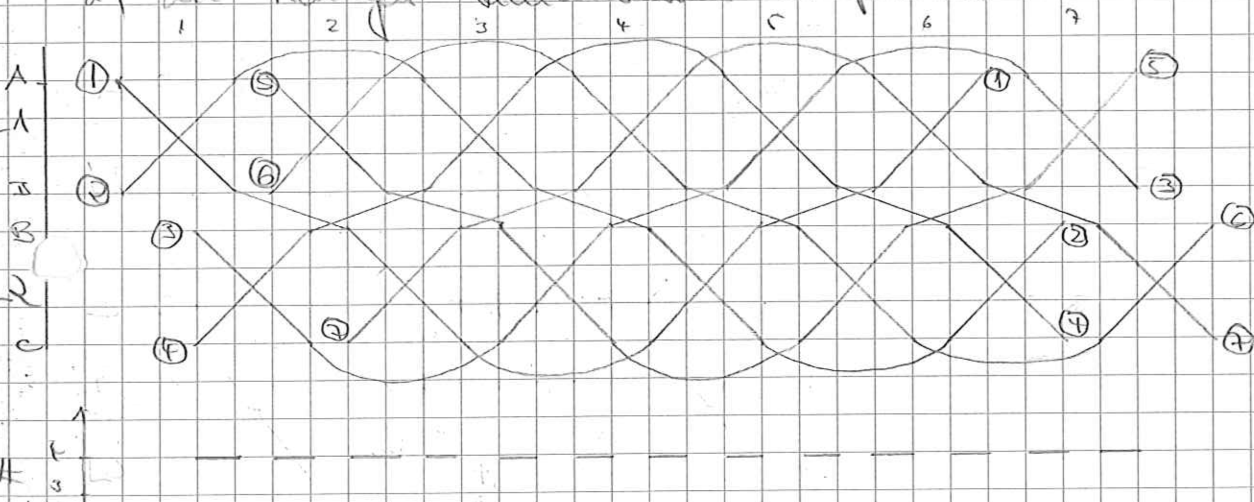
1. Umlaufplanung

Motivation: Fahrzeugkosten machen etwa 20-25% der operativen Kosten eines Nahverkehrsunternehmens aus:

Kosten	städt. DM	%	regional DM	%
Abschreibung	35.400	7,4	31.000	10,7
kalk. Zins	15.300	3,2	12.900	4,5
Materialien	14.000	2,9	10.000	3,5
Treibstoff	22.200	4,7	18.000	6,2
Instandhaltung	5.000	1,0	5.000	1,7
Fahrzeuge	91.900	19,3	75.900	26,3
Fahrer	349.600	73,5	195.000	67,5
Sonstiges	34.000	7,2	18.000	6,2
Summe	475.500	100,0	288.900	100,0

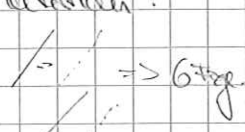
Idee: Bei der Umlaufplanung geht es darum,

a) die richtigen Linienwechsel zu finden



„Leerfahrt wurde nicht möglich"

Überlegen:

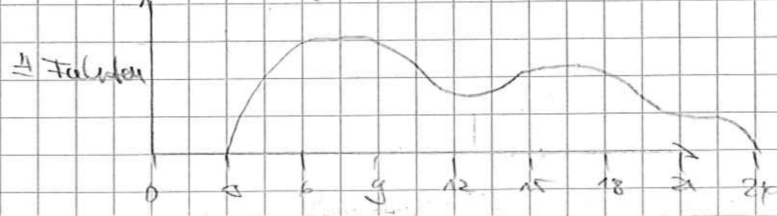


Schwankung verb. durch Wendezeiten

Fzg.bedarf ohne Linienwechsel: 8

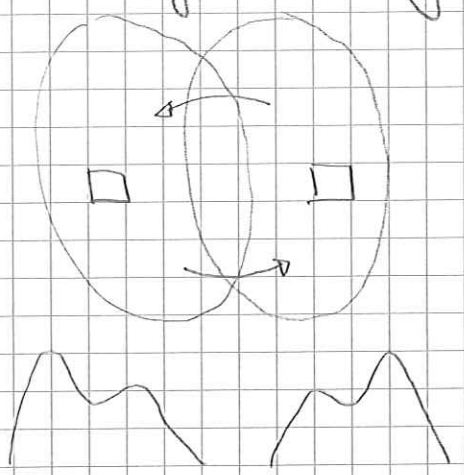
" mit " : 7

b) den Fzg.bedarf in der Spitze möglichst klein zu halten



„Ganglinie"

c) die richtige Zuordnung von Fahrten zu Fahrtenhöfen zu finden.



Bsp (untere Schranke):

$$\max_{t \in T} \#\text{Fahrten}(t) \leq \text{min. \# Fahrtenhöfe}$$

Diese Schranke ist, wie Bsp. a) zeigt, i.a. nicht scharf

2. Zuordnungsmethode

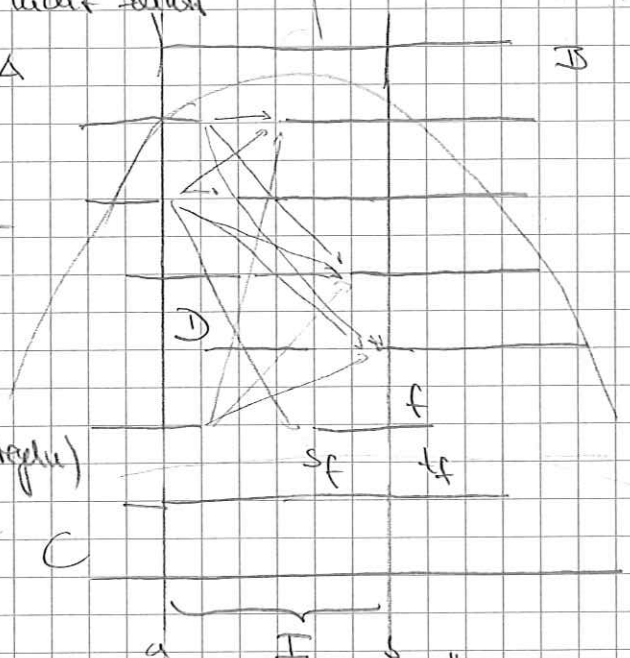
Idee: Teile in der Ideallösung eine Fahrzeugumlaufzuordnung von „Nachfolgefahrten" ab.

Alg. (Zuordnungsmethode).

Input: Menge von Fahrgastfahrten $\mathcal{F}$ (mit Vorsch. Regeln)

Start- und Endzeit s_f, t_f, $f \in \mathcal{F}$

Zeitintervall $I = [a, b]$



Output: Aufteilung von Fahrten in $\mathcal{F}$

1. $\mathcal{D} \leftarrow \{f \in \mathcal{F} : a \leq s_f \leq t_f \leq b\}$ „in I enthaltene Fahrten"

   $\mathcal{F} \leftarrow \mathcal{F} \setminus \mathcal{D}$

2. $C \leftarrow \{f \in \mathcal{F} : s_f \leq a \leq b \leq t_f\}$ „Fahrten, die I enthalten"

   $\mathcal{F} \leftarrow \mathcal{F} \setminus C$

3. $\mathcal{D}_A \leftarrow$ sinnvolle Teilmenge von $\mathcal{D}$ „verteile C auf A und B"

   $\mathcal{D}_B \leftarrow \mathcal{D} \setminus \mathcal{D}_A$

4. $A \leftarrow \{f \in \mathcal{F} : t_f \leq b\} \cup \mathcal{D}_A$ „in I hineinf. Fahrten"

   $B \leftarrow \{f \in \mathcal{F} : a \leq s_f\} \cup \mathcal{D}_B$ „aus I herausf. Fahrten"

5. $E \leftarrow \{fg \in A \times B,\ g \text{ kann auf } f \text{ folgen}\}$

   $G \leftarrow (A \cup B, E)$ „bipartiter Zuordnungsgraph"

6. $M \leftarrow$ max. Matchung in G

$A' \leftarrow \{ f \in A : \exists g : \{f,g\} \in M \}$

$B' \leftarrow \{ g \in B : \exists f : \{f,g\} \in M \}$

7. Gib A', B', M, C aus, dann Trg bed. ist $|F| - |M|$.

Prop.: Sei $[s_f, t_f] \cap \{a,b\} \neq \emptyset \quad \forall f \in F$ (*). Dann ist $|F| - |M|$ eine untere Schranke für das Fahrzeugbedarf in I.

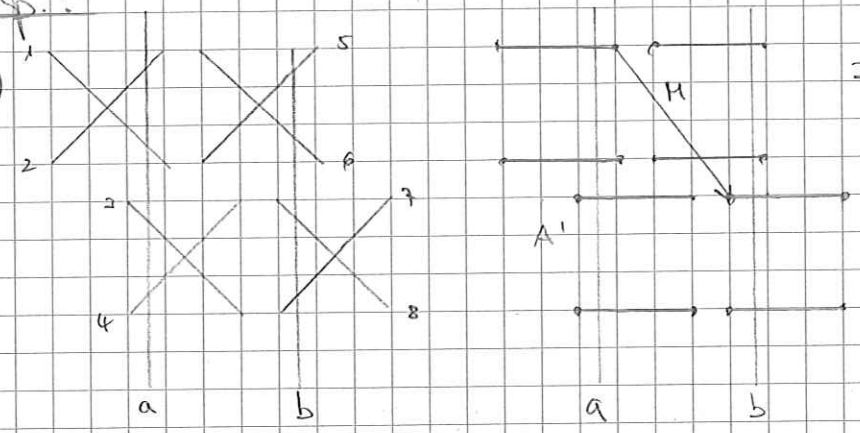
Bew.:

$$(*) \Rightarrow \begin{cases} B = \emptyset \\ s_f \leq a \leq t_f \quad \forall f \in A \\ s_f \leq b \leq t_f \quad \forall f \in B. \end{cases}$$

Der Trg.bedarf ist $|C| + |A| + |B| - |M| = |F| - |M|$. □

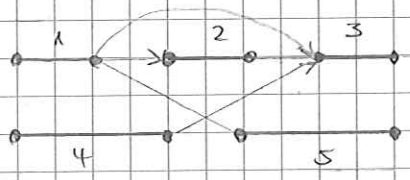
Bsp.:

a)



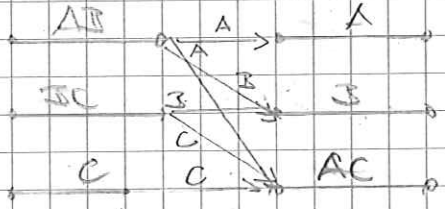
Zerlegungsmethode findet mit geeignetem Intervall I den min. Trg. bedarf.

b)



Wegen $|F| - |M| \leq 5 - 2 = 3$ liefert die Zerlegungsmethode hier den min. Trg. bedarf.

