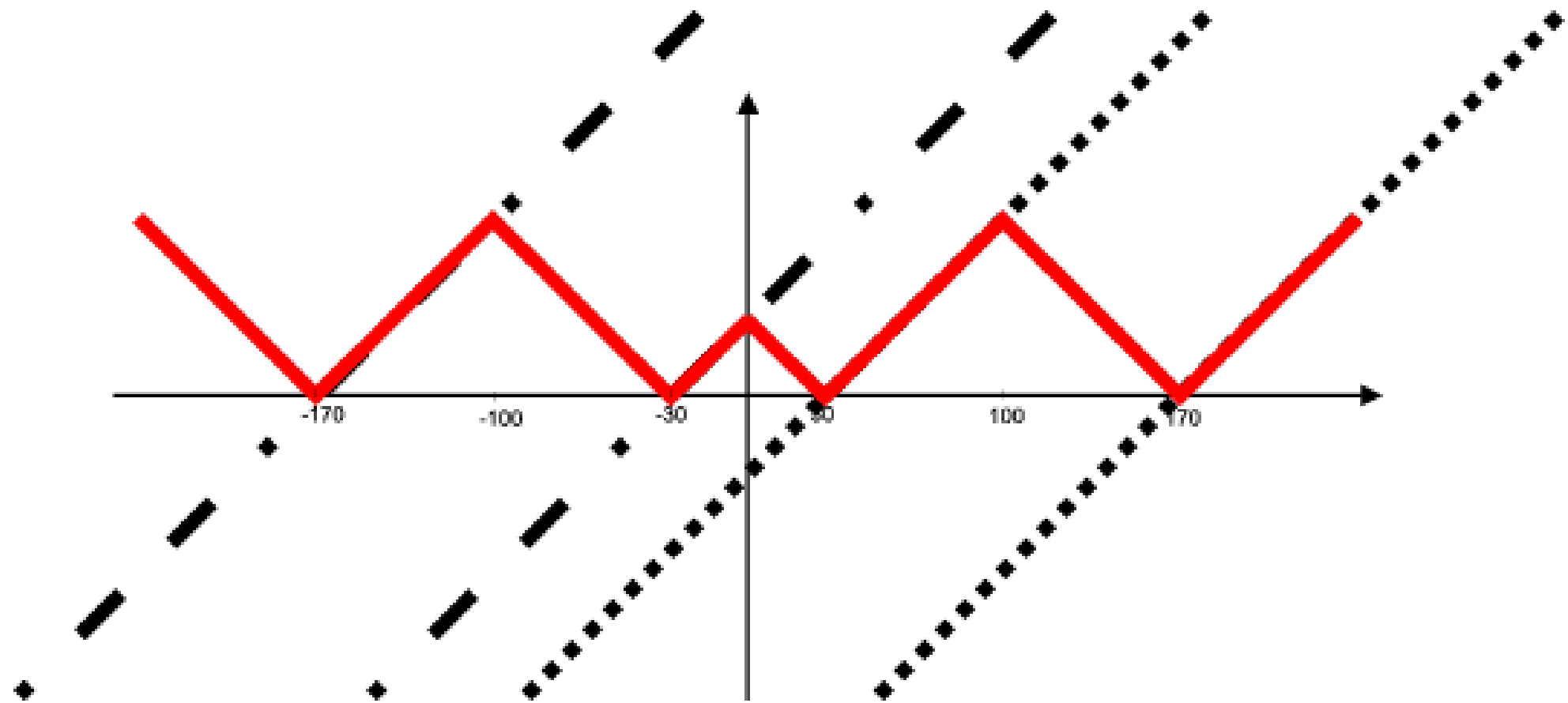
$B = \{100, 70, 50, 20\}$

Step 2



$B = \{100, 70, 50, 20\}$

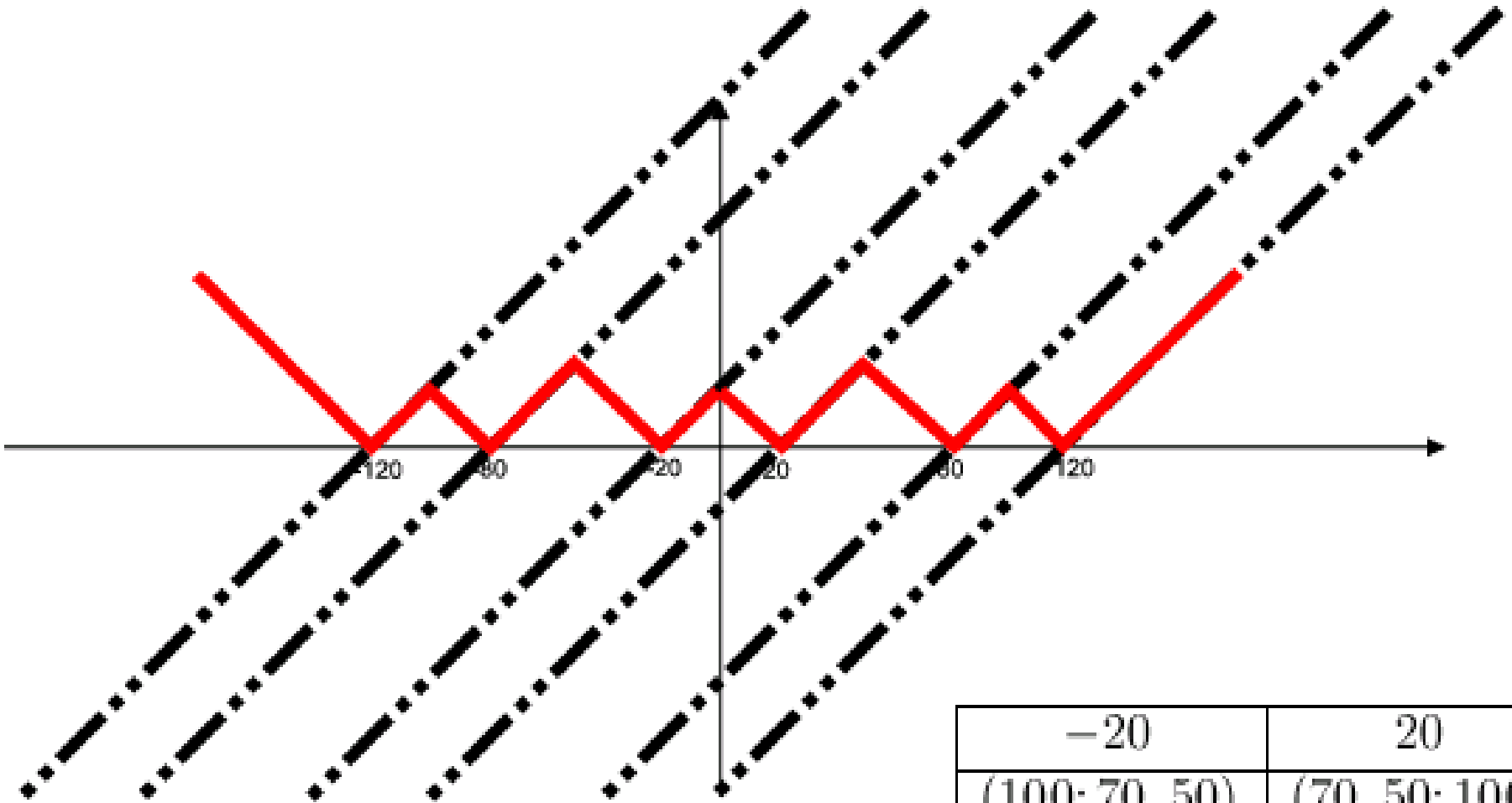
Step 2



-30	30
$(100; 70)$	$(70; 100)$

$B = \{100, 70, 50, 20\}$

Step 3



-20	20
$(100; 70, 50)$	$(70, 50; 100)$

$$40 \geq b_1 \geq b_2 \geq \ldots \geq b_n. \quad n = 4, 5, \ldots, 10.$$

1	2	3	4	5	6	7	8	9	10
4	123 410	9	307	328	20	443	640	2	63 684
5	1 086 008	16	444	512	40	564	1000	2	337 077
6	8 145 060	29	542	738	60	687	1440	4	1 140 166
7	53 524 680	48	633	1004	140	811	1960	11	2 799 418
8	314 457 495	76	725	1312	212	933	2560	23	5 348 746
9	1 677 106 640	115	814	1660	376	1053	3240	83	8 488 253
10	8 217 822 536	168	905	2050	500	1172	4000	416	11 426 171

1st column: dimensionality of the problem (*n*);
2ndcolumn: total number of solved instances for given *n* ($C_n^{b_{max}+n-1}$, where $b_{max} = 40$);
3rd column: average value of computational complexity of graphic algorithm;
4th column: average value of computational complexity of Balsub algorithm;
5th column: average value of computational complexity of dynamic programming algorithm;
6th column: maximal value of computational complexity of graphic algorithm;
7th column: maximal value of computational complexity of Balsub algorithm;
8th column: maximal value of computational complexity of dynamic programming algorithm;
9th column: amount of instances for which complexity of Balsub algorithm is less than the complexity of graphic algorithm;
10th column: amount of instances for which complexity of dynamic programming algorithm is less than complexity of Balsub algorithm.

Keller H., Pferschy U., Pisinger D. Knapsack problems. Springer, 2004.

Project investment problem

n potential projects

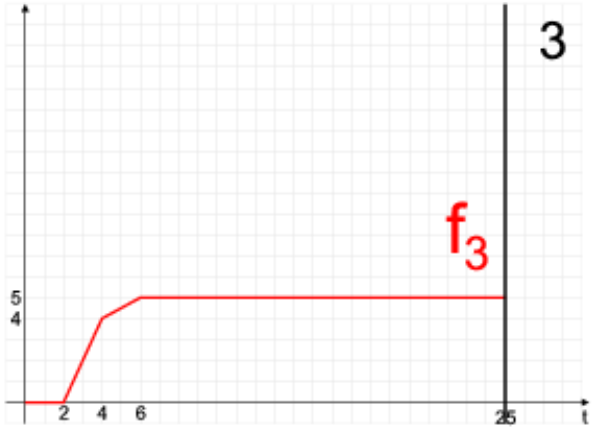
A – an investment budget (for all A from interval [A',A''])

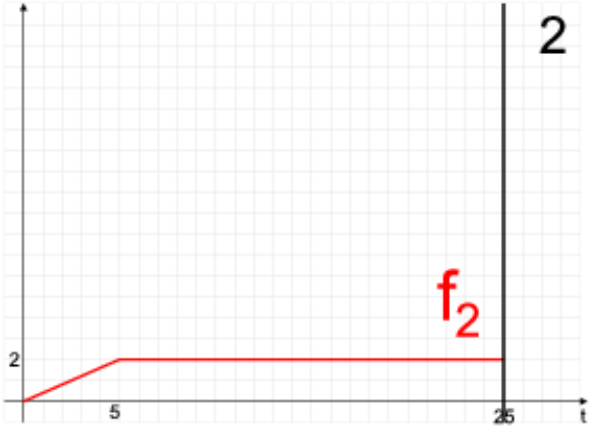
$f_j(t)$ -- a profit function of project j

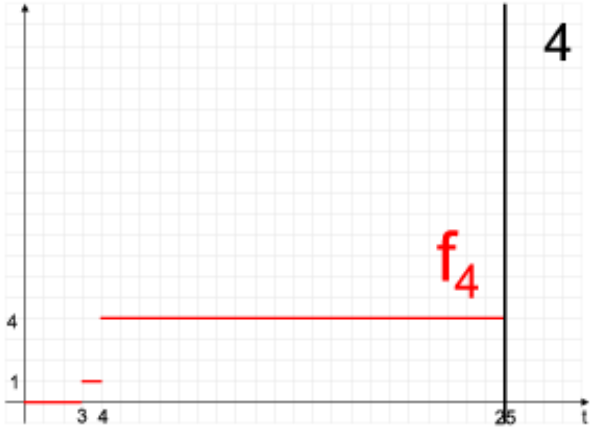
The goal is to define an amount t_j in [0,A] (integer) for each project to maximize the total profit.

$\sum t_j <= A$

Project investment problem







Graphical algorithm for the project investment problem

Dynamic programming algorithm $O(nA^2)$. Or $O(\sum k_j A)$

$$F_j(T) = \max_{t=0,1,\ldots,T} \{f_j(t) + F_{j-1}(T-t)\},\ T = A, A-1, \ldots, 1,$$

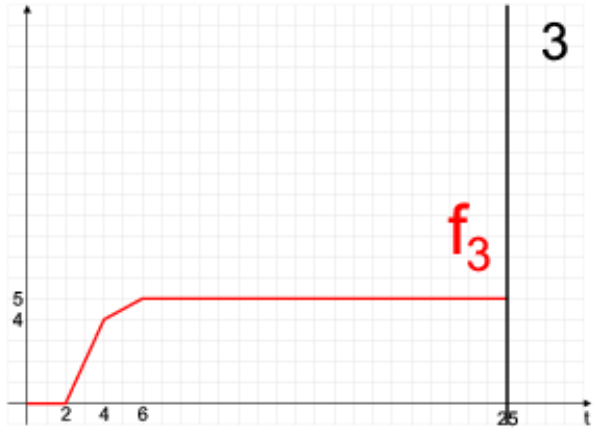
In Graphical Algorithm functions $f_j(t)$ and Bellman's functions (value function) $F_j(t)$ are saved in tabular form:

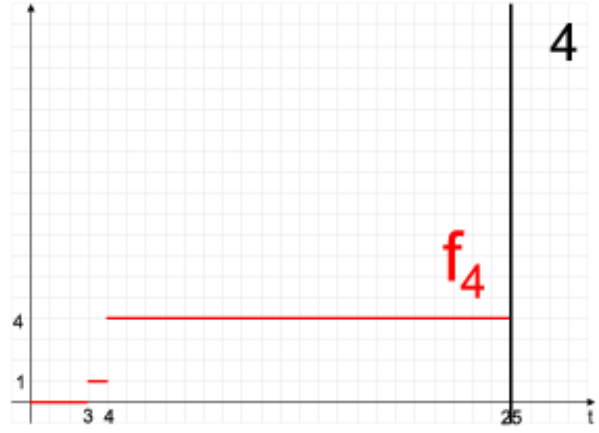
K	1	2	...	k_j
interval K	$[t_j^1, t_j^2)$	$[t_j^2, t_j^3)$	...	$[t_j^{k_j}, A)$
b_j^K	b_j^1	b_j^2	...	$b_j^{k_j}$
u_j^K	u_j^1	u_j^2	...	$u_j^{k_j}$

Running time for the 1st version of Graphical Algorithm $O(nk_{max}A \log(k_{max}A))$

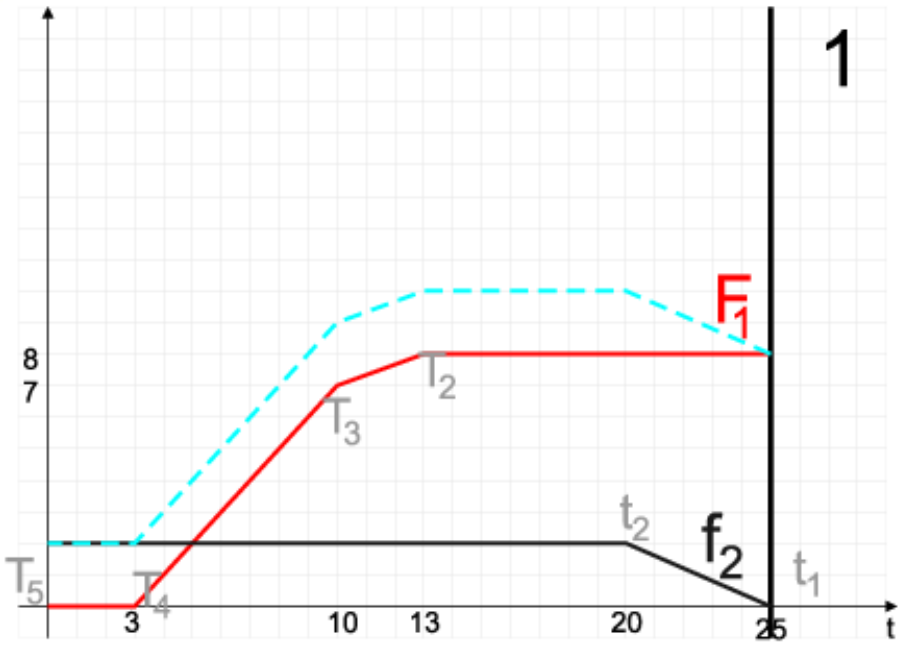
Running time for the 2nd version of Graphical Algorithm $O(\sum k_j A)$

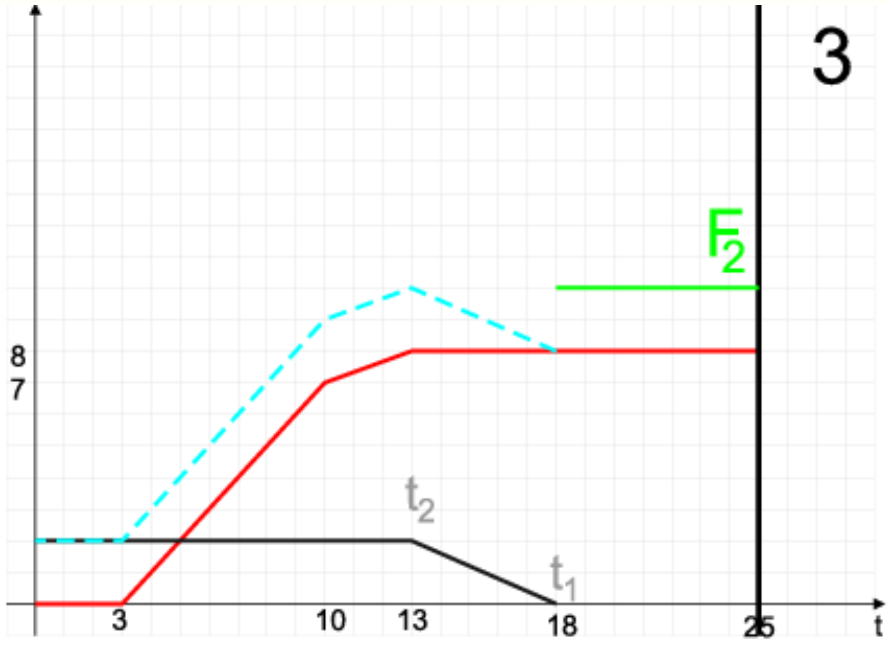
Graphical algorithm for Investments problem

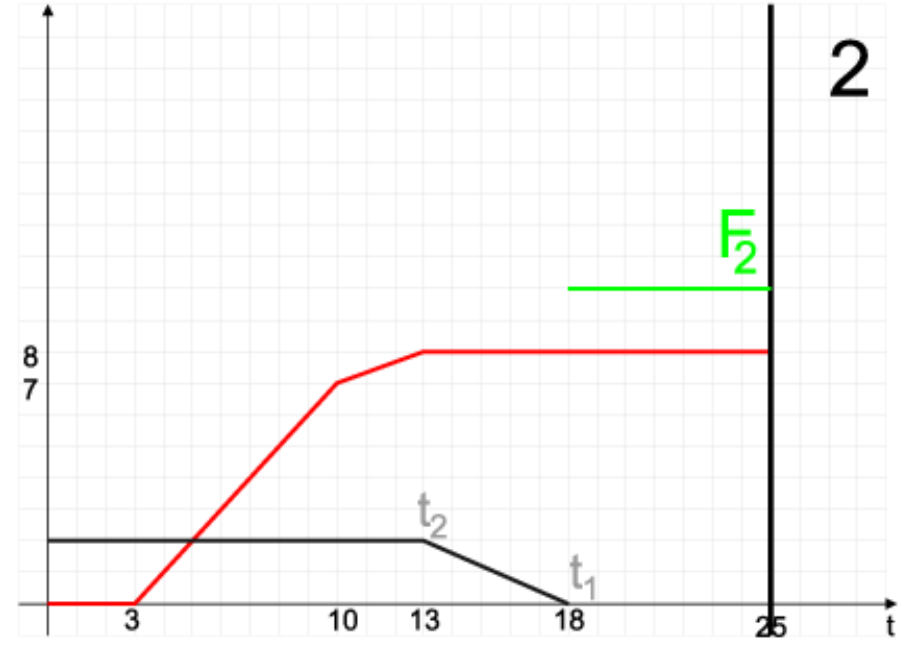


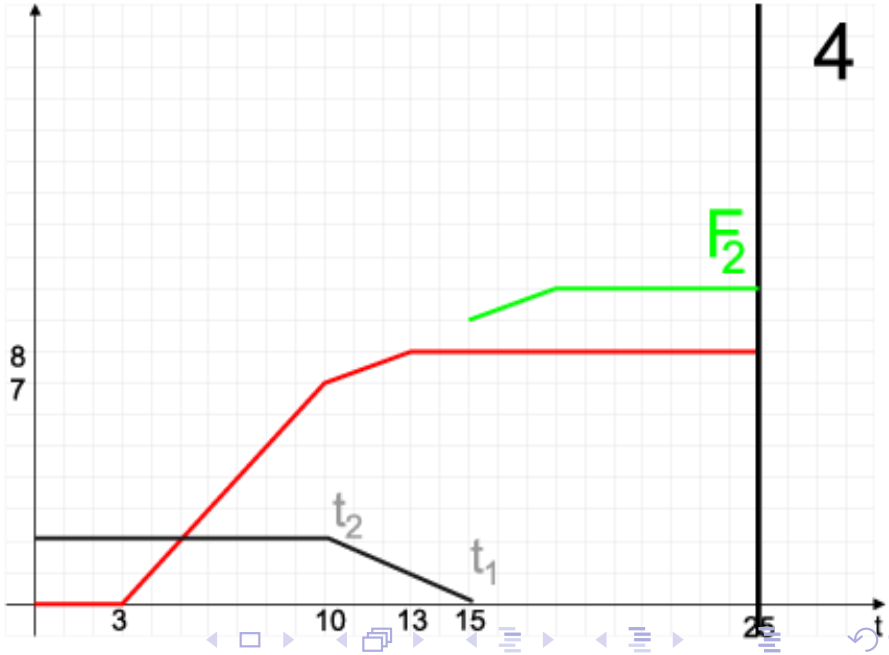


K	1	2	3	4
interval K	$[0, 3)$	$[3, 10)$	$[10, 13)$	$[13, 25]$
b_1^K	0	0	7	8
u_1^K	0	1	$\frac{1}{3}$	0


1
F1
8
7
T2
T3
T4
T5
t2
f2
t1
3
10
13
20
25
t

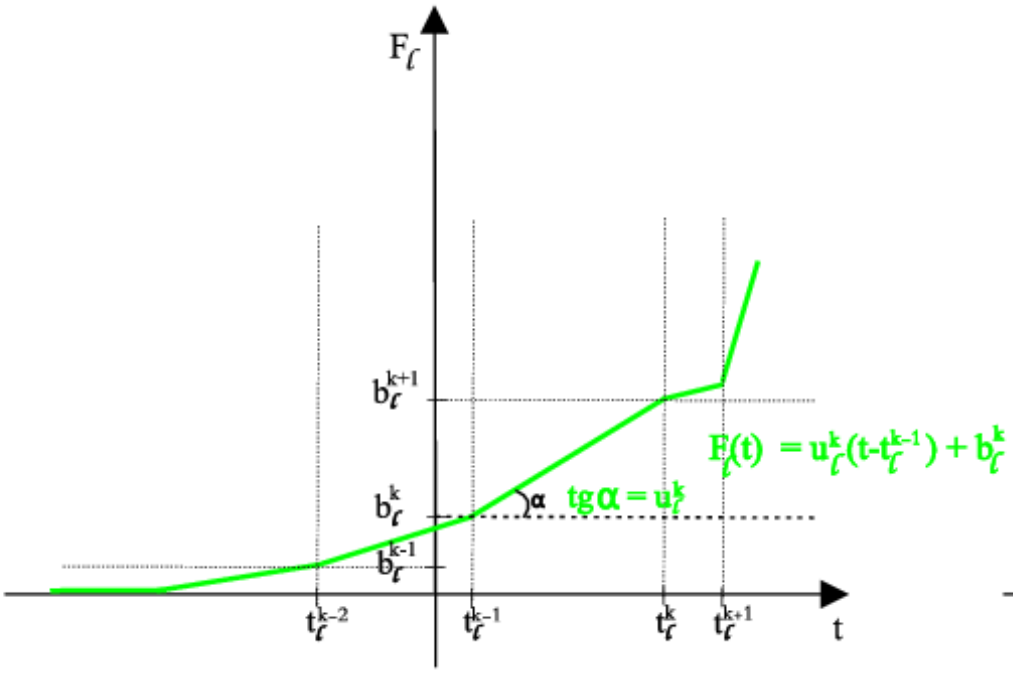

3
F2
8
7
t2
t1
3
10
13
18
25
t


2
F2
8
7
t2
t1
3
10
13
18
25
t

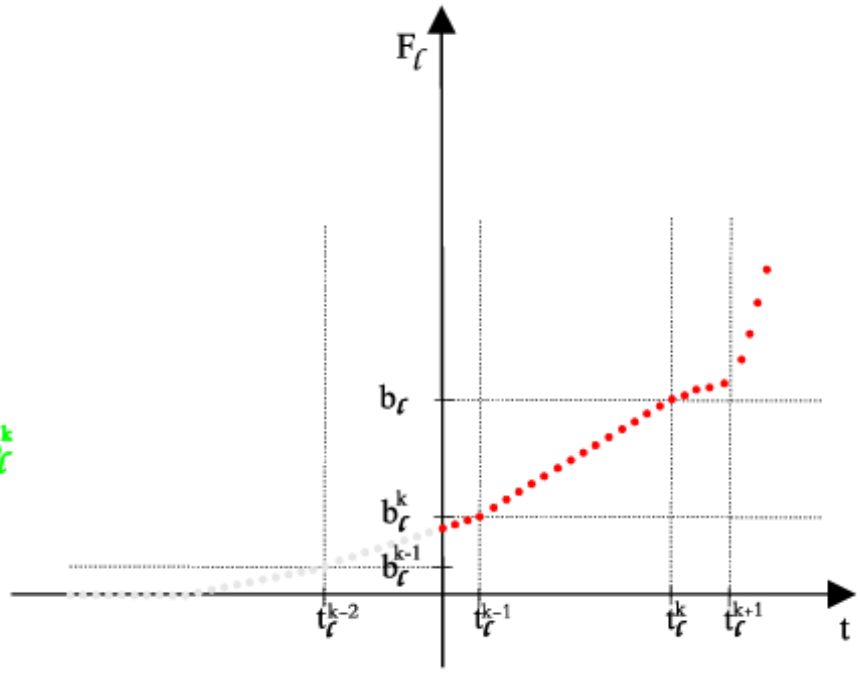

4
F2
8
7
t2
t1
3
10
13
15
25
t

FPTAS for 6 scheduling problems

(a)



