Informace k finální zkoušce (Kombinatorická optimalizace)

Tento dokument obsahuje shrnutí klíčových informací o finální zkoušce z předmětu Kombinatorická optimalizace (KO), získaných z proběhlé komunikace mezi studenty v závěru semestru (konec května a červen) a z nalezených poznámek v archivech.

1. Co se bude a NEBUDE zkoušet (oficiální informace z mailu od p. Hanzálka)

Následující témata **NEBUDOU** předmětem zkoušky: * **Flows/Matching:** Maďarský algoritmus (Flows_e.pdf 42-48). * **Scheduling:** Od "Feasibility test" do konce (Sched_e.pdf 71-78). * **Další výjimka ze Scheduling:** Redukce z `PSm, 1 |temp| Cmax` na `PS1 |temp| Cmax`. * **Constraint Programming (CSP):** Většina CSP ve zkoušce JE, na termínech se objevilo například oditerování algoritmu AC-3.

2. Struktura písemné zkoušky a reálná zadání

- Test se celkově skládá ze **6 úloh**. Zastoupení témat bývá zhruba 3 úlohy na ILP, SP, Flows a zbylé úlohy na Knapsack, TSP a CSP.
- Nejlepší přípravou je propočítat si stará zadání, příklady se často opakují.

Konkrétní příklady z reálných termínů (dle zápisků studentů):

Termín A: 1. ILP formulace max flow s extra podmínkami (např. přes množinu M, liché hodnoty, y větší než 1 pokud je x nenulové,...). 2. All-to-all nejdelší cesta přes Floyd-Warshallův algoritmus na grafu o 4 vrcholech. 3. Formulace distribuce komodit ze zdrojů do spotřebičů pomocí multicommodity network flow (byla to složitá úloha, kterou nakonec zjednodušovali). 4. Důkaz korektnosti Christofidesova algoritmu pro TSP. 5. Časově indexovaný model (Time-indexed scheduling model) pomocí ILP.

Termín B: 1. Graf s inicializačním tokem, udělat iterace Ford-Fulkersonova algoritmu. + Podotázka: Kolik iterací potřebujeme pro tento případ (maximálně)? 2. Zadáno auto (potřebuje 12 l), kamion (potřebuje 68 l) a barel s 80 l. K dispozici jsou láhve o objemu 9 l a 15 l. Zformulovat to jako *shortest path* problém (popsat jak vypadá graf, vrcholy, hrany, váhy atd.). 3. Odvodit (derivovat) aproximační faktor pro Christofidesův algoritmus (přepsat ze slajdů). 4. Vyřešit Knapsack problém pomocí Dynamického programování (DP - tabulka). + Podotázka: Je řešení jediné, nebo jich existuje víc? 5. Napsat ILP model pro časově indexovaný model (Time-indexed) pro `PS1 | temp | Cmax`.

Zkouškový materiál:

Termín: 02. 06. 2021

Zdrojový soubor: KO_02_06_2021-dacaa928922029f8.pdf

Původ: Sdíleno studenty na Discordu.

Penguins on ice

You are given a 2D grid of size $w \times h$. Each cell represents an ice cube floating on the water. There are n penguins standing at starting positions $(x_1^{(s)}, y_1^{(s)}), (x_2^{(s)}, y_2^{(s)}), \ldots, (x_n^{(s)}, y_n^{(s)})$ at time $t = 0$. The time is discretized; during each time interval, each penguin can either remain on its position or it can move (by one tile) to its 4-neighborhood tiles. When the penguin moves, the ice cube on which it was standing sinks down, i.e., it cannot be used anymore. At most one penguin at time can stand on one ice cube.

Design an ILP model deciding whether the penguins can move in such a way that at time $T_{\max}$, penguin i stands at ending position $\left(x_i^{(e)}, y_i^{(e)}\right) \quad \forall i \in \{1, \ldots, n\}$.

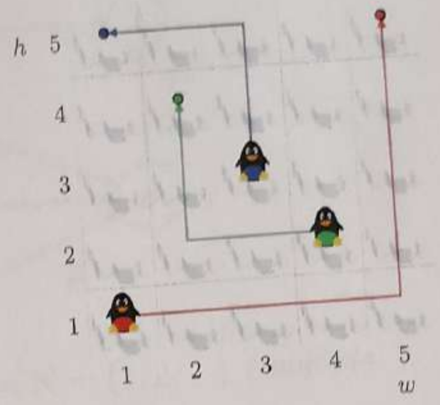


Figure: Illustration of an instance with 5×5 grid and three penguins starting at positions $(1,1)$, $(4,2)$ and $(3,3)$, respectively. The instance is feasible for $T_{\max} \geq 8$.

min 0

subject to:

$$\cancel{T_{max} = \left|x_i^{(s)} - x_i^{(e)}\right| + \left|y_i^{(s)} - y_i^{(e)}\right| \quad ; \forall i \in \{1,\ldots,n\}}$$

$$T_{max} = \sum_{k=s}^{T_{max}} \left|x_i^{(k)} - x_i^{(k+1)}\right| + \sum_{k=s}^{T_{max}} \left|y_i^{(k)} - y_i^{(k+1)}\right| \quad ; \forall i \in \{1,\ldots,n\}$$

$$M \cdot y + x_i^{(k)} = x_j^{(k)}$$

$$y_i^{(k)} + 1 \leq y_i^{(k)} + M \cdot y$$

$$y_i^{(k)} - 1 \geq y_j^{(k)} - M \cdot y$$

$$M \cdot y + y_i^{(k)} = y_j^{(k)}$$

$$x_i^{(k)} + 1 \leq x_j^{(k)} + M \cdot y$$

$$x_i^{(k)} - 1 \geq x_j - M \cdot y$$

$$\cancel{(x_i^{(k)} + 1) \leq (x_j^{(k)}) + M \cdot y}$$

$$\cancel{(x_i^{(k)} - 1) \geq (x_j^{(k)}) - (1-y) \cdot M}$$

$$\cancel{(y_i^{(k)} + 1) \leq (y_j^{(k)}) + M \cdot y}$$

$$\cancel{(y_i^{(k)} - 1) \geq y_j^{(k)} - (1-y) \cdot M}$$

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

$\forall k \in \{s, \ldots, e\}$

$M \in \mathbb{R}^+$

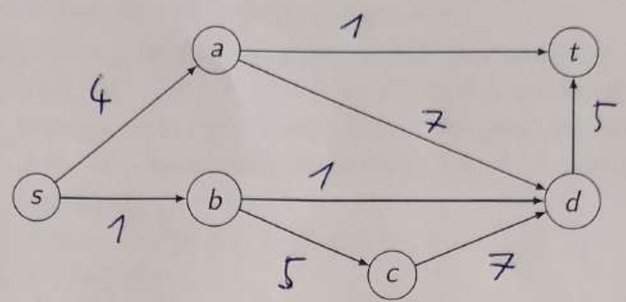
parameters:

$M \in \mathbb{R}^+, \; n \in \mathbb{N}^+, \; k \in \mathbb{N}^+, \; i \in \{1, \ldots, n\}, \; j \in \{1, \ldots, n\}$

Variables:

$y \in \{0, 1\}, \; x_i \in \{1, \ldots, w\}, \; y_i \in$

Edge costs assignment



Consider the graph above. Assign edge weights from $\mathbb{N} = \{1, 2, ...\}$ *(multiple edges can be assigned the same weight)* to each edge so that:

1. Dijkstra algorithm searching the shortest path from vertex s to vertex t must visit/expand the vertices in the following order: s, b, d, a, c, t.
2. The resulting shortest path is $s - a - t$.
3. The cost of the shortest path (sum of path's edge weights) is ≤ 5.

After you assigned the edge weights such that conditions 1–3 hold, mark the edges whose value can be changed to any value from $\mathbb{N}$ such that a run of Dijsktra algorithm still necessarily results in satisfying conditions 1–3.

changable edges : $c \rightarrow d$

$d \rightarrow t$

$a \rightarrow d$

Correctness of Dijkstra's Algorithm

Prove the correctness of Dijkstra's Algorithm

Input: digraph G, weights $c : E(G) \to \mathbb{R}_0^+$ and node $s \in V(G)$.

Output: Vectors l and p. For $v \in V(G)$, $l(v)$ is the length of the shortest path from s and $p(v)$ is the previous node in the path. If v is unreachable from s, $l(v) = \infty$ and $p(v)$ is undefined.

$l(s) := 0$; $l(v) := \infty$ for $v \neq s$; $R := \emptyset$;
while $R \neq V(G)$ **do**
  Find $v \in V(G) \setminus R$ such that $l(v) = \min_{w \in V(G) \setminus R} l(w)$;
  $R := R \cup \{v\}$;
  `// calculate` $l(w)$ `for all nodes on border of` R
  for $w \in V(G) \setminus R$ such that $(v, w) \in E(G)$ **do**
    if $l(w) > l(v) + c(v, w)$ **then**
      $l(w) := l(v) + c(v, w)$; $p(w) := v$;
    end
  end
end

if computed distance from source (s) to sink (t) is not ∞,
then there must exist path from s to t,
it is unreachable otherwise.

1) every node can be only visited if conected to sub graph of visited nodes (which includes s), therefore sink distance is ∞ only if disconected

2) values are updated by visited nodes distance + cost of edge
visited node distances are already shortest,
so any nodes visited from visited network will become shortest as well

School Bus Driver Assignment

A bus company has n morning runs and n afternoon runs that it needs to assign them to its n drivers. The runs are of different duration. If the total duration of the morning and afternoon runs assigned to a driver is more than a specified D, the driver receives a premium payment for each hour of overtime. The company would like to assign the runs to the drivers to minimize the total number of overtime hours.

a) Formulate this problem as an appropriate matching problem. Formally specify input parameters of the problem.

b) Suppose that we arrange the morning runs in the non-decreasing order of their duration and the afternoon runs in the non-increasing order of their duration. If we assign each driver i to the ith morning run and the ith afternoon run, do we obtain the optimal assignment? Outline the proof or find a counterexample.

a)

Problem: $P_m \mid d_j \mid \sum w \cdot C_{max}$

$n = m$ = number of resources/tasks

d_j = due time = D pro $\forall d_j$

w = weight = overtime or not

$w \in \mathbb{R}^+_1$

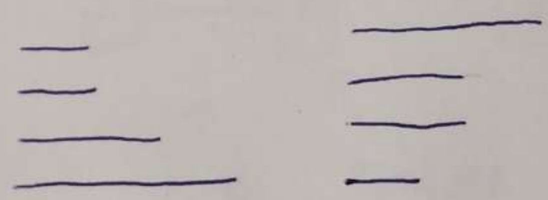
$d_j \in \mathbb{R}^+$

$m \in \mathbb{N}^+_1$

$C_{max} \in \mathbb{R}^+$

b)

morning runs: afternoon runs



- with this arrangement, the shortest run is assigned the longest, in proportion to how long or short it is
- this way we get pairs of runs whose length is the least different to other pairs (we minimize the differences)

$C_{i\,max} - C_{j\,max} = d$

- therefore, this approach is valid in order to get optimal solu

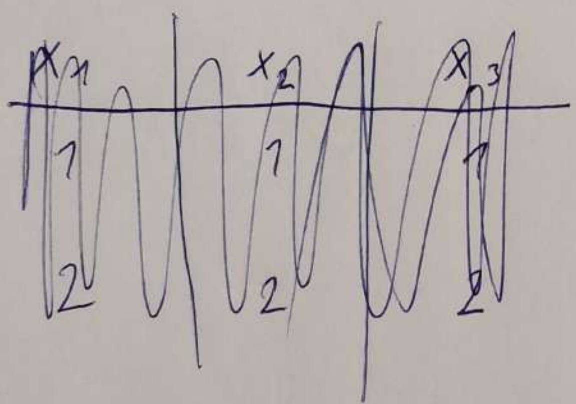
CSP - Arc Consistency Check Using AC-3

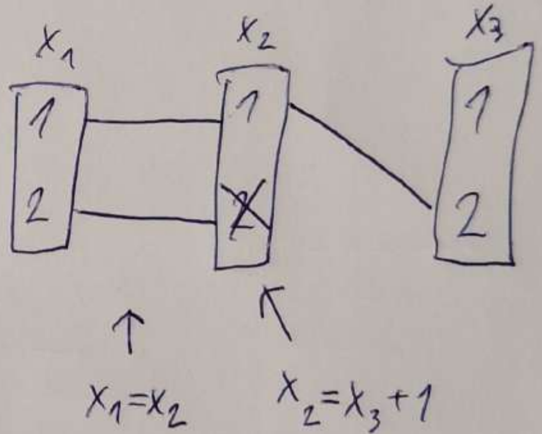
Let us consider the CSP problem given by:

- variables $X = \{x_1, x_2, x_3\}$,
- constraints $x_1 = x_2,\ x_2 = x_3 + 1$
- domains $D_1 = \{1, 2\}, D_2 = \{1, 2\}, D_3 = \{1, 2\}$.

Reduce the domains to be arc consistent.
Record the steps of the algorithm (the queue content prior to the revision of each arc).