Bem. a) Durch geeignete Def. von Teilaufgaben können auch die Zerlegungsmethoden durch Trg.typen behandelt werden:



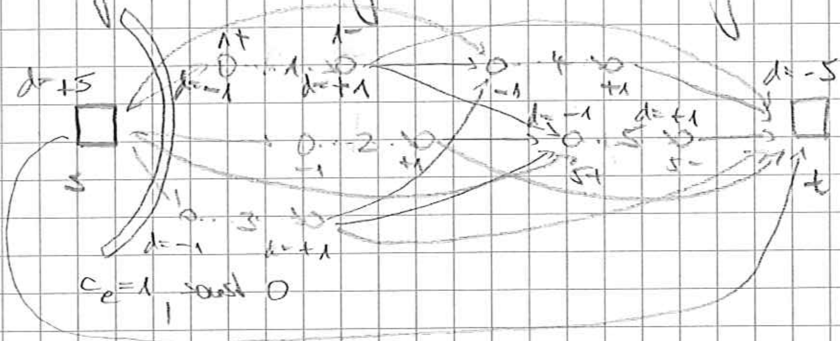
b) Die Zerlegungsmethode ist die wohl erfolgreichste Methode zur Lösung großer Umlaufplanungsprobleme. Sie wurde in den Anfangs-Jahren

in dem System HOT auf Großkundenbasis umgesetzt.

c) Direkte Def. von Kosten der Form $M \pm w_{fg}$, wobei M sehr groß ist, und Berechnung eines gewichteten Matchings können die Fzg.zahl minimiert und weitere Zielkriterien optimiert werden.

3. Ein-Depot-Umlaufplanung

Idee: Globale Planung durch Berechnung eines minimalen „Fzg.flusses".



azyklischer Digraph

Alg (Ein-Depot-Methode)

Input: Menge F von Fahrgastfahrten mit Start- und Endzeitpunkt $s_f, t_f \in \mathbb{R}$ $\forall f \in F$, $s_f < t_f$

Menge $M \subseteq F \times F$, $t_f \leq s_g$ $\forall fg \in M$ mögliche Fahrtverknüpfungen

Output: Menge K von disjunkten Fahrzeugketten (d.h. Folge von Fahrten und Verkn., Start und Ende eines Umlaufs), die alle Fahrten enthalten, mit minimaler Kardinalität.

1. $F^+ \leftarrow \{f^+, f \in F\}$, $F^- \leftarrow \{f^-, f \in F\}$

   $V \leftarrow \{s, t\} \dot\cup F^+ \dot\cup F^-$

   $A \leftarrow s \times F^+ \cup t \times F^- \cup \{f^- g^+ : fg \in M\} \cup \{st\}$

   $D \leftarrow (V, A)$

   $c_{uv} \leftarrow \begin{cases} 1, & uv = sf^+ \\ 0, & \text{sonst} \end{cases}$

   $d_v \leftarrow \begin{cases} |F|, & v = s \\ -|F|, & v = t \\ 1, & v = f^- \\ -1, & v = f^+ \end{cases}$

2. $x \leftarrow$ Opt. Min cost flow in D bzgl. Bedarf d und Kosten c

3. $P \leftarrow$ Pfadzerl. von x

4. $K_p \leftarrow (f_1, \dots, f_k)$ $\forall p = (s, f_1^+, f_1^-, f_2^+, f_2^-, f_3^+, \dots, f_{k-1}^-, f_k^+, f_k^-, t) \in P$

   $K \leftarrow \{K_p : p \in P\}$

5. Gib K aus.

Bew.: a) Wegen $s_f < t_f \; \forall f \in F$ und $t_f < s_g \; \forall fg \in M$ ist D azyklisch.

b) $|K| = |\{g \in F : \nexists f \in F, p \in T : f^- g^+ \in p\}|$ „Fahrten ohne Vorgängerfahrt"

Das Umlaufplan-Unterauflösungsproblem kann deshalb auch als Zuordnungsproblem in einem geeigneten bipartiten Graphen gelöst werden:

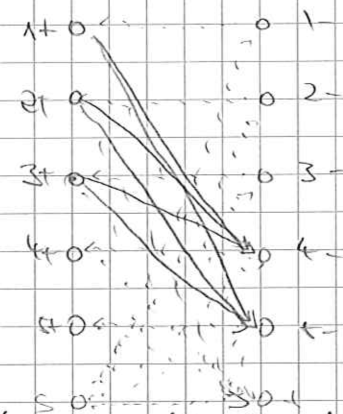
$U = F^- \dot\cup F^+$

$E = \{f^- g^+ : fg \in M\}$

$G = (U, E)$

$Y \leftarrow$ max. Matching in G

$K \leftarrow \{(f_1, \ldots, f_k)$ max. mit $f_i^- f_{i+1}^+ \in Y\}$



Die Ein-Depot-Planung stellt in diesem Sinne eine Erweiterung der Zuordnungsmethode dar.

08.01.07 (13)

3 Mehrdepotplanung

Idee: Behandlung von Fzgtypen durch Typgruppen auf Bögen als Mengenfunktion.

Prob. (Umlaufplanungsproblem, VSP)

Geg.: Menge T von Fzgtypen (endlich)

Kapaz. $u^d \in \mathbb{N}_0 \quad \forall d \in T$

Menge V von Fahrtenknoten mit Start- und Endz. $s_f, t_f \in \mathbb{R}$, $s_f < t_f \; \forall f$, darunter zwei „künstliche" Knoten $s \neq t$ mit $t_s < \min_{f \neq s} s_f$, $\max_{f \neq t} t_f < s_t$.

Menge $A \subseteq V^2$ von Fahrtenverknüpfungen mit $t_u \le s_v \quad \forall uv \in A$

Typgruppe $T_a \subseteq T \quad \forall a \in A$.

Kosten $c_a^d \in \mathbb{N}_0 \quad \forall a \in A, d \in T_a$

Def.: Planungsgraph $D = (V, A)$

Typgraph $D^d = (V, A^d)$, $A^d = \{a \in A : d \in T_a\}$

Umlauf p vom Typ d :⇔ st-Pfad in D^d

Menge P^d aller Umläufe vom Typ d

Menge $P = \dot\bigcup P^d$ aller Umläufe (Beachte: zu einem Umlauf gehört ein Typ!)

Umlaufkosten $c_p = \sum_{a \in p \in P^d} c_a^d$

Umlaufplan $U \subseteq P$: p, q knotendisj. bzgl. $V \setminus \{s,t\}$, überdeckt $V \setminus \{s,t\}$, $|U \cap P^d| \le u^d \; \forall d \in T$

Kosten $c_U = \sum_{p \in U} c_p$

Ges.: Umlaufplan mit min. Kosten.

Bem.: a) D ist azyklisch.

b) Durch geeignete Def. von Verknüpfungen und Kosten kann man vielfältige Anforderungen abdecken:

i) Min. # Fzge: $c_{uv} = \begin{cases} 1 & u = s,\ v \neq t \\ 0 & \text{sonst} \end{cases}$ (sei dieses $c := c^{\#}$)

ii) Min. # Fzge, dann Km-Leistung: $c_{uv} = \begin{cases} M + l_{uv} & , uv = s\cdot \\ l_{uv} & , \text{sonst} \end{cases}$

iii) Min # Linienwechsel: $c_{uv} \mathrel{+}= w_{uv}$, falls u,v zu versch. Linien geh.

iv) Min. kurze Wende: $c_{uv} \mathrel{+}= w_{uv}$, falls uv kurze Wende

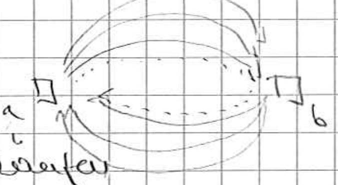
v) Mindestwendezeit T: Entferne uv mit $s_v - t_u < T$

vi) Fzgtypen T_f auf Teilfahrten f: $T_{fg} \leftarrow T_{fg} \cap T_f \quad \forall fg \in A$

Kosten c_f auf " f: $c_{fg} \leftarrow c_{fg} + c_f \quad \forall fg \in A$

c) Einige Anforderungen, die man nicht „einfach" abdecken kann:

i) Balancierter Depotausgleich:

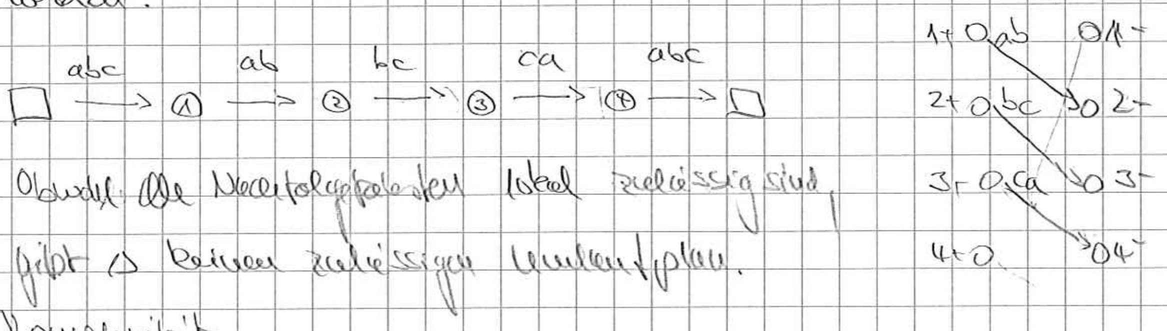


ii) Längenbeschr. auf einzelnen Umläufen

iii) " auf Mengen von Umläufen

iv) Lokale Typschranken

d) Außer den Kapazitäten kann das VSP nicht als Zuordnungsproblem gelöst werden:



Obwohl alle Nachfolgerpaare lokal zulässig sind, gibt es keinen zulässigen Umlaufplan.

4. Komplexität

Satz. Das VSP ist NP-hart.

Bew.: Betrachtung des zugeh. Entscheidungsproblems (VSP') (gibt es einen Umlaufplan mit Kosten ≤ k?) für den Fall $\mathcal{U} \equiv \mathbb{1}$ und $c = c^{\#}$. (gibt es eine Knotenüberdeckung mit k Knoten?)

Reduktion des ~~Vertex Cover~~ Dominating-Set-Entscheidungsproblems (DS') auf (VSP').