(b)



FPTAS for 6 scheduling problems

Problem	Time complexity of the GrA	Time complexity of the FPTAS	Time complexity of the classical DPA
$1\|\|\sum w_jU_j$	$O(\min\{2^n, n\cdot\min\{d_{max}, F_{opt}\}\})$ [5]	-	$O(nd_{max})$
$1\|d_j = d'_j + A\|\sum U_j$	$O(n^2)$ [5] (GrA)	-	$O(n\sum p_j)$
$1\|\|\sum GT_j$	$O(\min\{2^n, n\cdot\{d_{max}, nF^*\}\})$	$O(n^2\log\log n + \frac{n^2}{\varepsilon})$	$O(nd_{max})$
$1\|\|\sum T_j$ special case $B-1$	$O(\min\{2^n, n\cdot\min\{d_{max}, F^*\}\})$	$O(n^2/\varepsilon)$	$O(nd_{max})$
$1\|\|\sum T_j$ special case $B-1G$	$O(\min\{n^2\cdot\min\{d_{max}, F^*\}\})$	$O(n^3/\varepsilon)$	$O(n^2d_{max})$
$1\|d_j = d\|\sum w_jT_j$	$O(\min\{n^2\cdot\min\{d, F^*\}\})$	$O(n^3/\varepsilon)$	$O(n^2d_{max})$
$1(no\text{-}idle)\|\|\max\sum w_jT_j$	$O(\min\{2^n, n\cdot\min\{d_{max}, nF^*, \sum w_j\}\})$ [5]	$O(n^2\log\log n + \frac{n^2}{\varepsilon})$	$O(nd_{max})$
$1(no\text{-}idle)\|\|\max\sum T_j$	$O(n^2)$ [4] (GrA)	-	$O(nd_{max})$

Dynamic Programming Algorithms for the Problem $1|d_j=d|\sum w_jT_j$

Single machine

n jobs $\quad j = 1,2,\ldots,n$

p_j processing time $\quad d_j=d$ common due date

w_j weight

Tardiness of job j in schedule π : $T_j(\pi) = max\{0, C_j(\pi)-d\}$

Goal: Find a schedule π^* that minimizes $\sum w_jT_j$

Dynamic Programming Algorithms for the Problem $1|d_j=d|\sum w_jT_j$

Lemma 1: There exists an optimal schedule

$\pi = (G,x,H)$, where

all jobs from set *G are on-time* and processed in non-increasing order of the values p_j/w_j ;

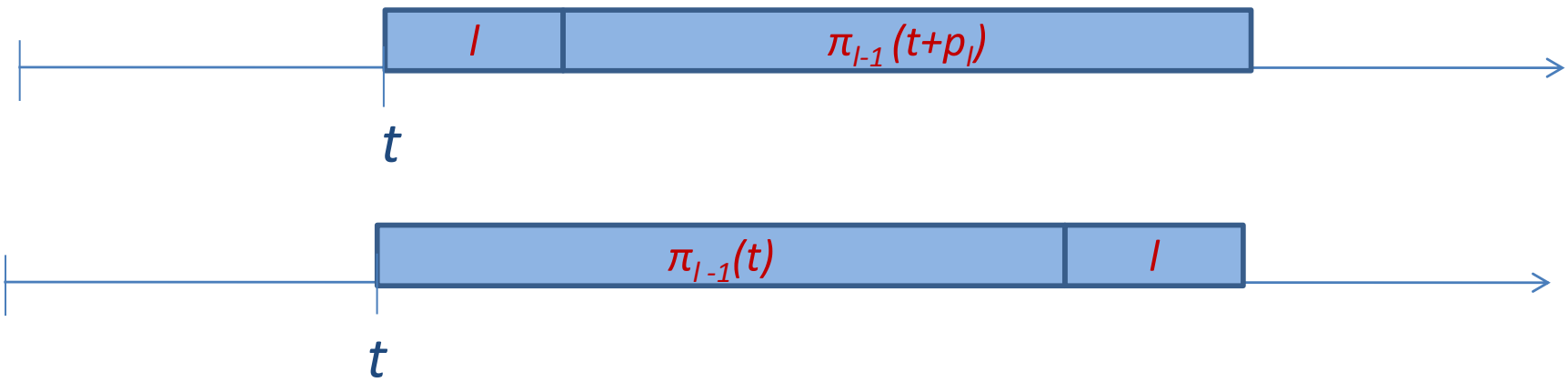
all jobs from set *H* are tardy and processed in non-decreasing order of the values p_j/w_j ;

the straddling job *x* starts before time *d* and is completed no earlier than time *d*.

First Dynamic Programming Algorithm for the Problem $1|d_j=d|\sum w_jT_j$

Let $x=1$ be the straddling job.

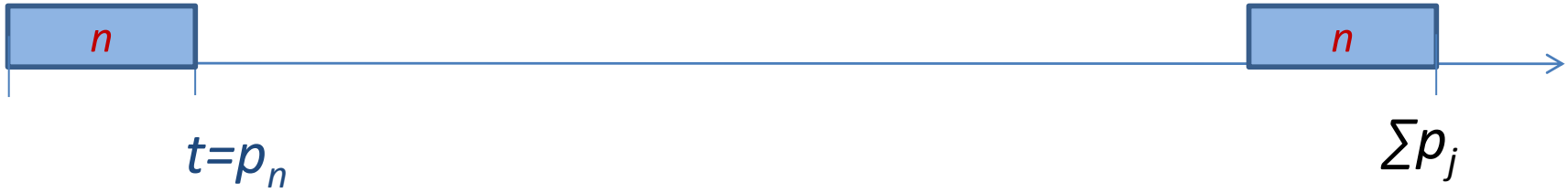
In step l, $l = 1,2,\ldots, n$ for each state $t=[0, \sum p_j]$ or $[0,d]$
we choose one of two positions for job l:



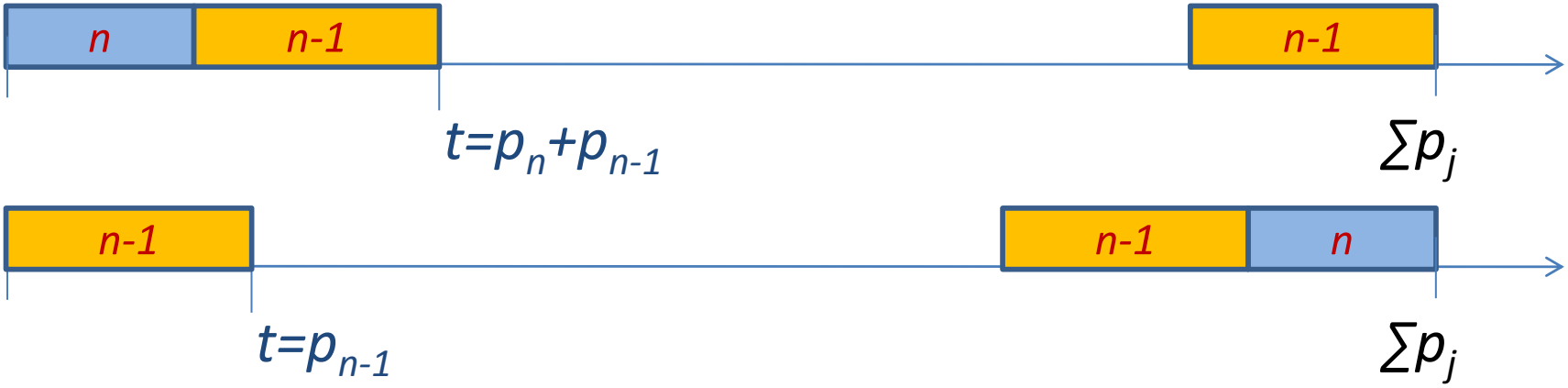
The running time is $O(nd)$ for each straddling job $x=1,2,\ldots,n$

The Second Dynamic Programming Algorithm for the Problem $1|d_j=d|\sum w_jT_j$

Let $x=1$ be the straddling job.



t is the total processing time of the jobs scheduled at the beginning of a schedule.
In step $l=n$, two states are saved: (p_n,F_1) and (p_n,F_2)



4 states are saved in step $l=n-1$

Comparison of Dynamic Programming Algorithms

In the first algorithm, all integer points (states) *t = [0,d]* are considered.
The running time is *O(nd)*.

In the second algorithm, only possible points *t = [0,d]* are considered, which are computed if the processing of the jobs starts at time 0.
The running time is *O(nd)* as well.

The second algorithm is faster (since it considers not all points *t*),
but **the first algorithm finds an optimal solution for each integer starting time from *[0,d]*.**

Graphical Algorithm

Dynamic Programming (Bellman 1954)

Functional equations:
consider in each step j all states $t \in [0, A] \cap Z$

$$f_j(t) = \min \begin{cases} \Phi^1(t) = \alpha_j(t) + f_{j-1}(t - a_j), & j = 1, 2, \ldots, n; \\ \Phi^2(t) = \beta_j(t) + f_{j-1}(t - b_j), & j = 1, 2, \ldots, n. \end{cases}$$

Idea of the graphical algorithm:
Combine several states into a new state

For $t \in [t_l, t_{l+1})$, we have
$f_j(t) = \varphi_{l+1}(t)$ and an optimal solution $X(t_l)$

Graphical Algorithm

Computations **in the first** dynamic programming algorithm

t	0	1	2	...	y	...	A
$f_j(t)$	$value_0$	$value_1$	$value_2$	...	$value_y$	...	$value_A$
optimal partial solution $X(t)$	$X(0)$	$X(1)$	$X(2)$	...	$X(y)$	...	$X(A)$

Computations in the graphical algorithm

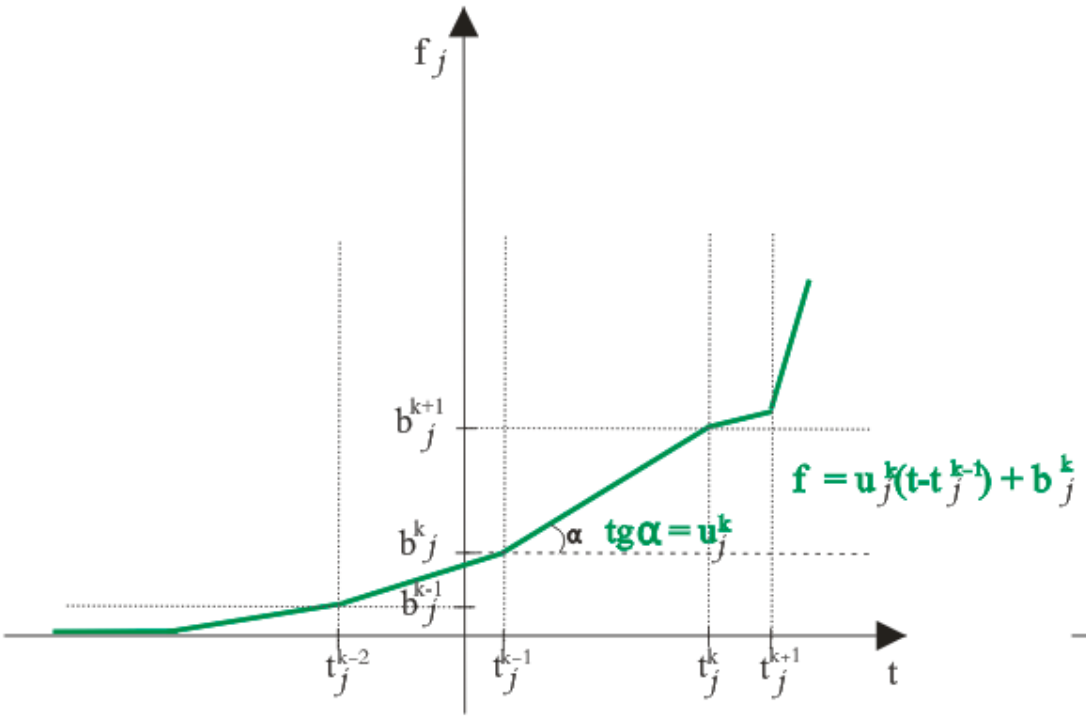
t	$[t_0, t_1)$	$[t_1, t_2)$	...	$[t_l, t_{l+1})$	...	$[t_{m_j-1}, t_{m_j}]$
$f_j(t)$	$\varphi_1(t)$	$\varphi_2(t)$	...	$\varphi_{l+1}(t)$	...	$\varphi_{m_j}(t)$
optimal partial solution $X(t)$	$X(t_0)$	$X(t_1)$	...	$X(t_l)$	...	$X(t_{m_j-1})$

For $t \in [t_l, t_{l+1})$, we have

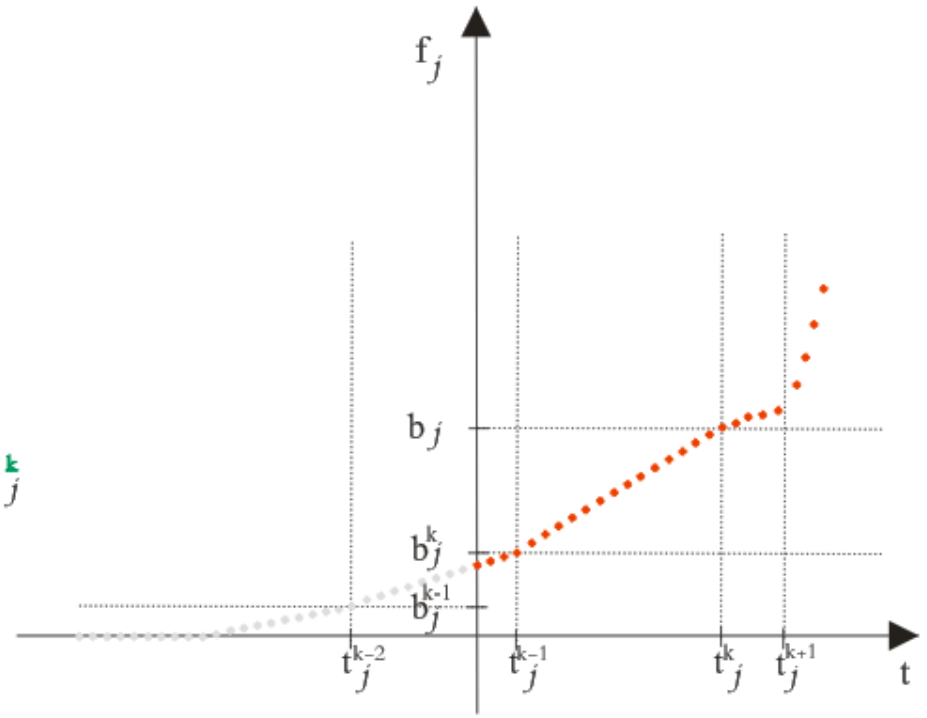
$f_j(t) = \varphi_{l+1}(t)$ and an optimal solution $X(t_l)$

Graphical Algorithm

(a)



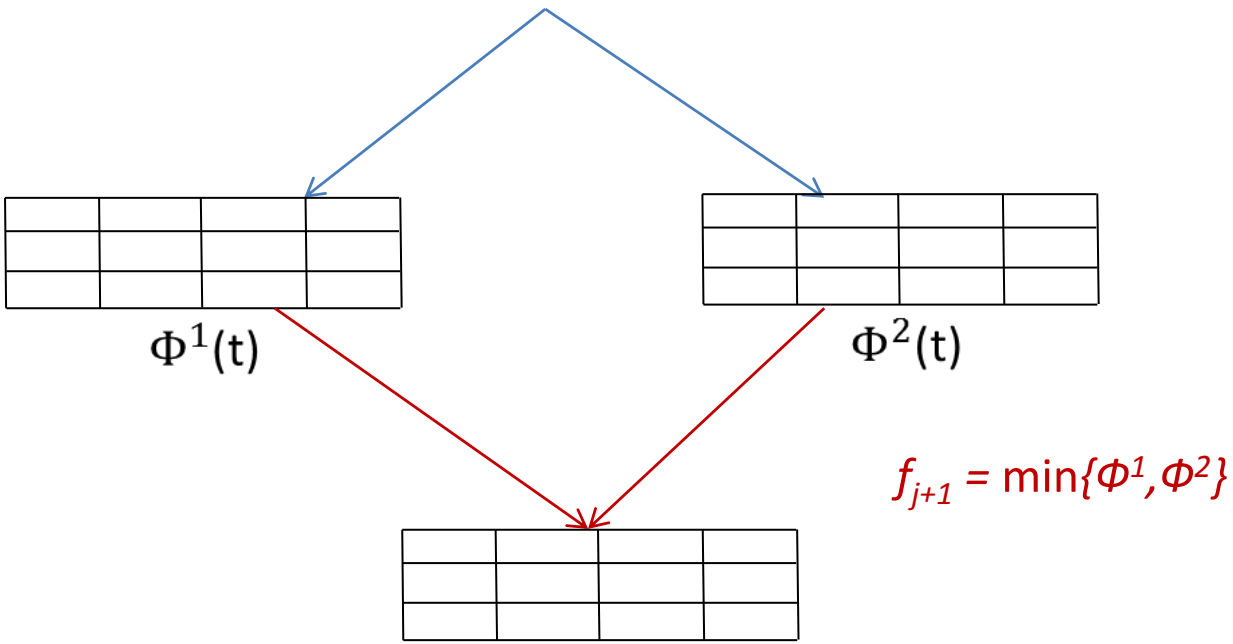
(b)



k	1	2	...	m_j+1	m_j+2
interval k	$(-\infty, t_j^1]$	$(t_j^1, t_j^2]$	...	$(t_j^{m_j}, t_j^{m_j+1}]$	$(t_j^{m_j+1}, +\infty)$
b_j^k	0	b_j^2	...	$b_j^{m_j+1}$	$+\infty$
u_j^k	0	u_j^2	...	$u_j^{m_j+1}$	0
π_j^k	π_j^1	π_j^2	...	$\pi_j^{m_j+1}$	$(1, 2, \ldots, j)$

Graphical Algorithm

k	1	2	...	m_j+1	m_j+2
interval k	$(-\infty, t_j^1]$	$(t_j^1, t_j^2]$	...	$(t_j^{m_j}, t_j^{m_j+1}]$	$(t_j^{m_j+1}, +\infty)$
b_j^k	0	b_j^2	...	$b_j^{m_j+1}$	$+\infty$
u_j^k	0	u_j^2	...	$u_j^{m_j+1}$	0
π_j^k	π_j^1	π_j^2	...	$\pi_j^{m_j+1}$	$(1, 2, \ldots, j)$



$f_{j+1} = \min\{\Phi^1, \Phi^2\}$

Graphical Algorithm

k	1	2	...	m_j+1	m_j+2
interval k	$(-\infty, t_j^1]$	$(t_j^1, t_j^2]$	...	$(t_j^{m_j}, t_j^{m_j+1}]$	$(t_j^{m_j+1}, +\infty)$
b_j^k	0	b_j^2	...	$b_j^{m_j+1}$	$+\infty$
u_j^k	0	u_j^2	...	$u_j^{m_j+1}$	0
π_j^k	π_j^1	π_j^2	...	$\pi_j^{m_j+1}$	$(1, 2, \ldots, j)$

In the table, $0<b_l^1<b_l^2<\ldots$ since function $F(t)$ is monotonic with t being the starting time.

Function $F_l(t)$ can be defined for all t from $(-\infty,+\infty)$.

Let *UB* be an upper bound on the optimal objective function value.
Then we have to save only the columns with $b_l^k<UB$.

The running time of the Graphical Algorithm is $O(n \min\{UB,d\})$ for each straddling job x.

FPTAS based on the Graphical Algorithm

k	1	2	...	m_j+1	m_j+2
interval k	$(-\infty, t_j^1]$	$(t_j^1, t_j^2]$	...	$(t_j^{m_j}, t_j^{m_j+1}]$	$(t_j^{m_j+1}, +\infty)$
b_j^k	0	b_j^2	...	$b_j^{m_j+1}$	$+\infty$
u_j^k	0	u_j^2	...	$u_j^{m_j+1}$	0
π_j^k	π_j^1	π_j^2	...	$\pi_j^{m_j+1}$	$(1, 2, \ldots, j)$

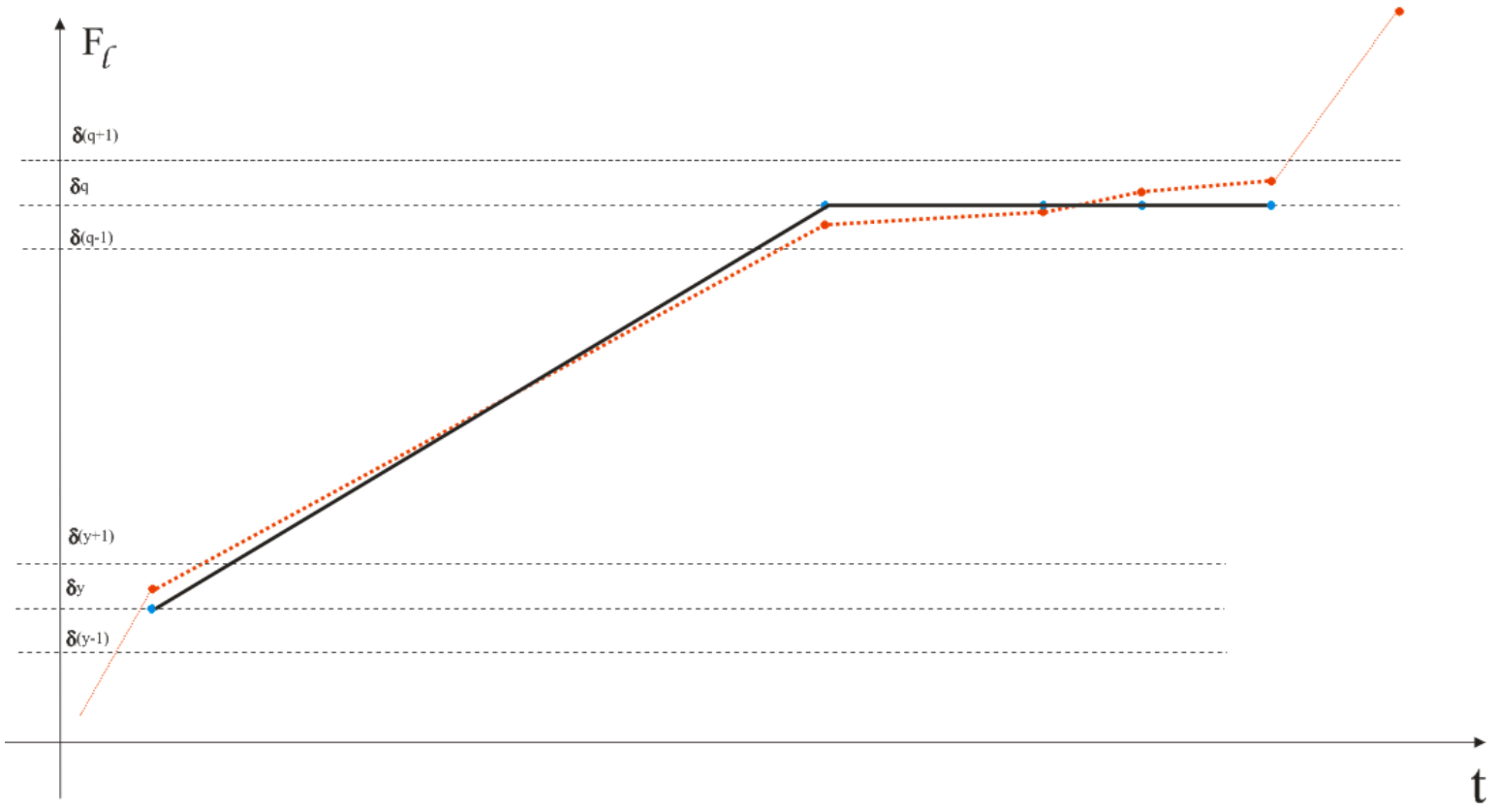
In the table, $0<b_j^1<b_j^2<\ldots$ since function $F(t)$ is monotonic with t being the starting time.

The running time of the Graphical Algorithm is $O(n \min\{UB,d\})$ for each straddling job x.