Init:

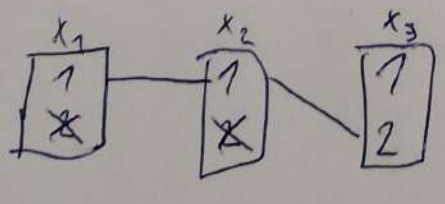




(neexistuje pro $x_2 = 2$)
x_3 nemá takovou doménu

Revisions

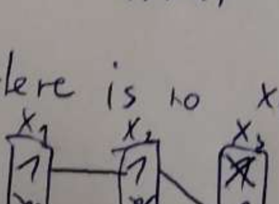
1) (x_1, x_2) → remove $x_1 = 2$, unsplittable by $x_2 = x_3 + 1 = x_1$



2) (x_2, x_1) → all consistent, no changes

3) (x_2, x_3) → all consistent (2 pruned in init), no changes

4) (x_3, x_2) → there is no x_2 for which $x_3 = 1$, prune



5) done

Time-indexed ILP Model for $PS1\,|temp|\,C_{max}$

Formulate $PS1\,|temp|\,C_{max}$ problem as a time-indexed ILP.

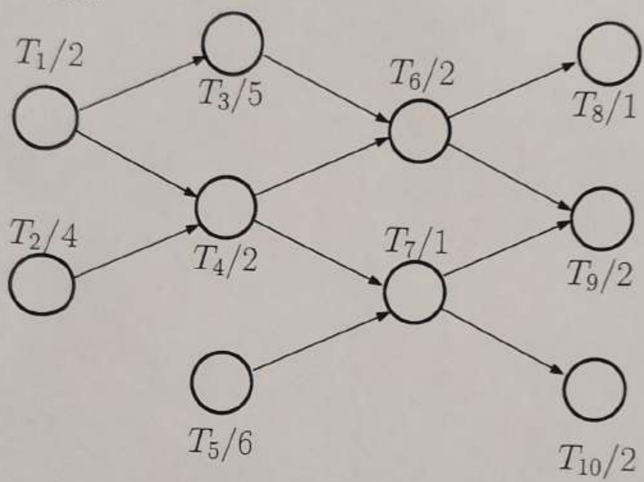
- Input: The number of non-preemptive tasks n and processing times $[p_1, p_2, ..., p_n]$. The temporal constraints defined by digraph G.
- Output: n-element vector s, where s_i is the start time of T_i

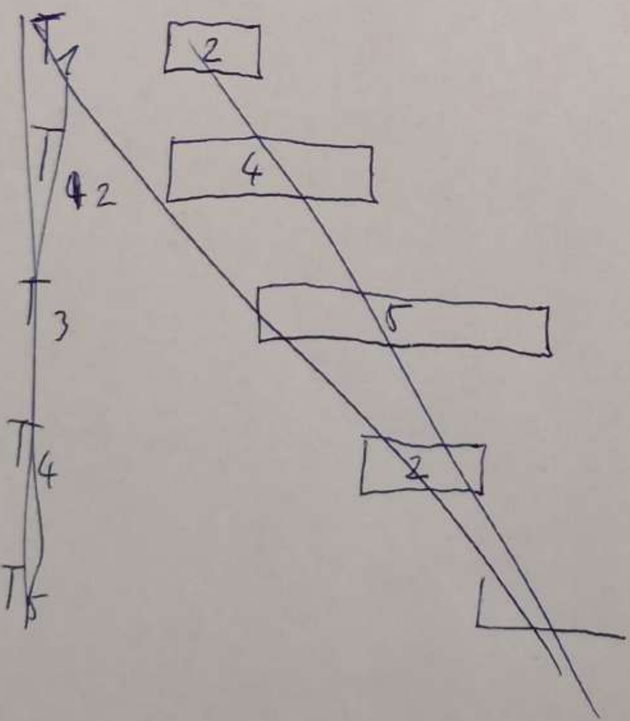
min C_{max}

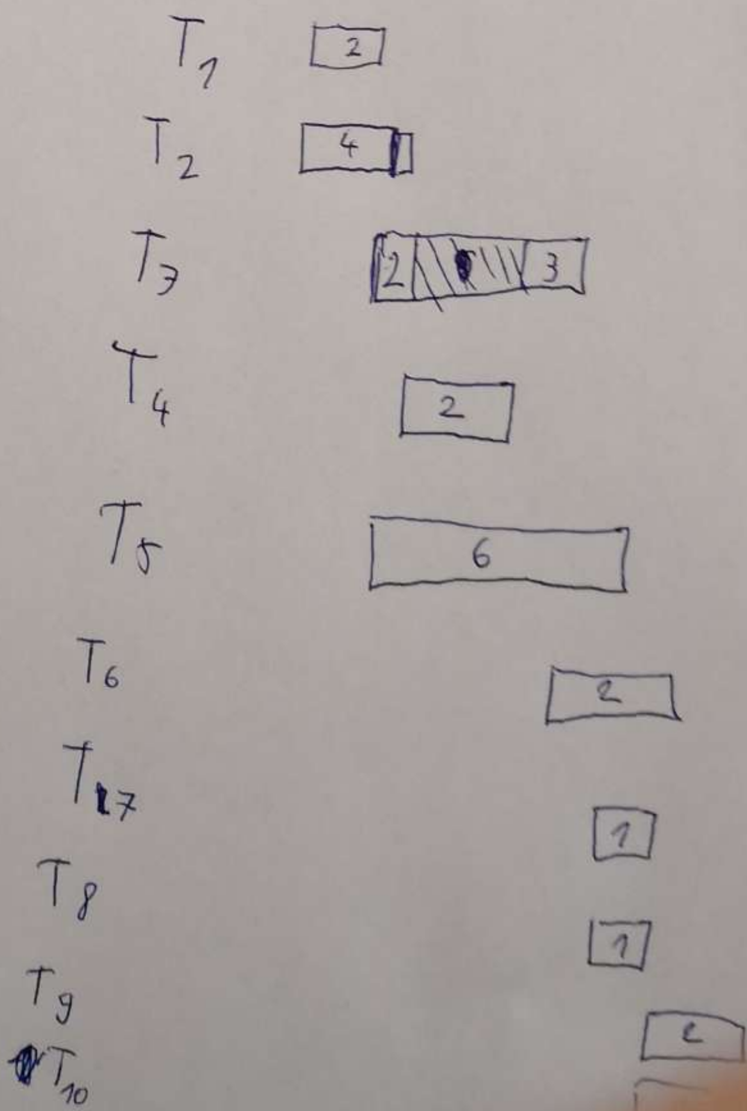
subject to :

Muntz&Coffman's Level Algorithm for $P\,|\text{pmtn}, \text{prec}|\,C_{max}$

Solve the following instance $P2\,|\text{pmtn}, \text{prec}|\,C_{max}$ by Muntz&Coffman's Level Algorithm. Record a list of levels at initialization and $\mathcal{Z}$, set of free tasks, for each iteration of the main loop. Precedence relations are given by the following graph. Processing time is behind the slash of each task. Draw the Gantt chart.







Zkouškový materiál:

Termín: 09. 07. 2021

Zdrojový soubor: KO_09_07_2021-60645c6598e3db40.pdf

Původ: Sdíleno studenty na Discordu.

Toys Assignment

(4.75)

There is a set of toys $T = \{t_1, t_2, \ldots, t_n\}$ scattered all around the room. By one of the walls, there is a row of boxes $B = \{b_1, b_2, \ldots, b_n\}$, where box b_i is located left to b_{i+1} for all $i \in \{1, 2, \ldots, n-1\}$. Distance of toy t_i to box b_j is $d_{i,j} \in \mathbb{R}_0^+ \quad \forall t_i \in T, b_j \in B$.

(a) Your goal is to design an ILP model to organize the toys to the boxes such that: (i) each box contains exactly one toy and each toy is stored in one box, (ii) the total distance between the toys and their assigned boxes is minimized, (iii) toy t_1 is assigned to the box immediately left to the box where t_2 is assigned, (iv) neither t_5 nor t_6 can be assigned to b_1 or b_n, (v) t_4 is in the box immediately next to t_5 or t_6.

(b) Disregard constraints (iii) and (v). Then, formalize the problem as an assignment problem (i.e., minimum weight perfect matching in complete bipartite graph whose sets have the same cardinality).

a)

(4.75)

$x_{ij} = 1$ iff toy i placed in box j ✓0.5

objective

$\min z = \sum_{i \in 1 \ldots n} \sum_{j \in 1 \ldots n} x_{ij} \cdot d_{ij}$ podmínka (ii) ✓0.5

subject to:

$\sum_{i \in 1 \ldots n} x_{ij} = 1 \quad | \; j \in 1 \ldots n$

$\sum_{j \in 1 \ldots n} x_{ij} = 1 \quad | \; i \in 1 \ldots n$

podmínka (i) ✓0.5

$x_{1,j} = x_{2,j+1} \quad | \; j \in 1 \ldots n-1$ } podmínka (iii) ✓1

$x_{51} = 0$

$x_{5n} = 0$

$x_{61} = 0$

$x_{6n} = 0$

podmínka (iv) ✓0.5

≤ ?

$y_2 \cdot M + y_1 \cdot M + x_{4j} \geq x_{5,j+1}$

$y_2 \cdot M + (1-y_1) \cdot M + x_{4j} \geq x_{5,j-1}$

$(1-y_2) \cdot M + y_1 \cdot M + x_{4j} \geq x_{6,j+1}$

$(1-y_2) \cdot M + (1-y_1) \cdot M + x_{4j} \geq x_{6,j-1}$

podmínka (v)

pokud tam t4 je → dej tam t5 nebo t6

Kvantifikátory?

1.75

…rameters

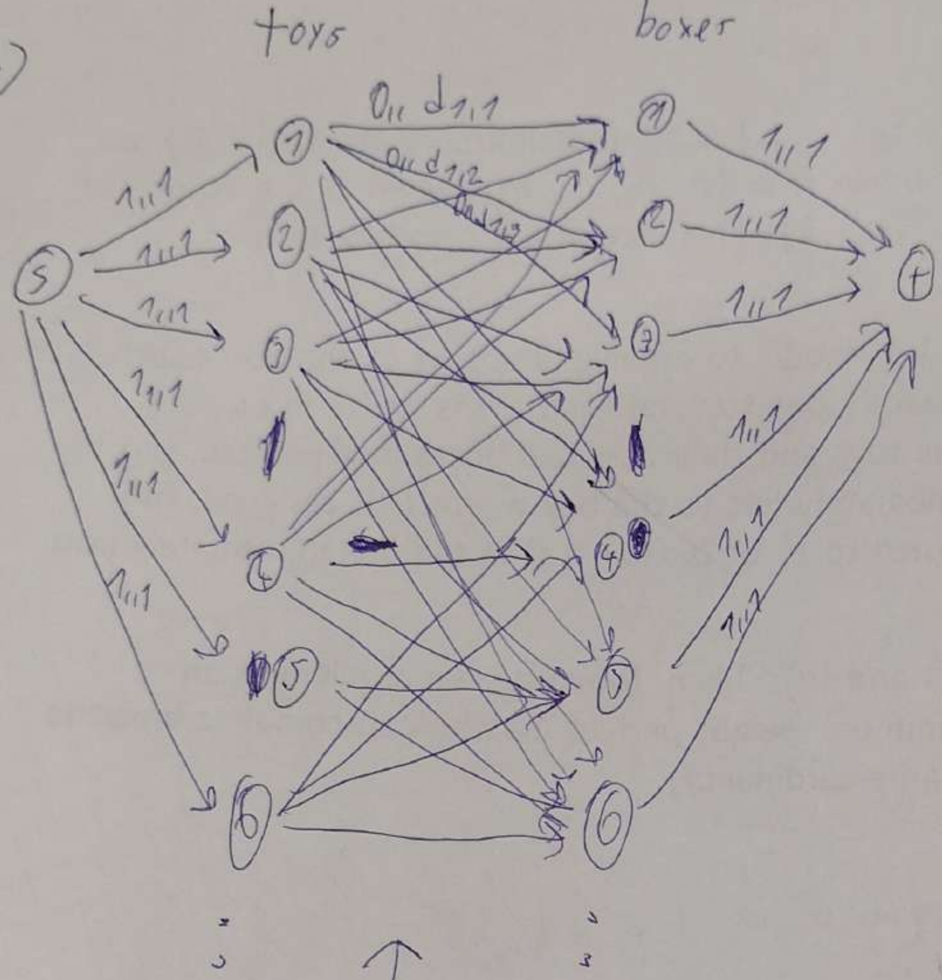
…$_{1 \ldots n, j \in 1 \ldots n} \in \mathbb{R}_0^+$

$\ldots \in \mathbb{R}^+, n \in \mathbb{Z}^+$

Variables

$x_{i \in 1 \ldots n, j \in 1 \ldots n} \in \{0,1\}$, $y_1, y_2 \in \{0,1\}$

b)



nechceme tohle ale párování

kompletní propojení,

s vyjímkou hran: $d_{5,1}$, $d_{5,n}$, $d_{6,1}$, $d_{6,n}$, $n \geq 6$

[illegible] kompletní graf

- z tohoto grafu můžeme najít reziduální graf, a v něm odstranit záporné cykly abychom dostali min flow

5,5

Constrained Shortest Path Problem

In a network G we associate two numbers with each arc: its length $c_{ij} \in \mathbb{R}$ and its traversal time $\tau_{ij} \in \mathbb{Z}_0^+$. We would like to determine a shortest-length path from the source node s to the sink node t with the additional constraint that the traversal time of the path does not exceed τ.

a) Formulate this problem as a shortest path problem (hint: use time expansion of the network).

b) Does the path necessarily exist if the network G is strongly connected?