Sei $G = (N, E)$, $k \in \mathbb{N}$ eine Instanz von (DS')

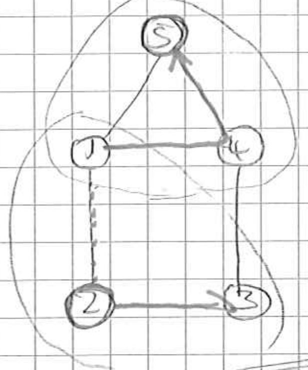
$N(1) = \{1,2,4,5\}$

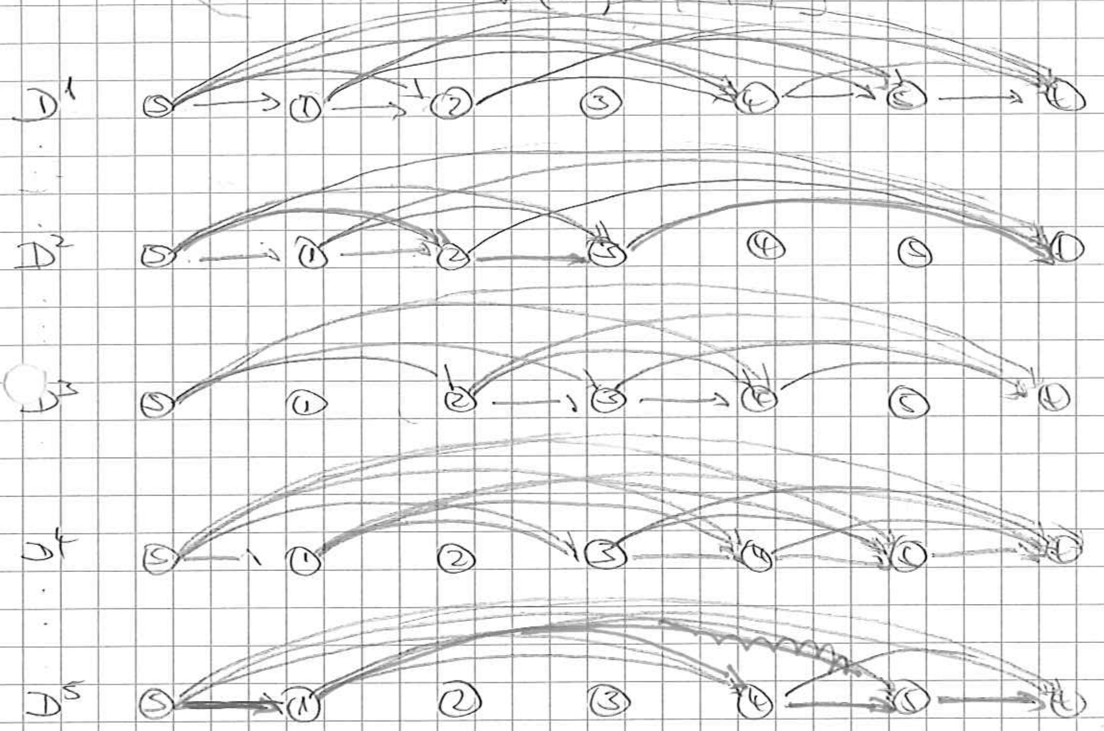
$N(2) = \{1,2,3\}$

$N(3) = \{2,3,4\}$

$N(4) = \{1,3,4,5\}$

$N(5) = \{1,4,5\}$





O.B.d.A. sei $N = \{1,\dots,n\}$. Setze

$T := \{1,\dots,n\} = N$

$u^d = 1 \quad \forall\, d \in T$

$V := \{0,\dots,n+1\}$ mit $s := 0$, $t := n+1$, $s_i := i$, $t_i := i+1$, $\forall i$

$N(i) := \{i\} \cup \mathcal{N}(i)$, $i \in N$

$A^i := \{ij \in (\{0\} \cup N(i) \cup \{n+1\})^2 : i < j\}$ (wohldef.!)

$A := \bigcup_{i \in N} A^i$

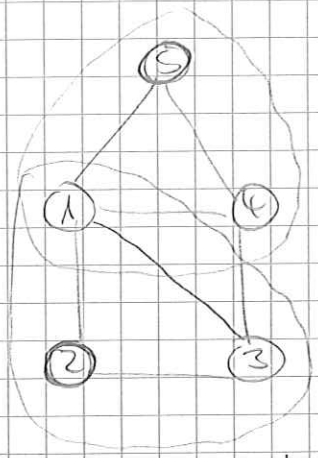
$T_a := \{i : a \in A^i\}$

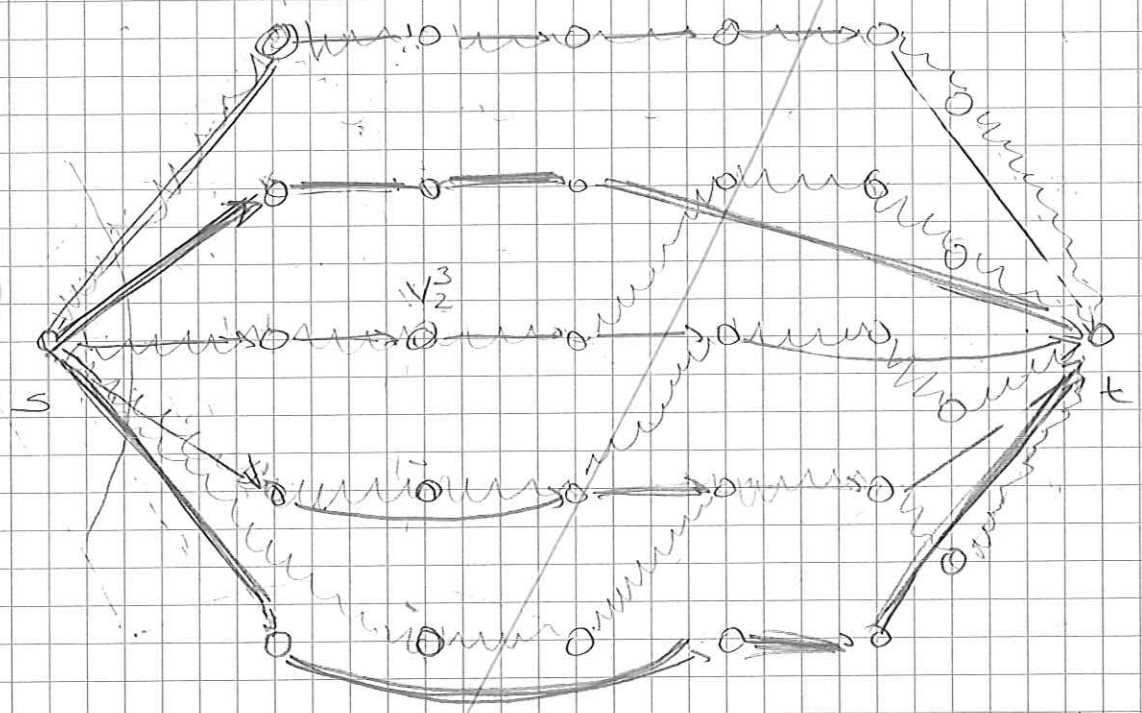
$c_{ij} := \begin{cases} 1, & i = s \\ 0, & \text{sonst} \end{cases}$

$D := (V, A)$

Diese Konstruktion hat folgende Eigenschaften:

i) Die Codierungslänge von K, T, V, A, u, c ist $O(|V|^3 + \log K)$, d.h. die Konstruktion ist polynomial in der Codierungslänge $O(|V|^2 + \log K)$ von V, K, E.





$V_0 \quad V_1 \quad V_2 \quad V_3 \quad V_4 \quad V_5 \quad V_6 \quad V_7$

O.E.d.A. sei $N = \{1, \dots, n\}$, $N(i) = \{j \mid j(i) = \{j_0 < \dots < j_{d(i)}\}$

$V_0 := \{s\}$.

$V_i := \{ v_i^j \mid j \in N \}$, $i \in N$

$V_{n+1} := \{ v_{n+1}^1, \dots, v_{n+1}^{n-1} \}$

$V_{n+2} := \{t\}$

$A_i := \{ s v_i^{j_0}, v_i^{j_0} v_i^{j_1}, \dots, v_i^{j_{d(i)-1}} v_i^{j_{d(i)}}, v_i^{j_{d(i)}} t \}$, $i \in N$

$A_i := \bigcup_{0 \le j < k \le n+2} V_j V_k$, $i = n+1, \dots, n+(n-1)$

$T := \{1, \dots, n, n+1, \dots, n+(n-1)\}$.

Diese Konstruktion hat folgende Eigenschaften:

a) Sie ist polynomial in G.

b) Der Digraph ist gekennzeichnet mit Knotenmengen $V_0, \dots, V_{n+2}$, sodass $V_1, \dots, V_n$ hat n Knoten, V_{n+1} hat $n-1$ Knoten, V_0, V_{n+2} hat 1 Knoten.

c) Für Arctyp $i \in N$ gibt es genau einen Knoten, und zwar A_i.

ii) Für jedes $i \in \mathbb{N}$ ist die Abb.

$$\pi_i : P^{N(i)} \to 2^{N(i)}, \quad p \mapsto p \cap N(i)$$

bijektiv, d.h. es besteht eine 1-1-Beziehung zwischen Unterräumen vom Typ i und Teilmengen von $N(i)$. Die Umkehrabb. ist

$$\pi_i^{-1} : 2^{N(i)} \to P^{N(i)}, \quad W(i) \supseteq \{j_1, \dots, j_k : j_1 < \dots < j_k\} \mapsto \{j_1, \dots, j_k\}^{\perp}.$$

iii) Wenn U ein Unterraumplan ist, dann ist

$$T(U) := \{ i \in T : P^i \cap U \neq \emptyset \}$$

eine Trägerüberdeckung von G mit $|T(U)| = |U|$. $(U \equiv \mathbb{1})$.

iv) Wenn $J = \{j_1 < \dots < j_k\}$ eine Trägerüberdeckung von G ist, dann ist mit der Rekursion

$$p_i := \pi_i\Big(N(j_i) \setminus \bigcup_{e<i} N_{j_e}\Big), \quad i = 1, \dots, k$$

$U(J) := \{p_1, \dots, p_k\}$ ein Unterraumplan mit $|J|$ ($\leq k$) Unterräumen

v) (i)(iv) $\Rightarrow$ Antwort VS' $= JA \iff$ Antwort VSP' $= JA$. □

Folg.: a) Da in VS' und VSP' keine Zahlen (außer k) auftreten, folgt, daß VSP sogar NP-vollständig im starken Sinn ist.

b) Da die Transformation Zielfunktionswerte erhält und VC nicht approximierbar ist, folgt, daß VSP ebenfalls nicht approximierbar ist. f. VC ist 2-approx

d) Umläufe der $n-1 = |V|-1$ Typen $n, \dots, n+(n-1)$ laufen beliebig von links nach rechts durch die Knotenschichten $V_0, \dots, V_{n+2}$, wobei in jeder Schicht höchstens 1 Knoten besucht wird.

e) Die $n-1$ Knoten der Schicht V_{n+1} können nur von Umläufen vom Typ $n+1, \dots, n+(n-1)$ besucht werden. Es muß folglich $n-1$ solche Umläufe geben, die in den Schichten $V_1, \dots, V_n$ jeweils mindestens einen Knoten übrig lassen, d.h. die Schichten $V_1, \dots, V_n$ werden nicht voll überdeckt.