To reduce the running time, we can round (approximate) the values $b_j^k<UB$ to get a polynomial number of different values b_j^k

Let $\delta = \frac{\varepsilon UB}{2n}$. Round b_j^k up or down to the nearest multiple of δ

FPTAS based on the Graphical Algorithm



no more than $\frac{UB}{\delta} = \frac{2n}{\varepsilon}$ different values $\overline{b_l^k}$ no more than $4\frac{n}{\varepsilon}$ columns

cumulative error will be no more than $n\delta \leq \varepsilon F(\pi^*)$

The running time of the FPTAS is $O(\frac{n^3}{\varepsilon})$

Comparison of Dynamic Programming and Graphical Algorithms

Note	Classical DPA	GrA	Alternative DPA
Can it solve instances with $p_j \notin Z$ and instances with large values p_j	no	yes	yes
states t considered	all $t \in [0, d] \bigcap Z$	only t, where the slope of the function $F_l(t)$ is changed	only t from the set Θ_l
The running time for the initial instance	$O(n \min\{d, UB\})$	$O(n \min\{d, UB\})$	$O(n \min\{d, UB\})$
- of the problem $1\|\|\sum GT_j$ is	$O(nd_{max})$	$O(n \min\{d_{max}, UB\})$	$O(n \min\{d_{max}, UB\})$
- of the problem $1(no\text{-}idle)\|\|\max \sum w_j T_j$ is	$O(n \min\{d_{max}, UB\})$	$O(n \min\{d_{max}, UB, \sum w_j\})$	$O(n \min\{d_{max}, UB\})$

$$\Theta_l = \{x_1 p_1 + x_2 p_2 + \cdots + x_l p_l | x_1, x_2, \ldots, x_l \in \{0, 1\}\}$$

Comparison of Dynamic Programming and Graphical Algorithms

Note	Classical DPA	GrA	Alternative DPA
It finds all optimal schedules for all starting times $t \in [0, d]$ in time	$O(nd)$	$O(nd)$	-
If finds all optimal schedules for all starting times $t \in (-\infty, t_n^{UB}]$ in time	$O(nUB)$	$O(nUB)$	-
It finds all optimal schedules for all starting times $t \in (-\infty, +\infty)$ in time	$O(nF(\pi', d))$	$O(nF(\pi', d))$	-
The running time of the FPTAS is	$O(\frac{n^3}{\varepsilon} \log \frac{n}{\varepsilon}))$	$O(n^3/\varepsilon)^*$	$O(n^3/\varepsilon)^{**}$

* In this time, for all $t \in (-\infty, t_n^{UB}]$ solutions can be found with an absolute error restricted by εLB. For all $t \in [t_n^{LB}, t_n^{UB}]$, $t_n^{LB} \leq 0 \leq t_n^{UB}$, solutions can be found with a relative error restricted by ε.

** An approximate solution is only found for the starting time $t = 0$.

Graphical Algorithms and the corresponding FPTAS

Problem	Time complexity of GrA	Time complexity of FPTAS	Time complexity of classical DPA
$1\|\|\sum w_jU_j$	$O(\min\{2^n, n\cdot\min\{d_{max}, F_{opt}\}\})$ [5]	-	$O(nd_{max})$
$1\|d_j = d'_j + A\|\sum U_j$	$O(n^2)$ [5]	-	$O(n\sum p_j)$
$1\|\|\sum GT_j$	$O(\min\{2^n, n\cdot\{d_{max}, nF^*\}\})$	$O(n^2\log\log n + \frac{n^2}{\varepsilon})$	$O(nd_{max})$
$1\|\|\sum T_j$ special case $B-1$	$O(\min\{2^n, n\cdot\min\{d_{max}, F^*\}\})$	$O(n^2/\varepsilon)$	$O(nd_{max})$
$1\|\|\sum T_j$ special case $B-1G$	$O(\min\{n^2\cdot\min\{d_{max}, F^*\}\})$	$O(n^3/\varepsilon)$	$O(n^2d_{max})$
$1\|d_j = d\|\sum w_jT_j$	$O(\min\{n^2\cdot\min\{d, F^*\}\})$	$O(n^3/\varepsilon)$	$O(n^2d_{max})$
$1(no\text{-}idle)\|\|\max\sum w_jT_j$	$O(\min\{2^n, n\cdot\min\{d_{max}, nF^*, \sum w_j\}\})$ [5]	$O(n^2\log\log n + \frac{n^2}{\varepsilon})$	$O(nd_{max})$
$1(no\text{-}idle)\|\|\max\sum T_j$	$O(n^2)$ [4]	-	$O(nd_{max})$

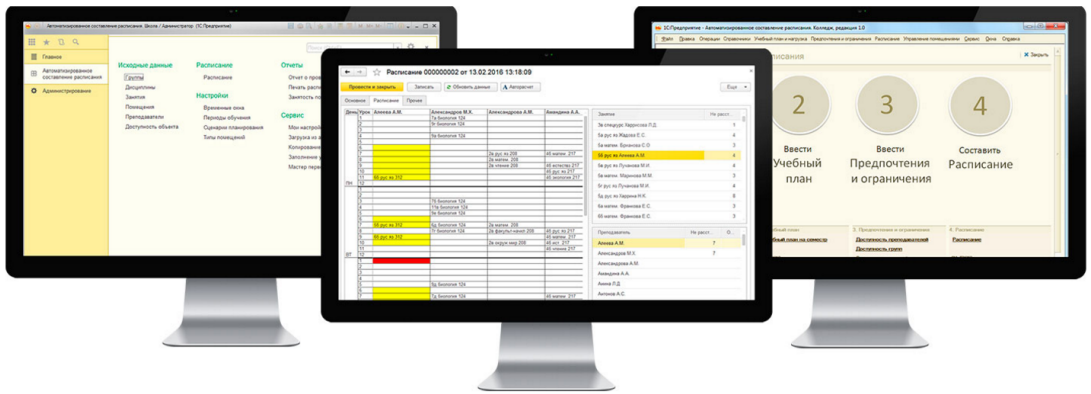
Section 3

Practical results

Education planning

1C Software product

1С: Автоматизированное составление расписания.
Университет / Колледж / Школа



На современной платформе 1С:8.3

Легкая в освоении

Математика для авторасчета расписания

Широкие настройки

Соответствие российским реалиям

Education planning

- Schedule construction in manual/automatic/mixed mode.
- 30 universities, 55 colleges, 160 schools

День	Интервал	АДФ 1 (50 чел.)	К2 10 (30 чел.)	К2 11 (56 чел.)	К2 23 (30 чел.)	914 к.к. (25 чел.)	СЗ
	08:00-09:35	4200а Нем.язык Свирид	900б Мат. анализ Петров	901а МСФО Иванов И.И.		4200а Франц.язык Расп	420
	09:50-11:25	4200а Линейная алгебра	900б Мат. анализ Петров	резерв под кафедру			420
	11:40-13:15	4200а Линейная алгебра	900б Мат. анализ Петров				420
	14:00-15:35	4201б 3D моделирование		резерв под кафедру			
	15:45-17:20						
1	17:30-19:05						
	08:00-09:35	4200а Линейная алгебра			905а Бухгалтерский учет		420
	09:50-11:25	4200а Линейная алгебра			905а Бухгалтерский учет		420
	11:40-13:15	4200а Линейная алгебра	4201б 3D моделирование				
	14:00-15:35	4200а Линейная алгебра	4201б 3D моделирование				
	15:45-17:20						
2	17:30-19:05						
	08:00-09:35	4200а Франц.язык Расп	900б Мат. анализ Петров	901а МСФО Иванов И.И	905а Бухгалтерский уче	4200а Нем.язык Свирид	900
	09:50-11:25	4200а Макроэкономика А					
	11:40-13:15	4200а Макроэкономика А					
	14:00-15:35			деканат совещание			
	15:45-17:20						
3	17:30-19:05						
	08:00-09:35		Несколько занятий				900
	09:50-11:25		4200а Макроэкономика		905а Бухгалтерский учет		900
	11:40-13:15				905а Бухгалтерский учет		
	14:00-15:35		901а МСФО Иванов И.И.				
	15:45-17:20						
4	17:30-19:05						
	08:00-09:35		901а МСФО Иванов И.И.				
	09:50-11:25						
	11:40-13:15						

Education planning

The goal

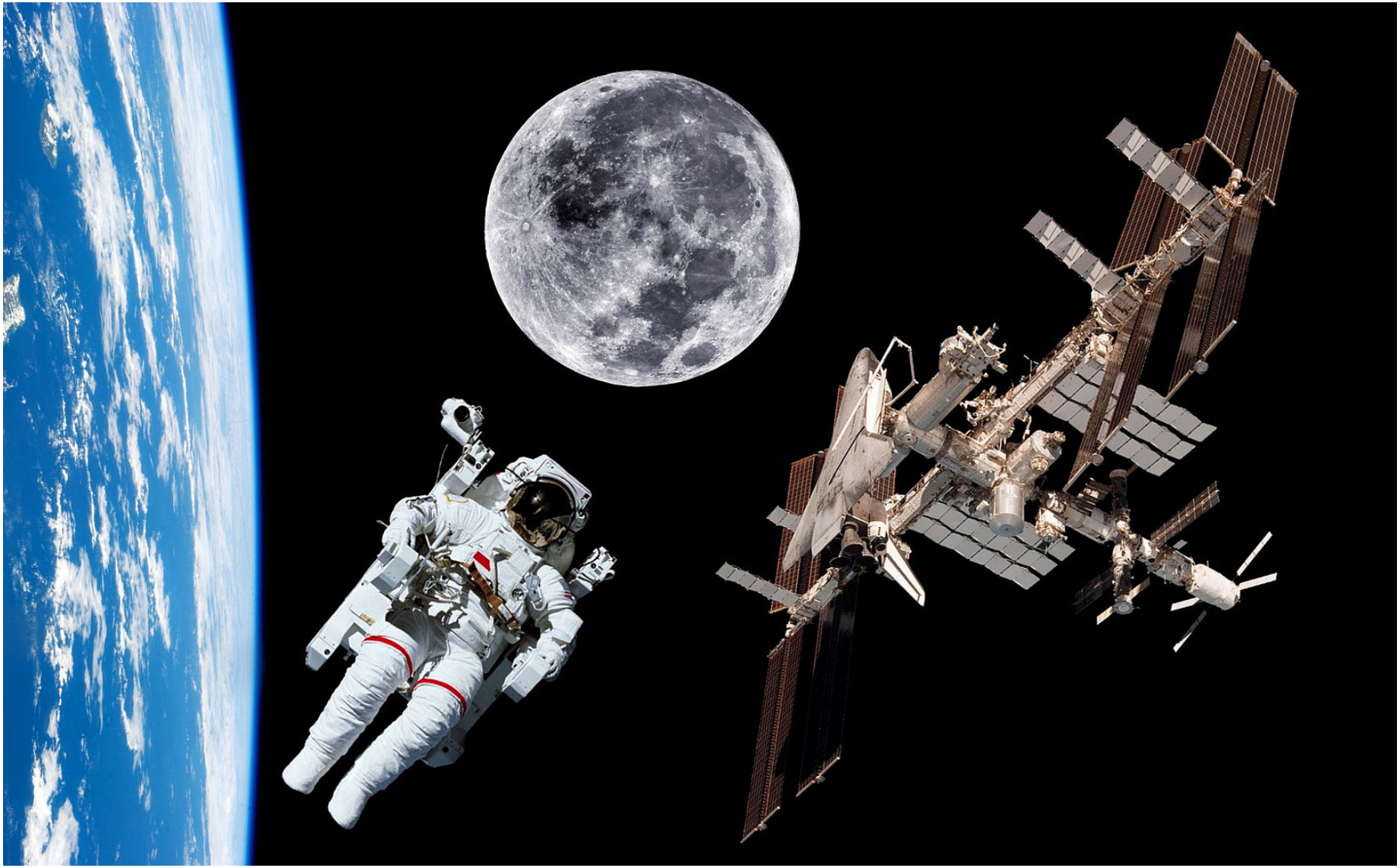
- To construct a feasible schedule that fits in all constraints,
- or an optimal schedule that minimizes the number of
  - windows (blank spaces) in a schedule;
  - transitions between buildings during a day;
  - unfulfilled staff wishes;
  - used rooms;

Mathematical problem

- Timetabling (over 1600 papers on similar problems on ScienceDirect.com).
- Problem is NP-hard.
- Fast metaheuristic ant-colony based solution approach was proposed.

Cosmonaut training scheduling problem

Cosmonaut training scheduling problem



Cosmonaut training scheduling problem statement

- Set of on-board systems.
- Sets of cosmonauts and crews.
- Set of resources (equipment, teachers, etc.).
- Dates of starts.

It is necessary to prepare appropriate crews to dates of their starts.

Our goals

- to develop mathematical model
- to find approaches to solve it
- to implement Planner system
- to reduce labor costs
- to form new and reschedule available timetable

Cosmonauts Training Scheduling Problem

Mathematical formulation — RCPSP (Resource-Constrained Project Scheduling Problem).

- Resource constraints.
- Precedence constraints.
- More than 4000 publications are devoted to this problem at scholar.google.ru.
- NP-hard in strong sense, there are no pseudo-polynomial algorithms.

Methods for solving RCPSP

- Dynamic programming.
- Methods of Integer Linear Programming.
- Methods of Constraint Programming.
- Heuristic algorithms.

Volume planning problem

Problem statement

- set of on-board systems (near 140);
- required number of cosmonauts of different skills for each on-board system.

Goal: to distribute training qualifications between cosmonauts, minimizing the difference between the maximum and minimum total time of training of cosmonauts.

Results

- heuristic greedy algorithm;
- branch and bound method (CPLEX).

Initial data
for volume planning problem

	требемое количество квалификаций							часы на подготовку					
	Корабль К1			*Корабль К2*				*неопытный*			*опытный*		
	С	О	П	С	О	П		С	О	П	С	О	П
Срочное покидание в аварийных ситуациях	0	3	0	0	3	0		23	23	22	23	23	22
Система инвентарного учета	0	0	3	0	1	2		0	17	2	0	9	0
Информационно-управляющая система	2	0	0	1	0	0		12	12	1	4	4	1
Бортовая вычислительная система	1	0	2	1	0	2		15	11	5,5	2	2	2
Система управления бортовым комплексом/ бортовой аппаратурой	1	0	2	1	0	2		28	22	9,5	2	2	2
Система бортовых измерений	1	0	2	1	0	2		15	13	2	4	4	0
Средства радиосвязи	1	1	1	1	0	2		35	28	11,25	4	4	2
Телевизионная система	1	0	2	1	0	2		11	11	2	4	4	0
Система обеспечения жизнедеятельности	1	1	1	1	0	2		70	57,25	26	12	12	5
Система энергоснабжения	1	0	2	1	0	2		20	18	2	8	6	0
Система управления движением и навигацией	1	1	1	1	1	1		38	17,5	2	8	8	1
Двигательные установки	1	0	0	1	0	0		4	0	0	2	0	0
Оптико-визуальные системы	1	0	0	1	0	0		0	4	0	0	2	0
Курс	1	0	0	1	0	0		7	7	0	2	2	0
Система стыковки	1	0	2	1	0	2		16	13	4	4	4	2
Конструкция и компоновка	0	2	1	0	2	1							
Система обеспечения теплового режима	1	1	1	1	0	2		43	24	6,5	8	8	2
Фотоаппаратура	0	1	2	0	1	2		0	19	0	0	8	0
Видеоаппаратура, аудиоаппаратура	0	1	2	0	1	2		0	28	0	0	12	0
ЛРС	1	0	0	1	0	0		22	14	5	11	8	5
Оборудование для ВКД (скафандр Орлан, шлюзовой отсек, инструменты для ВКД)	0	2	0	0	1	0		0	60	8	0	32	8

The experimental results for volume planning problem

№	Опыт	Жадный алгоритм			CPLEX		
		max	min	δ	max	min	δ
1	3 неоп.	889.5	887.0	2.5	888.05	887.75	0.3
	3 оп.	570.5	569	1.5	570	569.5	0.5
	1 оп., 2 неоп.	721.7	694.5	27.2	697.25	695.25	2
	2 оп., 1 неоп.	669.7	598.0	71.7	616.5	612.75	3.75
2	3 неоп.	266.25	265	1.25	265.75	265.2	0.55
	3 оп.	234.2	233	1.2	233.75	233.25	0.5
	1 оп., 2 неоп.	245.5	244.0	1.5	244.45	244	0.45
	2 оп., 1 неоп.	235.0	233.25	1.75	233.75	233.25	0.5
3	3 неоп.	660.2	659.5	0.7	659.85	659.75	0.1
	3 оп.	353.5	353.05	0.45	353.5	353	0.5
	1 оп.,2 неоп.	497.95	493.5	4.45	484.05	481.75	2.3
	2 оп., 1 неоп.	398.05	394.0	4.05	393.5	392.5	1
4	3 неоп.	925.75	924.2	1.55	925	924.8	0.2
	3 оп.	587	586.5	0.5	587	586.5	0.5
	1 оп., 2 неоп.	774.5	694.5	80.0	731.5	730.75	0.75
	2 неоп., 1 оп.	649.2	648.5	0.7	628.75	628	0.75

Measure of unsolvability

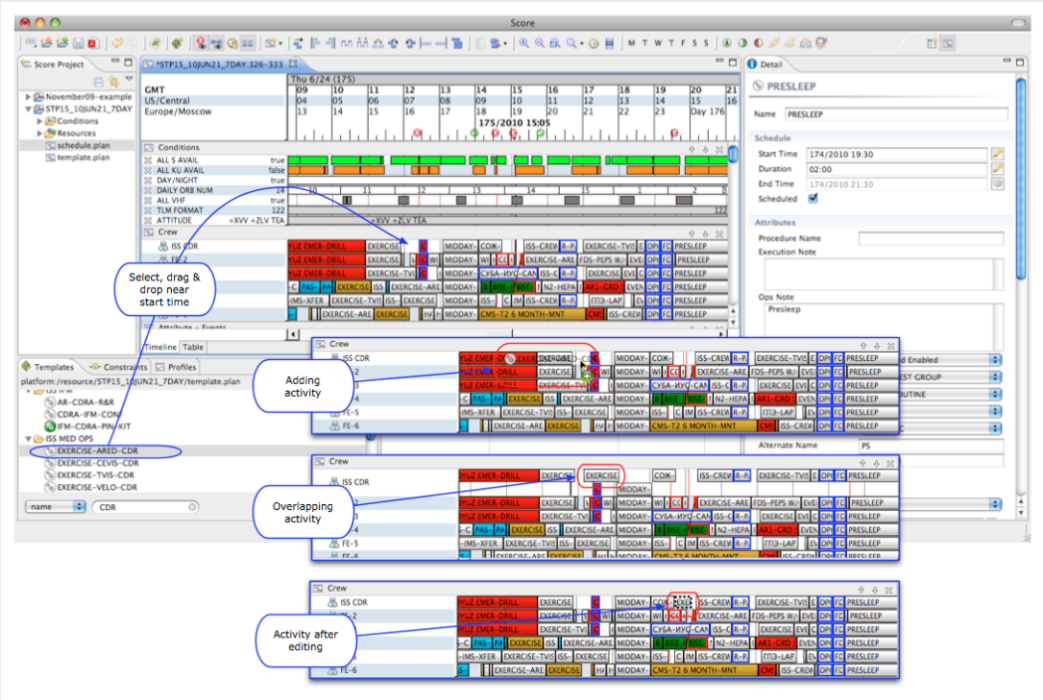
Timetabling problem

- Planing horizon is about 3 years.
- Each cosmonaut has an individual learning plan.
- 10 crews are studying simultaneously.
- There are main and backup crews.

BQ W:10 D:5	BT W:11 D:1	BU W:11 D:2	BV W:11 D:3	BW W:11 D:4	BX W:11 D:5	CA W:12 D:1	CB W:12 D:2	CC W:12 D:3	CD W:12 D:4	CE W:12 D:5	CH W:13 D:1	CI W:13 D:2	CJ W:13 D:3
ТПК СУС С1 БК#7	ТПК СУС Л ОРРЕЗСП БК#7	ТПК СУС С2 БК#7	ТПК СУС Л НАДБЕ32 БК#7	ТПК СУС С3 БК#7	ТПК СУС Л РАБАВНОС БК#7	ТПК СУС ПЗ ЦШТСП БК#7	ТПК СУС ПЗ РЕЗСП БК#7	ТПК СУС К1 БК#7	ТПК СУС К2 БК#7	ТПК СУД СБЛ Л ОРГРЕЖ БК#8	ТПК СУД СБЛ Л КОРРЦМ БК#8	ТПК СУД СБЛ Л РАБКУРС БК#8	ТПК СУД СБЛ Л ДФРС БК#8
	ТПК ССВП Л ПЛУСИС БК#10	ТПК СУС Л НАДБЕ31 БК#7			ТПК ССВП Л РЕЗРЕЖРАБ БК#10				ТПК СУС Э БК#7	Физ-ра		ТПК СУД СБЛ Л ОРЛСК БК#8	Англ.яз
Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед
	ТПК СУС Л ОРСРСП БК#7	ТПК ССВП Л КОМУПР БК#10		ТПК ССВП Л ШТРЕЖРАБ БК#10	ТПК ССВП Л НШС БК#10	ТПК ССВП ПЗ РАБССВП БК#10	Англ.яз	ТПК ССВП С БК#10		ТПК ССВП К БК#10	Англ.яз	ТПК СКГС Л ОБЩСВ БК#10	ТПК СКГС ПЗ РБКЛ БК#10
Физ-ра	Англ.яз	Англ.яз	Физ-ра	Физ-ра		Англ.яз	ТПК ССВП ПЗ РУЧОП БК#10	Физ-ра	ТПК СУД СБЛ Л ОС БК#8		ТПК ССВП Э БК#10	Физ-ра	ТПК СКГС К БК#10
													ТПК СМО ПЗ БК#20
ТПК СУС С1 БК#7	ТПК СУС Л ОРРЕЗСП БК#7	ТПК СУС С2 БК#7	ТПК СУС Л НАДБЕ32 БК#7	ТПК СУС С3 БК#7	ТПК СУС Л РАБАВНОС БК#7	ТПК СУС ПЗ ЦШТСП БК#7	ТПК СУС ПЗ РЕЗСП БК#7	ТПК СУС К1 БК#7	ТПК СУС К2 БК#7	ТПК СУД СБЛ Л ОРГРЕЖ БК#8	ТПК СУД СБЛ Л КОРРЦМ БК#8	ТПК СУД СБЛ Л РАБКУРС БК#8	ТПК СУД СБЛ Л ДФРС БК#8
	ТПК ССВП Л ПЛУСИС БК#10	ТПК СУС Л НАДБЕ31 БК#7			ТПК ССВП Л РЕЗРЕЖРАБ БК#10				Физ-ра			ТПК СУД СБЛ Л ОРЛСК БК#8	Физ-ра
Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед
	ТПК СУС Л ОРСРСП БК#7	ТПК ССВП Л КОМУПР БК#10	Англ.яз	ТПК ССВП Л ШТРЕЖРАБ БК#10	ТПК ССВП Л НШС БК#10	ТПК ССВП ПЗ РАБССВП БК#10	Англ.яз	ТПК ССВП С БК#10	ТПК СУС Э БК#7	ТПК ССВП К БК#10	ТПК ССВП Э БК#10	ТПК СКГС Л ОБЩСВ БК#10	ТПК СКГС ПЗ РБКЛ БК#10
Физ-ра	Англ.яз	Физ-ра		Физ-ра		Англ.яз	ТПК ССВП ПЗ РУЧОП БК#10	Физ-ра	ТПК СУД СБЛ Л ОС БК#8		Англ.яз	Физ-ра	ТПК СКГС К БК#10
													ТПК СМО ПЗ БК#20
	Англ.яз	ТПК КИК ПЗ ОПЕР БИ2 БК#1	ТПК КИК К БИ2 БК#1	ТПК КИК ПЗ ПРИМ БК#1	ТПК СРС Л УПРАВЛЕН БК#13	ТПК СУД АК Л ОСАК БК#5	ТПК СУД АК ПЗ НБИ2 БК#5	ТПК СУД ДК ПЗ НБИ2 БК#6	ТПК СУД ДК ПЗ НБИ2 БК#6	ТПК СУС Л НБИ2 БК#7	ТПК СУС Л ЦШС БК#7	ТПК СУС ПЗ НБИ2 БК#7	ТПК КСОЖ Л СОГС БК#12
	ТПК КИК Л ОЗН БИ2 БК#1	ТПК ПК Л НАЗН БК#2	Физ-ра		ТПК СРС Л ОРГАНРСВ БК#13	ТПК СУД АК Л РРАК БК#5	Физ-ра	ТПК СРС ПЗ БК#13		ТПК СУС Л АСАБНО БК#7			
Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед	Обед
	ТПК КИК Л ОТС БИ2 БК#1	Англ.яз	ТПК ПК З БК#2	ТПК СРС Л НАЗНАЧ БК#13	ТПК КДУ Л НБИ2 БК#3	ТПК СРС Л НШС БК#13	ТПК СУД ДК Л НБИ2 БК#6	Англ.яз	ТПК СРС Э БК#13		Англ.яз	ТПК СУД СБЛ Л ОП БИ2 БК#8	ТПК СУД СБЛ Л ДУ БИ2 БК#8
			ТПК КИК З БК#1						ТПК СРС ПЗ ИРИДИУМ БК#13				
	ТПК ПК Л ОБЩПРЕД БК#2	ТПК ПК ПЗ РАБПУЛ БК#2	Физ-ра			Англ.яз		ТПК СРС К ПРЕДСТРАЗ БК#13	Физ-ра		ТПК КСОЖ Л ВВЕДЕНИЕ БК#12	Физ-ра	
		ТПК ПК К БК#2											

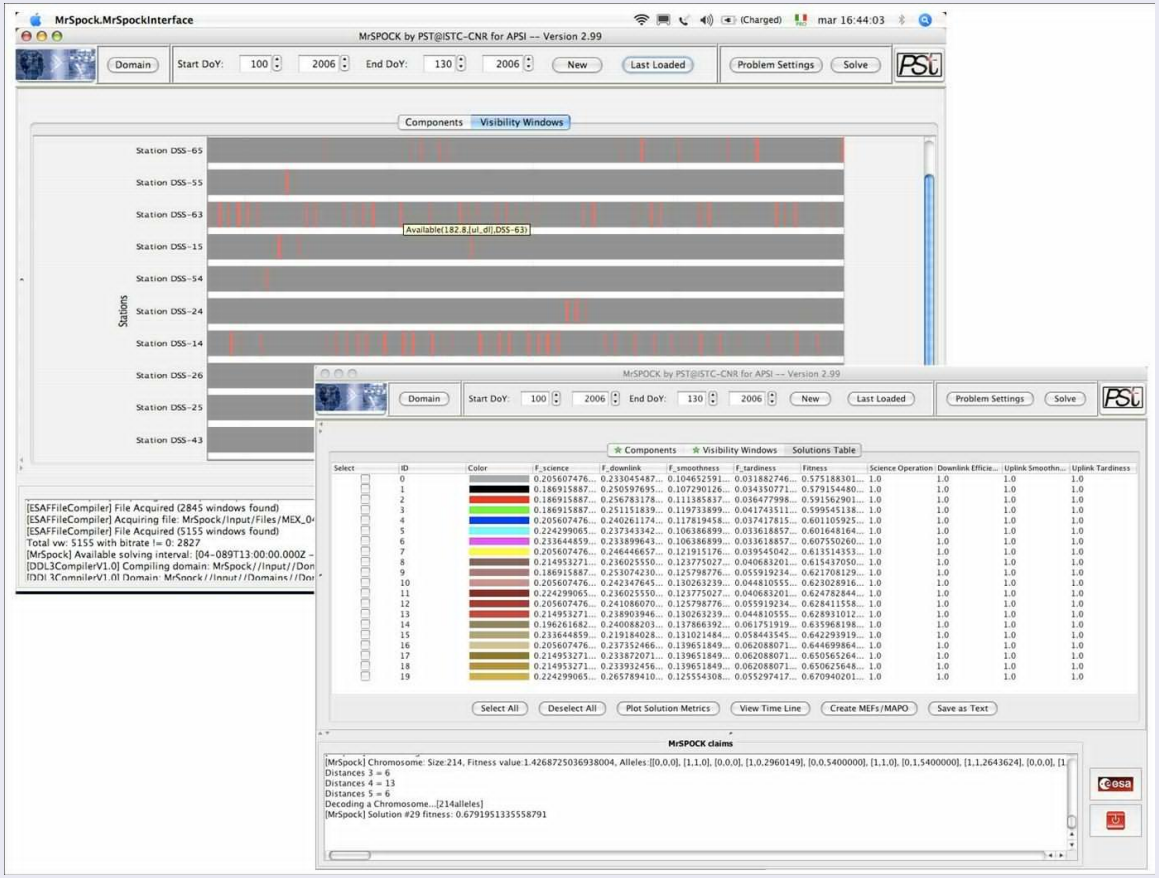
Review of other space agencies systems

NASA – TAMS, FOCAS, STAR



KAREN AU, SAMUEL SANTIAGO, RICHARD PAPASIN, MAY WINDERM, TRISTAN LE. *Streamlining Space Training Mission Operations with Web Technologies. An Approach to Developing Integral Business Applications for Large Organizations* // IEEE 4th International Conference on. Space Mission Challenges for Information Technology (SMC-IT), 2011, pp.159-166.