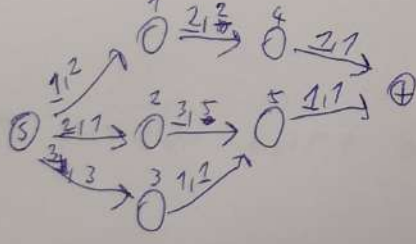
b) it doesnt necessarily exist, its possible that ~~the shortest~~ all paths ~~path~~ violate the constraint of not exceeding τ ✓

a) ~~We can draw the graph in a way, that~~

We draw the graph on discretised timeline, ending at τ.
At each time t of the time line, we draw all nodes between s and t.
Then we conect each node the way it was in original, but corecting to the nodes at time $(\tau_{ij} + t)$

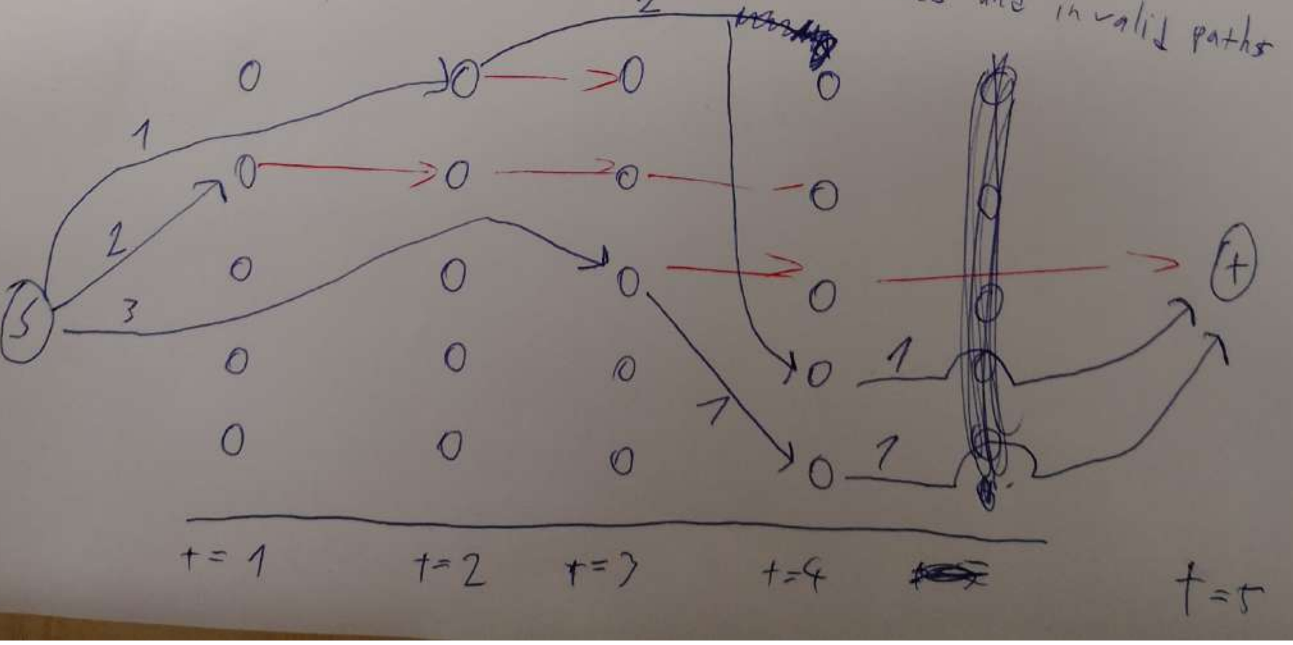
kam hledáme cestu?
co když jsme v t dřív?

potřebujeme nekonečné hrany



$\tau = 5$

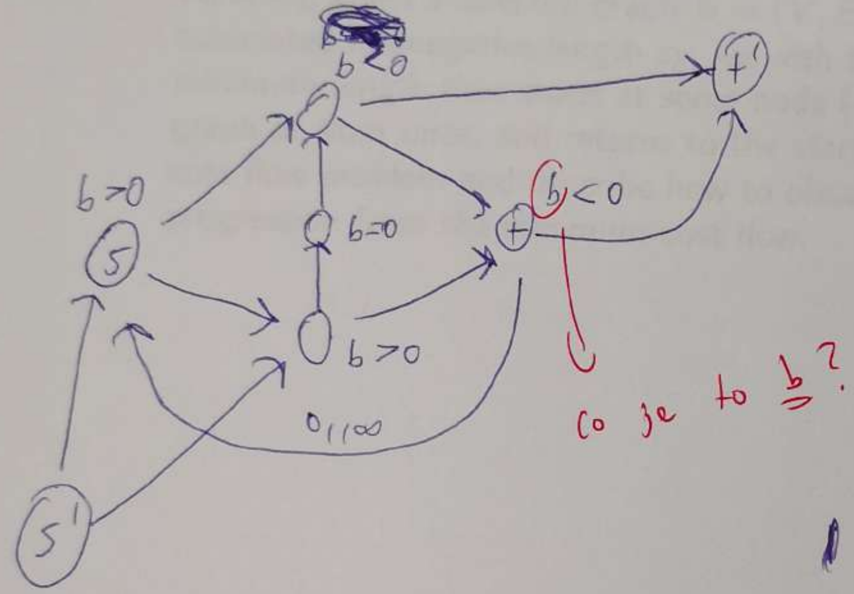
* remove all disconected nodes and invalid paths



Initial Feasible Flow for Ford-Fulkerson Algorithm

2b

Transform the problem of finding an initial feasible flow (with non-zero lower bound problem (G, l, u, s, t)) for Ford-Fulkerson algorithm to problem (G', u', b) i.e., the feasible flow decision problem with zero lower bound.



// this problem finds initial flow, when max flow is found on it

- in this graph, all edges should have $LB = 0$

$UB = u_f$

where $u_f = u(e) - l(e)$

- init flow exist only if LB of preceding edge is not ??

$LB \geq UB$

Directed Chinese Postman Problem

Leaving from their post office, a postal carrier needs to pass every street in the given direction, delivering and collecting letters, and then returning to the post office. The carrier would like to cover this route by traveling the minimum possible distance.

Formally, given a directed graph $G = (V, E)$, where each edge (i, j) has an associated nonnegative length c_{ij}, we wish to find an edge progression of minimum length that starts at some node (the post office), visits each arc of the graph at least once, and returns to the starting node. Formulate as a minimum cost flow problem and describe how to obtain the minimum-length edge progression from the minimum-cost flow.

Knapsack

0,5

Using dynamic programming, solve the following instance of Knapsack Problem:

- number of items: $n = 7$
- knapsack capacity: $W = 5$
- costs $\boldsymbol{c} = (2, 2, 2, 2, 4, 3, 1)$
- weights: $\boldsymbol{w} = (1, 1, 2, 2, 3, 4, 1)$

a) Compute the optimal solution (objective value and items in knapsack) of this instance of Knapsack Problem. Write down all iterations of the algorithm. Is this solution unique and why?

b) What can you say about the computational complexity of the algorithm for instances where $W \leq 10n$?

podíly: $\left(\frac{2}{1}, \frac{2}{1}, 1, 1, \frac{4}{3}, \frac{3}{4}, 1\right)$

Není DP a ani není naznačen žádný postup.

$x_{ij} = 1$ iff item i put in j position

1) $x_{11} = 1$ $w = 1$ $c = 2$

2) $x_{22} = 1$ $w = 2$ $c = 4$

3) $x_{53} = 1$ $w = 5$ $c = 8$ →

This solution is unique, there is to 2 or more last items with equal ~~...~~ price that dont fit into the bag

Scheduling on One Resource Minimizing $\sum w_j C_j$

2

a) Using ILP formulate scheduling of nonpreemptive tasks with precedence relations on one resource while minimizing weighted sum of completion times, i.e., $1|prec|\sum w_j C_j$.
Task T_j has general processing time p_j. Precedence relations are encoded in $e_{ij} \in \{0,1\}$ such that $e_{ij} = 1$ iff there is a directed edge from T_i to T_j in the precedence graph G or $i = j$.
b) Explain in detail, how to use relaxed LP solution in Branch and Bound method for $1|prec|\sum w_j C_j$?

$x_{ij} = 1$ if task i precedes task j in schedule

Objective

$$\min z = \sum_{j \in 1 \dots n} w_j C_j$$

subject to:

✓ $x_{ij} \geq e_{ij}$ // precedence graph implication

✓ $1 \leq x_{ij} + x_{jk} + x_{kj} \leq 2$ // no cycles

$p_i \cdot x_{ij} \leq C_j$ // set C_j

to nestačí! C_j je suma processing

Parameters

$w_j \in R^+$ $p_i \in \mathbb{Z}^+$

$e_{ij} \in \{0,1\}$ $C_j \in \mathbb{Z}^+$

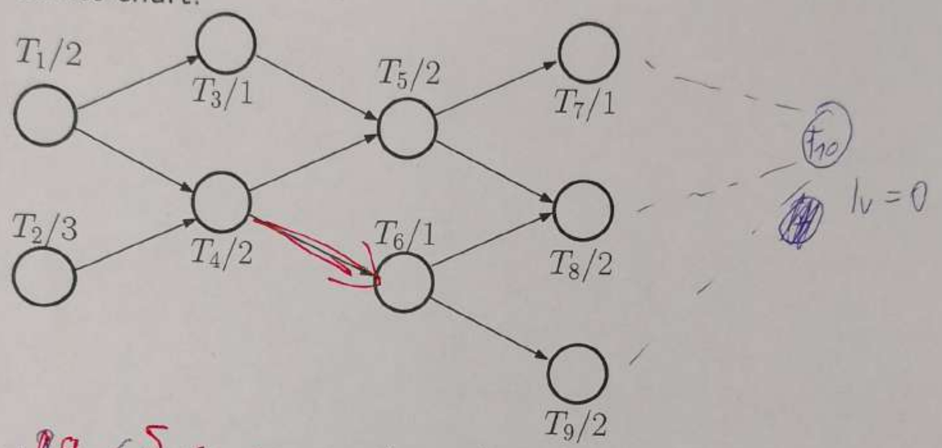
Variables

$x_{ij} \in \{0,1\}$

Muntz&Coffman's Level Algorithm for $P\,|\text{pmtn, prec}|\,C_{max}$

(16)

Solve the following instance of $P2\,|\text{pmtn, prec}|\,C_{max}$ problem by Muntz&Coffman's Level Algorithm. Record a list of levels at initialization and $\mathcal{Z}$, set of free tasks, for each iteration of the main loop. Precedence relations are given by the following graph. Processing time is behind the slash of each task label. Draw the Gantt chart.



Init:

level = (8, 9, 5, 6, 4, 3, 1, 2, 2)

$Z = \{T_1, T_2\}$

1) $h=2$ $\quad Z = T_1, T_2$

$S = T_2 \quad \beta = 1$

$h = 1 \quad Z = \{T_1\} \quad \beta = 1$

$\delta = 2 \quad S = T_1$

2) level = (4, 6, 4, 5, 3, 3, 1, 2, 2)

$h = 1 \quad Z = \{T_2, T_3\}$

$S = T_2 \quad \beta = 1$

$h = 1 \quad S = T_3 \quad \beta = 1$

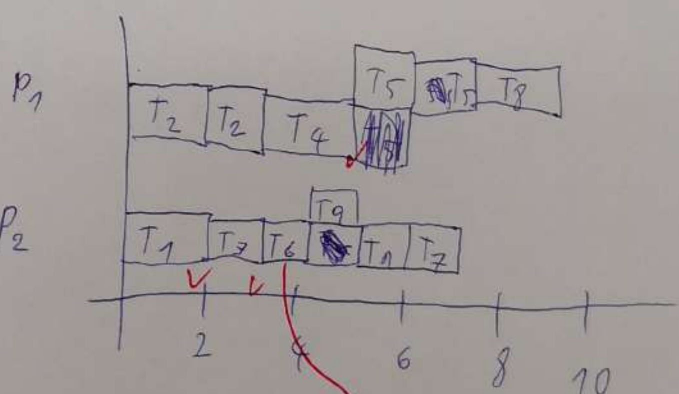
$\delta = 1$

3) level = (4, 5, 3, 5, 3, 3, 1, 2, 2)

$h=2 \quad Z = \{T_4\}$

$S = T_4$

$\delta = 2$



4) level = (4, 5, 3, 3, 3, 3, 1, 2, 2)

$h=2 \quad Z = \{T_5, T_6\}$

$S = T_5 \quad \beta = 1$

$h = 1 \quad S = T_6 \quad \beta = 1$

$\delta = 1$

T_6 nemůže začít před T_4 !