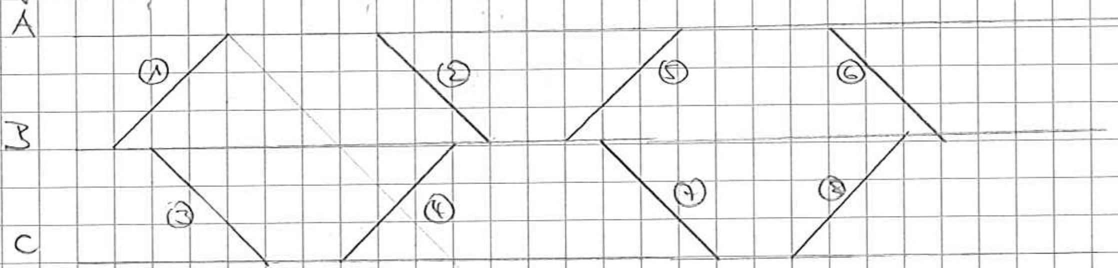
f) Jeder übriggelassene Knoten wird von einem Umlauf vom Typ $i \in N$ überdeckt, da damit die Schicht überdeckt.

g) Die Überdeckung der Schichten durch Umläufe vom Typ $i \in N$ entspricht einer Überdeckung von V durch Knoten $i \in N$. □

Bem.: Der Beweis zeigt auch, daß das VSP stark NP-hart ist und, da VC nicht approximiert werden kann, auch APX-hart ist.

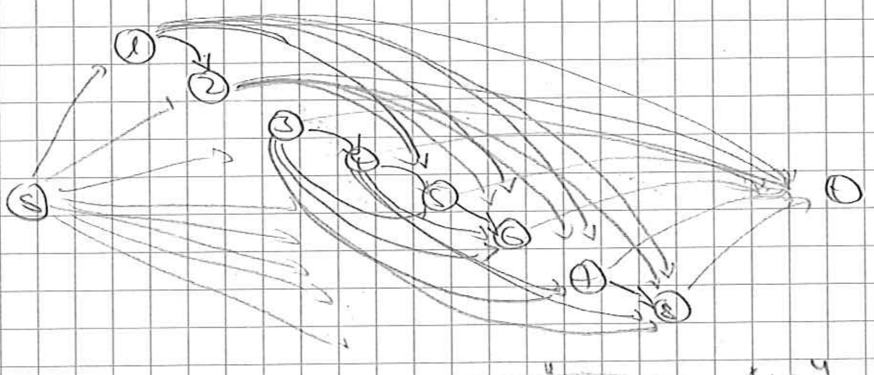
5. Planungsgraphen

Bsp.:



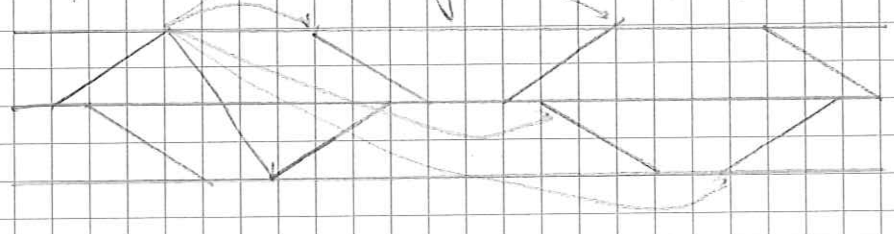
8 Touren

a) Standard-Planungsgraph "Leg-on-Node"

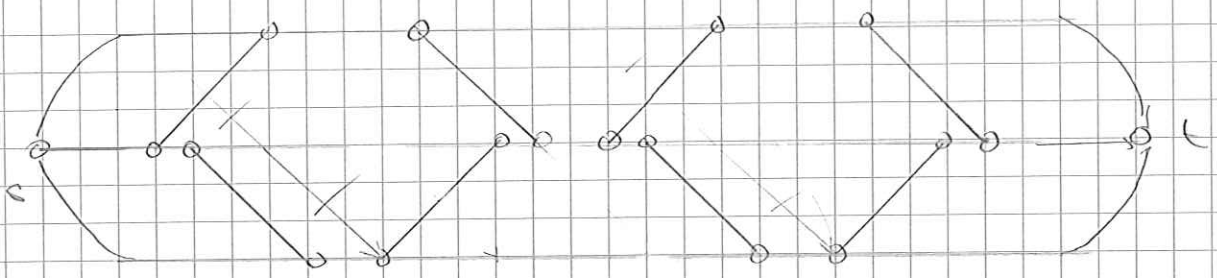


"Leg-on-Arc" etc.

b)



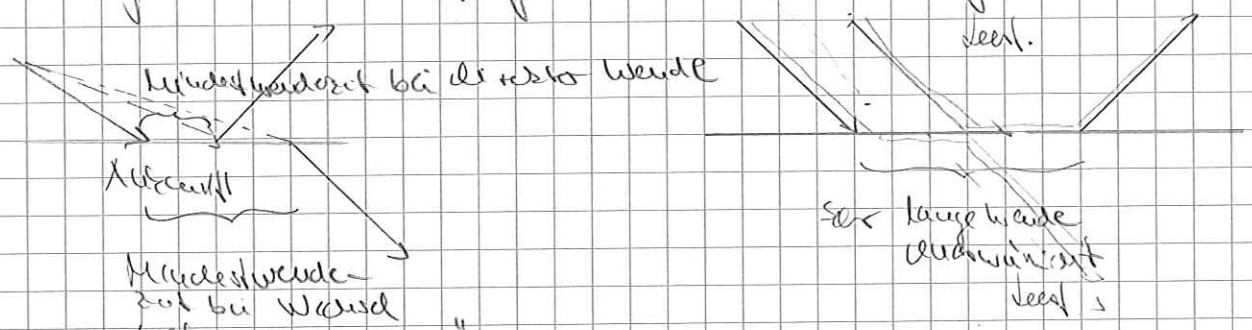
b) Zeitlinien - Prozessgraph



Bem.: a) Die Größe des Zeitlinienprozessgraphen ist der Komplexität Darstellung:

	$\lvert V \rvert$	$\lvert A \rvert$
Standard	$O(\lvert F \rvert)$	$O(\lvert F \rvert^2)$
Zeitlinien	$O(\lvert F \rvert)$	$O(\lvert F \rvert \lvert L \rvert)$

b) Voraussetzung für den Zeitlinienprozessgraph ist, dass: Dauer und Reihenfolge von Verrichtungen nur von Orten abhängen.



Man kann diese Voraussetzung so replizieren, indem man einen kürzeren Übergang zwischen den Prozessgraphen definiert.

6 IP Modelle

Def. (IP-Formulierung des VSP).

a) Zugformulierung

Mit Variablen $x_a^d \in \{0,1\}$ $\forall a \in A, d \in T$ kann das VSP wie folgt formuliert werden:

(VSP_1) min $\sum_{d \in T} \sum_{a \in A^d} c_a^d x_a^d$

(i)	$x^d(\delta^-(v)) - x^d(\delta^+(v)) = 0$	$\forall d \in T,\ s \neq v \neq t$	„Flusserhaltungsbed."
(ii)	$x(\delta^-(v)) = 1$	$\forall s \neq v \neq t$	„Flussbedingung" (Graph)
(iv)	$x^d(\delta^+(s)) \leq u^d$	$\forall d \in T$	„Typbeperikaten"
(vi)	$x_a^d \in \{0,1\}$	$\forall a \in A, d \in T$	„Ganzzahligkeit"

b) Pfadformulierung

Mit Variablen $y_p \in \{0,1\}$, $p \in P = \dot{\bigcup}_{d \in T} P^d$ erhält man die äquivalente Formulierung

$$(VSP_2) \quad \min \sum_{d \in T} \sum_{p \in P^d} \overbrace{\sum_{a \in p} c_a^d}^{= c_p} y_p$$

(i) $y(\delta^-(v)) = 1 \quad \forall\, s \neq v \neq t$ „Flussbedingung"

(ii) $y(P^d) \leq u^d \quad \forall\, d \in T$ „Depotkapazitäten"

(iii) $y_p \in \{0,1\} \quad \forall\, p \in P$ „Ganzzahligkeit"

Bem: Die Äquivalenz von (VSP_1) und (VSP_2) bzgl. Zielfunktionswerten und Lösungen wurde in der VL über Multi-Commodity-Flow und column-generation gezeigt. Dort wurde erg., daß man Probleme mit vielen Spalten am besten per CG löst. Bei wenigen Spalten lohnt sich dies weniger. In der Umlaufoptimierung versucht man meist, die Flussform zu lösen und durch die Pfadformulierung auszuwerten.

7. Zwischenspiel: Lagrange-Relaxierung

Def. (Lagrange-Relaxierung) Betrachte ein opt. Problem

(P) $\min c^T x$, $Dx = d$, $x \in X$,

mit $D \in \mathbb{R}^{m \times n}$, $X \subseteq \mathbb{R}^n$ abgeschlossen. Dann heißt

(L) $\max_\lambda f(\lambda) := \max_\lambda \min (c^T - \lambda^T D) x + \lambda^T d$, $x \in X$

Lagrange-Relaxierung von (P) bzgl. $Dx = d$,

$f : \mathbb{R}^m \to \mathbb{R}$, $\lambda \mapsto f(\lambda)$

Lagrangefunktion von (P) bzgl. $Dx = d$,

$f(\lambda) = \min (c^T - \lambda^T D) x + \lambda^T d$, $\quad \in \mathbb{R}^m$

Subproblem von (L).

Satz: Sei $v(P) = \min c^T x$, $Dx = d$, $x \in X \in \mathbb{R}_\infty$.

a) $\max f(\lambda) \leq v(P)$.

b) Sei $X = \{Ax = b, x \geq 0\}$ und $\{Dx = d, Ax = b, x \geq 0\} \neq \emptyset$. Dann ist $\max f(\lambda) = v(P)$.

Insbesondere ist der Wert der Lagrange-Relaxierung eines LPs gleich

dann wird der LP-Relaxierung (falls diese zulässig ist).

c) Sei X {Polytop / endliche} und $X \cap \{Dx = d\} \neq \emptyset$. Dann ist f

i) konkav

ii) stückweise affin.

iii) nach oben beschränkt.

Bew.:

a) $f(\lambda) = \min_{x \in X} (\bar{c}^T - \lambda^T D)x + \lambda^T d = \min_{x \in X} \bar{c}^T x - \lambda^T (Dx - d) \leq \min_{\substack{x \in X \\ Dx = d}} \bar{c}^T x - \lambda^T (Dx - d) = \nu(P)$

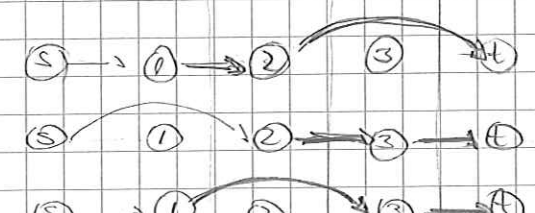
b) $\min \{\bar{c}^T x : Ax = b,\ Dx = d,\ x \geq 0\} = \max \{\mu^T b + \lambda^T d : \mu^T A + \lambda^T D \leq \bar{c}^T\} = \max_{\lambda} \lambda^T d + \max_{\mu} \{\mu^T b : \mu^T A \leq \bar{c}^T - \lambda^T D\}$

$\max_{\lambda} f(\lambda) = \max_{\lambda} \min_{\substack{Ax = b \\ x \geq 0}} (\bar{c}^T - \lambda^T D)x + \lambda^T d = \max_{\lambda} \lambda^T d + \min_{\substack{Ax = b \\ x \geq 0}} (\bar{c}^T - \lambda^T D)x$

c) Sei $X = \{x_1, \dots, x_k\}$, also $X = \operatorname{conv}\{x_1, \dots, x_k\}$:

$f(\lambda) = \min_{x \in X} (\bar{c}^T - \lambda^T D)x + \lambda^T d = \min_{i=1,\dots,k} \underbrace{\bar{c}^T x_i - \lambda^T (Dx_i - d)}_{\text{affine Funktion}} \leq \nu(P) < \infty$, $x_i \in \{x : Dx = d\}$

Bsp.:



$\min x^1_{s1} + x^2_{s2} + x^3_{s1}$

$Dx = d$: $x^1_{12} + x^1_{2t} + x^2_{23} + x^2_{3t} + x^3_{13} + x^3_{3t} = 3$

$x \in X \Leftrightarrow \begin{cases} Ax = b \\ x \geq 0 \end{cases} \begin{cases} x^1_{s1} = x^1_{12} = x^1_{2t} \\ x^2_{s2} = x^2_{23} = x^2_{3t} \\ x^3_{s1} = x^3_{13} = x^3_{3t} \\ 0 \leq x \leq 1 \end{cases}$

opt. Lsg $x^* = 1/2$.