EKA



SPAGNULO, M., FLEETER, R., BALDUCCINI, M., NASINI, F. *Space Program Management : Methods and Tools* // Spagnulo, M., Fleeter, R., Balduccini, M., Nasini, F., Springer-Verlag New York - 2013. - 352 c.

Problem statement

- K – a number of cosmonauts;
- J_k – each cosmonaut k has his own set of training tasks;
- p_j – execution time of task $j \in J$;
- R – set of resources.

The goal is

to form a training schedule for each cosmonaut

Time intervals

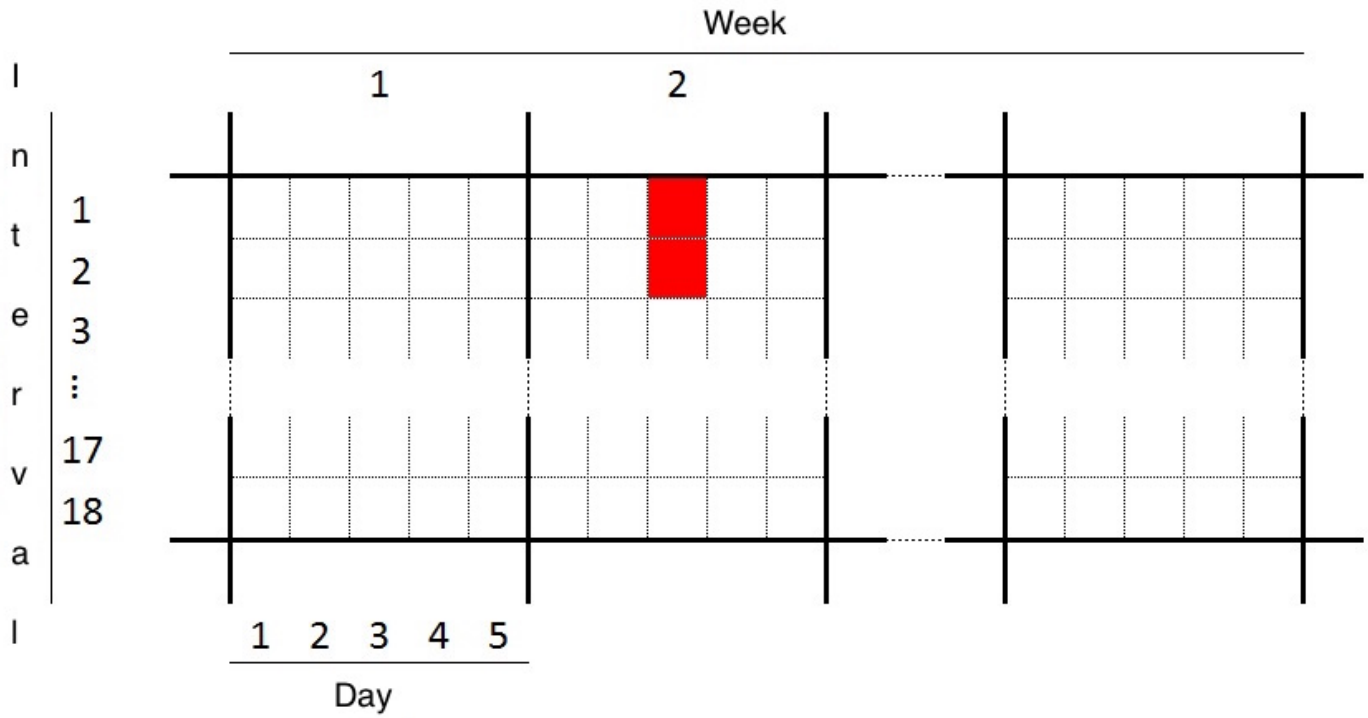
- W – set of planning weeks, where $|W| = 156$ weeks (3 years);
- $D_w = \{1,2,3,4,5\}$ – set of work days per week, $w \in W$;
- $H_{wd} = \{1, \ldots, 18\}$ – set of half-hour intervals of day $d \in D_w$ of week $w \in W$.

$$Y = \{(w, d, h) | w \in W, d \in D_w, h \in H_{wd}\}, \quad |Y| \approx 14040$$

$t(w, d, h)$ – considering time moment.

Variables

$$x_{jwdh} = \begin{cases} 1, & \text{iff task } j \text{ is started} \\ & \text{from interval } h \text{ of day } d \text{ of week } w; \\ 0, & \text{else.} \end{cases}$$



Constraints

Precedence relations between the tasks (academic plan)

$$\sum_{(w,d,h)\in Y} t(w,d,h)(x_{j_2 wdh} - x_{j_1 wdh}) \geq p_{j_1}, \tag{13}$$

$$\forall (j_1, j_2) \in \Gamma_k.$$

The resource limits (teachers, simulators, trainers)

$$\sum_{j\in J} rc_{jr} \sum_{\substack{h' > 0,\\ h - p_j + 1 \leq h' \leq h}} x_{jwdh'} \leq ra_{rwdh}, \tag{14}$$

$$\forall r \in R,\ \forall (w,d,h) \in Y. \qquad |Y| \approx 14040,\ |R| \approx 100.$$

Constraints

No more than ... (frequency of classes)

$$\sum_{j \in J^F} \sum_{d \in D_w} \sum_{h \in H_{wd}} x_{jwdh} \leq 2, \quad \forall w \in W. \tag{15}$$

Each cosmonaut may have no more than 2 physical trainings per week.

Excluding some time intervals

$$\sum_{j \in J_{[h_1;h_2]}} \sum_{h_1 - p_j + 1 \leq h \leq h_2} x_{jwdh} = 0, \tag{16}$$

$$\forall w \in W, \quad \forall d \in D_w;$$

$[h_1; h_2]$ – time period when performing task j is forbidden.

It is forbidden to practice in the hyperbaric chamber after lunch.

Comparison of two approaches to solving the scheduling problem for 1 crew

N	CPLEX MIP				CPLEX CP			
	Time, c	Var.	Constr.	Iter.	Time, c	Var.	Constr.	Branch.
1	09.06	26820	37620	21922	0.250	291	2170	1272
2	30.75	52680	60066	54234	0.329	363	2788	1512
3	559.84	73500	87846	5019412	0.438	492	3548	2008
4	375.834	108720	121578	2032790	0.703	606	4263	2784
5	374.63	115200	125466	2022320	0.610	642	4348	2912
7	346.30	144480	157920	820534	0.640	654	4374	2648
10	6657.98	204000	210646	16 917 014	1.317	852	5738	3 448

N is a number of on-board systems.

Results

Results

- Schedule for 1 crew for 1 year 3 moths

Our plans

- Schedule for 2 crew for 2 year

Railway operational and maintenance scheduling

Railway scheduling pioneers

Frank, O., Two-Way Traffic on a Single Line of Railway, Oper. Res., 1966, vol. 14, no. 5, pp. 801–811.

Szpigel, B., Optimal Train Scheduling on a Single Line Railway, Oper. Res., 1973, pp. 344–351.

Relation between railway planning problems and classical scheduling problems

- track segments = «machines»
- trains = «jobs»

Existing approaches and solution methods

1. Considering in terms of job-shop.

Szpigel B. Optimal train scheduling on a single line railway. Oper Res, 344 - 351, 1973.

Sotskov Y. Shifting bottleneck algorithm for train scheduling in a single-track railway. Proccedings of the 14th IFAC Symposium on Information Control Problems. Part 1. Bucharest/Romania. 87 - 92. 2012.

Gafarov E.R., Dolgui A., Lazarev A.A. Two-Station Single-Track Railway Scheduling Problem With Trains of Equal Speed. Computers and Industrial Engineering. 85:260 - 267. 2015.

Harbering J., Ranade A., Schmidt M. Single Track Train Scheduling. Institute of Numerical and Applied Mathematics. preprint. 18. 2015.

Existing approaches and solution methods

2. Integer linear programming

Brannlund U., Lindberg P.O, Nou A. and Nilsson J.E.
Railway Timetabling Using Lagrangian Relaxation.
Transportation Science 32(4):358 - 369. 1998.

Lazarev, A.A. and Musatova, E.G.
Integer Formulations of the Problem of Railway Train Formation and Timetabling,
Upravlen. Bol'shimi Sist., 2012, no. 38, pp. 161–169.

Exicting approaches and solution methods

3. Heuristics

Sotskov Y.
Shifting bottleneck algorithm for train scheduling in a single-track railway.
Proccedings of the 14th IFAC Symposium on Information Control Problems. Part 1. Bucharest/Romania. 87 - 92. 2012.

Mu S., Maged D.
Scheduling freight trains traveling on complex networks.
Transportation Research Part B: Methodological. 45(7):1103 - 1123. 2011.

Carey M., and Lockwood D.
A model, algorithms and strategy for train pathing.
The Journal of Operational Research Society. 8(46):988 - 1005. 1995.

Exicting approaches and solution methods

Allocation of polynomially solvable cases of railway scheduling problems

Gafarov E.R., Dolgui A., Lazarev A.A.
Two-Station Single-Track Railway Scheduling Problem With Trains of Equal Speed.
Computers and Industrial Engineering. 85:260 - 267. 2015.

Harbering J., Ranade A., Schmidt M.
Single Track Train Scheduling.
Institute of Numerical and Applied Mathematics. preprint. 18. 2015.

Disser Y., Klimm M., Lubbecke E.
Scheduling Bidirectional Traffic on a Path.
In Proceedings of the 42nd International Colloquium on Automata, Languages, and Programming (ICALP). 406 - 418. 2015.

Laboratory projects in railway scheduling

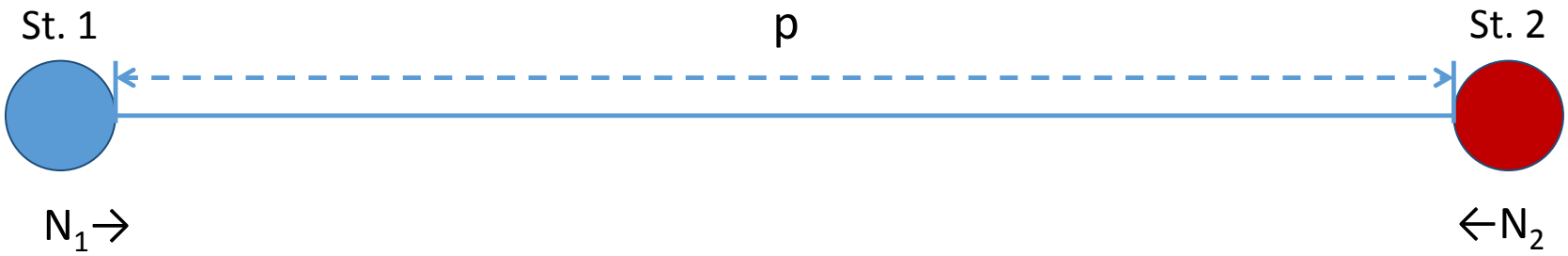
Small-scale problems

- Scheduling problem on single railway tracks.
- Goal – the development of exact polynomially solvable algorithms with small computational complexity.
- Solution approach – dynamical programming.

Large-scale problems

- The freight car routing problem.
- Goal – the construction of operational plan with feasible solution time.
- Solution approach – integer linear programming, LP-relaxation, column generation.

Two-station single track railway scheduling problem



Initial data

- $|N_1| = n$, $|N_2| = n'$, $N = N_1 \cup N_2$, $|N| = n + n'$.
- All trains have equal speed, track traversing time – p.
- Minimal time between the departure of two trains from one station – β.
- The transportation starts at time $t = 0$.

Denote the problem as $STR2$ (Single Track Railway Scheduling Problem).

Problem formulation

Schedule

In schedule σ, for each train $i \in N$

- $S_i(\sigma)$ – it's departure time;
- $C_i(\sigma)$ – arrival time, $C_i(\sigma) = S_i(\sigma) + p$.

Objective function

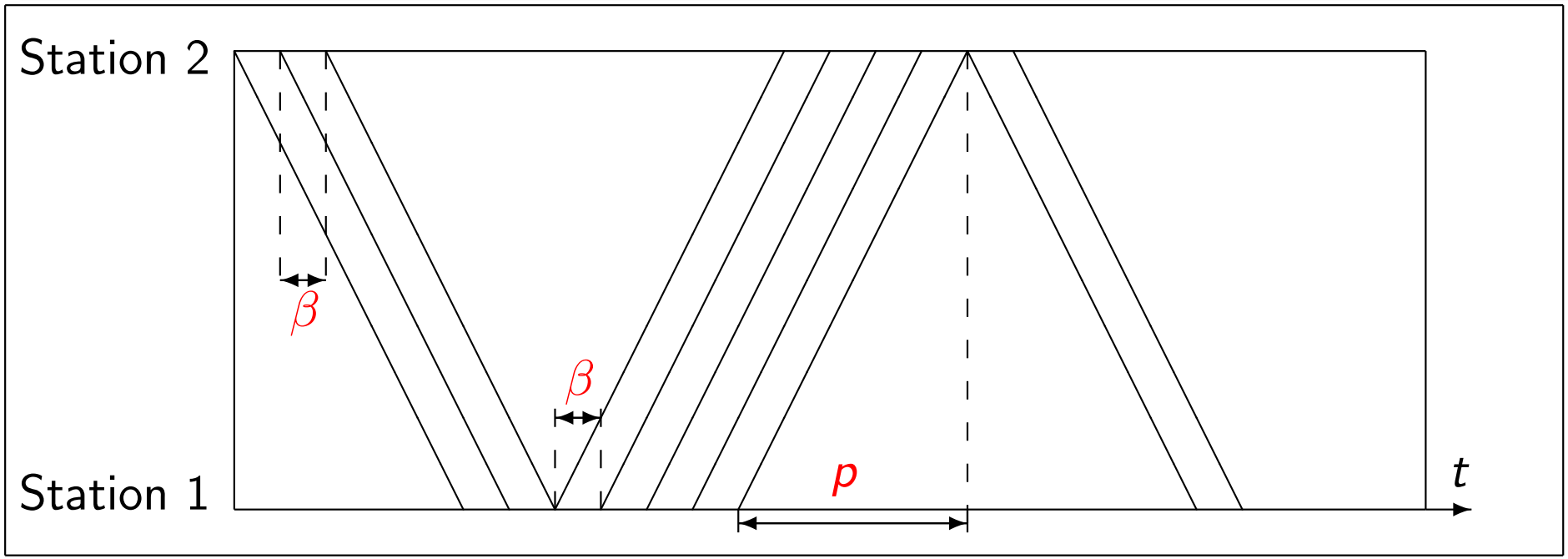
- Family of objective functions.
- The approach will be demonstrated on the maximum lateness objective function $L_{max}(\sigma)$,

$$L_{max}(\sigma) = \max_{i \in N} L_i = \max_{i \in N} \{ C_i(\sigma) - d_i \}.$$

Dynamic programming approach

Assumption

We will consider schedule schedule σ which possess the following property: for any point in time t such that $0 \leq t \leq C_{max}(\sigma)$ there exists at least one train $i \in N$ satisfying the condition $S_i(\sigma) \leq t \leq C_i(\sigma)$.



Dynamic programming approach

Assumption

Train departure order is specified.

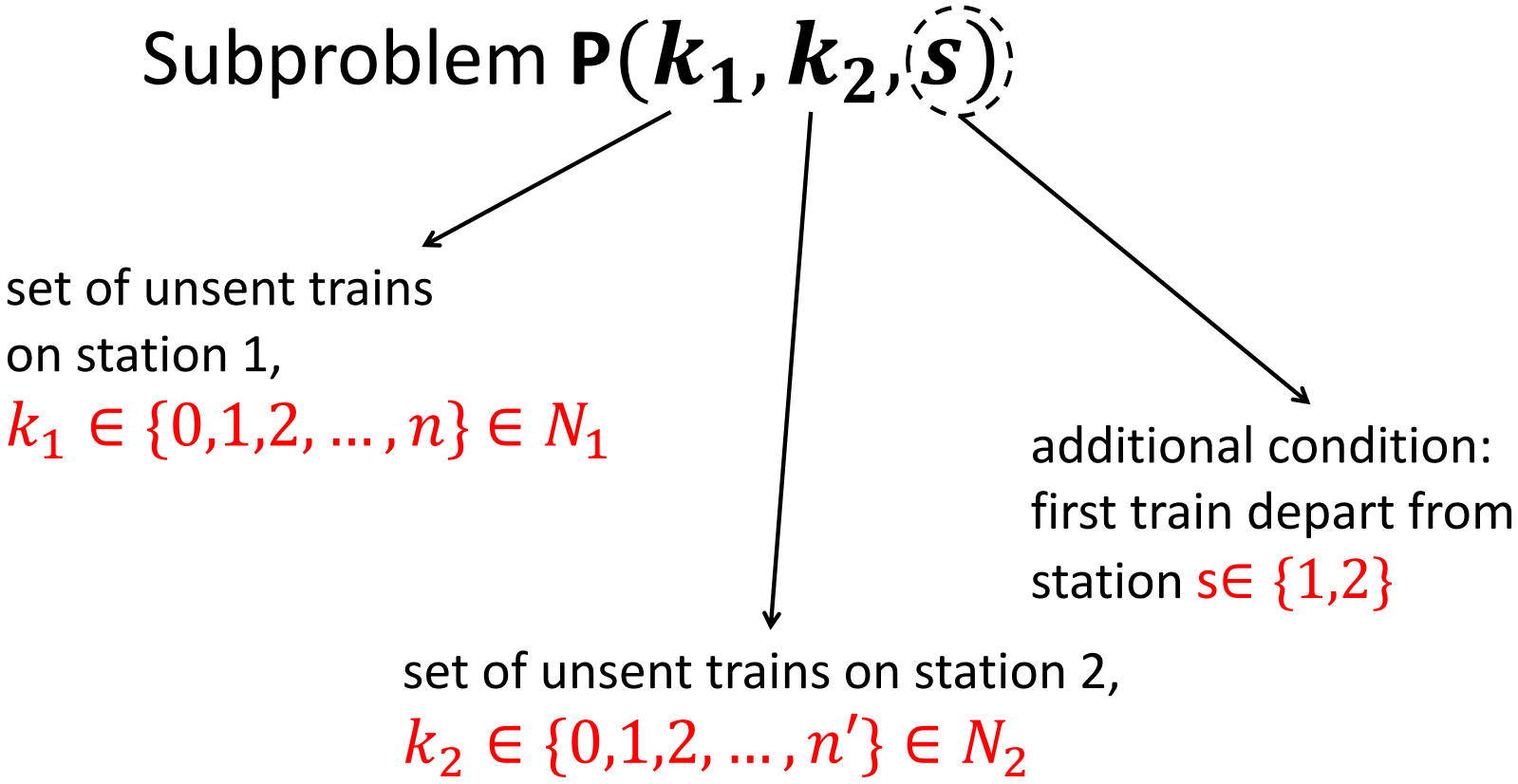
Maximum lateness L_{max}

For objective function $L_{max}(\sigma) = \max\limits_{i \in N}\{C_i(\sigma) - d_i\}$ there exists an optimal schedule σ in which trains depart from each station in a nondecreasing order of due dates d_i.

Numbering of trains

On each station trains are numbered in the decreasing order of their departure times, $i > j$ implies that, in any schedule σ, $S_i(\sigma) < S_j(\sigma)$.