5) level = (4, 5, 3, 3, 2, 2, 1, 2, 2)

$h=2 \quad Z = (T_5, T_9)$

$S = T_5 \quad \beta = 1$

$h = 1 \quad S = T_9 \quad \beta = 1 \qquad \delta = 1$

Zkouškový materiál:

Termín: 21. 06. 2021

Zdrojový soubor: KO_21_06_2021-4387223f7ceac7a4.pdf

Původ: Sdíleno studenty na Discordu.

4.5

Minimization of discontinuous functions

Consider the following mathematical model of two variables x_1, x_2.

$$\text{Minimize } z = f(x_1)$$

subject to the restrictions

1. $|x_1 - x_2| = 0$ or 6.
2. $0 \le x_1 \le 20$, $0 \le x_2 \le 20$.
3. $x_{i \in 1,2} \in \mathbb{Z}_0^+$

where

$$f(x_1) = \begin{cases} 7 + 5x_1, & \text{if } x_1 > 0. \\ 0, & \text{if } x_1 = 0. \end{cases}$$

Formulate this problem as an ILP problem.

Objective

$\min\ z = (7 + 5x_1) \cdot (1 - y_2)$ 2b

Conditions:

$x_1 \ge 0$
$x_1 \le 20$
$x_2 \ge 0$
$x_2 \le 20$
2. podmínka
1b

$x_1 = 0 \rightarrow y_2 = 1$
$x_1 = 1 \rightarrow y_2 = 0$

$y_2 \cdot M + x_1 \ge 1$
$x_1 \le 0 + (1 - y_2) \cdot M$
rozdělení $f(x_1)$

$y_1 \cdot M + x_1 - x_2 \ge 0$
$x_1 - x_2 \le 0 + y_1 \cdot M$
$(1 - y_1) \cdot M + x_1 - x_2 \ge 6$
$x_1 - x_2 \le 6 + (1 - y_1) \cdot M$

~~1. podmínka~~

y

$x_2 - x_1 \ge 0$
$x_2 - x_1 \le 0$
$x_2 - x_1 \ge 6$
$x_2 - x_1 \le 6$

$y_1 = 0 \rightarrow |0|$
$y_1 = 1$
→ je součástí modelu?
pokud ano model infeasible
pokud ne absolutní hodnota pro 6 nefunguje

1.5b

Parameters

$M \in \mathbb{R}$

Variables

$x_{i \in 1,2} \in \mathbb{Z}_0^+$, $y_1, y_2 \in \{0, 1\}$ ← 3. podmínka

Volíte proměnné a Rosta (6)

Wedding plan

You are planning a wedding. There is a set of potential guests $G = \{g_1, g_2, \dots, g_n\}$. You want to invite as many guests as possible; however, there are some constraints. A meal per a single guest costs m CZK (Czech koruna). A price for renting a place differs, depending on the number of invited guests: for $[0, 9]$ guests, the price is p_1 CZK, for $[10, 19]$ guests, the price is p_2 CZK, and for $[20, n]$ guests the price is p_3 CZK. You have B CZK; this budget cannot be exceeded. Furthermore, for the sake of justice, the number of guests invited from the groom's side and the bride's side can differ by at most $D_{\max}$. Assume functions $br : G \to \{0, 1\}$ and $gr : G \to \{0, 1\}$, where $br(g_i) = 1$ iff g_i is from the bride's side, and $gr(g_i) = 1$ iff g_i is from the groom's side. (Note that there might be some guest g_j such that $br(g_j) = gr(g_j) = 1$).

Design an ILP model deciding which guests will be invited while respecting all the constraints.

Objective

$\max \sum_{i=1}^{n} g_i$ ✓1

Parameters

$M \in \mathbb{R},\ m \in \mathbb{R},\ p_1, p_2, p_3 \in \mathbb{R},\ B \in \mathbb{R},\ D_{max} \in \mathbb{Z}_0^+$

$gr_{i \in 1 \dots n} \in \{0,1\},\ br_{i \in 1 \dots n} \in \{0,1\},\ n \in \mathbb{N}^+$

✓0.5

Variables

$g_{i \in 1 \dots n} \in \{0,1\},\ Y_1, Y_2, Y_3 \in \{0,1\}$

Conditions

$$\left.\begin{aligned}
&Y_1 + Y_2 + Y_3 \geq 2\\
&Y_1 + Y_2 + Y_3 \leq 2\\
&Y_1 \cdot M + n \geq 0\\
&n \leq 9 + Y_1 \cdot M\\
&Y_2 \cdot M + n \geq 10\\
&n \leq 19 + Y_2 \cdot M\\
&Y_3 \cdot M + n \geq 20
\end{aligned}\right\}$$

Set up Y switches for not rented place

$Y_k = 1$ iff not rented

± 1

místo ? $n \sim \sum_{i=1}^{n} x_i$

pokud to dobře chápu, p_3 není shora omezené

2

$$m \cdot \sum_{i=1}^{n} g_i + (1 - Y_1) \cdot p_1 + (1-Y_2) \cdot p_2 + (1-Y_3) \cdot p_3 \leq B$$

budget not exceeded

✓0.5

$$\sum_{i=1}^{n} br_i \cdot g_i - \sum_{i=1}^{n} gr_i \cdot g_i \leq D_{max}$$

$$\sum_{i=1}^{n} gr_i \cdot g_i - \sum_{i=1}^{n} br_i \cdot g_i \leq D_{max}$$

✓2

invided from gr/br side dont differ more than Dmax

(4)

Handling negative cycles in the graph

Consider having a strongly connected graph G, with a set of edges E, and with arbitrary edge weights from $\mathbb{Z}$; $c(e)$ denotes the weight of edge e. *(For the following questions, consider $P \neq NP$.)*

1. How would you check if G contains a negative cycle in a polynomial time?
2. Is finding the shortest path between any two vertices in G always solvable in polynomial time?
3. Consider an integer h, $h \geq |\min_{e \in E} c(e)|$. Consider graph G', which is identical to G, but every edge weight is increased by h. Does the shortest path between any two vertices in G' correspond to the shortest path between the same vertices in G? If yes, explain why; if no, give a counterexample of graphs G and G'.
4. Assuming that $c(e) < 0, \forall e \in E(G)$, can you always find the longest path between any two vertices in G in polynomial time? If yes, explain how to do it; if no, briefly describe why it is not possible.

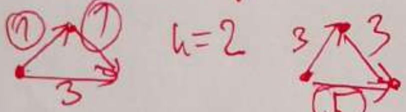
(and incorrect)
very confusing wording

1) I wold run Bellman Ford algorithm, relaxing all edges |V-1| times.
I woud then relax the edges once more, and if any computed distances change, there is a negative cycle.

2) ~~Yes~~, since we are working with finite number of V and E, which are iterated in SPT algorithms.

3) ~~yes it d~~ What about negative cycles.

Yes ~~it~~ does.



We can say that SPT computation is decided by the weight difference between pairs of edges in the graph, i.e. $c(e_i) - c(e_j) = d$ $\quad i,j \in 1 \dots |E|$

Then if we add h to each weight, we get $c(e_i) + h - c(e_j) - h = d$

$$c(e_i) - c(e_j) + h - h = d$$

$$c(e_i) - c(e_j) = d \quad i,j \in 1 \dots |E|$$

Which is the same difference for any pair of edges. Therefore SPT of G corresponds to SPT of G'.

4) Yes we can.
We need to create G' identical to G, where each c(e) is multiplied by (-1).
Then we apply SPT on G', ~~[illegible]~~ and apply the resulting evaluation back to G.

Pruned Chessboard Problem

A chessboard consists of n^2 squares arranged in n rows and n columns. A *domino* is a wooden or plastic piece consisting of two squares joined on a side. The goal is to fully cover (i.e., some domino covers each square) the chessboard with dominos (i.e., each domino covers two squares of the board, and no two dominos overlap). A *pruned board* is a chessboard with some squares removed.
Suppose that we want to know whether it is possible to fully cover a pruned board and if not, to find the maximum number of dominos we can place on the pruned board so that each domino covers two squares of the board and no two dominos overlap. Formulate this problem as an appropriate matching problem.

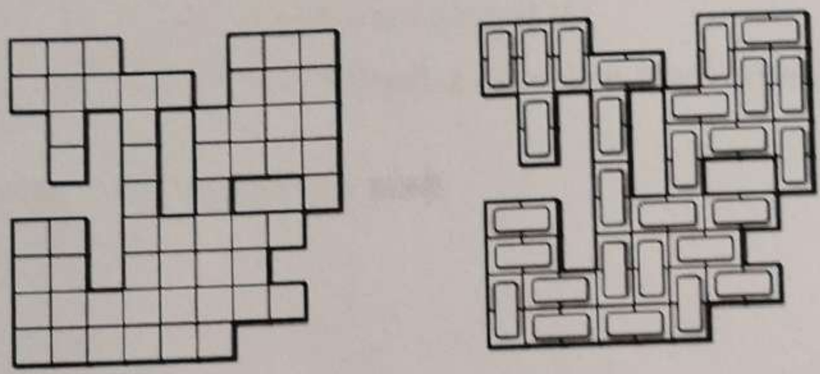


Figure: Pruned chessboard and a solution.

Approximation Factor of Christofides' Algorithm

Derive Approximation Factor of Christofides' Algorithm:

Input: An instance (K_n, c) of **metric TSP**.

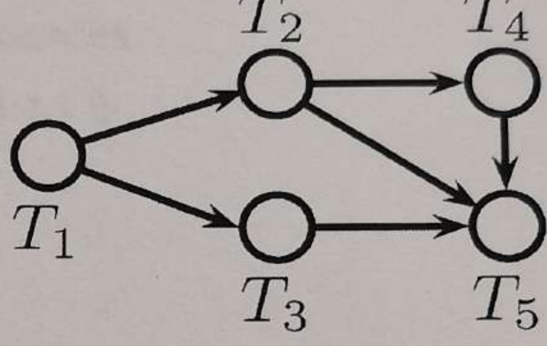
Output: Hamiltonian circuit H.

1. Find a minimum weight spanning tree T in K_n;
2. Let W be the set of vertices having an **odd degree** in T;
3. Find a **minimum weight matching** M of nodes from W in K_n;
4. Merge of T and M forms a multigraph $(V(K_n), E(T) \cup M)$ in which we find the Eulerian circuit L;
5. Transform the Eulerian circuit L into the Hamiltonian circuit H in the complete graph K_n ;

Chetto, Silly, Bouchentouf algorithm for
$1 \mid \text{pmtn}, \text{prec}, r_j, d_j = \widetilde{d}_j \mid L_{max}$

a) Using Chetto, Silly, Bouchentouf algorithm solve the following instance of mono-processor scheduling of preemptive tasks with precedence relations, release dates and due-dates equal to deadlines while minimizing the maximum lateness. Indicate values of main variables in separate steps of the algorithm. Draw the Gantt chart.

$r = (0, 3, 2, 6, 3)$
$p = (2, 3, 2, 2, 3)$
$\widetilde{d} = d = (3, 7, 13, 9, 13)$



b) Calculate the L_{max} value of the optimal solution.

Scheduling on Parallel Identical Resources [Rothkopf]

Solve the following instance $P||C_{max}$ by dynamic programming. Record the main variables for each iteration of the main loop. Draw the Gantt chart.

- processing time $p = (2, 2, 1, 2)$
- $R = 2$, two parallel identical resources
- UB, upper bound C_{max}, is equal to 5

Zkouškový materiál:

Zkoušky z roku 2023 (Rotováno)

Zdrojový soubor: 2023_rotated.pdf

Původ: Staženo ze zip archivu 'zkousky.zip' sdíleného na Discordu.

4

Loot distribution

k loupežníkům ⇒ k skrýši

Alibaba and his k men have raided an ancient tomb. Inside, there were n treasures $T = \{t_1, \ldots, t_n\}$. Each treasure t_i has some weight $w_i \in \mathbb{R}^+$ and price $p_i \in \mathbb{R}^+$. Each treasure belongs to one of four groups $G = \{1, 2, 3, 4\}$, where $g_i \in G$ denotes the group of treasure t_i. Each of Alibaba's k men will move part of the loot to one of the Alibaba's hideouts. The maximal weight that man $j \in \{1, 2, \ldots, k\}$ can carry is $u_j \in \mathbb{R}^+$.

Alibaba wants to:

a) ▶ transport at least one treasure from group 1 and at least one treasure from group 2 to any of his hideouts;

b) ▶ transport any treasure from group 4 only if at least some treasure from group 3 is being transported to any of the hideouts; ($trans_4$, $trans_3$)

c) ▶ distribute the loot such that the maximal difference in the total prices transported by the individual men is at most $P_{max} \geq \max\{p_i\} - \min\{p_i\}$; $\forall \ell$ k nese p_i

d) ▶ maximize the total price of the transferred treasures

Design an ILP model to solve the given loot distribution problem.

$t_i \Rightarrow g_i$ — předpočítám si binární vektory q: $q_{i,1}; q_{i,2}, q_{i,3}, q_{i,4}$ ... i-th treasure patří skupině 1,2,3 "až" 4.