$\nu(P) = 3/2$

(L) $\max_{\lambda} \min x^1_{s1} + x^2_{s2} + x^3_{s1} - \lambda(x^1_{12} + x^1_{2t} + x^2_{23} + x^2_{3t} + x^3_{13} + x^3_{3t} - 3)$

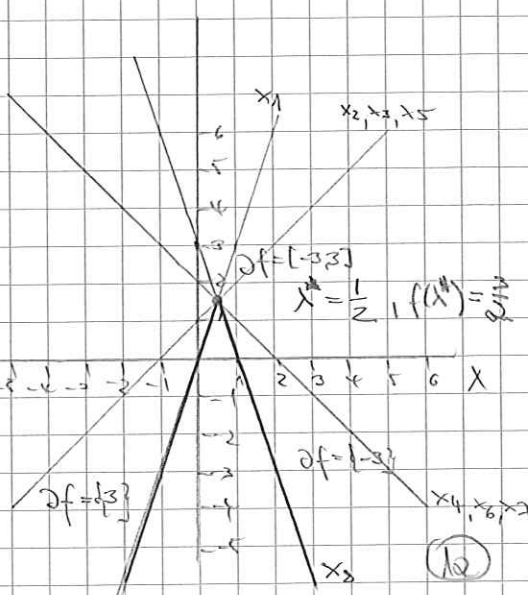
$x^1_{s1} = x^1_{12} = x^1_{2t}$

$x^2_{s2} = x^2_{23} = x^2_{3t}$

$x^3_{s1} = x^3_{13} = x^3_{3t}$

$0 \leq x \leq 1$

	p_1	p_2	p_3	$\bar{c}^T x$	$- \lambda (Dx - d)$	$f(\lambda)$
x_1	0	0	0	0	$- \lambda(-3)$	3λ ✓
x_2	0	0	1	1	$- \lambda(-1)$	$\lambda + 1$ ✓
x_3	0	1	0	1	$- \lambda(-1)$	$\lambda + 1$ ✓
x_4	0	1	1	2	$- \lambda(+1)$	$-\lambda + 2$ ✓
x_5	1	0	0	1	$- \lambda(-1)$	$\lambda + 1$ ✓
x_6	1	0	1	2	$- \lambda(+1)$	$-\lambda + 2$ ✓
x_7	1	1	0	2	$- \lambda(+1)$	$-\lambda + 2$ ✓
x_8	1	1	1	3	$- \lambda(+3)$	$-3\lambda + 3$



Satz: $\partial f(x_0) = \{f'(x_0)\}$

Bew.: $f(x_0+\lambda d) \geq f(x_0) + u^T(x_0+\lambda d - x_0) = f(x_0) + \lambda u^T d$

$\Rightarrow \frac{f(x_0+\lambda d) - f(x_0)}{\lambda} \geq u^T d$

$\Rightarrow f'_d(x_0) \geq u^T d$

$\Rightarrow f'_{-d}(x_0) \leq -u^T d \Rightarrow f'_d(x_0) = -f'_{-d}(x_0) \geq u^T d$

$\Rightarrow f'_d(x_0) = u^T d$

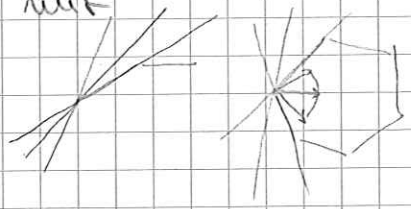
$\Rightarrow f'(x_0) = u.$

Bemerke: Für $f(\lambda)$ sind alle bis hierhin Schritte optimal! Löst man $f(\lambda)$ mit Verfahren, erhält man immer eine gemeinsame Lösung, die für (P) nur optimal ist, d.h. eine Lagrangerelaxierung liefert i.a. nur einen Zielfunktionswert, aber keine Lösung.

Def a) Sei $f: \mathbb{R}^m \to \mathbb{R}$ konvex. Dann heißt $u \in \mathbb{R}^m$ mit

$$f(\lambda) \geq f(\lambda_0) + u^T(\lambda - \lambda_0) \quad \forall \lambda$$

Subgradient von f in λ_0



b) $\partial f(\lambda_0) = \{u \in \mathbb{R}^m : u \text{ ist Subgradient von } f \text{ in } \lambda_0\}$

Subdifferential von f in λ_0

Prop.: Sei $f: \mathbb{R}^m \to \mathbb{R}$ konvex und diffbar in $\lambda_0 \in \mathbb{R}^m$. Dann ist

$$\partial f(\lambda_0) = \{f'(\lambda_0)\}.$$

Bew.: Übung

Satz: Sei $f: \mathbb{R}^m \to \mathbb{R}$ konvex. Dann gilt:
$f(\lambda^*) = \min f(\lambda) \Leftrightarrow 0 \in \partial f(\lambda^*)$.

Bew.: "$\Rightarrow$": $f(\lambda) \geq f(\lambda^*) = f(\lambda^*) + 0^T(\lambda - \lambda^*)$
$\Leftarrow$ $f(\lambda) \geq f(\lambda^*) + 0^T(\lambda - \lambda^*) = f(\lambda^*)$ □

Prop.: Sei $X = \left\{ \begin{matrix} \text{conv}\{x_1, \dots, x_k\} \\ \{x_1, \dots, x_k\} \end{matrix} \right\} \neq \emptyset$. Dann ist

$$\partial f(\lambda_0) = -\text{conv}\{Dx_i - d : x_i \in \operatorname{argmin} f(\lambda_0)\}.$$

Bew.: Sei $X(\lambda) = \{x_i : f(\lambda) = c^T x_i - \lambda_0^T(Dx_i - d)\} \quad \forall \lambda \in \mathbb{R}^m$, $u_i := -(Dx_i - d)$

"$\supseteq$": $\forall x_j \in X(\lambda_0)$ gilt:

$$f(\lambda_0) + u_j^T(\lambda - \lambda_0) = \underbrace{c^T x_j - \lambda_0^T(Dx_j - d)}_{f(\lambda_0)} - \underbrace{(Dx_j - d)^T(\lambda - \lambda_0)}_{u_j^T(\lambda - \lambda_0)}$$

$$= c^T x_j - \lambda^T(Dx_j - d)$$

$$\geq \min_{i=1} c^T x_i - \lambda^T(Dx_i - d)$$

$$= f(\lambda).$$

$\Rightarrow \text{conv } X(\lambda_0) \subseteq \partial f(\lambda_0)$.

"$\subseteq$" Wähle $\varepsilon := \left\{ \min_{x_i \notin X(\lambda_0)} c^T x_i - \lambda_0^T(Dx_i - d) - f(\lambda_0), 1 \right\} > 0$,

ist $X(\lambda) \subseteq X(\lambda_0) \quad \forall \lambda \in U(\lambda_0)$ für eine gewisse Umgebung von λ_0.

Sei $u \notin -\text{conv}\{Dx_i - d : x_i \in X(\lambda_0)\}$. Dann ex.

$$u^T\pi < u_i^T\pi \quad \forall x_i \in X(\lambda_0). \qquad \text{(Trennsatz)}$$

$$f(\lambda_0 + \varepsilon\pi) = \min_{x_i \in X(\lambda_0)} c^T x_i - (\lambda_0 + \varepsilon\pi)^T(Dx_i - d)$$

$$= \min_{x_i \in X(\lambda_0)} c^T x_i - \lambda_0^T(Dx_i - d) + \varepsilon\pi^T u_i$$

$$= f(\lambda_0) + \min_{x_i \in X(\lambda_0)} \varepsilon\pi^T u_i > f(\lambda_0) + \varepsilon\pi^T u = f(\lambda_0) + u^T(\lambda_0 + \varepsilon\pi - \lambda_0)$$

↯

 □

Alg.: Subgradientenverfahren

Input: $f: \mathbb{R}^m \to \mathbb{R}$ konkav

$\lambda_0 \in \mathbb{R}^m$ Startpunkt

$(\alpha_k)_{k=1}^{\infty}$, $\alpha_k > 0$ Schrittweiten

Output: $(\lambda_k)_{k=1}^{b}$

1. $k \leftarrow 0$, $u_0 \leftarrow u \in \partial f(\lambda_0)$
2. $\lambda_{k+1} \leftarrow \lambda_k + \alpha_k u_k$

   $u_{k+1} \leftarrow u \in \partial f(\lambda_{k+1})$
3. goto 2.

Satz: Sei $f: \mathbb{R}^m \to \mathbb{R}$ konkav und es gelte

i) $f^* = \max f \in \mathbb{R}$ (endliches Max.)

ii) $\|u\|_2 \leq L \quad \forall u \in \partial f$ (Lipschitzbedingung)

iii) $\sum \alpha_k \to +\infty$

iv) $\sum \alpha_k^2 < \infty$.

Dann gilt $\lim_{k\to\infty} \max_{j=0}^{k} f(\lambda_j) = f^*$.