Dynamic programming approach

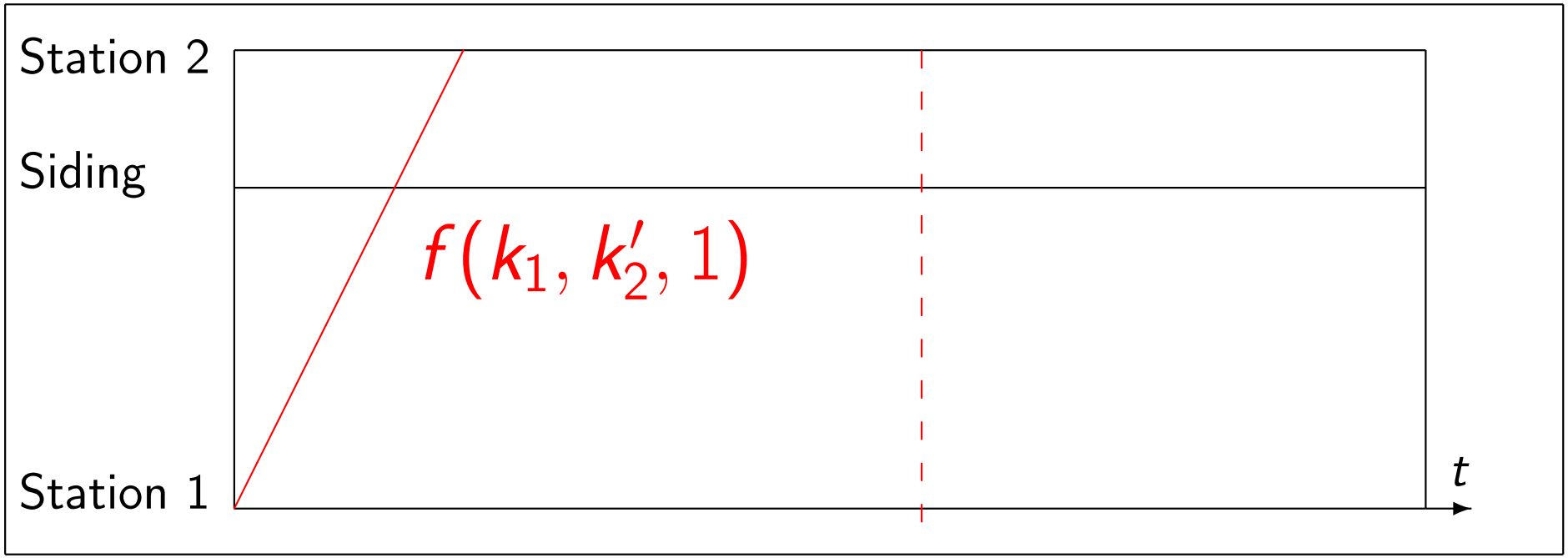


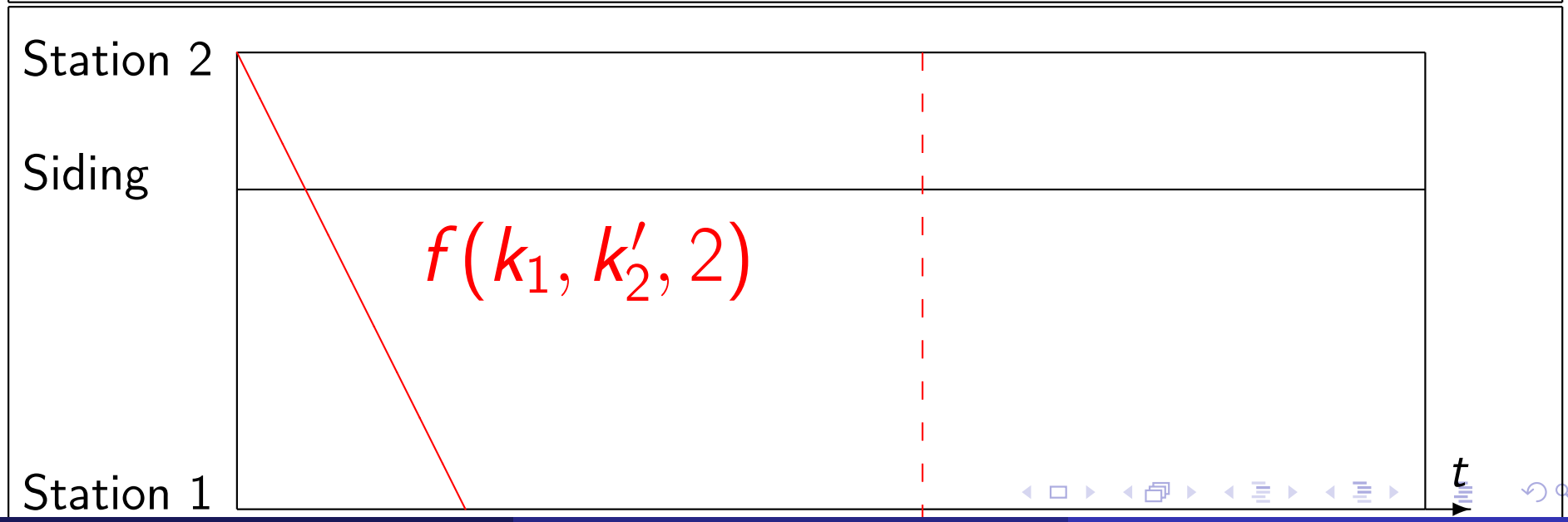
Optimal value of the objective function for $P(k_1, k_2', s)$

$$f(k_1, k_2', s) = F(\sigma^*),$$

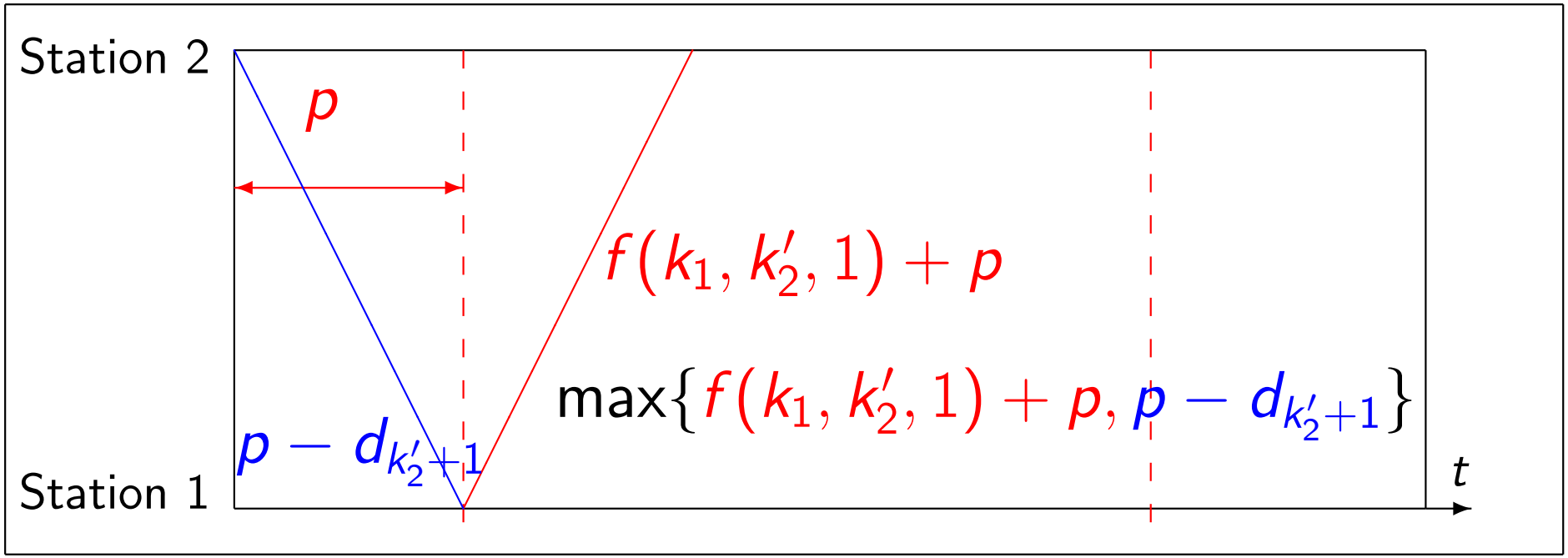
where σ^* is an optimal schedule for $P(k_1, k_2', s)$.

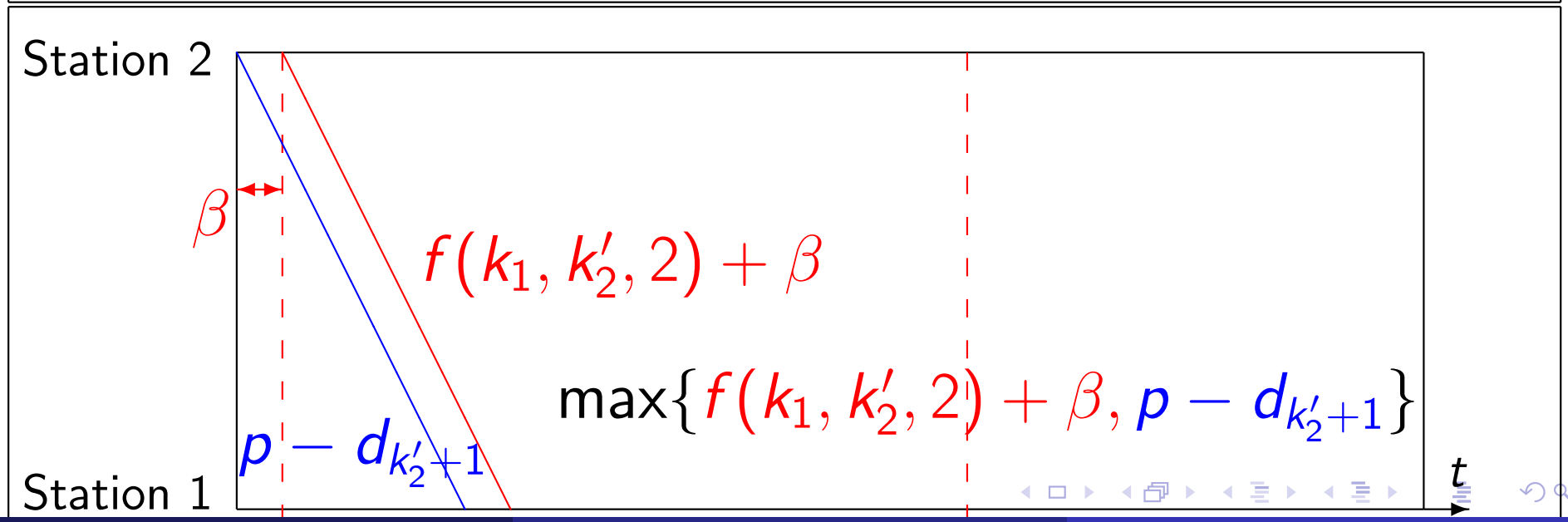
Solution algorithm





Solution algorithm





Dynamic programming approach

$$f(k_1, k_2' + 1, 2) = \max \begin{cases} p - d_{k_2'+1}; \\ \min \begin{cases} f(k_1, k_2', 1) + p; \\ f(k_1, k_2', 2) + \beta; \end{cases} \end{cases}$$

$$\text{for each } k_2' \in \{1', ..., n' - 1'\}, \ k_1 \neq 0.$$

Dynamic programming approach

Setting

$$f(1, 0', 1) = p - d_1$$

$$f(0, 1', 2) = p - d_{1'}$$

Bellman equation

$$f(k_1 + 1, k_2', 1) = \max \begin{cases} p - d_{k_1+1}; \\ \min \begin{cases} f(k_1, k_2', 1) + \beta; \\ f(k_1, k_2', 2) + p. \end{cases} \end{cases} \qquad k_1 \in \{1, ..., n-1\},\ k_2' \neq 0'$$

$$f(k_1, k_2' + 1, 2) = \max \begin{cases} p - d_{k_2'+1}; \\ \min \begin{cases} f(k_1, k_2', 1) + p; \\ f(k_1, k_2', 2) + \beta. \end{cases} \end{cases} \qquad k_2' \in \{1', ..., n' - 1'\},\ k_1 \neq 0$$

Dynamic programming approach

Optimal objective function value of the original problem

$$\min\{f(n, n', 1), f(n, n', 2)\}$$

Computational complexity

$$O((n + n')^2)$$

Value of $f(k_1, k_2', s)$ is computed for:

- each pair of k_1, $k_1 \in \{1, ..., n\}$), and k_2', $k_2 \in \{1, ..., n'\}$.

Dynamic programming approach

Other objective functions

This solution procedure can applied to a set of objective functions, for example for

$$\sum w_i C_i(\sigma) = \sum_{i \in N} w_i C_i(\sigma)$$

Condition

- "Shifted" schedule σ_t of schedule σ, $C_i(\sigma) - C_i(\sigma_t) = t$ for all $i \in N$.
- There exists $G(k_1, k_2', s)$ so that $F(\sigma_t) = F(\sigma) + G(k_1, k_2', t)$.
  - for L_{max}: $G(k_1, k_2', t) = t$;
  - for $\sum w_i C_i(\sigma)$: $G(k_1, k_2', t) = \sum_{i=1}^{k_1} w_i t + \sum_{j=1'}^{k_2'} w_j t$.

Dynamic programming approach

General form of objective functions

$$\bigodot_{i \in N} \varphi_i(C_i(\sigma)),$$

where

- $\varphi_i(\cdot)$ – nondecreasing function, defined for each train $i \in N$,
- $\odot$ – some commutative and associative operation such,
- for any numbers a_1, a_2, b_1, b_2, $\odot$ satisfy $a_1 \leq a_2$ and $b_1 \leq b_2$,

$$a_1 \odot b_1 \leq a_2 \odot b_2.$$

Dynamic programming approach

Solution procedure

$$STR2||\bigodot_{i\in N}\varphi_i(C_i(\sigma))$$

- Specified train departure order on each station.
- Polynomial set of possible departure times T, $|T| = O((n+n')^2)$.
- Subproblem: $P(k_1, k_2', s, t)$, $f(k_1, k_2', s, t)$ is calculated for
  - each pair of k_1, $k_1 \in \{1, ..., n\}$;
  - each pair of k_2', $k_2 \in \{1, ..., n'\}$;
  - all $t \in T$.

Computational complexity – $O((n+n')^4)$.

Dynamic programming approach

Minimization of maximum cost functions

$$F_{max}(\sigma) = \max_{i \in N} \varphi_i(C_i(\sigma))$$

- No specified order of train departure on each station.

Iterative optimization procedure

dynamic programming algorithm for $STR2||L_{max}$

general optimisation scheme, presented by Zinder and Shkurba[1]

[1] Zinder, Y. and Shkurba, V. Effective iterative algorithms in scheduling theory. Cybernetics, 21(1), 86–90. 1985.

Dynamic programming approach

Iterative optimisation procedure

Algorithm 1 Solution method for the train scheduling problem $STR2||F_{max}$

1: $V := \max_{i \in N} \varphi_i(p)$ (lower bound)
2: **for** $i := 1$ **to** $n + n'$ **do** — Due date setting
3: **if** $\varphi_i(\tau_r) \leq V$ **then**
4: $d_i := \tau_r$
5: **else**
6: choose τ_k so that $\varphi_i(\tau_k) \leq V < \varphi_i(\tau_{k+1})$
7: $d_i := \tau_k$
8: **end if**
9: **end for**
10: construct schedule σ by solving $STR2||L_{max}$
11: $L := L_{max}(\sigma)$
12: **if** $L > 0$ **then** — Lower bound checking
13: $V := \min_{i \in \{j:\, j \in N,\ (d_j + L) \in T'\}} \varphi_i(d_i + L)$ (lower bound)
14: **go to** 2
15: **else**
16: **return** σ is an optimal value
17: **end if**

Computational complexity

$O((n + n')^5 \log(n + n'))$

Results for STR2

Dynamic programming procedure for a set of objective functions

$$F(\sigma) = \bigodot_{i \in N} \varphi_i(C_i(\sigma))$$

Computational complexity is $O((n + n')^4)$,
can be reduced for a subset of objective functions – $O((n + n')^2)$.

Iterative optimisation procedure for maximum cost functions

$$F_{max}(\sigma) = \max_{i \in N} \varphi_i(C_i(\sigma))$$

Computational complexity is $O((n + n')^5 \log(n + n'))$.

Results for STR2

Solution algorithm complexity

Problem	Complexity
$STR2 \mid\mid L_{max}$	$O(n^2)$
$STR2 \mid\mid \sum w_j C_j$	$O(n^2)$
$STR2 \mid\mid \max\limits_{j \in N} \varphi_j(C_j(\sigma))$	$O(n^5 \log n)$
$STR2 \mid p(j), \lambda \mid L_{max}$	$O(n^\lambda)$
$STR2 \mid p(j), \lambda \mid \sum w_j C_j$	$O(n^\lambda)$
$STR2 \mid p(j), \lambda \mid \sum U_j(\sigma)$	$O(n^{2\lambda})$
$STR2 \mid p(j), \lambda \mid \bigodot\limits_j \varphi(C_j)$	$O(n^{\alpha^2+\alpha} n^\lambda)$
$STR2 \mid p(j), \lambda, V \mid \max\limits_{j \in N} \varphi_j(C_j(\sigma))$	$O(q^2 \log q n^{2\alpha^2+2\alpha+1} n^\lambda \log n)$

λ – the number of subsets with possible fixed departure order $p(j)$ – different train traversing times V – feasible intervals of movement

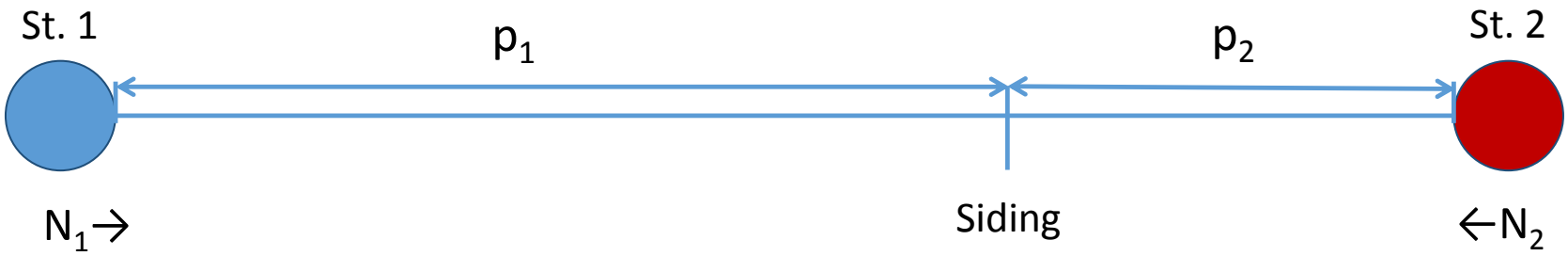
Single track railway scheduling problem with a siding

What is the siding?

Main track

Additional track

Single track railway scheduling problem with a siding



Initial data

- One siding, capacity is one train.
- $|N_1| = n_1$, $|N_2| = n_2$, all trains have equal speed.
- Traversing times: p_1, p_2, $p_1 \geq p_2$.
- For each train i from station s, $i \in N_s$, $s \in \{1, 2\}$, due date d_s^i and cost coefficient w_s^i are given;
- Release times: $r_s^i = 0$, $i \in N_s$, $s \in \{1, 2\}$.

Denote the problem as STR2S (STR2 with a siding).

Single track railway scheduling problem with a siding

Schedule

We need to construct optimal schedule σ, i.e. to set for each train number i moving from station s, $i \in N_s$, $s \in \{1, 2\}$, it's departure time $S_s^i(\sigma)$, stop time in the siding $\tau_s^i(\sigma)$ and arrival time $C_s^i(\sigma)$.

Objective function

Minimizing maximum lateness

$$L_{max} = \max_{i \in N_s, s \in \{1,2\}} \{L_s^i\},$$

where

$$L_s^i = C_s^i - d_s^i,$$

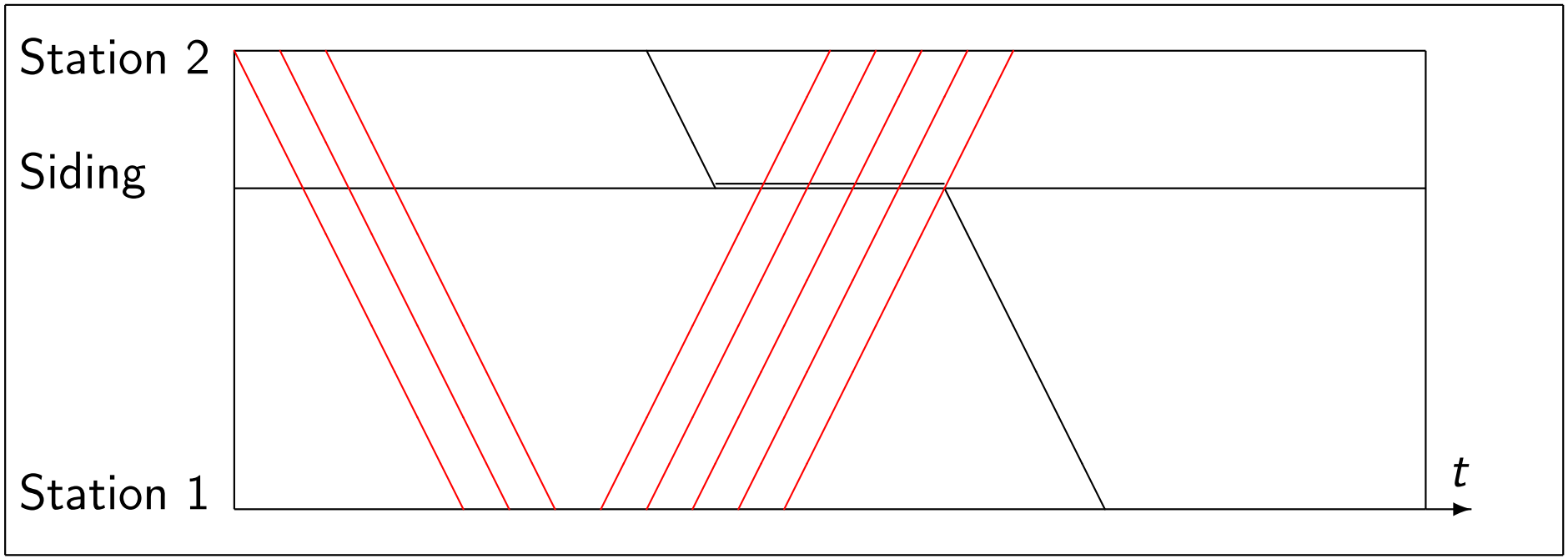
and weighted sum of arrival moments

$$\sum w_j C_j = \sum_{i \in N_s,\ s \in \{1,2\}} w_s^i C_s^i.$$

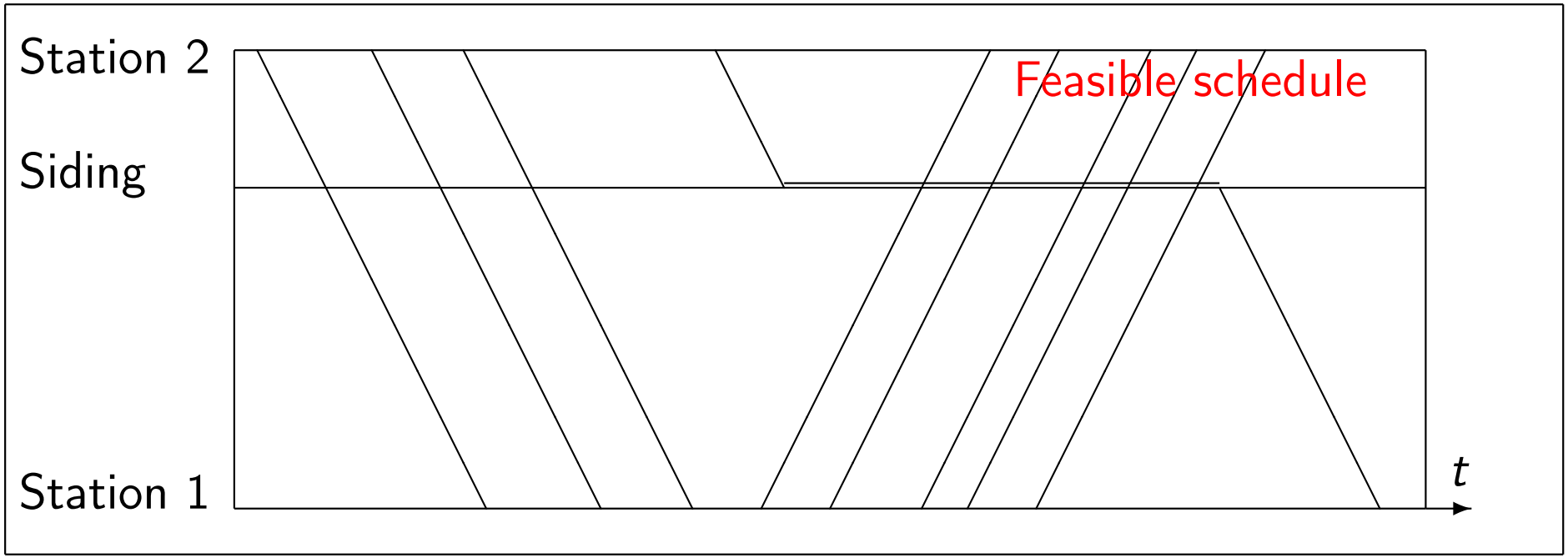
Schedule properties for presented model

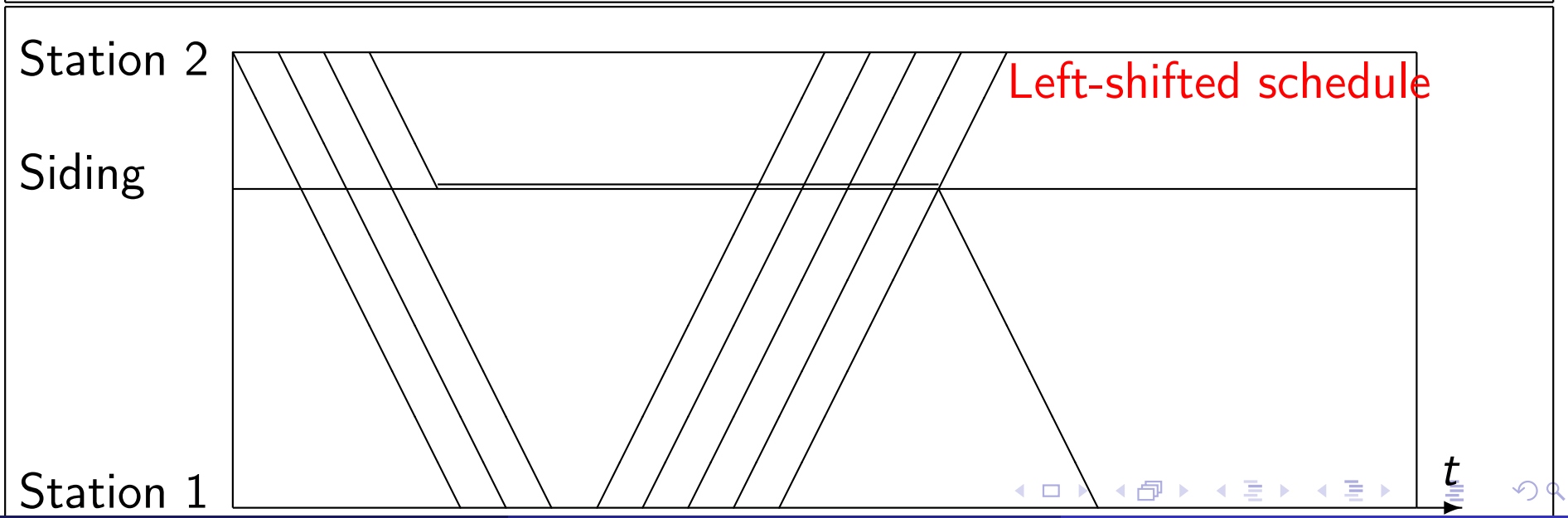
Express

Express is the train i moving from station s, $i \in N_s$, $s \in \{1, 2\}$, if it doesn't stop in the siding, i.e. $\tau_s^i = 0$.