Vars:

~~$m = [m_1, \ldots, m_k] \in \{0,1\}^k$ ... will i-th man move~~

$m^{\ell i} \in \{0,1\}^{k \times n}$ ... k-th man will move n-th part

~~$trans_4, trans_3 \in \{1,0\}$~~

$y, x \in \{0,1\}$

~~Param:~~

Conds:

~~a) $\sum_{\ell \in k} \sum_{i \in n} (g_i = 1) \cdot m^{\ell i} \geq 1$~~

max. weight

$$\sum_{i=1..n} m^{\ell i} \cdot w_i \leq u_\ell \; \forall \ell \in 1..$$ ✓ 1.6

a) $$\sum_{\ell=1..k} \sum_{i=1..n} q_{i,1} \cdot m^{\ell,i} \geq 1$$ ✓

$$\sum_{\ell=1..k} \sum_{i=1..n} q_{i,2} \cdot m^{\ell i} \geq 1$$ ✓ 1.6

b) $$\sum_{\ell} \sum_{i} q_{i4} \cdot m^{\ell i} \; {}^{+y \cdot M} \leq \cancel{trans_4}\; 1 \; ; \; \sum_{\ell} \sum_{i} q_{i3}\, m^{\ell i} \; {}^{+x \cdot M} \geq \cancel{trans_3}\; 1$$

~~$trans_4 = trans_3$~~ $y = x$ 0

z=1 ... y=1 ale musí nést G4 i bez...

Objective

d) maximize $$\sum_{\ell=1..k} \sum_{i=1..n} m^{\ell i} \cdot p_i$$ 1.6

pokračování 2. strana 1. list

Correctness of Dijkstra's Algorithm

Prove the correctness of Dijkstra's Algorithm

Input: digraph G, weights $c : E(G) \to \mathbb{R}_0^+$ and node $s \in V(G)$.

Output: Vectors l and p. For $v \in V(G)$, $l(v)$ is the length of the shortest path from s and $p(v)$ is the previous node in the path. If v is unreachable from s, $l(v) = \infty$ and $p(v)$ is undefined.

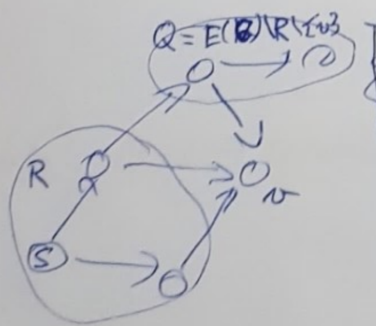
$l(s) := 0$; $l(v) := \infty$ for $v \neq s$; $R := \emptyset$;
while $R \neq V(G)$ **do**
  Find $v \in V(G) \setminus R$ such that $l(v) = \min_{w \in V(G) \setminus R} l(w)$;
  $R := R \cup \{v\}$;
  `// calculate` $l(w)$ `for all nodes on border of` R
  for $w \in V(G) \setminus R$ such that $(v, w) \in E(G)$ **do**
    if $l(w) > l(v) + c(v, w)$ **then**
      $l(w) := l(v) + c(v, w)$; $p(w) := v$;
    end
  end
end

Dokážeme indukcí:

První krok: Pokud $R_0 = \emptyset$, tak R obsahuje optimální vzdálenosti z s.

Indukční předpoklad: ~~Když R~~ R_n obsahuje pouze uzly jejichž vzdálenost od s je optimální.

Chci dokázat: když ↗, tak R_{n+1} je také optimální.



R je optimální, tedy platí Bellmanova rovnice $l(v) = \min_{w \in pred(v)} \{ l(w) + c(w, v) \}$

cena hrany(w, v)

pro předchůdce z Q: ~~[illegible]~~ je ~~[illegible]~~, protože hrany musí mít nezáporné c, tak cesta vedoucí přes uzly z Q, nebude optimální

$\Rightarrow R_{n+1} = R_n \cup \{v\}$ bude optimální.

6

Multiprocessor Scheduling problem with preemption, release date and deadline

We have n **tasks** which we want to assign to R **identical resources** (processors). Each task has its own **processing time** p_j, **release date** r_j and **deadline** $\tilde{d}_j$. **Preemption** is allowed (including migration from one resource to another). Every processor will execute no more than one task at a moment and no task will be executed simultaneously on more than one processor.

2 parallel identical resources:

$M =$

úkol	T_1	T_2	T_3	T_4	T_5
p_j	2.1	3.2	4.1	1.6	2
r_j	0	2	0	5	5
$\tilde{d}_j$	5	6	6	7	7

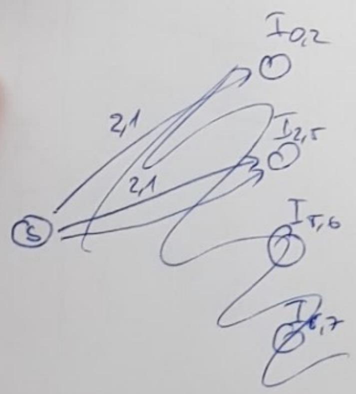
Formulate as Maximum flow problem.

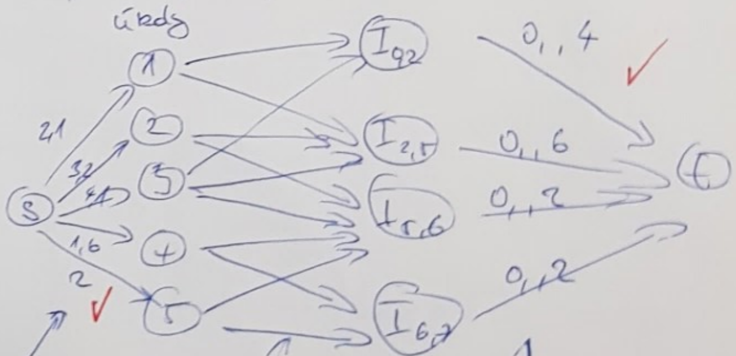
$R = 2$

Důležité body určující intervaly: set($M[r_j : d_j, :]$) =

$= \{0, 2, 5, 6, 7\}$ ✓

Intervaly: $\langle 0; 2)$ $I_{0,2}$, $\langle 2; 5)$ $I_{2,5}$, $\langle 5; 6)$ $I_{5,6}$, $\langle 6; 7)$ $I_{6,7}$ ✓





stejný ub i LB

✓ ... feasible

opt. flow → schedule

✓

LB = 0; UB = vstupní tok do úkolu.

upper bounds: množství volných jednotek v intervalu. → délka I · R

Knapsack

6b

Using dynamic programming, solve the following instance of Knapsack Problem:

- number of items: $n = 7$
- knapsack capacity: $W = 5$
- costs $\mathbf{c} = (2, 2, 2, 2, 4, 3, 1)$
- weights: $\mathbf{w} = (1, 1, 2, 2, 3, 4, 1)$

a) Compute the optimal solution (objective value and items in knapsack) of this instance of Knapsack Problem. Write down all iterations of the algorithm. Is this solution unique and why?

b) What can you say about the computational complexity of the algorithm for instances where $W \leq 10n$?

může předmět buďto vložit, nebo nevložit

beru maximum

item \ w	0	1	2	3	4	5
	0	$-\infty$	$-\infty$			
I_1	0	2	$-\infty$	$-\infty$	$-\infty$	$-\infty$
I_2	0	2	4	$-\infty$	$-\infty$	$-\infty$
I_3	0	2	4	4	6	$-\infty$
I_4	0	2	4	4	6	6
I_5	0	2	4	4	6	8
I_6	0	2	4	4	6	8
I_7	0	2	4	5	6	8

← maximální cena

Když zde neplníte konkrétní čísla, je nutné uvést, že opt. řešení nemusí být v pravém dolním rohu.

a) řešení: ~~{I_1, I_2, I_5}~~ {I_1, I_2, I_5}, $c_{opt} = 6$

Řešení je unikátní: existuje jen jedna cesta z $(I_1, 0)$ do optima

b) $O(n \cdot W) = O(n \cdot 10n) = O(n^2)$

Složitost bude záviset jen na počtu těsně, ale stále bude polynomiální.

Project Scheduling with Temporal Constraints

4b

A construction company has obtained a contract on a project consisting of 4 non-preemptive activities A, B, C and D. The following temporal constraints apply:

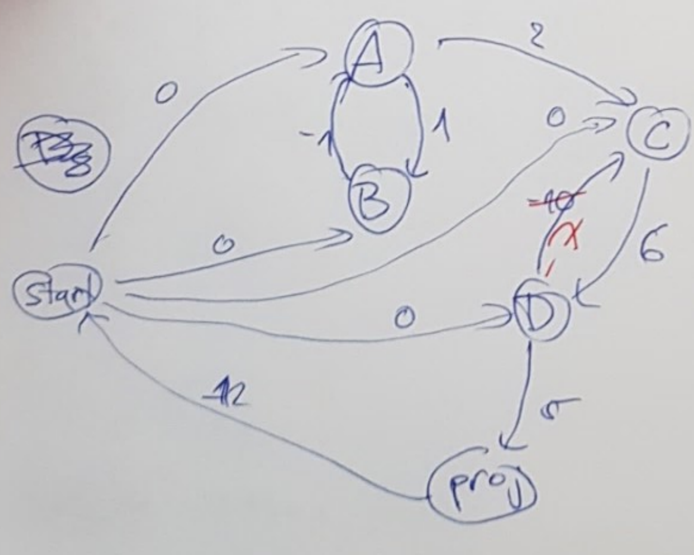
- a) Activity durations are $p_A = 1$, $p_B = 2$, $p_C = 3$ and $p_D = 2$.
- b) B must start exactly 1 time unit after start of A.
- c) C can start at least 2 time units after start of A.
- d) After the completion of C, there are at least 3 more time units before D can start.
- e) D can be started not later than 7 units after start of C.
- f) We suppose that the project can be terminated 3 time units after the completion time of D.
- g) The maximum duration of the project is 12 time units.

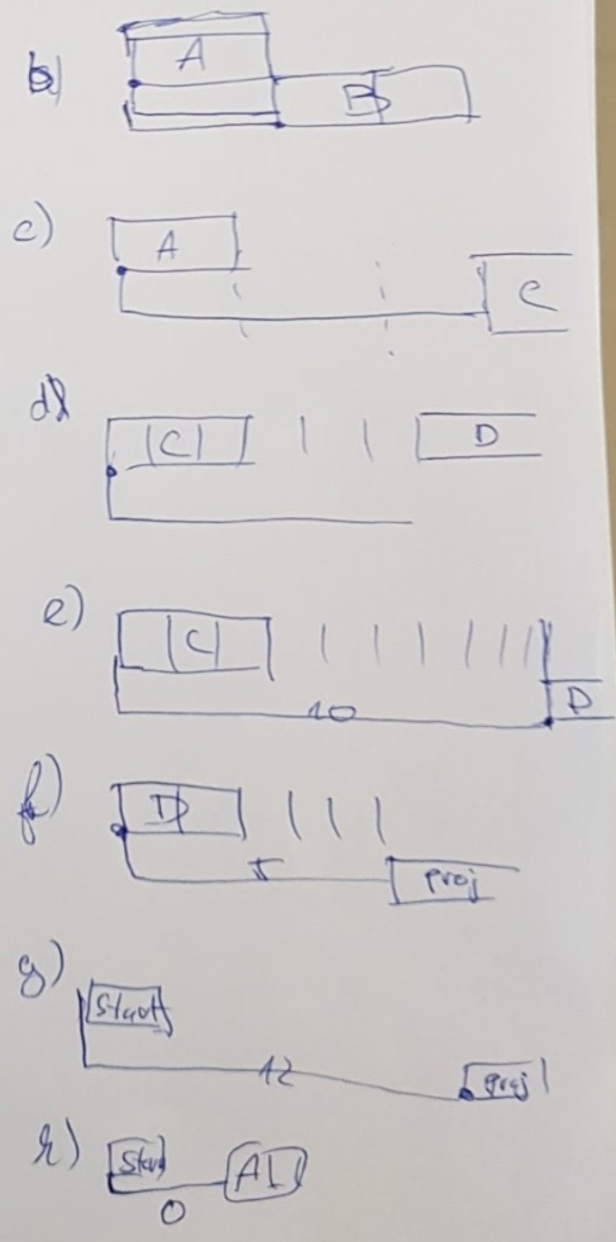
a) Draw a directed graph with temporal constraints for this problem.

b) Is this instance schedulable? Why? ✓

Hint: Add "dummy" activities with zero duration for beginning and end of the project. ~~Bs, De~~

	A	B	C	D
p_i	1	2	3	2





Maximization of absolute value

Consider the following mathematical model of four variables x_1, x_2, x_3 and x_4.