Beh.: Sei $\lambda^* \in \operatorname{argmax} f$.

$$\|\lambda_{k+1} - \lambda^*\|_2^2 = \|\lambda_k + \alpha_k u_k - \lambda^*\|_2^2$$

$$= \|\lambda_k - \lambda^*\|_2^2 + 2\alpha_k u_k (\lambda_k - \lambda^*) + \alpha_k^2 \|u_k\|_2^2$$

$f(\lambda^*) \leq f(\lambda_k) + u_k(\lambda^* - \lambda_k)$

$\Rightarrow f(\lambda^*) - f(\lambda^k) \leq u_k(\lambda^k - \lambda^k)$

$\Rightarrow f^* - f(\lambda_k) \leq -u_k(\lambda^k - \lambda^*)$

$$\leq \|\lambda_k - \lambda^*\|_2^2 - 2\alpha_k \left[f^* - f(\lambda_k)\right] + \alpha_k^2 \|u_k\|_2^2$$

$$\leq \|\lambda_0 - \lambda^*\|_2^2 - 2\sum_{j=0}^{k} \alpha_j \left[f^* - f(\lambda_k)\right] + \sum_{j=0}^{k} \alpha_k^2 \|u_k\|_2^2$$

$$\Rightarrow \underbrace{\|\lambda_{k+1} - \lambda^*\|_2^2}_{\geq 0} + 2\sum_{j=0}^{k} \alpha_j \left[f^* - f(\lambda_k)\right] \leq \|\lambda_0 - \lambda^*\|_2^2 + \sum_{j=0}^{k} \alpha_k^2 \|u_k\|_2^2$$

$$\Rightarrow f^* - \max_{j=0}^{k} f(\lambda_j) \leq \frac{\|\lambda_0 - \lambda^*\|_2^2 + \sum_{j=0}^{k} \alpha_k^2 L^2}{2\sum_{j=0}^{k} \alpha_j} \xrightarrow{k\to\infty} 0. \quad \square$$

Bsp.: Umlaufplanung

$$(VSP_1) \quad \min \sum_d \sum_{a \in A^d} c_a^d x_a^d$$

(i) $x^d(\delta^-(v)) - x^d(\delta^+(v)) = 0 \quad \forall d \in T, \ s \neq v \neq t$

[(ii) $x(\delta^-(v)) - x(\delta^+(v)) = 0 \quad \forall s \neq v \neq t$ „ggf. Flugbahnkopplung"]

(iii) $x(\delta^-(v)) = 1 \quad \forall s \neq v \neq t$

(iv) $x^d(\delta^+(s)) + s_d \leq u^d \quad \forall d \in T$

[(v) $x(\delta^+(s)) \leq \sum_d u^d$ „ggf. Depotkap."]

(vi) $0 \leq x_a^d \leq 1 \quad \forall d \in T, a \in A^d$

(vii) $s_d \geq 0$

a) (VSP_1) ohne (ii) und (v). Lagrange-Relaxierung der Flugbed. (iii):

$$\max_{\lambda} \min \sum_d \sum_{a \in A^d} c_a^d x_a^d - \sum_{s \neq v \neq t} \lambda_v \left[x(\delta^-(v)) - 1 \right]$$

(i) $x^d(\delta^-(v)) - x^d(\delta^+(v)) = 0 \quad \forall d \in T, \ s \neq v \neq t$

(iv) $x^d(\delta^+(s)) \leq u^d \quad \forall d \in T$

(vi) $0 \leq x_a^d \leq 1 \quad \forall d \in T, a \in A^d$

Das Subproblem dekomponiert in unabhängige Min-Cost-Flow-Probleme für die einzelnen Depots bei stets ganzzahligen Lösungen, wenn Kanten sind ohne Kapazität.

b) (NSP_1) mit (ii) und (v). Lagrange-Relaxierung von (i) und (iv):

$$(LVSP_1^{i,iv}) \quad \max_{\mu, \pi} \min \sum_d \sum_{a \in A^d} c_a^d x_a^d - \sum_{\substack{d \in T \\ s \neq v \neq t}} \mu_d^v \left[x^d(\delta^-(v)) - x^d(\delta^+(v)) \right] - \sum_d \pi^d \left[x^d(\delta^+(s)) - u^d \right]$$

(ii) $x(\delta^-(v)) - x(\delta^+(v)) = 0 \quad \forall s \neq v \neq t$

(iii) $x(\delta^-(v)) = 1 \quad \forall s \neq v \neq t$

(v) $x(\delta^+(s)) \leq \sum_d u^d$

(vi) $0 \leq x \leq 1$

(vii) $s \geq 0$

Das Subproblem ist ein einziges Min-Cost-Flow-Problem, in dem Flugtypen ignoriert werden (→ Variablen zusammenfassen) d.h. es reduziert Depotkapazität.

c) Beide Lagrange-Relaxierungen haben denselben Wert wie die LP-Relaxierung von (NSP_1).

3. Heuristiken

Idee: a) 2 Basisoperationen

i) Schedule : Touren → Cluster } Zuordnungen
ii) Cluster : Touren → Depots }

b) Annahmen

i) $\forall v \in V : \exists d \in T : d_{0v}, d_{v0} \in A$ } Zulässigkeit
ii) "weitere Depotkapazitäten" u^d }

Alg: Cluster first – Schedule second (CFSS)

Input: VSP durch T, V, A, u, c

Output: Δ-Touren $p_1 \dots p_\ell$: $v \in p_i$ $\forall v \in V \setminus \{s,t\}$, $|\{p_i \in P^d\}| \leq u^d$ $\forall d \in T$

0. forall $d \in T$: $V^d \leftarrow \emptyset$

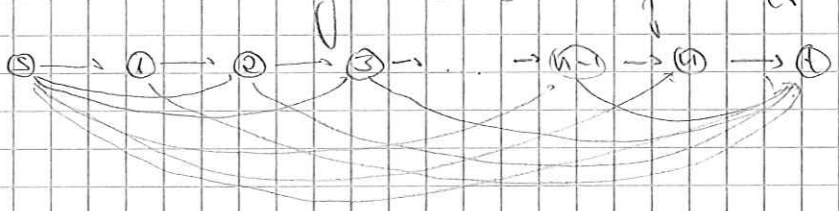
1. forall $v \in V \setminus \{s,t\}$: // „cluster"

$d \leftarrow \underset{d \in T}{\operatorname{argmin}} \{ c^d_{dv} + c^d_{vd} \}$; // „nearest depot"

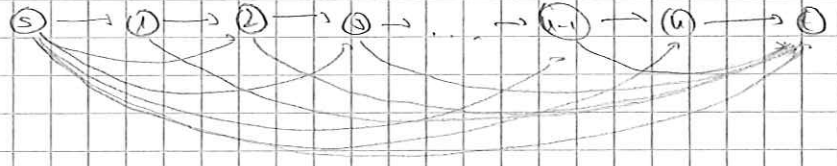
$V^d \leftarrow V^d \cup \{v\}$;

2. $p_1 \dots p_\ell \leftarrow \bigcup_{d \in T} \operatorname{argmin} \text{SDVSP}(V^d, A^d, c^d, u^d)$ // „schedule"

Bsp.: CFSS kann beliebig schlechte Lösungen liefern:

D_1 

$c^1_{s_i} := c^1_{it} := \begin{cases} 0, & i \text{ ungerade} \\ 1, & i \text{ gerade} \end{cases}$

D_0 

$c^2_{s_i} := c^2_{it} := \begin{cases} 1, & i \text{ gerade} \\ 0, & i \text{ ungerade} \end{cases}$

CFSS : $V^1 = \{1, 3, 5, \dots\}$, $V^2 = \{2, 4, 6, \dots\}$

$P_{CFSS} = \{p_1, \dots p_n : p_i = s\, i\, t \in P^{i \bmod 2}\}$

$P_{opt} = \{s\, 1\, 2\, 3 \dots n-1\, n\, t\}$

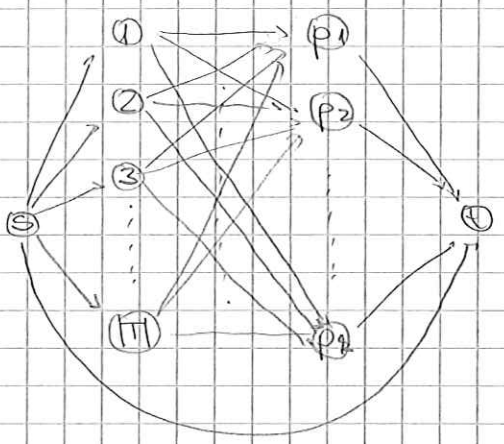
$|P_{CFSS}| / |P_{opt}| = n/1 = n \xrightarrow{n \to \infty} \infty$. □

Alg. (Löbel [1997]): Schedule first – clusters second – reschedule

Input: VSP durch $T, V, A, c, u, \varepsilon > 0, M > 0$

Output: St-Pfade $p_1, \dots, p_k$ mit $v \in p_i$ $\forall v \in V \setminus \{s,t\}$, $\left| \{ p_i \in p^d \} \right| \leq u^d \ \forall d \in T$

1. $p_1, \dots, p_k \leftarrow$ argmin $(LVSP_1^{i,iv})$ „Schedule"
2. forall $d \in T$: $\hat{A}^d \leftarrow \{ a \in A^d : \bar{c}_a^d < \varepsilon \}$
3. forall $p_1, \dots, p_k$, $d \in T$: $c_i^d \leftarrow \left| p_i - p_i \cap \hat{A}^d \right| \cdot M + \sum_{a \in p_i \cap \hat{A}^d} c_a^d$



$b_s = k$, $b_t = -k$, $b_v = 0 \ \forall v \neq s,t$

$$c_{uv} = \begin{cases} c_i^d, & uv = dp_i \\ MM, & uv = st \\ 0, & \text{sonst} \end{cases}$$

$$u_{uv} = \begin{cases} u^d, & uv = sd \\ \sum u^d, & uv = st \\ \Lambda, & \text{sonst} \end{cases}$$

4. $MCF \leftarrow$
5. $x \leftarrow$ argmin MCF „capacity estimate"
6. for $i = 1, \dots, k$: „best depot"

$$d_i \leftarrow \begin{cases} d, & \text{falls } x_{dp_i} = 1 \\ \underset{d \in T}{\operatorname{argmax}} \left| p_i \cap \hat{A}^d \right|, & \text{falls } x_{dp_i} = 0 \ \forall d \in T \end{cases}$$

7. forall $d \in T$: $V_d \leftarrow \emptyset$
8. for $i = 1, \dots, k$: $V_{d_i} \leftarrow p_i \cap V^{d_i}$ „clusters"
9. forall $v \notin \bigcup_{d \in T} V^d$: $V^d \leftarrow V^d \cup \{v\}$, $d \leftarrow \underset{d \in T}{\operatorname{argmin}} \{ c_{dv}^d + c_{vd}^d \}$
10. $p_1, \dots, p_k \leftarrow$ argmin $\sum_{d \in T} VSP(V^d, A^d, c^d, u^d)$ „reschedule"

Bem.: Die „cluster"-Schritte 7–9 sind hier in einer vereinfachten Form dargestellt. Löbel [1997] analysiert die Umläufe genauer, um Depots, die nicht bindende Depotkapazität verwenden, zu bestimmen und in einer übergeordneten Tabu-Search-Heuristik (2nd order) zu eliminieren.

Sensitivitätsanalyse

Idee: Senke den Frg.bedarf durch Verschieben von Fahrten (in der Spitze).
Die Fahrtenverschiebungen werden als „Waren" auf den Fahrzeiten aufgefaßt,
daher das neue „Sensitivitätsanalyse" für diesen Ansatz.