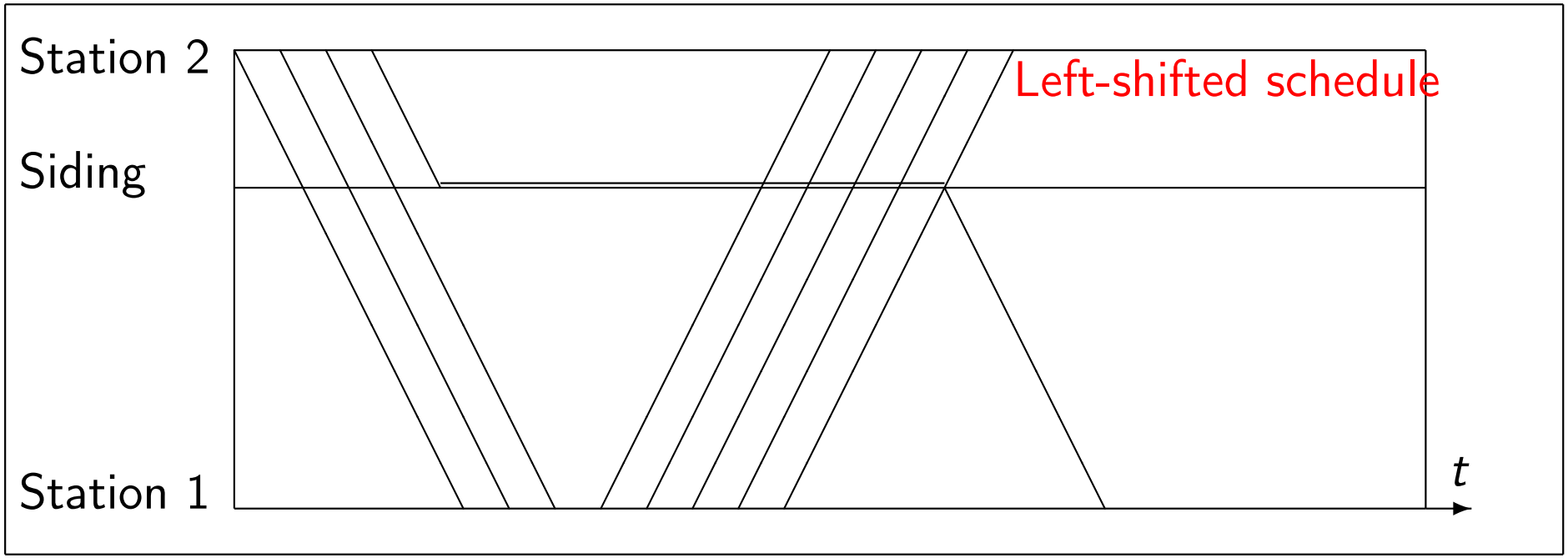


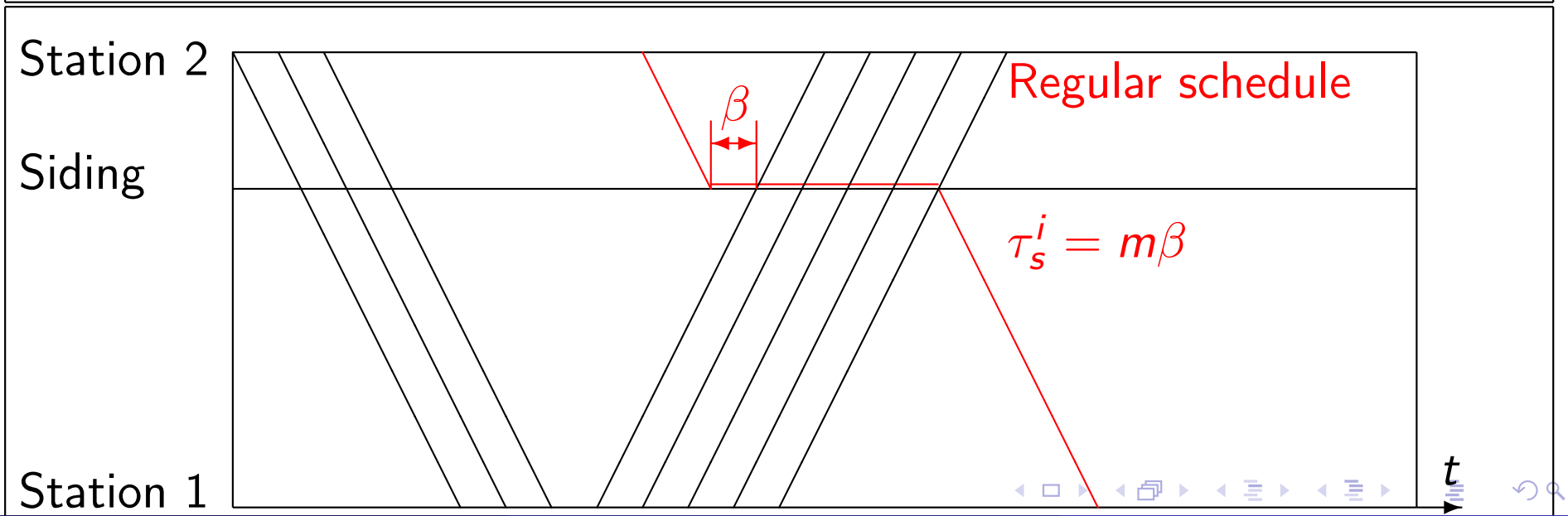
Schedule properties for presented model



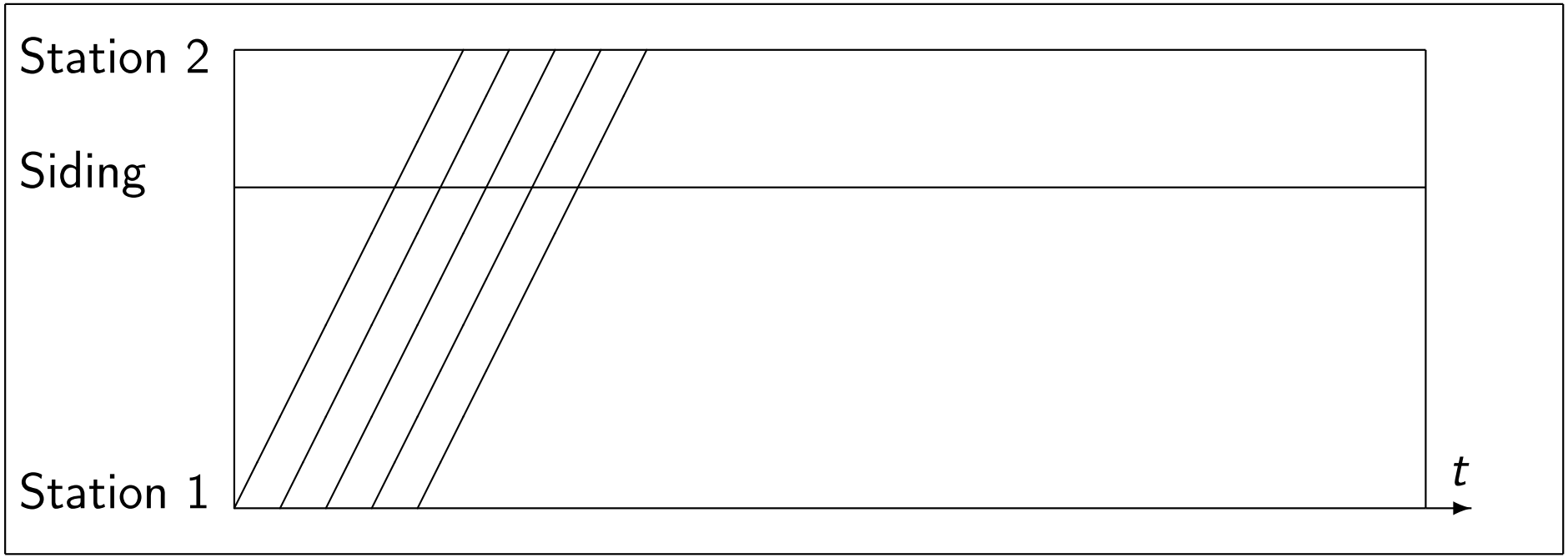


Schedule properties for presented model



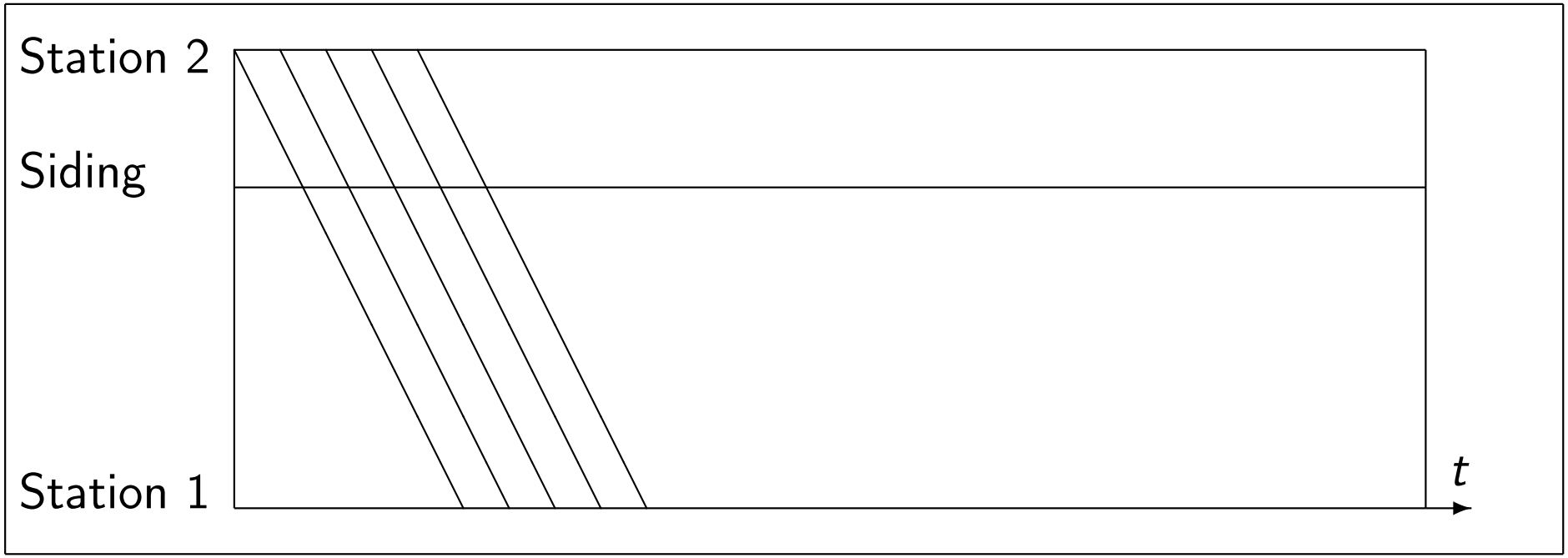


States



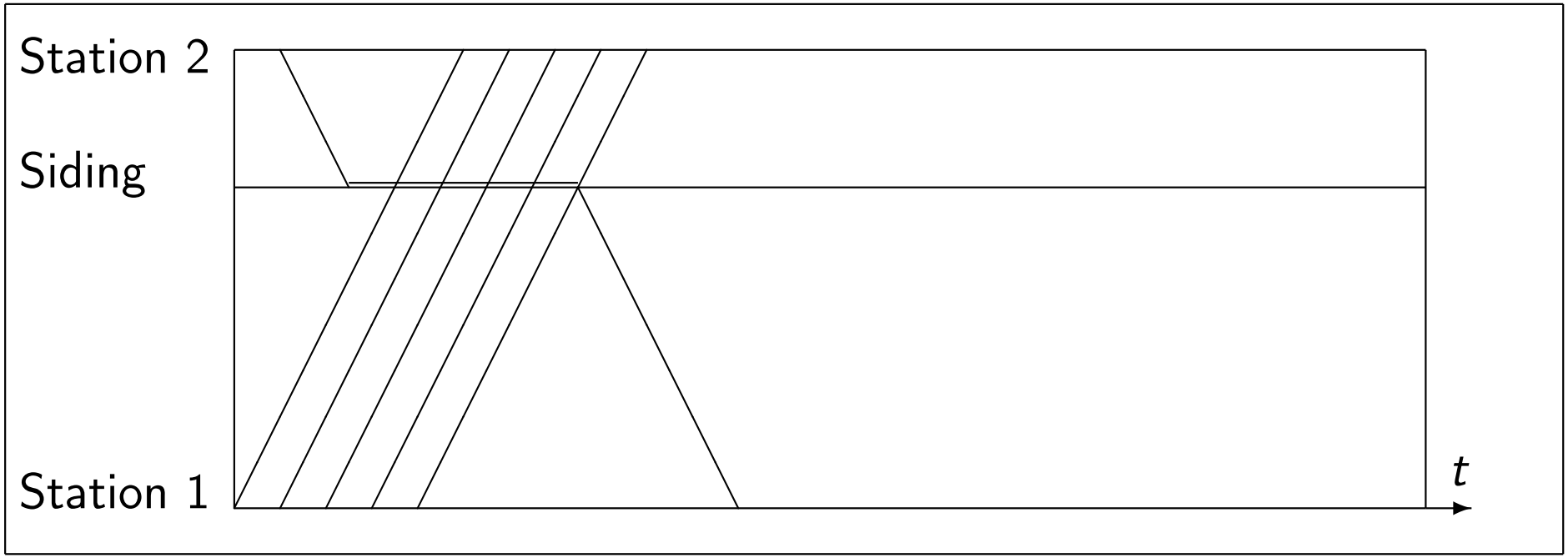
1) Batch moving from station 1 with empty siding.

States



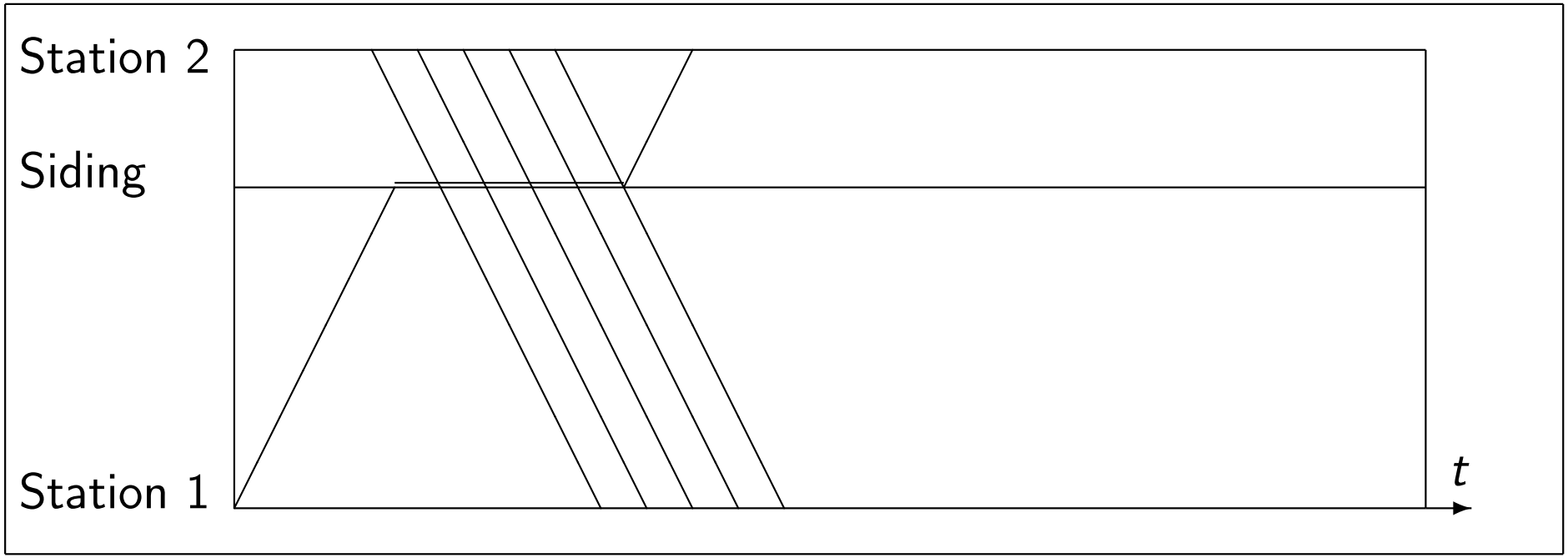
2) Batch moving from station 2 with empty siding.

States



3) Batch moving from station 1 with occupied siding.

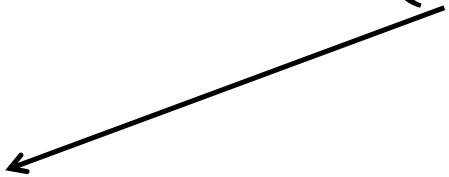
States



4) Batch moving from station 2 with occupied siding.

States

Express state $(\boldsymbol{s}, \boldsymbol{b})$



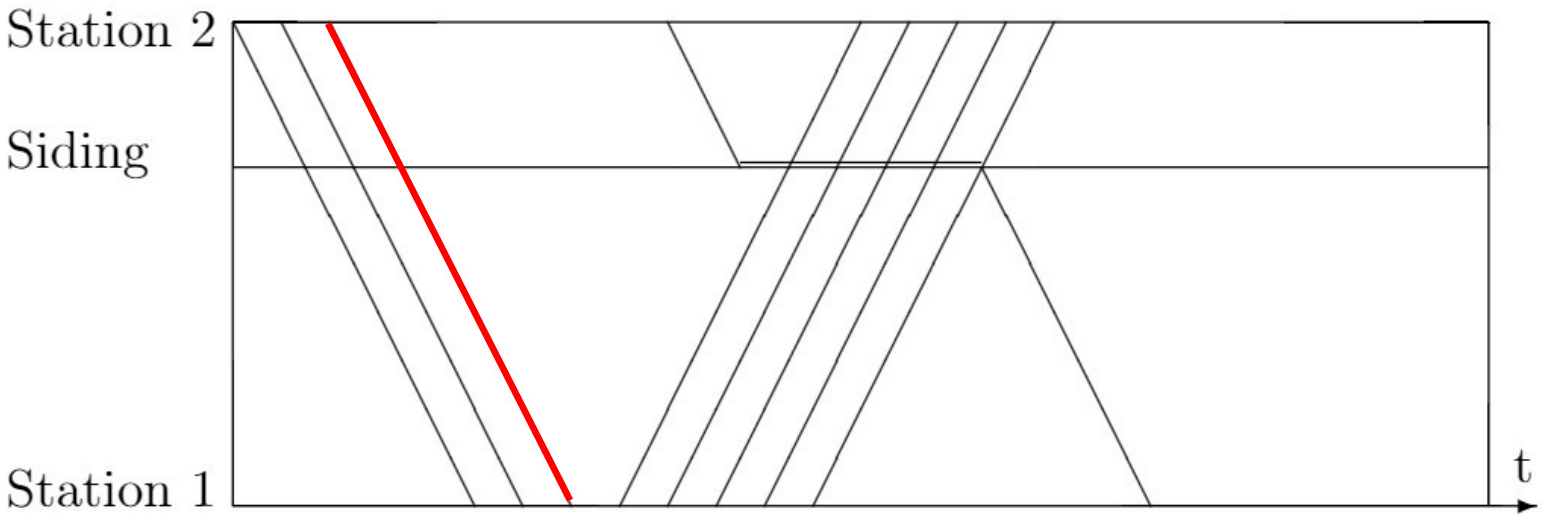
express departure station,
$s \in \{1,2\}$

States

Express state (s, b)

express departure station,
$s \in \{1,2\}$

«0»

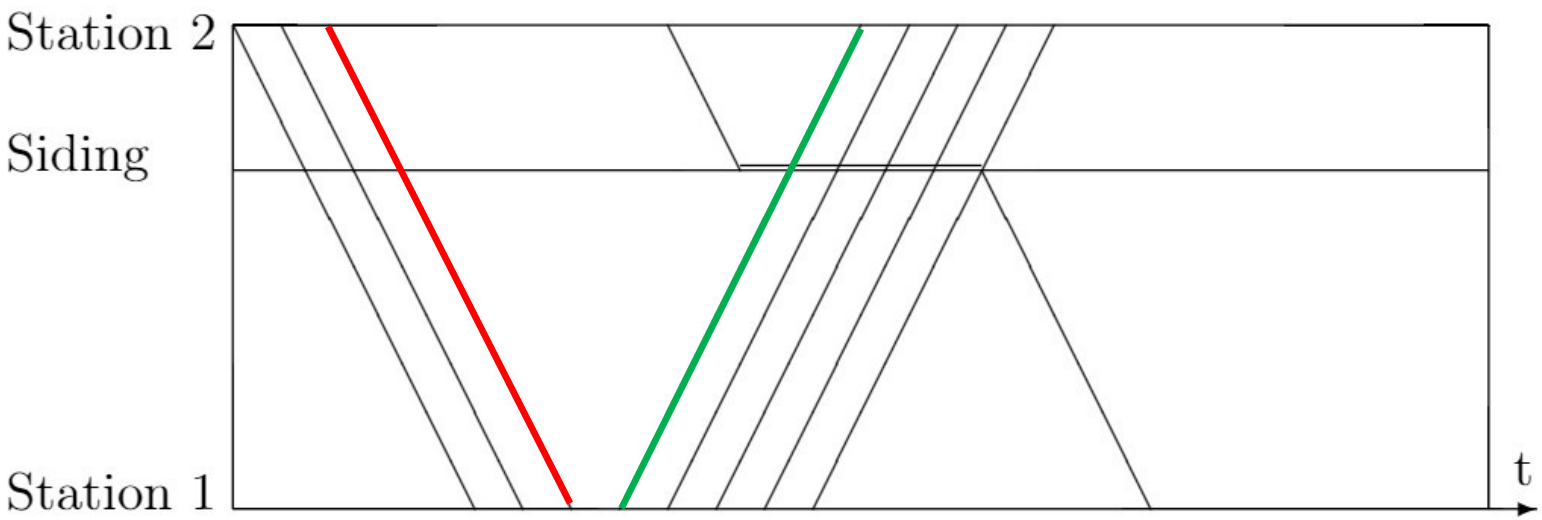


States

Express state (s, b)

express departure station,
$s \in \{1,2\}$

«0» «1»

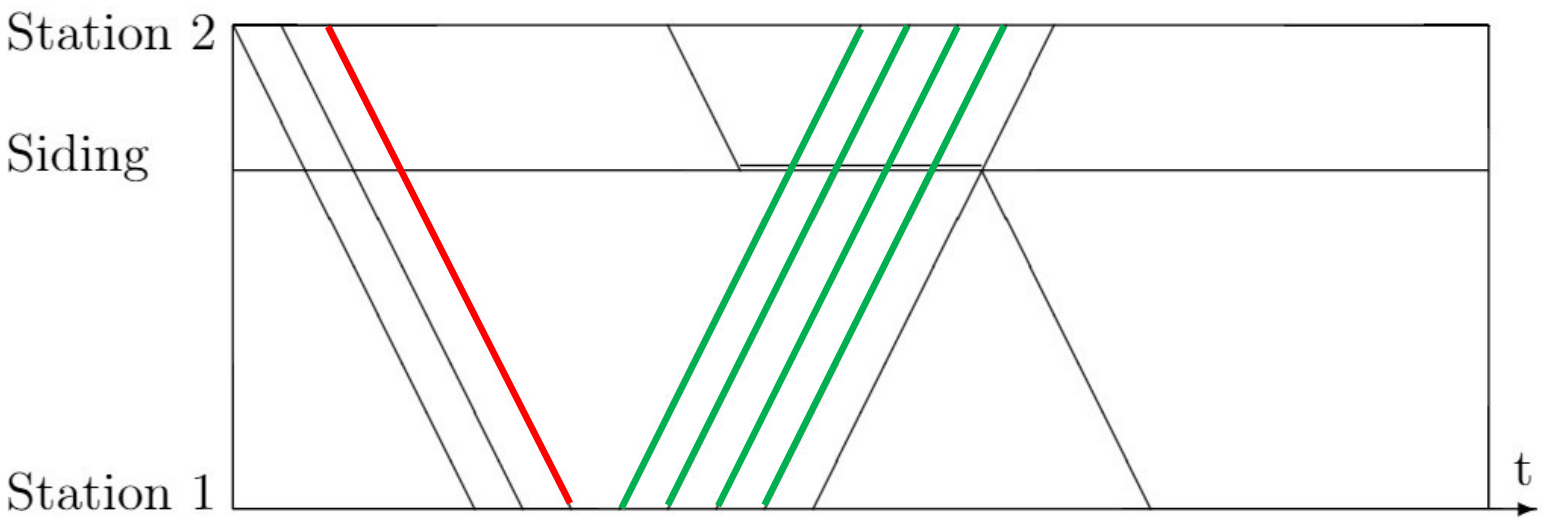


States

Express state (s, b)

express departure station,
$s \in \{1,2\}$

«0» «1»...«1»