$|x_1| = \max\{x_1; -x_1\}$
$|x_2| = \max\{x_2; -x_2\}$

$$\text{Maximize } |x_1| + |x_2|$$

subject to the restrictions

4.5b

1. at least one of these two constraints must hold:
   - $x_1 + x_2 + x_3 + x_4 \leq 1000$
   - $3 \cdot x_1 + 5 \cdot x_2 + x_3 + 2 \cdot x_4 \leq 500$
2. x_1 is at least $\max\{2 \cdot x_2, x_3\}$.
3. $\frac{x_1}{x_4}$ equals to 4. $x_1 = 4x_4$; $(x_4 \neq 0) \rightarrow (x_4 > 0) \vee (x_4 < 0)$ není nutné díky
4. $x_{i\in\{1,2,3,4\}} \in \mathbb{Z}$
5. $-5000 \leq x_{i\in\{1,2,3\}}$, $x_4 \geq \cancel{1}$, $x_{i\in\{1,2,3,4\}} \leq 1000$.

Formulate by the **Integer Linear Programming**.

konstanta M
$M = 4 \cdot 1000$

Vars:
$y \in \{0,1\}$
~~$z \in \{0,1\}$~~

4.)
$x_{1,\dots,4} \in \mathbb{Z}$
$|x_1| ; |x_2| \in \mathbb{Z}$

Constraints:

1.) $x_1 + x_2 + x_3 + x_4 \leq 1000 + My$ 1.5b
$3x_1 + 5x_2 + x_3 + 2x_4 \leq 500 + M(1-y)$

2.) $x_1 \geq 2 \cdot x_2$ 1b
$x_1 \geq x_3$

3.) $x_1 = 4 \cdot x_4$ 1b

~~x_4 $\Rightarrow$ $y \cdot M + x_4 > 0$~~
~~$x_4 < 0 + (1-y)M)$~~

Obj cstr: $|x_1| \geq x_1$; $|x_1| \geq -x_1$; $|x_2| \geq x_2$; $|x_2| \geq -x_2$ X

5.) $-5000 \leq x_1$; $-5000 \leq x_2$; $-5000 \leq x_3$; $x_4 \geq 1$
$x_1 \leq 1000$; $x_2 \leq 1000$; $x_3 \leq 1000$; $x_4 \leq 1000$ 1b

Obj: maximize $|x_1| + |x_2|$

Zkouškový materiál:

Zkoušky z roku 2023

Zdrojový soubor: KO Zkousky 2023.pdf

Původ: Staženo ze zip archivu 'zkousky.zip' sdíleného na Discordu.

KO Zkousky 2023

29.5.

1. ILP - max absolutni hodnoty
2. ILP - loot distribution
3. Formulace $2P|r_j|d_j|preempt$ pomocí max flow
4. Důkaz korektnosti Djikstry
5. Knapsack
6. Nakreslit graf tasků s TC

12.6.

1. Pouziti floyda pro nalezeni nejdelsich cest mezi vsemi uzly v grafu, ukazat 3 iterace.
2. Dukaz aproximacniho faktoru pro Christophidesuv algoritmus, pseudokod zadany.
3. Scheduling za pomoci Chetto, silly, bouchentouff
4. Nakreslit graf pro ulohu multikomoditnich toku, to bylo docela random a Hanzalek v pulce zkousky prisel na to ze to nejde vyresit tak to zjednodusil a pak zjistil, ze ani ta zjednodusena verze nejde poradne vyresit :DD
5. ILP pro max flow plus extra podminky (formulace logickych podminek typu OR, implikace, atd. podobne jako u toho ILP s investicema do baraku)
6. time indexed ILP formulation for $PS1|temp|C_{max}$

29.6.

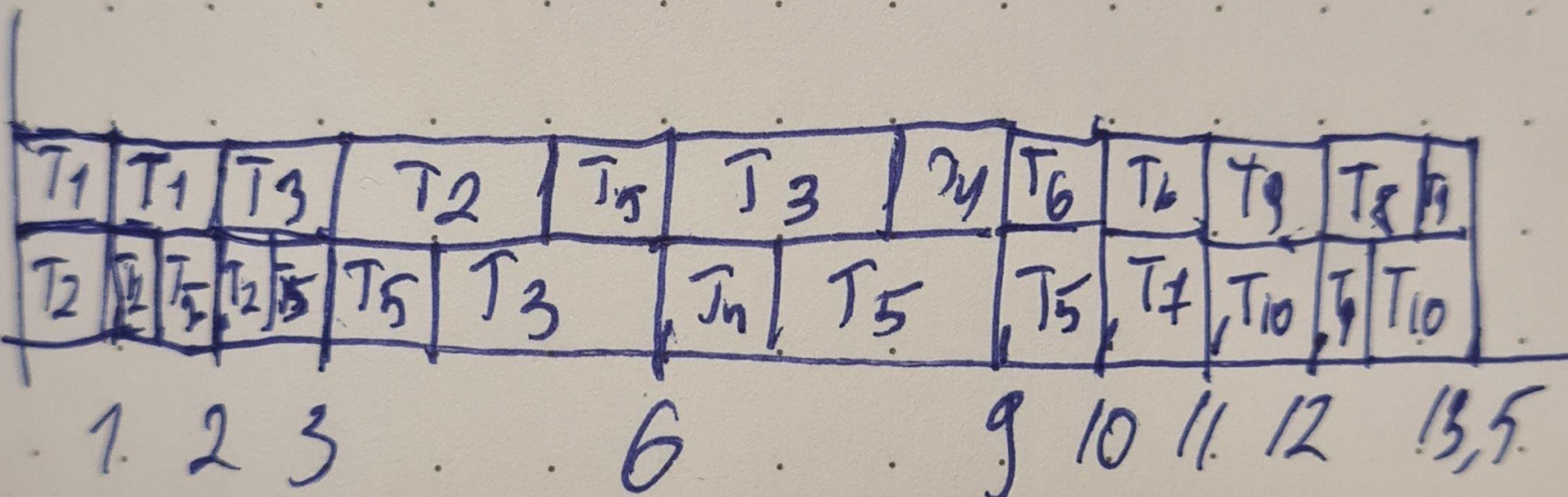
1. ILP - svatba, maximalizace poctu hostu, ale nesmi se prekrocit budget B, za kazdeho hosta stoji jidlo m a pronajem prostoru je podle poctu hostu p_1 pro $[0, 9]$, p_2 pro $[10, 19]$, p_3 pro $[20, n]$, dale jsou funkce $br(x)$ a $gr(x)$, ktere vrati 1 pokud je dany clovek bud ze strany nevesty nebo zenicha (muze byt oboji) a rozdil mezi poctem hostu ze strany nevesty a zenicha muze byt max D_{max}
2. SPT - graf kde ma kazda hrana w - delku cesty na vzdalenost a τ - delku cesty na cas, formulovat jako problem nejkratsich cest (nakreslit priklad grafu), ve kterem pujde najit cesta ktera se dostane z vrcholu s do vrcholu t a nebude ji to trvat vic nez cas T, (rozvinuti grafu do casove osy)
3. FLOWS - Ford-Fulkerson s inicialnim tokem
    1. a) rucne to doiterovat a v kazde iteraci ukazat zlepsujici cestu
    2. b) rict kolik bude max potreba iteraci pri danem inicialnim toku pro libovolne zvolene poradi zlepsujicich cest
4. TSP - dukaz pro neexistenci r-aproximacniho pro obecne TSP (polynomialnim prevodem z problemu hamiltonovskeho cyklu)
5. SCHED - Hornuv algoritmus
6. SCHED - ILP formulace pro $1|prec|\sum w_j C_j$, rict jak se LP relaxace toho ILP da pouzit pri branch and bound algoritmu

Zkouškový materiál:

Zkouškové fotky (zadání, řešení, poznámky na papír)

Zdrojový soubor: Zkouskove_fotky.pdf

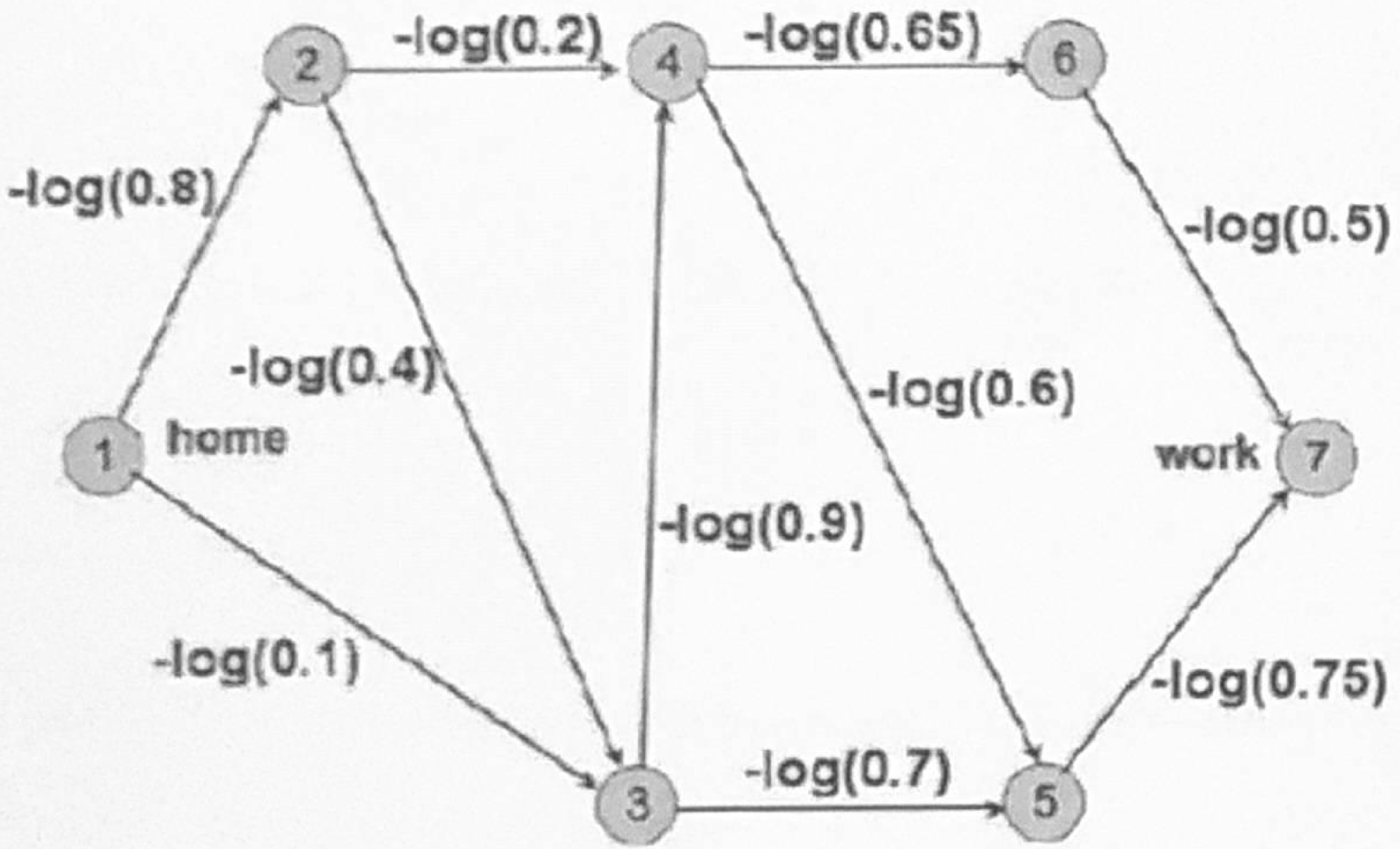
Původ: Generováno umělou inteligencí z posbíraných fotek zkoušek a poznámek studentů z května/června 2025.


T1
T1
T3
T2
T3
T6
T2
T5
T3
T5
T5
T7
T10
T10
1
2
3
6
9
10
11
12
13,5

$$
\begin{aligned}
\max\ z = \quad & z_1 + z_2 & \\
s.t. \quad & x_1 + M \cdot y_1 & \geq z_1, \\
& -x_1 + M \cdot (1 - y_1) & \geq z_1, \\
& x_2 + M \cdot y_2 & \geq z_2, \\
& -x_2 + M \cdot (1 - y_2) & \geq z_2, \\
& x_1 + x_2 + x_3 + x_4 & \leq 1000 + M \cdot y_3, \\
& 3 \cdot x_1 + 5 \cdot x_2 + x_3 + 2 \cdot x_4 & \leq 500 + M \cdot (1 - y_3), \\
& x_1 & \geq 2 \cdot x_2, \\
& x_1 & \geq x_3, \\
& x_1 & = 4 \cdot x_4, \\
& x_{i \in \{1,2,3,4\}} & \in \mathbb{Z} \\
& z_{j \in \{1,2\}} & \in \mathbb{R}_{\geq 0} \\
& y_{k \in \{1,2,3\}} & \in \{0, 1\} \\
& -5000 \leq x_{i \in \{1,2,3\}},\ x_4 \geq 1,\ x_{i \in \{1,2,3,4\}} & \leq 1000,
\end{aligned}
$$

$$
\begin{aligned}
\max\ z = \quad & z_1 + z_2 & & \\
s.t. \quad & x_1 + M \cdot y_1 & & \geq z_1, \\
& -x_1 + M \cdot (1 - y_1) & & \geq z_1, \\
& x_2 + M \cdot y_2 & & \geq z_2, \\
& -x_2 + M \cdot (1 - y_2) & & \geq z_2, \\
& x_1 + x_2 + x_3 + x_4 & & \leq 1000 + M \cdot y_3, \\
& 3 \cdot x_1 + 5 \cdot x_2 + x_3 + 2 \cdot x_4 & & \leq 500 + M \cdot (1 - y_3), \\
& x_1 & & \geq 2 \cdot x_2, \\
& x_1 & & \geq x_3, \\
& x_1 & & = 4 \cdot x_4, \\
& x_{i \in \{1,2,3,4\}} & & \in \mathbb{Z} \\
& z_{j \in \{1,2\}} & & \in \mathbb{R}_{\geq 0} \\
& y_{k \in \{1,2,3\}} & & \in \{0, 1\} \\
& -5000 \leq x_{i \in \{1,2,3\}},\ x_4 \geq 1,\ x_{i \in \{1,2,3,4\}} \leq 1000, & &
\end{aligned}
$$

We have to multiply probabilities that Mr.Krocan will NOT be stopped on a previous road AND on the subsequent road, i.e., we change probabilities on roads to $p'_{ij} = 1 - p_{ij}$. Then we have to maximize the probability that he will NOT be stopped.
To formulate the problem by SPT we set $c_{ij} = -logp'_{ij}$.