Def.: Umlaufplanungsproblem mit Fahrtenverschiebungen

ggg.: T „Depots"

V $s,t \in V$ „Fahrten, s,t"

$V^d \subseteq V$ $\forall d \in T$, $s,t \in V^d$

$A^d \subseteq V^d \setminus \{t\} \times V^d \setminus \{s\}$

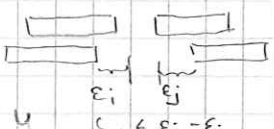
$D^d = (V^d, A^d)$ „Raumgraph für Depot d"

$\delta^d_{uv} \in \mathbb{Z}$ $\forall uv \in A^d$, $u \neq s$, $v \neq t$ „Fahrtverschiebungen"

$c^d_{uv} \in \mathbb{Z}$ $\forall uv \in A^d$ „Zeiten"

$c^d_v \in \mathbb{Z}$ $\forall v \in V \setminus \{s,t\}$, $d \in T$

$\Delta \in \mathbb{N}_0$ „max. Verschiebung"



Def.: $P^d := \{(p,\varepsilon) : p = s, v_1, \dots, v_k, t \subset P^d$ st-Pfad, $-\Delta \leq \varepsilon_1, \dots, \varepsilon_k \leq \Delta$:
$s^d_{v_i v_{i+1}} \leq \varepsilon_{i+1} - \varepsilon_i$, $i = 1, \dots, k-1\}$, $d \in T$

$P := \bigcup_{d \in T} P^d$

$c_p := \sum_{uv \in A^p} c^d_{uv} + \sum_{s,t \neq v \in p} \varepsilon_v c^d_v$

$U \subseteq P$: $\bigcup_{p \in U} p \setminus \{s,t\} = V$ „Umlaufplan"

$c_U := \sum_{p \in U} c_p$

ggg.: Umlaufpläne mit minimalen Kosten.

Def. (Bökler [2006]): IP-Formulierung für das Umlaufplanungsproblem mit Fahrtenverschiebung

$$(TSVSP_1) \quad \min \sum_{d \in T} \sum_{a \in A^d} c^d_a x^d_a + \sum_{v \in V \setminus \{s,t\}} c^d_v \varepsilon^d_v$$

$$(i) \quad x^d(\delta^-(v)) - x^d(\delta^+(v)) = 0 \quad \forall d \in T,\ s \neq v \neq t$$

$$(ii) \quad x(\delta^+(v)) = 1 \quad \forall\ s \neq v \neq t$$

$$(iii) \quad x^d_{uv}(\delta^d_{uv} + 2\Delta) - 2\Delta \leq \varepsilon^d_v - \varepsilon^d_u \quad \forall d \in T,\ u \neq s,\ v \neq t$$

$$(iv) \quad x^d(\delta^+(s)) \leq u^d \quad \forall d \in T$$

(v) $-\Delta \leq \varepsilon_v^d \leq \Delta \quad \forall d \in T,\ s+v \neq t$

(vi) $x_{uv}^d \in \{0,1\} \quad \forall uv \in A^d$

(vii) $\varepsilon_v^d \in \mathbb{R} \quad \forall d \in T,\ s+v \neq t$

Satz (Bökler [2006]): Diskretisierung

(x,ε) Lsg von (TSVSP) $\Rightarrow (x, \lceil \varepsilon \rceil)$ Lsg von (TSVSP).

Insbesondere gibt es eine Lösung mit ganz. Verbindungen, falls es eine Lsg gibt.

Bew.: Einsetzen. □

Satz (Bökler [2006]) (TSVSP,1) ist NP-hart, sogar für $|T| = 1$.

Bew.: $|T| > 1$ Reduktion auf VSP durch geeignete Def. von s und Δ.

$|T| = 1$ " auf PARTITION, siehe Bökler [2006]

Bem.: Man kann noch andere Modelle betrachten, in denen Fahrten jeweils Verbindung kapazitiert werden. Diese Modelle führen zu besseren unteren Schranken.

Ü: $\dot\cup P^M = \cup P^K$. VSP → DST

Ü: ∅ des min. Dienstmix ... $\sum x_k \le 4k$ MRCSP CP

top Sort – azykl.

Ü: Pause als Lin. NB

9. Dienstplanung

Def. (Dienstplanungsproblem (DSP)):

geg.: $D = (V, A)$ Dienstplanungsgraph (azyklisch)

$D_k = (V_k, A_k)$ für Dienstarten

$V = T \dot\cup S \dot\cup \{s,t\}$

T Menge von Dienstelementen („Tasks")

S Menge von Ergänzungselementen („Supplementary Tasks")

$s = \text{top min } V$ künstlicher Knoten („Dienstbeginn") (eindeutig)

$t = \text{top max } V$ künstlicher Knoten („Dienstende") (eindeutig)

A Verknüpfungen („links")

R Menge von Resourcen für einzelne Dienste

K Menge von Dienstarten (→ Gemeinsamkeiten, ähnlich bei VSP!)

M Menge von Resourcen für den Dienstmix

$c \in \mathbb{Q}^{A \times K}$ Kosten

$w \in \mathbb{Q}^{A \times R}$ Resourcenverbrauch für Dienste

$w \in \mathbb{Q}^{A \times M}$ Resourcenverbrauch für Dienstarten ?

$u \in \mathbb{Q}^{R \times M}$ Resourcenlimits

Pausenregeln

Def.: p st-Pfad in D

P_{st} Menge der st-Pfade in D

$w_p^r = \sum_{a \in p} w_a^r$ Verbrauch von Resource r durch p, $r \in R$

$P^k := \{ p \in P_{st} : w_p^r \le u^r \ \forall r \in R \}$ Menge der zulässigen Dienste der Dienstart k, $k \in K$

$P^K := \dot\cup P^k$ Menge aller zulässigen Dienste, o.B.d.A. $\dot\cup P^k = \cup P^k$, d.h. jeder st-Pfad ist für max. eine Dienstart gültig.

$c_p = \sum_{a \in p} c_a^k$, $p \in P^k$ Kosten eines Dienstes der Dienstart k, $k \in K$

$c_P = \sum_{p \in P} c_p$, $P \in P^K$

$w_P^m = \sum_{p \in P} w_p^m$, $P \in P^K$

$P \in P^K$: (zulässiger Dienstplan)

i) $\exists!\, p \ni v \in T$

ii) $w_P^m \le u^m \ \forall m \in M$

$\mathcal{P} = \{ P \text{ Dienstplan} \}$

ges.: $\min_{P \in \mathcal{P}} c_P$ Kostenminimaler Dienstplan

Item.: i) Typische Dienstarten sind Früh-, Mittel-, Spät, optische Dienste

ii) Typische Dienstmerkmale sind Dienstzeit, Arbeitszeit, Dienstart.

iii) Typische Dienstmixbed. sind Schranken auf die absolute / relative Anzahl der gleichen Spät, usw. Dienste und Schranken an die durchschnittliche Arbeits- und Dienstzeit.

Item.: i) Nebenbedingungstypologie:

Dienstzeit — max. Teil eines Dienstes auf einem Tag

Dienstteil — max Teil " " " Zwischen zwei Dienstenden/Beginnen bei gl. Dienst

Beob. i) Das DSP ist ein Pfadüberdeckungsproblem.

ii) VSP lässt sich auf DSP zurückführen, indem für jeden Typ. eine Ressource r mit $u^r = 0$ und

$$w^r_{uv} = \begin{cases} 0, & uv \in A^r \\ 1, & \text{sonst} \end{cases}$$

eingeführt wird; analog werden die Depotkapazitäten durch Dienstmix-Bedingungen abgebildet. Damit folgt:

iii) DSP $\subseteq$ NPH.

Bem ii) (Airline-Terminologie): In anderen Industrien werden teilweise andere Bezeichnungen verwendet, z.B. im Verkehrsbetrieb:

ÖV	Verkehrsbetrieb
task	leg
Dienst	pairing
Dienstteil	flight segment (Teil eines Pairings, an einem Tag)
Dienstart	Pairing type
Mix-Bedingung	Max (überflüssig (da Bed. häufig Personalverfügbarkeit an Flughäfen beschreiben)

Def (IP-Formulierung des DSP): Mit Variablen $x_p \in \{0,1\}$ $\forall p \in P^K$ kann das DSP wie folgt formuliert werden:

(DSP) $$\min \sum_{p \in P^K} c_p x_p$$

(i) $$\sum_{p \in V} x_p = 1 \quad \forall v \in T$$ „Tourenüberdeckungsbed."

(ii) $$\sum_{p \in P^K} w^u_p x_p \le u^u \quad \forall u \in M$$ „Dienstmixbedingungen"

(iii) $$x_p \in \{0,1\} \quad \forall p \in P^K$$ „Ganzzahligkeit"

mit

$$a_{vp} = \begin{cases} 1, & v \in p \\ 0, & v \notin p \end{cases}, \quad v \in T, p \in P^K.$$

$A := (a_{vp})_{v \in T, p \in P^K}$ „Task-Dienst-Inzidenzmatrix"

$W^M := (w^u_p)$, $u^M := (u^u)$

$c := (c_p)$

Ergibt sich die Behauptung

(DSP) $\min c^T x \;,\; Ax = \mathbb{1} \;,\; W^M x \le u^M \;,\; x \in \{0,1\}^{P^K}$.

"Set-Partitioning-Problems mit Resource Constraints".

Def. (Spaltenerzeugung): Sei $P^{k'} \subseteq P^k$, $c' = (c_{pk'})$, usw.

(MLP) $\min c^T x \;,\; Ax = \mathbb{1} \;,\; W^M x \le u^M \;,\; 0 \le x \le \mathbb{1}$ "Master LP"

(RMLP) $\min c'^T x'$ "Restricted Master LP"

(π) $A' x' = \mathbb{1}$

(μ) $W^{M'} x' \le u^M$

$0' \le x' \le \mathbb{1}'$ Welche $p \in S$?

$\bar{c}_p := c_p - \pi^T A_{\cdot p} - \mu^T W^M_p$ "reduzierte Kosten von p bzgl. π und μ"

(PRICE) $\min_{p \in P^K} \bar{c}_p = \min_{k \in K} \min_{p \in P^k} \bar{c}_p$ "Pricingproblem"

Bem.: Das Pricingproblem ist ein ressourcenbeschränktes Kürzeste-Wege-Problem mit zusätzlichen linearen Nebenbedingungen.