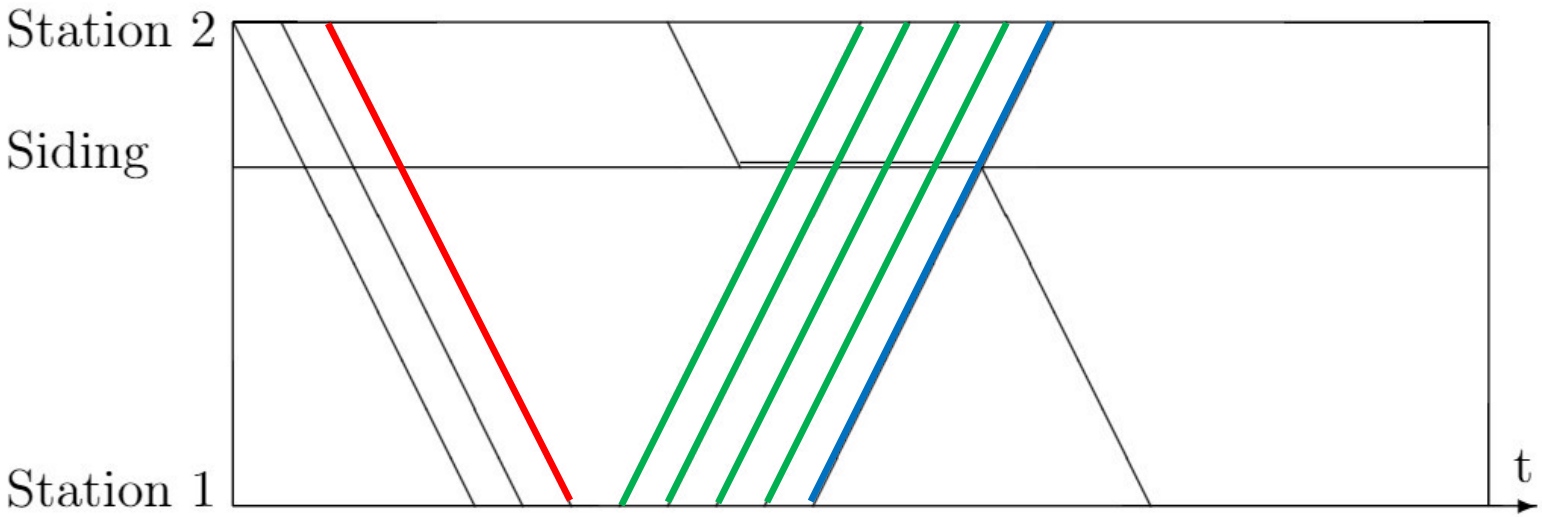


States

Express state (s, b)

express departure station,
$s \in \{1,2\}$

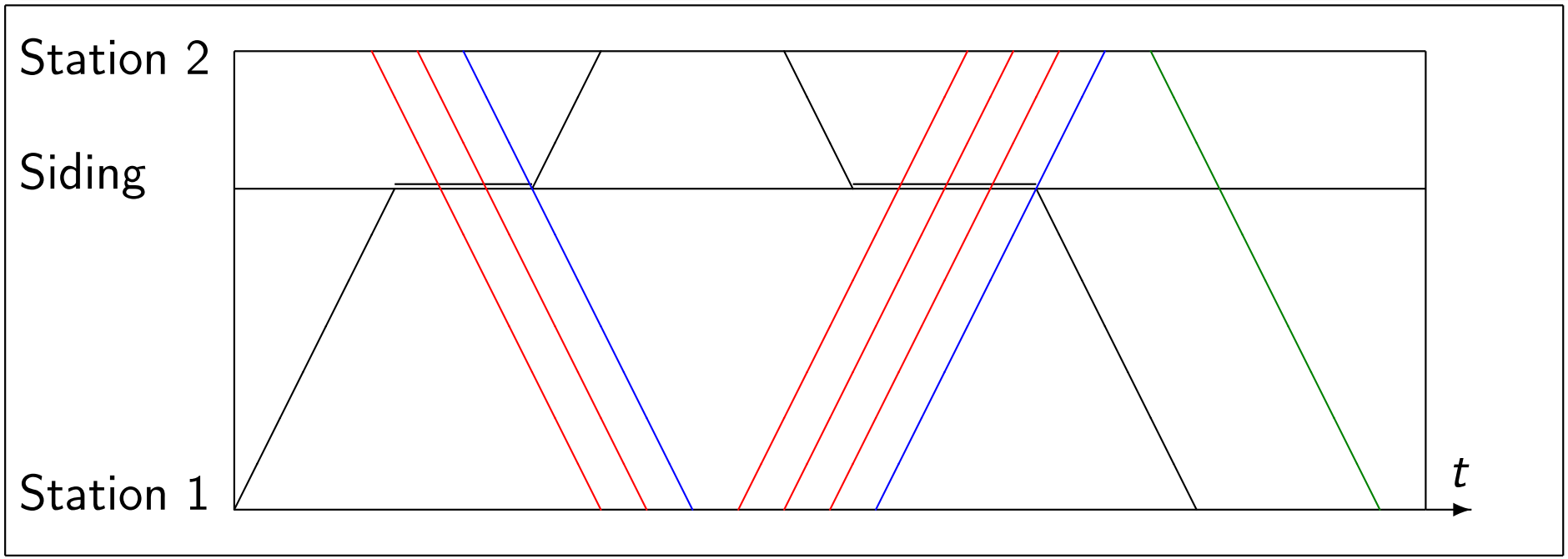
«0» «1»...«1»«2»



Regular schedule and expresses states sequences

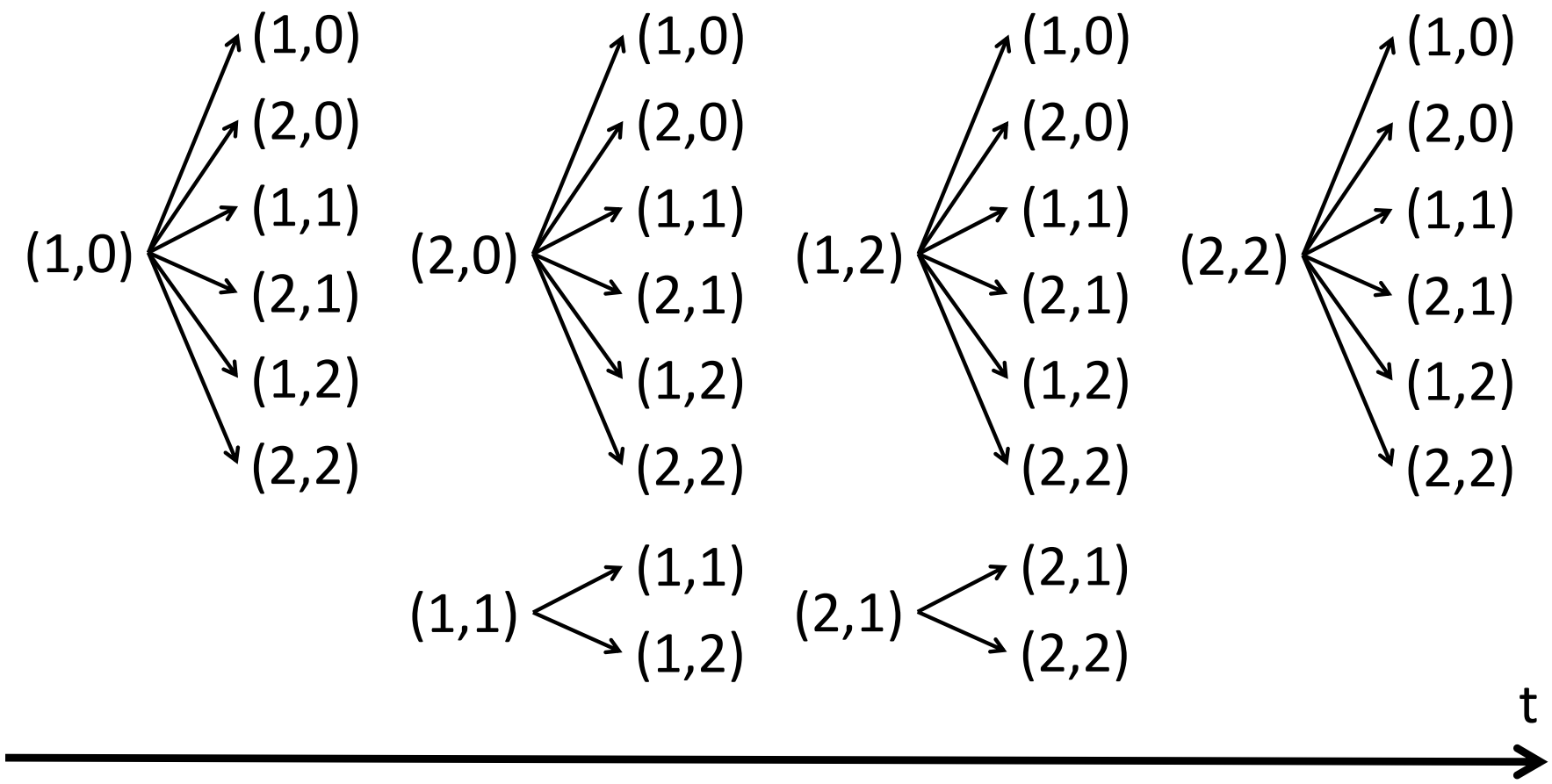
Theorem 1.

For each regular schedule there exists one and only one sequence of expresses states.

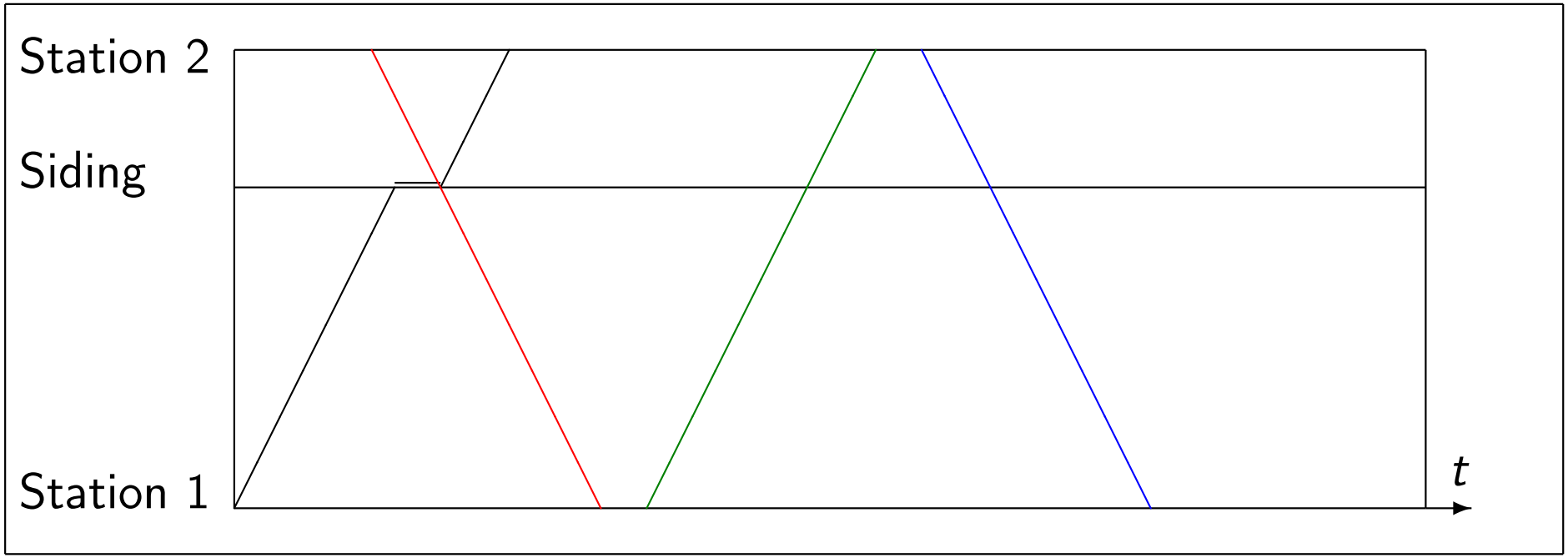


(2,1) (2,1) (2,2) (1,1) (1,1) (1,1) (1,2) (2,0)

States

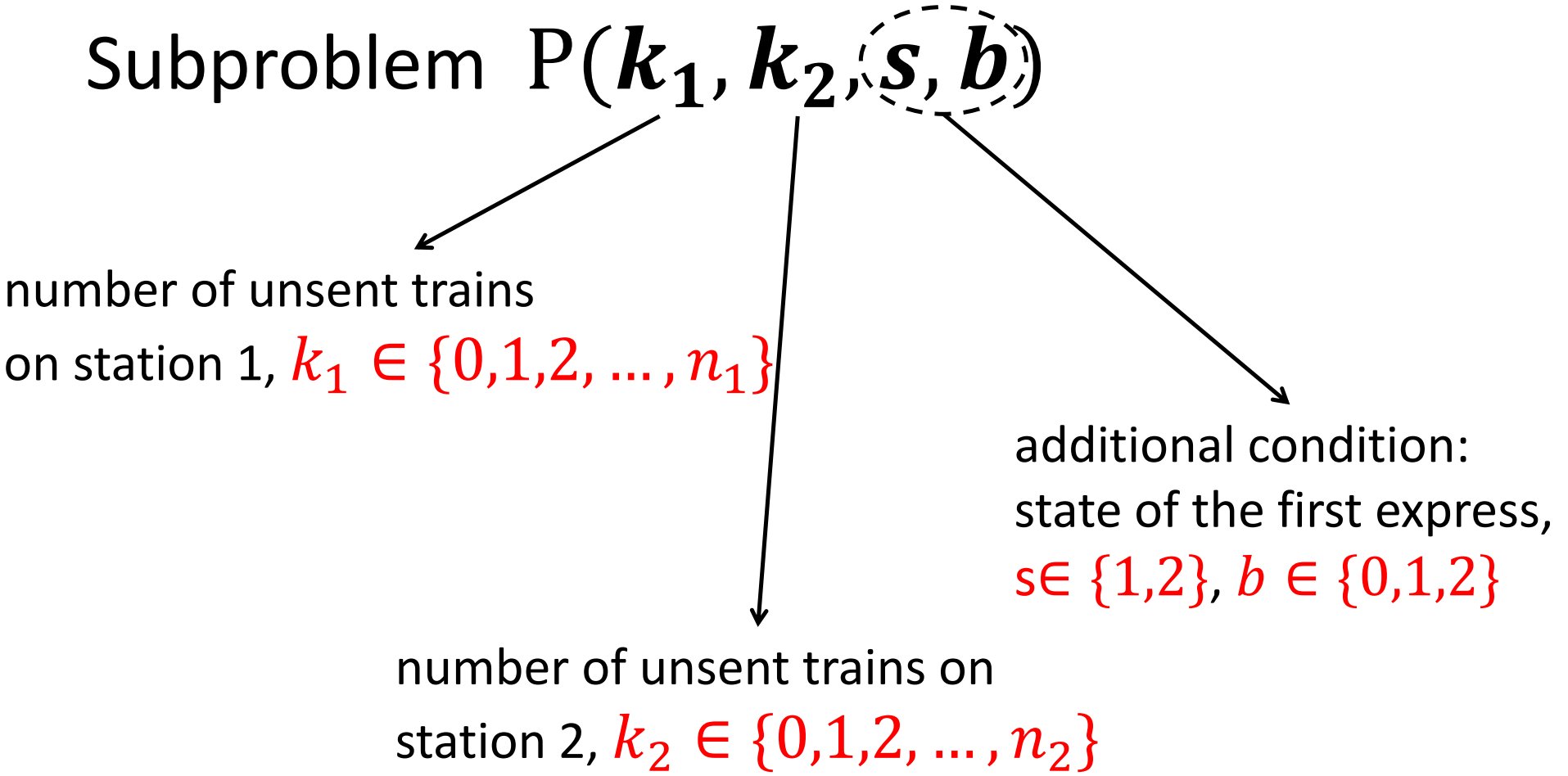

(1,0)
(1,0)
(2,0)
(1,1)
(2,1)
(1,2)
(2,2)
(2,0)
(1,0)
(2,0)
(1,1)
(2,1)
(1,2)
(2,2)
(1,1)
(1,1)
(1,2)
(1,2)
(1,0)
(2,0)
(1,1)
(2,1)
(1,2)
(2,2)
(2,1)
(2,1)
(2,2)
(2,2)
(1,0)
(2,0)
(1,1)
(2,1)
(1,2)
(2,2)
t

States



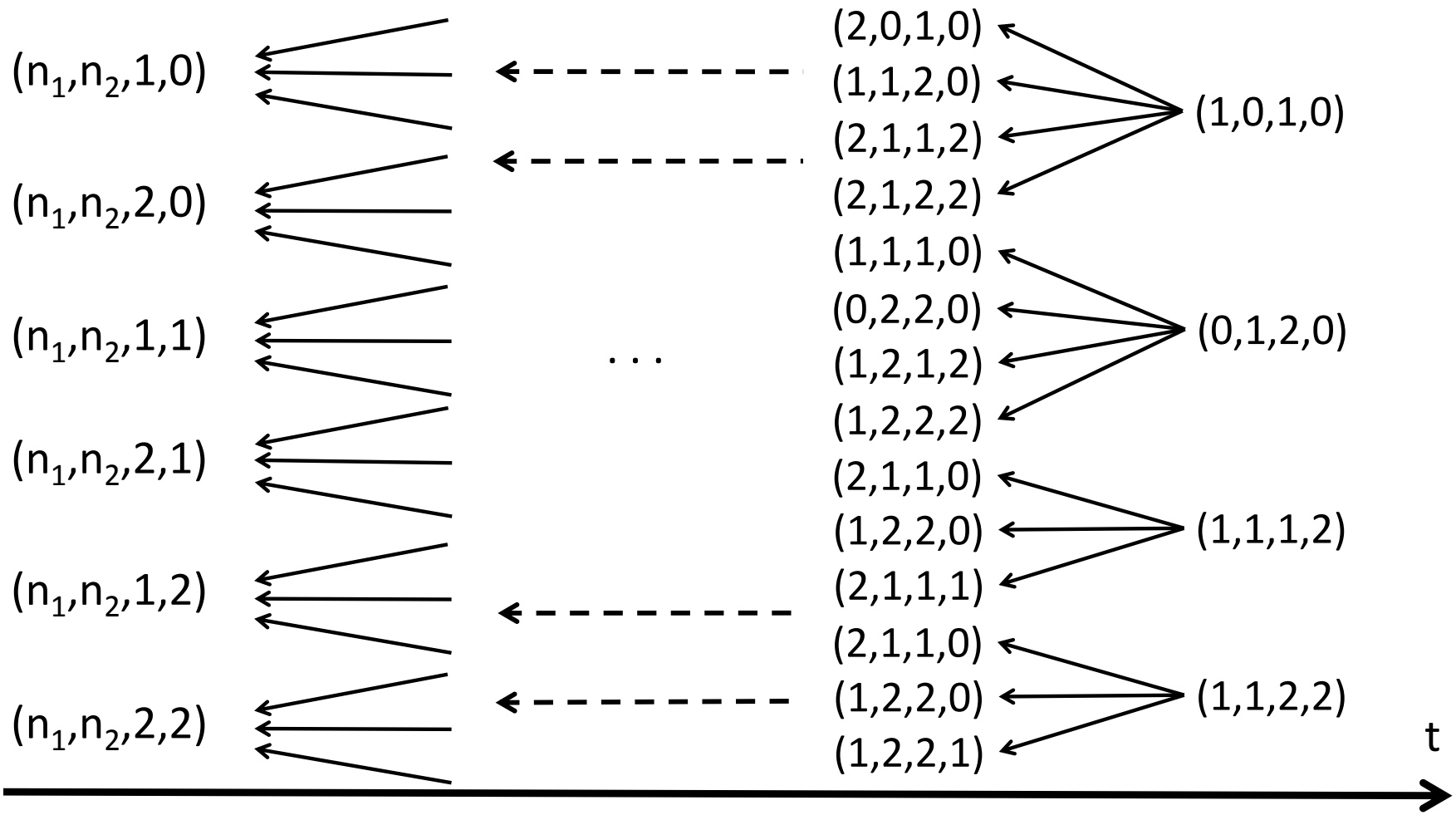
(2,2) (1,0) (2,0)

Solution algorithm



Number of different subproblems – $O(n^2)$

Solution algorithm



Solution algorithm

Initial values

$$F(1,0,1,0)) = p_1 + p_2 - d_1^1;$$

$$F(0,1,2,0)) = p_1 + p_2 - d_2^1;$$

$$F(1,1,1,2) = \max \begin{cases} 2p_1 - d_2^1; \\ p_2 + p_1 - d_1^1; \end{cases}$$

$$F(1,1,2,2) = \max \begin{cases} 2p_2 - d_1^1; \\ p_2 + p_1 - d_2^1. \end{cases}$$

Exclusion of impossible subtasks

- $F(0, k_2, 1, 0) = \infty$;
- $F(k_1, 0, 2, 0) = \infty$;
- $F(k_1, k_2, s, b) = \infty$ if $k_1 = 0$ or $k_2 = 0$, where $(s, b) \notin \{(1,0),(2,0)\}$.

Solution algorithm

Bellman equation

Optimal objective function value in the subproblem $P(k_1, k_2, s, b)$

$$F(k_1, k_2, s, b) = \min_{(k_1', k_2', s', b') \in T(k_1, k_2, s, b)} \max \begin{cases} H(k_1, k_2, s, b); \\ F(k_1', k_2', s', b') + g((s, b), (s', b')) \end{cases}$$

Objective function value of express in state (s, b) and skipping train

$$H(k_1, k_2, s, b) = \begin{cases} \max\{p_1 + p_2 - d_s^{k_s}; 2p_s - d_{\bar{s}}^{k_{\bar{s}}}\}, & \text{if } b = 2, \\ p_1 + p_2 - d_s^{k_s} & \text{otherwise.} \end{cases}$$

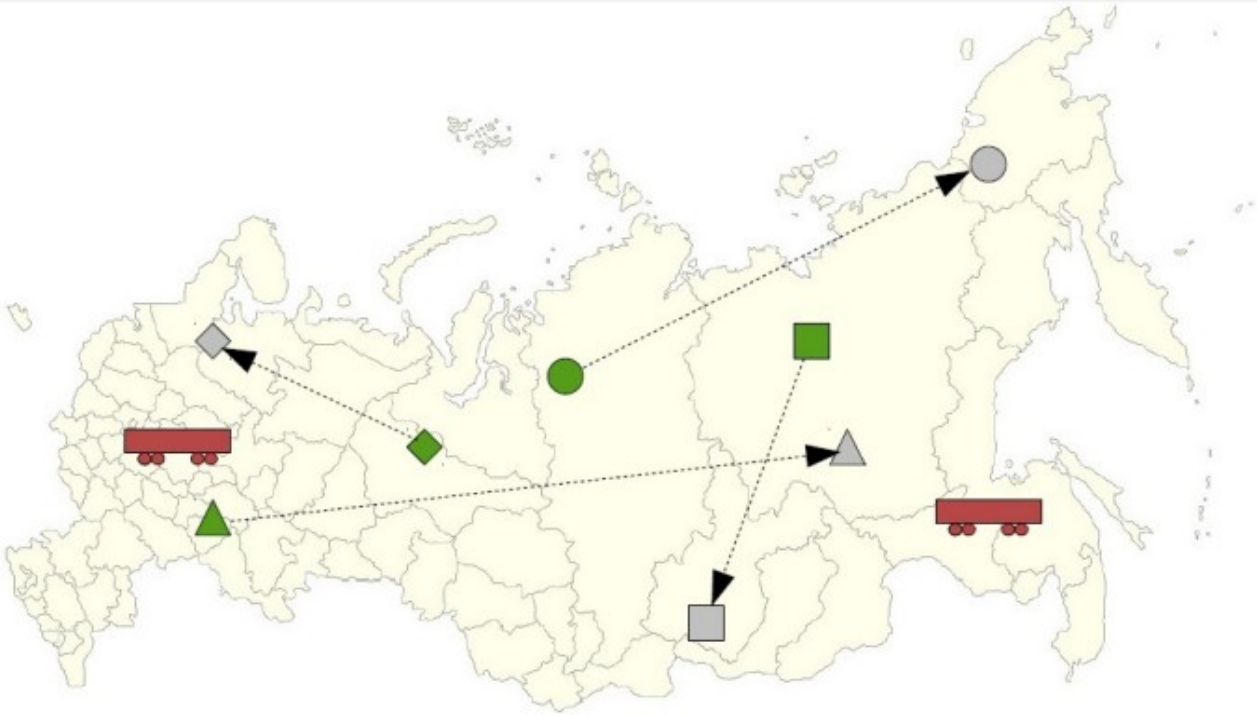
Results for STR2S

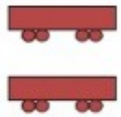
- Exact solution algorithm based on the dynamical programming method was proposed for the described problem.
- Presented algorithm allows to construct set of optimal schedules in $O(n^2)$ operations.

Algorithm for $\sum w_j C_j$

For objective function $\sum w_j C_j$ algorithm is the same, some operations and variables changes.