The best path is [home - 2 - 3 - 5 - work] with probability about 16.8% that mr. Krocan will not be stopped.

Ford-Fulkerson Algorithm - Solution

Flow04b

a) Maximum flow is 9.

b) The bound on the termination from given initial flow is 4 – each augmenting path has capacity at least one, the initial flow is 5 and the upper bound on the maximum flow is 9 (easily seen from the upper bounds from source edges).

Approximation Factor of Christofides' Algorithm - Solution

TSP03a

It is a $\frac{3}{2}$ approximation algorithm for the **metric TSP**:

- 1. due to the triangle inequality the skipped nodes do not prolong the route, i.e $c(E(L)) \geq c(E(H))$
- 2. while deleting one edge in the circuit, we create the tree. Therefore, inequality $OPT(K_n, c) \geq c(E(T))$ holds
- 3. since the perfect matching M considers every second edge in the alternating path and being the minimum weight matching it chooses the smaller half, $\frac{OPT(K_n,c)}{2} \geq c(M)$ holds
- 4. due to the construction of L it holds $c(M) + c(E(T) = c(E(L))$
- finally we obtain:

$$\frac{3}{2} OPT(K_n, c) \overset{2.,3.}{\geq} c(E(T)) + c(E(M)) \overset{4.}{=} c(E(L)) \overset{1.}{\geq} c(E(H))$$

$X = \{x_1, x_2, x_3\}$, constraints $x_1 < x_2,\ x_2 = x_3 - 1$, domains

$D_1 = \{1, 2, 3\}, D_2 = \{1, 2, 3\}, D_3 = \{1, 2, 3\}$

Initialization: $Q = \{(x_1, x_2), (x_2, x_1), (x_2, x_3), (x_3, x_2)\}$

revise (x_1, x_2)

$D_1 = \{1, 2\}^{1)}, D_2 = \{1, 2, 3\}, D_3 = \{1, 2, 3\}$

$Q = \{(x_2, x_1), (x_2, x_3), (x_3, x_2)\}$

revise (x_2, x_1)

$D_1 = \{1, 2\}, D_2 = \{2, 3\}^{2)}, D_3 = \{1, 2, 3\}$

$Q = \{(x_2, x_3), (x_3, x_2)\}$

revise (x_2, x_3)

$D_1 = \{1, 2\}, D_2 = \{2\}^{3)}, D_3 = \{1, 2, 3\}$

$Q = \{(x_3, x_2), (x_1, x_2)\}$

revise (x_3, x_2)

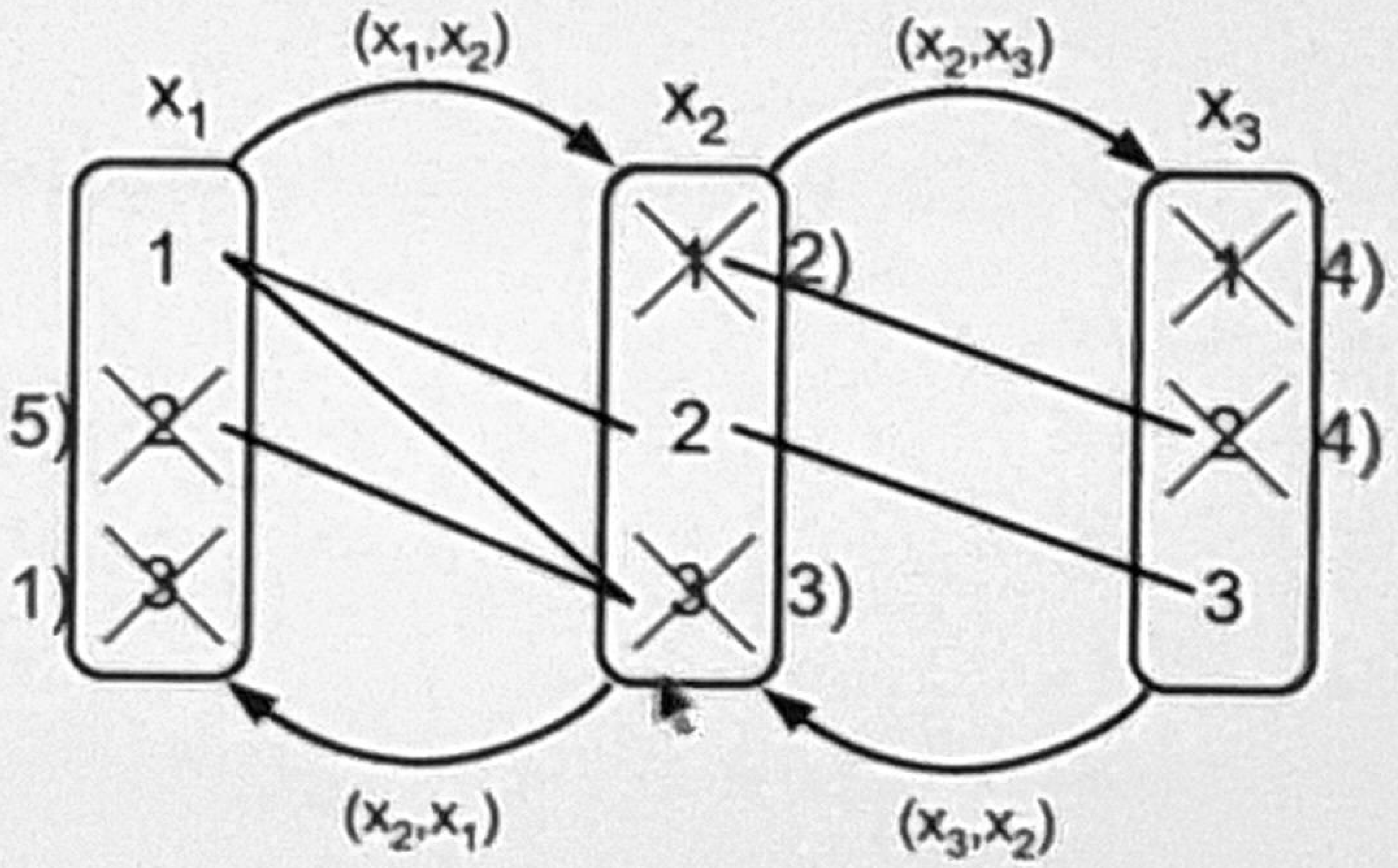
$D_1 = \{1, 2\}, D_2 = \{2\}, D_3 = \{3\}^{4)}$

$Q = \{(x_1, x_2)\}$

revise (x_1, x_2)

$D_1 = \{1\}^{5)}, D_2 = \{2\}, D_3 = \{3\}$

$Q = \emptyset$



Time-indexed ILP Model for $PS1|temp|C_{max}$ - Solution

SCH07a

$$\min C_{max}$$

$$\begin{array}{ll}
\sum_{t=0}^{UB-1} (t \cdot x_{it}) + l_{ij} \leq \sum_{t=0}^{UB-1} (t \cdot x_{jt}) & \forall l_{ij} \neq -\infty \text{ a } i \neq j \text{ prec. const.} \\
\sum_{i=1}^{n} \left(\sum_{k=\max(0,t-p_i+1)}^{t} x_{ik} \right) \leq 1 & \forall t \in \{0, \dots UB-1\} \text{ resource} \\
\sum_{t=0}^{UB-1} x_{it} = 1 & \forall i \in \{1, \dots n\} \;\; T_i \text{ is scheduled} \\
\sum_{t=0}^{UB-1} (t \cdot x_{it}) + p_i \leq C_{max} & \forall i \in \{1, \dots n\}
\end{array}$$

variables: $x_{it} \in \{0,1\}$, $C_{max} \in \{0, \dots UB\}$

UB - upper bound of C_{max} (e.g. $UB = \sum_{i=1}^{n} \max\left\{p_i, \max_{i,j\in\{1,\dots,n\}} l_{ij}\right\}$)

Start time of T_i is $s_i = \sum_{t=0}^{UB-1} (t \cdot x_{it})$.

Model contains $n \cdot UB + 1$ variables and $|E| + UB + 2n$ constraints. Constant $|E|$

4-Partition Problem - Solution

ILP03a

- $x_{ij} = 1$ iff banknote $i \in S_j$

$$
\begin{array}{lll}
\text{min} & 0 & \\
\text{subject to:} & & \\
& \sum_{i \in 1..n} x_{ij} * p_i = h_j & j \in 1..4 \\
& \sum_{j \in 1..4} x_{ij} = 1 & i \in 1..n \\
& h_{min} \leq h_j & j \in 1..4 \\
& h_{max} \geq h_j & j \in 1..4 \\
& h_{max} - h_{min} \leq 0.1 & \sum_{i \in 1..n} p_i \\
\text{parameters:} & p_{i \in 1..n} \in \mathbb{Z}^+ & \\
\text{variables:} & x_{i \in 1..n, j \in 1..4} \in \{0, 1\} \; h_{max}, h_{min}, h_{j \in 1..4} \in \mathbb{Z}^+ &
\end{array}
$$

Bratley's algorithm - Solution

SCH03b

a) Schedule:

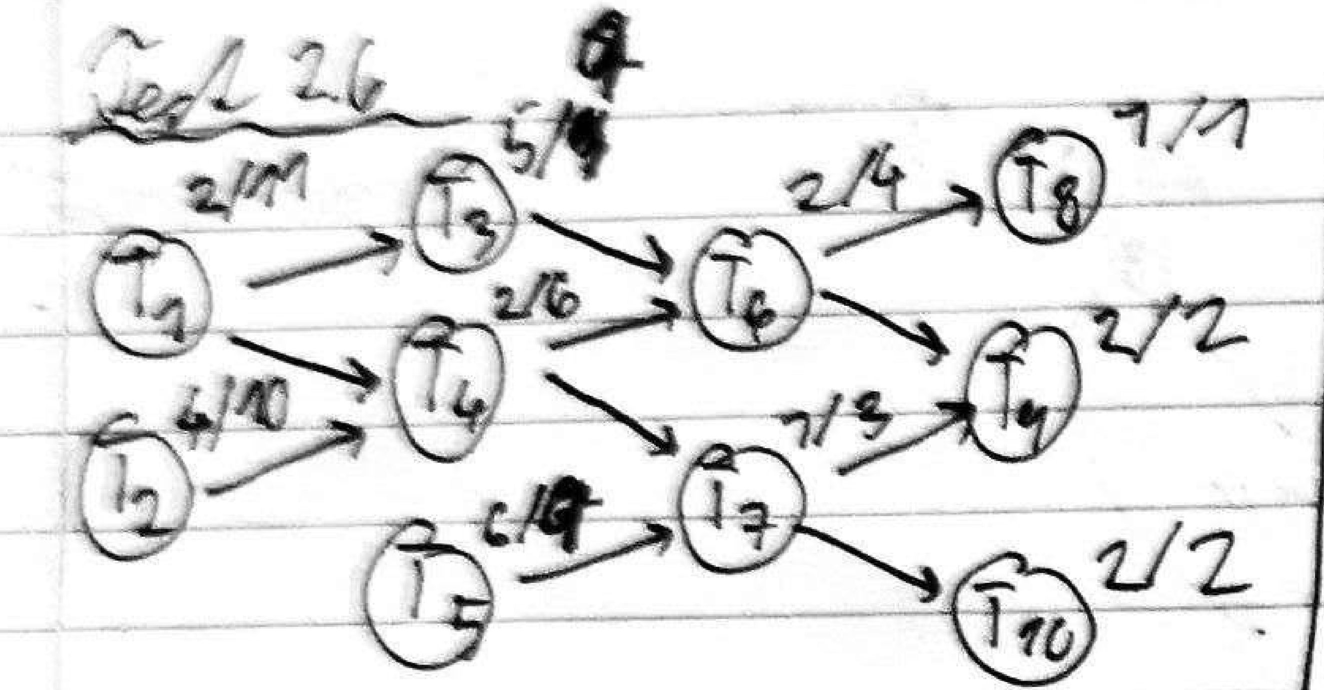
P_1	T_4	T_2	T_2	–	T_3	T_3	T_1	T_1	T_1	
t	0	1	2	3	4	5	6	7	8	9

$C_{max} = 9$

b) This solution is optimal since there is BRTP which starts at $r_3 = 4$.