The freight car routing problem: overview

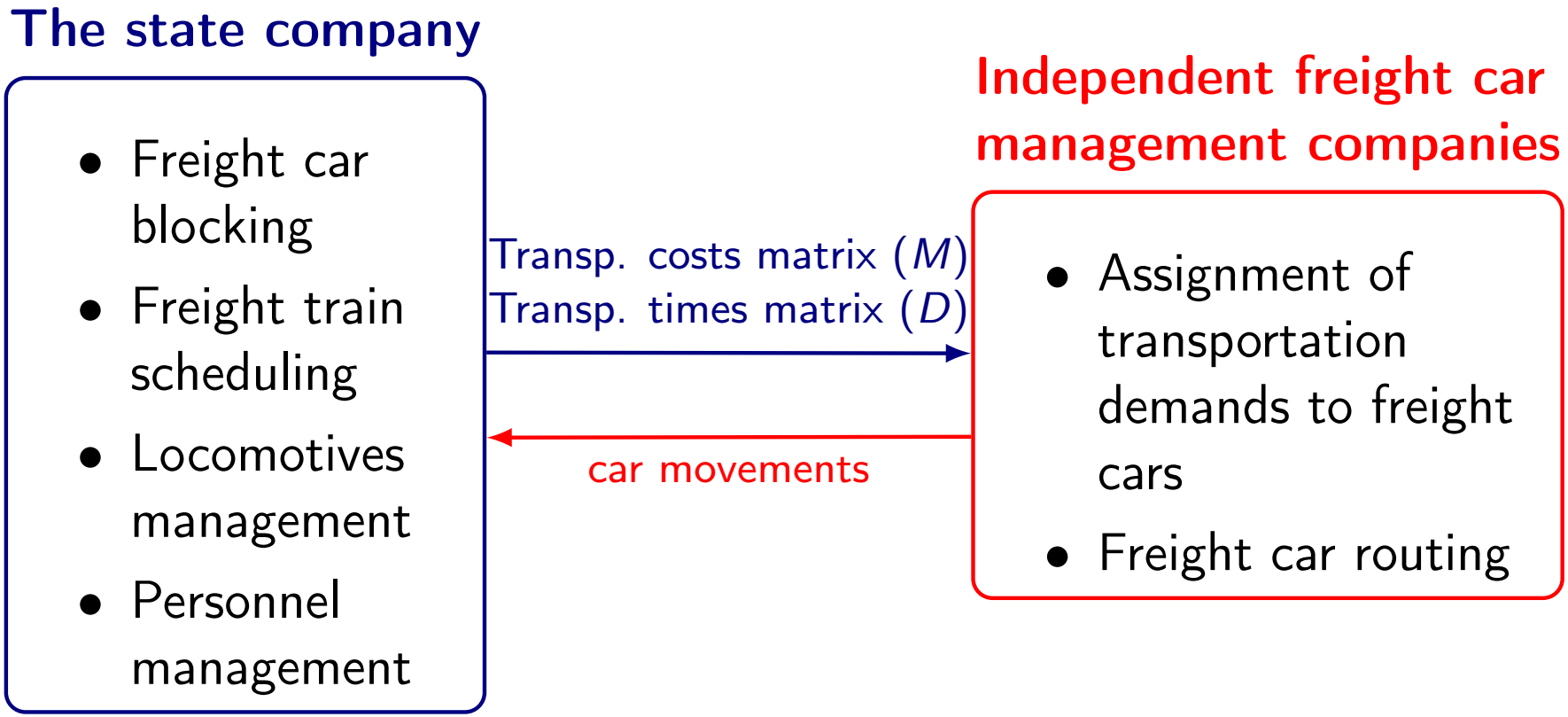




initial car distribution

transportation demands

Specificity of freight rail transportation in Russia



Distances are large, and average freight train speed is low ($\approx$ 300 km/day): discretization in periods of 1 day is reasonable

The freight car routing problem: input and output

Input

- Railroad network (stations)
- Initial locations of cars (sources)
- Transportation demands and associated profits
- Costs: transfer costs and standing (waiting) daily rates;

Output: operational plan

- A set of accepted demands and their execution dates
- Empty and loaded cars movements to meet the demands (car routing)

Objective

Maximize the total net profit

Similar works in the literature

[Fukasawa, Poggi, Porto, Uchoa, ATMOS02]

- Train schedule is known
- Cars should be assigned to trains to be transported
- Discretization by the moments of arrival and departure of trains.
- Smaller time horizon (7 days)

Other works

- [Holmberg, Joborn, Lundren, TS98]
- [Löbel, MS98]
- [Campetella, Lulli, Pietropaoli, Ricciardi, ATMOS06]
- [Caprara, Malaguti, Toth, TS11]

Data: overview

- T — planning horizon (set of time periods);
- I — set of stations;
- C — set of car types;
- K — set of product types;
- Q — set of demands;
- S — set of sources (initial car locations);
- M — empty transfer cost function;
- D — empty transfer duration function;

Demands data

For each order $q \in Q$

- origin and destination stations;
- product type
- set of car types, which can be used for this demand — $C_q \subseteq C$
- maximum (minimum) number of cars, needed to fulfill (partially) the demand — $n_q^{\max}(n_q^{\min})$
- time window for starting the transportation
- profit vector (for delivery of one car with the product), depends on the period on which the transportation is started
- transportation time of the demand
- daily standing rates charged for one car waiting before loading (after unloading) the product at origin (destination) station

Sources and car types data

For each source $s \in S$

- station where cars are located
- type of cars
- period, starting from which cars can be used
- daily standing rate charged for cars
- type of the latest delivered product
- number of cars in the source — $\vec{n_s} \in \mathbb{N}$

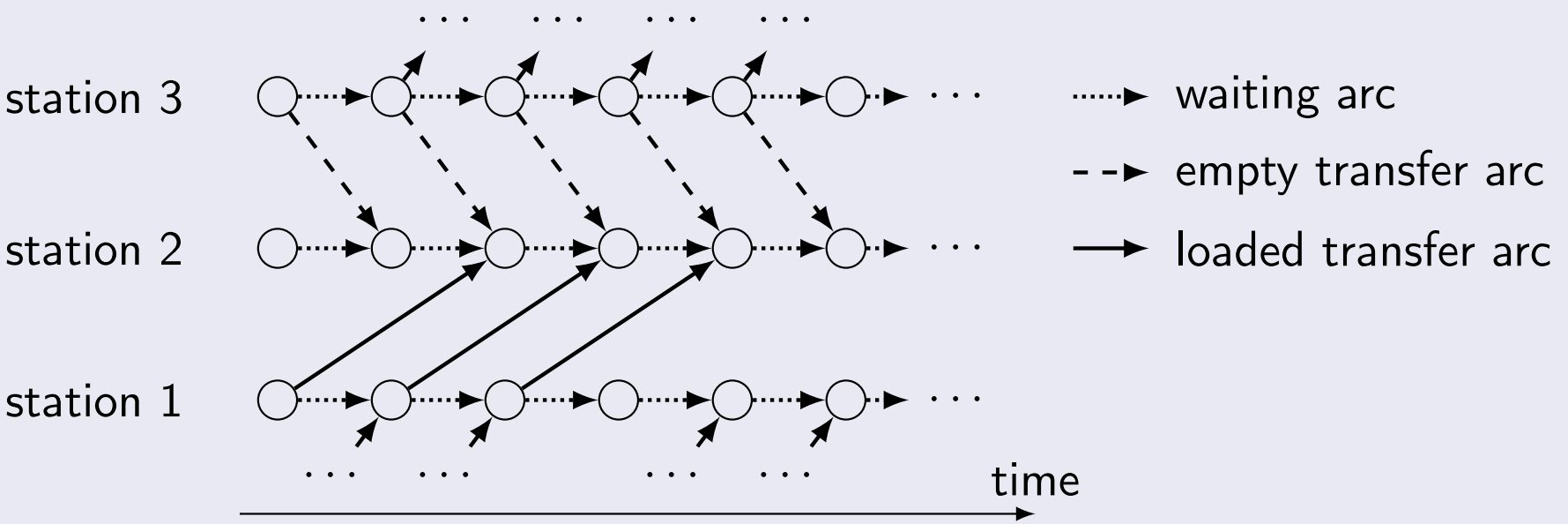
For each car type $c \in C$

- Q_c — set of demands, which a car of type c can fulfill
- S_c — set of sources for car type c

Commodity graph

Commodity $c \in C$ represents the flow (movements) of cars of type c.

Graph $G_c = (V_c, A_c)$ for commodity $c \in C$:



Each vertex $v \in V_c$ represent location of cars of type c on a certain station at a certain time standing at a certain rate

g_a — cost of arc $a \in A_c$

Multi-commodity flow formulation

Variables

- $x_a \in \mathbb{Z}_+$ — flow size along arc $a \in A_c$, $c \in C$
- $y_q \in \{0, 1\}$ — demand $q \in Q$ is accepted or not

$$\min \sum_{c \in C} \sum_{a \in A_c} g_a x_a$$

$$\sum_{c \in C_q} \sum_{a \in A_{cq}} x_a \leq n_q^{\max} y_q \quad \forall q \in Q$$

$$\sum_{c \in C_q} \sum_{a \in A_{cq}} x_a \geq n_q^{\min} y_q \quad \forall q \in Q$$

$$\sum_{a \in \delta^-(v)} x_a - \sum_{a \in \delta^+(v)} x_a = \vec{n}_v \quad \forall c \in C, v \in V_c$$

$$x_a \in \mathbb{Z}_+ \quad \forall c \in C, a \in V_c$$

$$y_q \in \{0, 1\} \quad \forall q \in Q$$

We concentrate on solving its LP-relaxation

Multi-commodity flow formulation

Variables

- $x_a \in \mathbb{Z}_+$ — flow size along arc $a \in A_c$, $c \in C$
- $y_q \in \{0, 1\}$ — demand $q \in Q$ is accepted or not

$$\min \sum_{c \in C} \sum_{a \in A_c} g_a x_a$$

$$\sum_{c \in C_q} \sum_{a \in A_{cq}} x_a \leq n_q^{\max} y_q \quad \forall q \in Q$$

$$\sum_{c \in C_q} \sum_{a \in A_{cq}} x_a \geq n_q^{\min} y_q \quad \forall q \in Q$$

$$\sum_{a \in \delta^-(v)} x_a - \sum_{a \in \delta^+(v)} x_a = \vec{n}_v \quad \forall c \in C, v \in V_c$$

$$x_a \in \mathbb{Z}_+ \quad \forall c \in C, a \in V_c$$

$$y_q \in \{0, 1\} \quad \forall q \in Q$$

We concentrate on solving its LP-relaxation

Multi-commodity flow formulation

Variables

- $x_a \in \mathbb{Z}_+$ — flow size along arc $a \in A_c$, $c \in C$
- $y_q \in \{0, 1\}$ — demand $q \in Q$ is accepted or not

$$\min \sum_{c \in C} \sum_{a \in A_c} g_a x_a$$

$$\sum_{c \in C_q} \sum_{a \in A_{cq}} x_a \leq n_q^{\max} y_q \quad \forall q \in Q$$

$$\sum_{c \in C_q} \sum_{a \in A_{cq}} x_a \geq n_q^{\min} y_q \quad \forall q \in Q$$

$$\sum_{a \in \delta^-(v)} x_a - \sum_{a \in \delta^+(v)} x_a = \vec{n}_v \quad \forall c \in C, v \in V_c$$

$$0 \leq x_a \quad \forall c \in C, a \in V_c$$

$$0 \leq y_q \leq 1 \quad \forall q \in Q$$

We concentrate on solving its LP-relaxation

Path reformulation

- P_s — set of paths (car routes) from source $s \in S$

Variables

- $\lambda_s \in \mathbb{Z}_+$ — flow size along path $p \in P_s$, $s \in S$

$$\min \sum_{c \in C} \sum_{s \in S_c} \sum_{p \in P_s} g_p^{path} \lambda_p$$

$$\sum_{c \in C_q} \sum_{s \in S_c} \sum_{p \in P_s:\ q \in Q_p^{path}} \lambda_a \le n_q^{\max} y_q \quad \forall q \in Q$$

$$\sum_{c \in C_q} \sum_{s \in S_c} \sum_{p \in P_s:\ q \in Q_p^{path}} \lambda_a \ge n_q^{\min} y_q \quad \forall q \in Q$$

$$\sum_{p \in P_s} \lambda_p = \vec{n}_s \qquad \forall c \in C, s \in S_c$$

$$\lambda_p \in \mathbb{Z}_+ \qquad \forall c \in C, s \in S_c, p \in P_s$$

$$y_q \in \{0, 1\} \qquad \forall q \in Q$$

Column generation for path reformulation

- Pricing problem decomposes to shortest path problems, one for each source
  - slow: number of sources are thousands
- To accelerate, for each commodity $c \in C$, we search for a shortest path in-tree to the terminal vertex from all sources in S_c
  - drawback: some demands are severely "overcovered", bad convergence
- We developed iterative procedure which removes covered demands and cars assigned to them, and the repeats search for a shortest path in-tree

Flow enumeration reformulation

- F_c — set of fixed flows for commodity $c \in C$

Variables

- $\omega_f \in \{0, 1\}$ — commodity c is routed accordity to flow $f \in F_c$ or not

$$\min \sum_{c \in C} \sum_{f \in F_s} g_f^{flow} \omega_f$$

$$\sum_{c \in C_q} \sum_{f \in F_c} \sum_{a \in A_{cq}} f_a \omega_f \leq n_q^{\max} y_q \quad \forall q \in Q$$

$$\sum_{c \in C_q} \sum_{f \in F_c} \sum_{a \in A_{cq}} f_a \omega_f \geq n_q^{\min} y_q \quad \forall q \in Q$$

$$\sum_{f \in F_c} \omega_f = 1 \quad \forall c \in C$$

$$\omega_p \in \{0, 1\} \quad \forall c \in C, f \in F_c$$

$$y_q \in \{0, 1\} \quad \forall q \in Q$$

Approach CGEF

- Pricing problem decomposes to minimum cost flow problems, one for each commodity
  - slow: very bad convergence
- "Column generation for extended formulations" (CGEF) approach: we disaggregate the pricing problem solution to arc flow variables, which are added to the master.
- The master then becomes the multi-commodity flow formulation with restricter number of arc flow variables, i.e. "improving" variables are generated dynamically

Proposition

If an arc flow variable x has a negative reduced cost, there exists a negative reduced cost pricing problem solution in which $x > 0$.
(consequence of the theorem by S. and Vanderbeck)

Tested approaches

- DIRECT: solution of the multi-commodity flow formulation by the *Clp* LP solver
  - Problem specific solver source code modifications
  - Problem specific preprocessing is applied (not public)
  - Tested inside the company

- COLGEN: solution of the path reformulation by column generation (*BaPCod* library and *Cplex* LP solver)
  - Initialization of the master by "doing nothing" routes
  - Stabilization by dual prices smoothing
  - Restricted master clean-up

- COLGENEF: "dynamic" solution of multi-commodity flow formulation by the CGEF approach (*BaPCod* library, *Lemon* min-cost flow solver and *Cplex* LP solver)
  - Initialization of the master by all waiting arcs
  - Only trivial preprocessing is applied

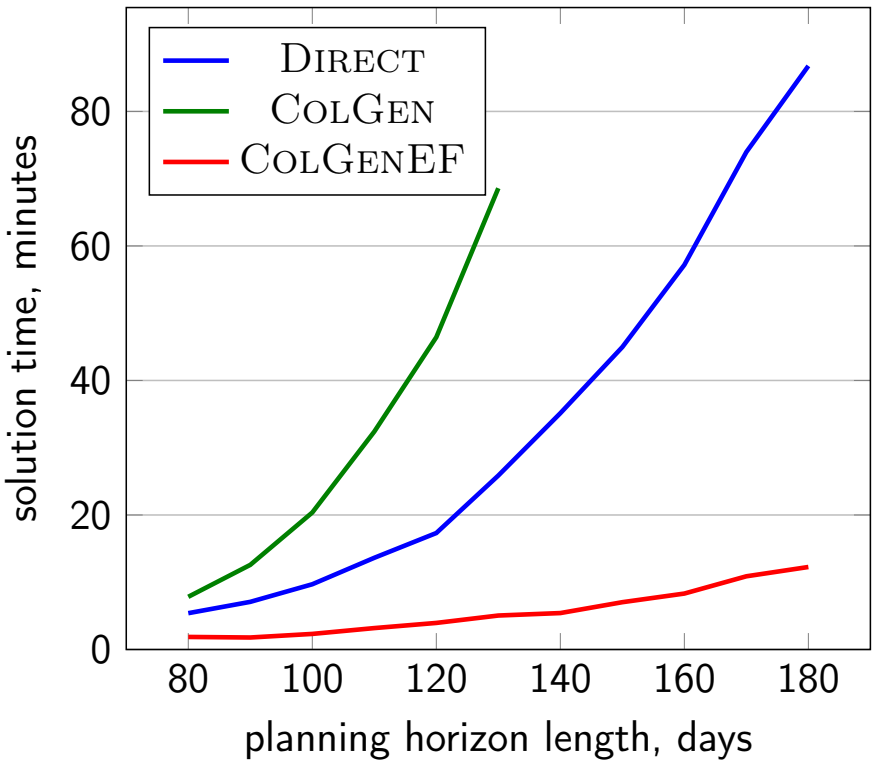
First test set of real-life instances

Instance name	x3	x3double	5k0711q
Number of stations	371	371	1'900
Number of demands	1'684	3'368	7'424
Number of car types	17	17	1
Number of cars	1'013	1'013	15'008
Number of sources	791	791	11'215
Time horizon, days	37	74	35
Number of vertices, thousands	62	152	22
Number of arcs, thousands	794	2'846	1'843
Solution time for DIRECT	20s	1h34m	55s
Solution time for COLGEN	22s	7m53s	8m59s
Solution time for COLGENEF	3m55s	>2h	43s

Real-life instances with larger planning horizon

1'025 stations, up to 6'800 demands, 11 car types, 12'651 cars, and 8'232 sources.

Up to $\approx$ 300 thousands nodes and 10 millions arcs.



Horizon	DIRECT	COLGENEF
80	5m24s	1m52s
90	7m05s	1m47s
100	9m42s	2m19s
110	13m38s	3m11s
120	17m19s	3m57s
130	25m52s	5m03s
140	35m08s	5m25s
150	44m58s	7m02s
160	57m11s	8m19s
170	1h13m58s	10m53s
180	1h26m46s	12m16s

Convergence of COLGENEF in less than 15 iterations.

About 3% of arc flow variables at the last iteration.

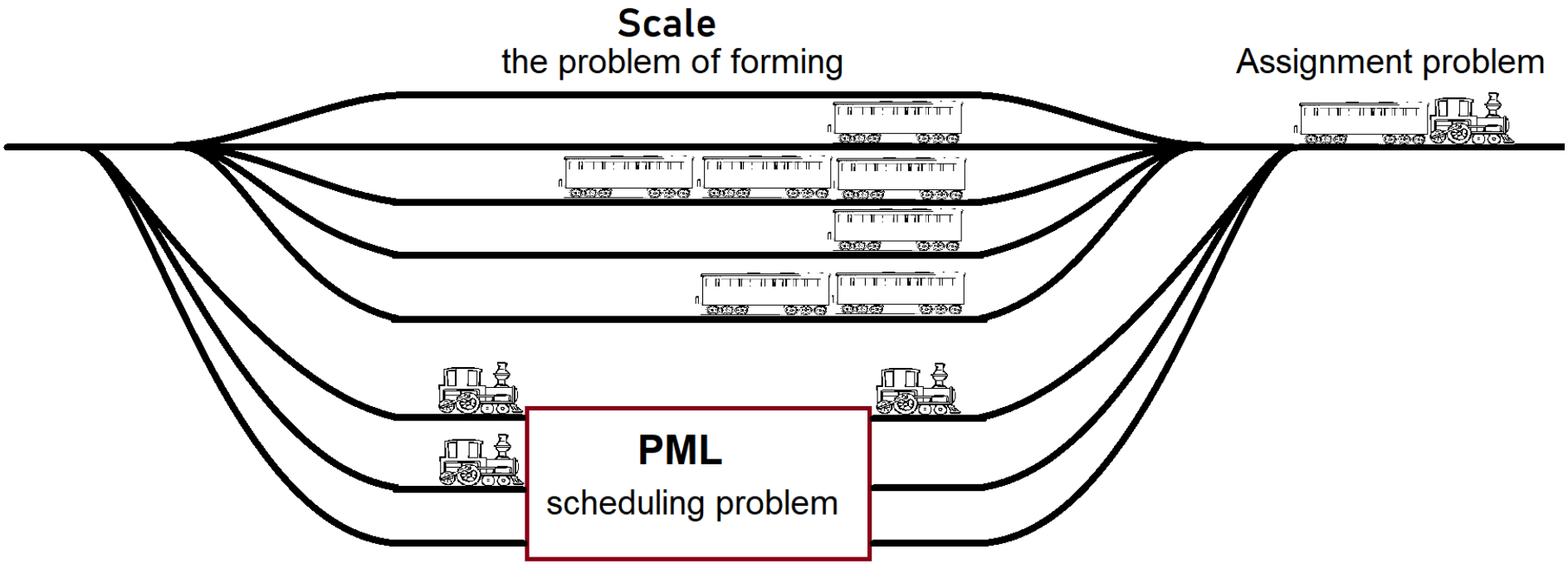
Conclusions

- Three approaches tested for a freight car routing problem on real-life instances
- Approach ColGen is the best for instances with small number of sources
- Problem-specific preprocessing is important: good results for Direct
- Approach ColGenEF is the best for large instances
- Combination of ColGenEF and problem-specific preprocessing would allow to increase discretization and improve solutions quality

Problems of the marshalling yard

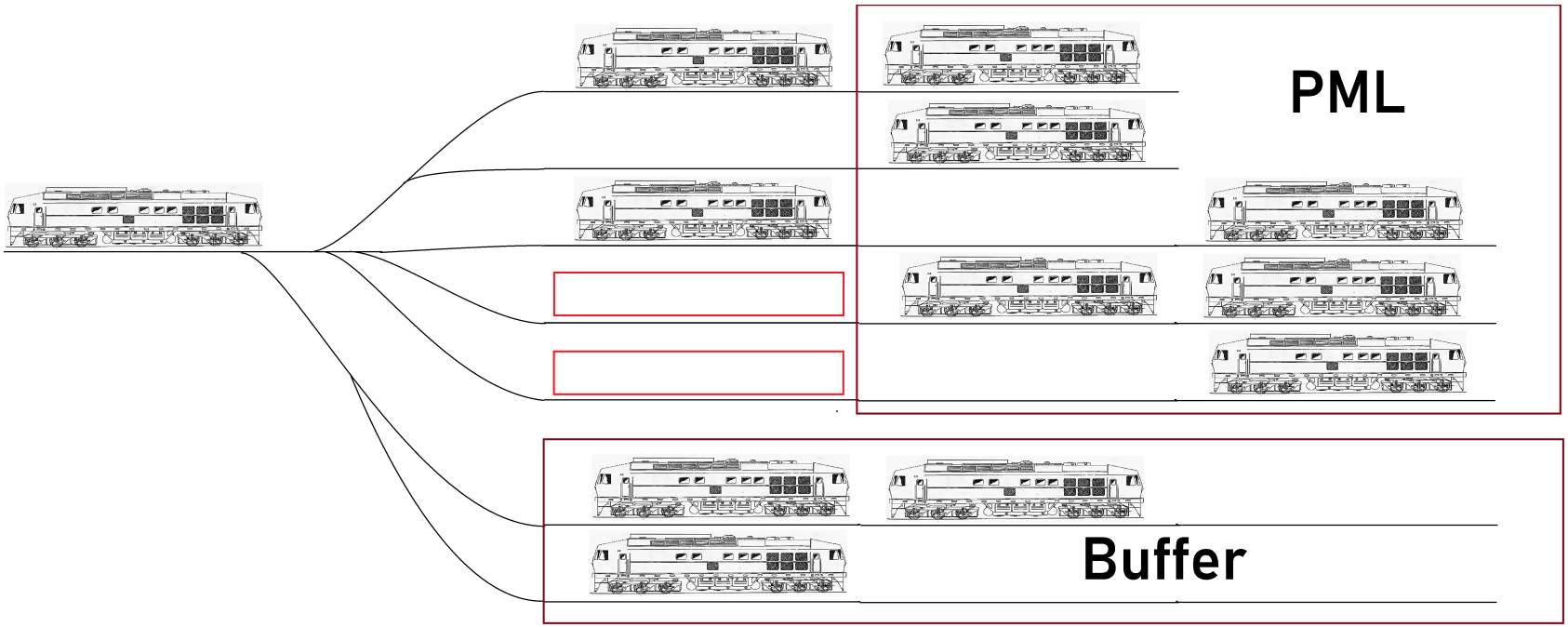
Three problems of the marshalling yard:

- trains must be disbanded and new ones formed;
- locomotives must undergo maintenance in the PML;
- each train must be assigned by a locomotive.



PML problem

You must specify the order of maintenance of locomotives, specifying the start times of service for each locomotive and a service position where the locomotive will be served.



Done work

The considered objective functions:

- total idle time;
- total waiting time;
- maximum waiting time;
- makespan.

Obtained results:

- for dynamic programming $O((\sum_s n_s)^m n_1^{m+1} \dots n_s^{m+1})$ of states must be checked;
- CP model for IBM ILOG CPLEX optimizer is developed. Finding of an approximate solution takes more than 4 hours;
- a heuristic algorithm is developed that gives a solution with the value of the objective function 20% more than that of the IBM ILOG CPLEX optimizer;
- the algorithm of local search is applied to the schedule received by heuristic algorithm. The value of the objective function decreased by 1%.

Section 4

About the author

About the author

Alexander Lazarev – the specialist in scheduling theory, optimization and discrete mathematics, author of more then 200 publications on scheduling problems, applications and combinatorial optimization, including three monographs. The Head of the laboratory of scheduling theory and discrete optimization in V. A. Trapeznikov Institute of Control Sciences of Russian Academy of Sciences, doctor of physical and mathematical sciences, professor (Lomonosov Moscow State University and Institute of Physics and Technology State University).

- 211 publications, h-index in Scopus 7, in WoS 25, 47 citations.
- Directed four PhD and one doctor of physical and mathematical sciences.
- Dissertation board ICS RAS D002 member.
- «Operational research» department editor in abstract journal «Mathematics» VINITI.
- Editorial board member of journals «Automation and remote control», «Control sciences» and «Large-scale Systems Control».