For polynomially bounded R, (i.e., $R = poly(n)$) the problem is

Test 2.6



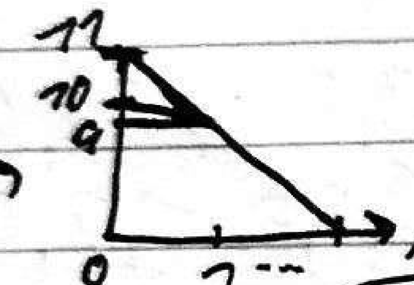
$P2|pmtn, prec|C_{max}$

$t=0$ $\quad r=(2,4,5,2,6,2,1,1,2,2)$ $\quad Z=\{T_1,T_2,T_5\}$

$\quad l=(11,10,9,6,9,4,3,1,2,2)$

$h=2$ $\quad S=\{T_1\}$ $\quad \beta=1$

$h=1$ $\quad S=\{T_2\}$ $\quad \beta=1$



! $g=1 \rightarrow$ case ②

$t=1$ $\quad r=(1,3,5,2,6,2,1,1,2,2)$ $\quad Z=\{T_1,T_2,T_5\}$

$\quad l=(10,9,9,6,9,4,3,1,2,2)$

$h=2$ $\quad S=\{T_1\}$ $\quad \beta=1$

$h=1$ $\quad S=\{T_2,T_5\}$ $\quad \beta=\frac{1}{2}$

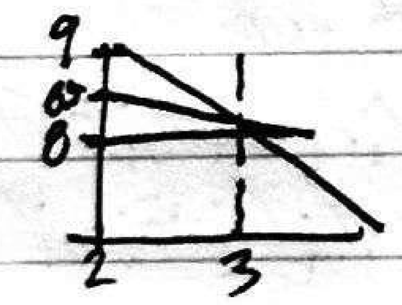
$g=1 \rightarrow$ dodělá se task T_1

$t=2$ $\quad r=(\_;2.5;5;2;5.5;2;1;1;2;2)$ $\quad Z=\{T_2,T_3,T_5\}$ (8.5, 9, 8.5)

$\quad l=(\_;8.5;9;6;8.5;4;3;1;2;2)$

$h=2$ $\quad S=\{T_3\}$ $\quad \beta=1$

$h=1$ $\quad S=\{T_2,T_5\}$ $\quad \beta=\frac{1}{2}$



! $g=1 \rightarrow$ case 2 znovu...

$t=3$ $\quad r=(\_;2;4;2;5;2;1;1;2;2)$ $\quad Z=\{T_2,T_3,T_5\}$ (8, 8, 8)

$\quad l=(\_;8;8;6;8;4;3;1;2;2)$

$h=2$ $\quad S=\{T_2,T_3,T_5\}$ $\quad \beta=\frac{2}{3}$ $\rightarrow h=0$

$g=3 \rightarrow$ dokončí se task T2

$t=6$ $\quad r=(\_,\_,2,2,3,2,1,1,2,2)$ $\quad Z=\{T_3,T_4,T_5\}$ (6, 6, 6)

$\quad l=(\_,\_,6,6,6,4,3,1,2,2)$

$h=2$ $\quad S=\{T_3,T_4,T_5\}$ $\quad \beta=\frac{2}{3}$ $\rightarrow h=0$

$g=3 \rightarrow$ dokončí se task T_3 a T_4

$t=9$ $\quad r=(\_;\_;\_;\_;1;2;1;1;2;2)$ $\quad Z=\{T_5,T_6\}$ (4, 4)

$\quad l=(\_;\_;\_;\_;4;4;3;1;2;2)$

$h=2$ $\quad S=\{T_5,T_6\}$ $\quad \beta=1$ $\rightarrow h=0$

$g=1 \rightarrow$ dodělá se task T_5

$t=10$ $\quad r=(\_,\_,\_,\_,\_,1,1,1,2,2)$ $\quad Z=\{T_6,T_7\}$ (3, 3)

$\quad l=(\_,\_,\_,\_,\_,3,3,1,2,2)$

$h=2$ $\quad S=\{T_6,T_7\}$ $\quad \beta=1$ $\rightarrow h=0$

$g=1 \rightarrow$ dodělá se task T_6 a T_7

$t=11$ $\quad r=(\_,\_,\_,\_,\_,\_,\_,1,2,2)$ $\quad Z=\{T_8,T_9,T_{10}\}$ (1, 2, 2)

$\quad l=(\_,\_,\_,\_,\_,\_,\_,1,2,2)$

$h=2$ $\quad S=\{T_9,T_{10}\}$ $\quad \beta=1$ $\rightarrow h=0$

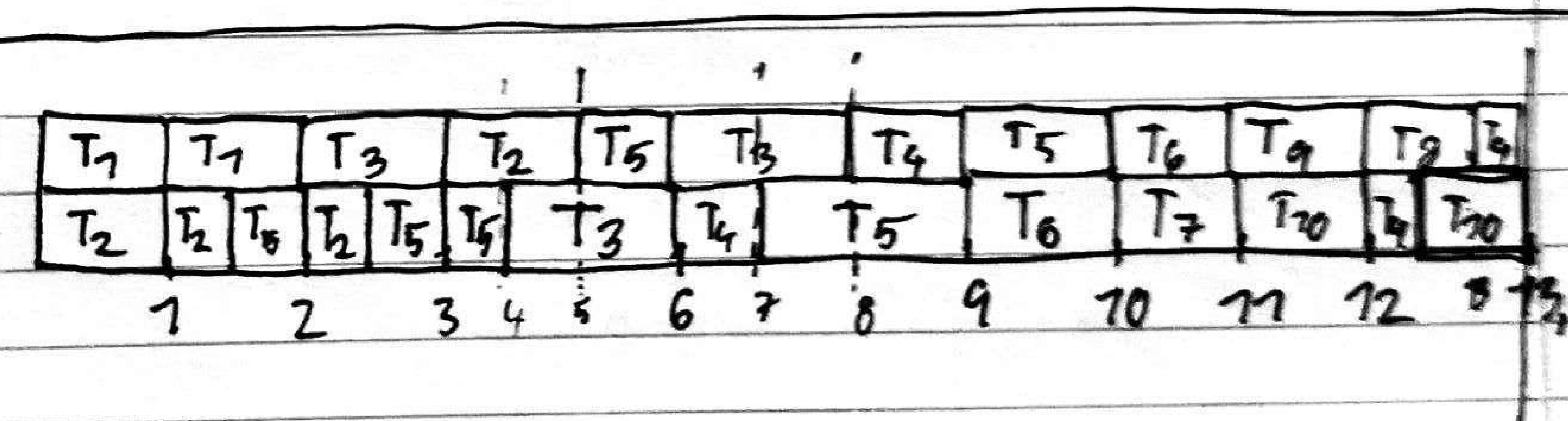
$g=1 \rightarrow$ case 2

$t=12$ $\quad r=(\_,\_,\_,\_,\_,\_,\_,1,1,1)$ $\quad Z=\{T_8,T_9,T_{10}\}$

$\quad l=(\_,\_,\_,\_,\_,\_,\_,1,1,1)$

$h=2$ $\quad S=\{T_8,T_9,T_{10}\}$ $\quad \beta=\frac{2}{3}$ $\rightarrow h=0$

$g=\frac{3}{2} \rightarrow$ dodělá se T_8,T_9,T_{10} $\rightarrow$ konec $\rightarrow$ $t=13{,}5$




T1 T1 T3 T2 T5 T3 T4 T5 T6 T9
T2 T2 T5 T2 T5 T5 T3 T4 T5 T6 T7 T10 T9 T10
1 2 3 4 5 6 7 8 9 10 11 12 13 13,5

Knapsack v testu

$n = 7$ $\quad$ $W = 5$

$I = (1, 2, 3, 4, 5, 6, 7)$

$c = (2, 2, 2, 2, 4, 3, 7)$

$w = (1, 1, 2, 2, 3, 4, 1)$

a)

$I_n \backslash w$	0	1	2	3	4	5
0	0	$-\infty$	$-\infty$	$-\infty$	$-\infty$	$-\infty$
1	0	$\boxed{2}$	$-\infty$	$-\infty$	$-\infty$	$-\infty$
2	0	2	$\boxed{4}$	$-\infty$	$-\infty$	$-\infty$
3	0	2	4	4	6	$-\infty$
4	0	2	$\boxed{4}$	4	6	6
5	0	2	4	4	6	$\boxed{8}$
6	0	2	4	4	6	8
7	0	2	4	5	6	8

$I = (I_1, I_2, I_5)$

Je to jedinečné, protože během zpětného trasování jsme v každém kroku měli jen jednu možnost jak dosáhnout hodnoty v předch. řádku

b) What can you say about the computational complexity of the algorithm for instances where $W \leq 10n$

Stand. alg má čas. náročnost $O(n \cdot W)$

Protože je dané, že $W \leq 10n$, tak platí $O(n \cdot 10n) = O(10n^2)$

Čas. složitost bude tedy $O(n^2)$ a stále polynomiální

- *Nejkratší cesty v obecném grafu.* Je dán orientovaný graf s ohodnocením hran a. Jsou dány vrcholy r a v. Je dáno číslo k. Existuje orientovaná cesta z vrcholu r do vrcholu v délky menší nebo rovno k?

Knapsack

Using dynamic programming, solve the following instance of Knapsack Problem:

- number of items: $n = 7$
- knapsack capacity: $W = 5$
- costs $\mathbf{c} = (2, 2, 2, 2, 4, 3, 1)$
- weights: $\mathbf{w} = (1, 1, 2, 2, 3, 4, 1)$

66

a) Compute the optimal solution (objective value and items in knapsack) of this instance of Knapsack Problem. Write down all iterations of the algorithm. Is this solution unique and why?

b) What can you say about the computational complexity of the algorithm for instances where $W \leq 10n$?

předmět buďto vložit, nebo nevložit

5,5

Constrained Shortest Path Problem

In a network G we associate two numbers with each arc: its length $c_{ij} \in \mathbb{R}$ and its traversal time $\tau_{ij} \in \mathbb{Z}_0^+$. We would like to determine a shortest-length path from the source node s to the sink node t with the additional constraint that the traversal time of the path does not exceed τ.

a) Formulate this problem as a shortest path problem (hint: use time expansion of the network).

b) Does the path necessarily exist if the network G is strongly connected?

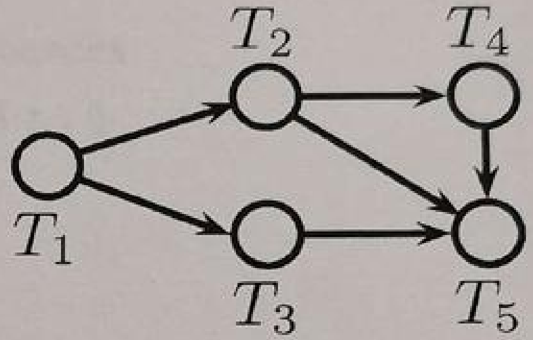
all paths

) it doesnt necessarily exist, its possible that

Chetto, Silly, Bouchentouf algorithm for $1 \left| \text{pmtn, prec}, r_j, d_j = \widetilde{d}_j \right| L_{max}$

a) Using Chetto, Silly, Bouchentouf algorithm solve the following instance of mono-processor scheduling of preemptive tasks with precedence relations, release dates and due-dates equal to deadlines while minimizing the maximum lateness. Indicate values of main variables in separate steps of the algorithm. Draw the Gantt chart.

$r = (0, 3, 2, 6, 3)$
$p = (2, 3, 2, 2, 3)$
$\widetilde{d} = d = (3, 7, 13, 9, 13)$



b) Calculate the L_{max} value of the optimal solution.

3. Ústní zkouška

Ústní zkouška se typicky koná den nebo dva po písemce (např. písemka v pondělí, ústní ve středu od 13:00). * **Povinnost:** Ústní probíhá tak, že by se tam měl ukázat každý (minimálně přijít sdělit, že se nechce nechat zkoušet a bere body z testu). * **Průběh:** Zkoušející (prof. Hanzálek) zadá otázky ze seznamu (většinou 4 otázky). Každý student si vezme papír, sedne si a může se libovolně dlouho připravovat, zatímco zkoušející odbavuje ostatní. * **Hodnocení:** Za každou otázku lze získat většinou cca 3 body. Hodnocení je vstřícné, většině studentů zkoušející najde body a známku podstatně zlepší.

Konkrétní otázky z ústní zkoušky (zápisky studentů):

- Branch & Bound (algoritmus / princip)
- Floyd-Warshallův algoritmus
- Bratleyho algoritmus (Scheduling)
- Formulace Shortest Path Tree (SPT) jako ILP / pomocí Floyda
- Flows – zaokrouhlování hodnot v tabulce
- Scheduling – formulace pro problém typu `P | | Cmax`

Všechny související fotky zadání, řešení z písemek a studijní PDF (jako např. známý Dumbshit guide nebo archiv starých zkoušek) jsou roztříděny ve složce `exam/` a `exam/photos/`.