Bew.: $\bar{c}_p = c_p - \pi^T A_{\cdot p} - \mu^T W^M_p = \sum_{a \in p} c^k_a - \sum_{v \in T \cap p} \pi_v - \sum_{m \in M} \mu^m w^m_p$

$= \sum_{a \in p} c^k_a - \sum_{v \in T \cap p} \pi_v - \sum_{m \in M} \mu^m \sum_{a \in p} w^m_a \;,\; p \in P^k$

Sei $\tilde{c}_{w'} := c^k_a - \mu^m w^m_a - \begin{cases} \pi_v, & v \in T \\ 0, & v \notin T \end{cases}$ $\forall w \in A$, dann gilt:

$\min_{p \in P^k} \bar{c}_p = \min \sum \tilde{c}_a x_a$

$x(\delta^+(v)) = \delta_{vs} - \delta_{vt} \quad \forall v \in V$

$w^r(x) \le u^r \quad \forall r \in R$

$x \in \{0,1\}^A$. □

05.02.07
12

10. Dienstplanungsprobleme

Bsp.: Airline-Crew-Scheduling

$V := \{ FRA, TXL, MUC \}$ "Flughäfen"

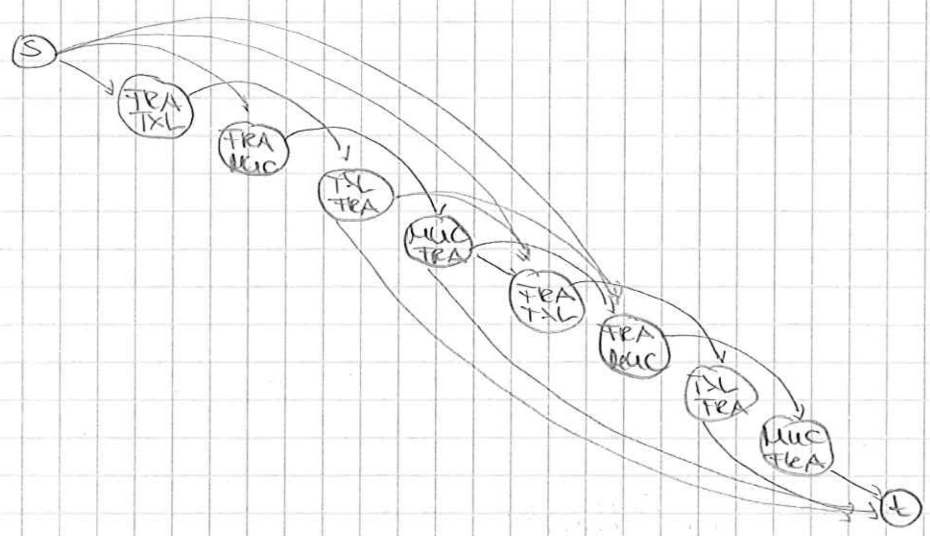
$T := \{ FRA \}$ "Heimatbasis"

$\mathcal{A} := \{ 1, \dots, 8 \}$ "Flüge"

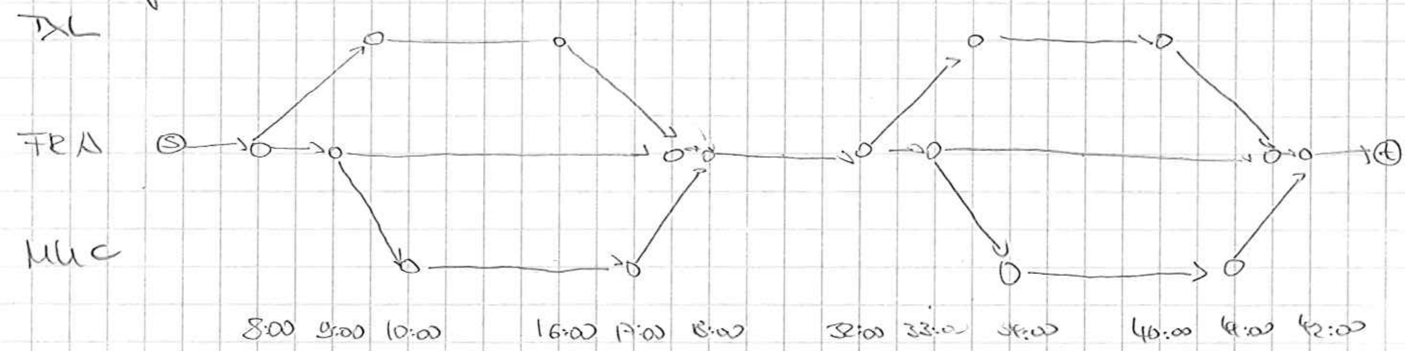
	o	d	s	t
1	FRA	TXL	8:00	9:30
2	FRA	MUC	9:00	10:00
3	TXL	FRA	16:00	17:30
4	MUC	FRA	17:00	18:00
5	FRA	TXL	32:00	33:30
6	FRA	MUC	33:00	34:00
7	TXL	FRA	40:00	41:30
8	MUC	FRA	41:00	42:00

$\mathcal{R}$ = { Pairing Start- u. Ende in $\mathcal{B}$, keine Übernachtungen in $\mathcal{A} \setminus \mathcal{B}$ } „Regeln"

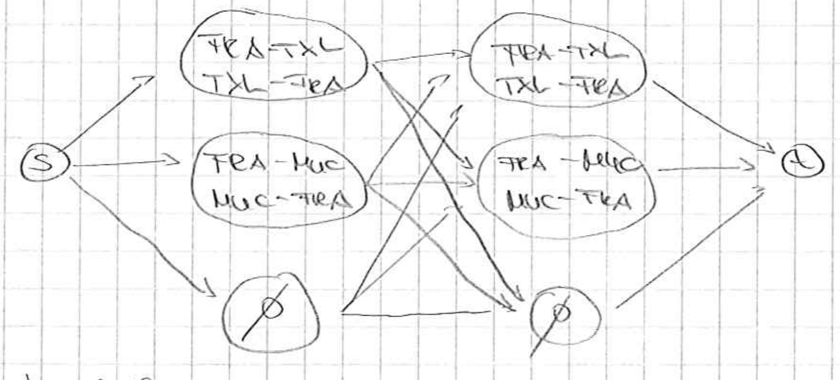
i) Leg-on-Node-Netzwerk



ii) Leg-on-Arc-Netzwerk



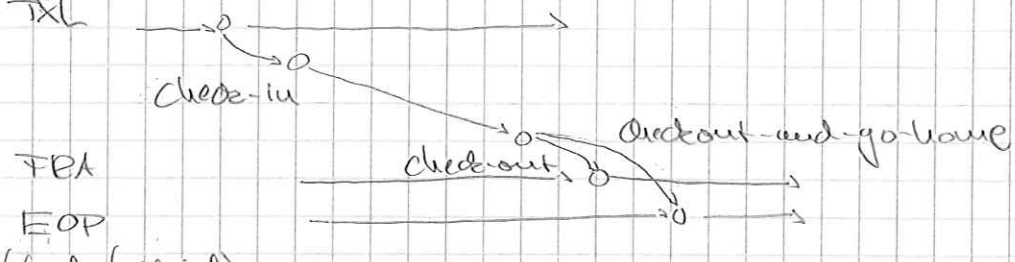
iii) Duty-Period-Netzwerk



iv) Vergleich

Typ	Leg-on-Node	Leg-on-Arc	Duty-Period
$\lvert V \rvert$	$O(\lvert\mathcal{F}\rvert)$	$O(\lvert\mathcal{F}\rvert)$	$O(\lvert\mathcal{F}\rvert^{L})$ (L = max # legs / dp)
$\lvert A \rvert$	$O(\lvert\mathcal{F}\rvert^{2})$	$O(\lvert\mathcal{F}\rvert)$	$O(\lvert\mathcal{F}\rvert^{2L})$
Vorteil	genaue Kontrolle über Folgeflüge	kompakt	Duty-Period-Regeln erledigt, bzw. häufig leichter
Nachteil	sehr viele Verb.! (insb. bei Einbez. von Transporten)	weniger Kontrolle über Folgeflüge	groß

v) Ergänzungselemente



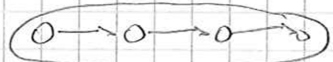
vi) Flug und Bodentransport mit eigenen und fremden Flügen, Übernachtungen etc. werden ähnlich modelliert.

vii) Large-Scale-Techniken:

Next-availables (Ryan):

Betrachte nur 2-3 „beste" nach jedes Flugs

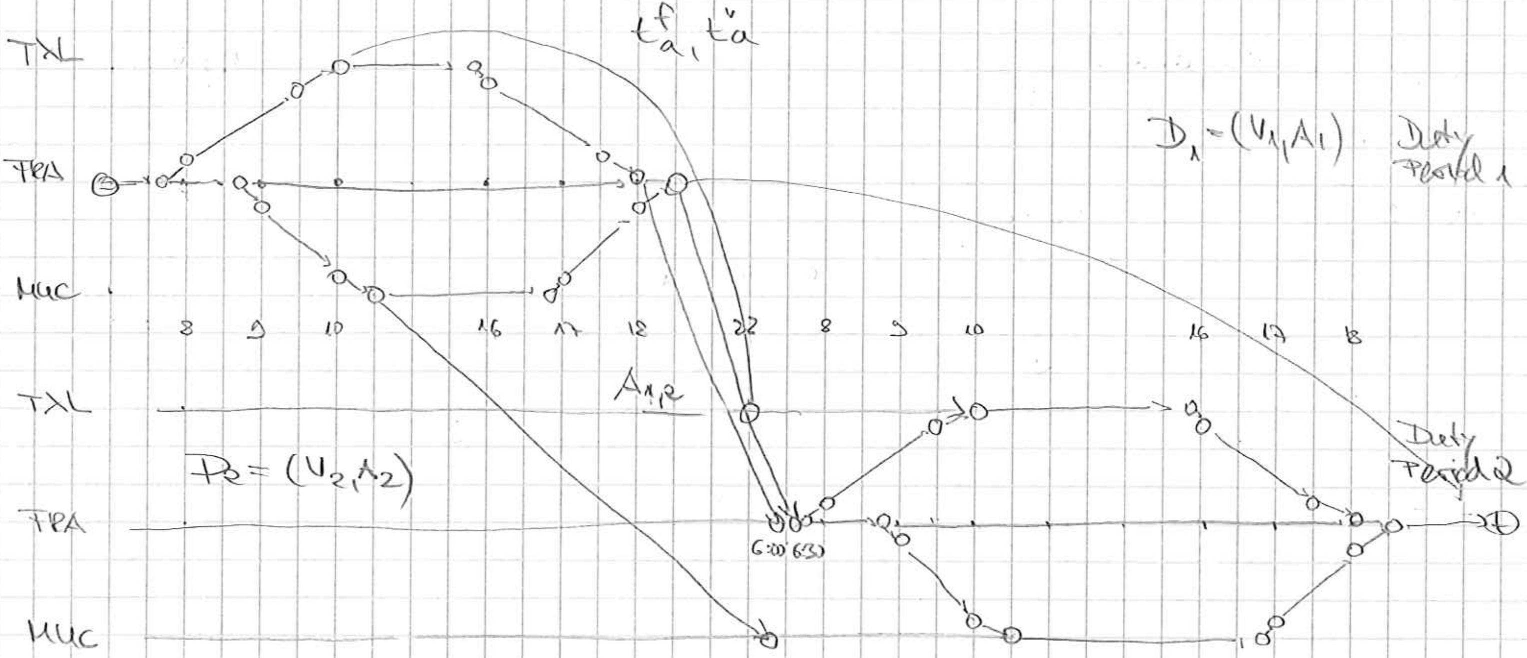
Aggregation (Desrosiers et al.)



temporäres Fixieren von Verbindungen

Bsp. 2. DV nach BO

Größe	Zeichen	Regel
Blockzeit	t^b	
Check-in	t^{cin}	$\geq 0{:}30$
Check-out	t^{cout}	$\geq 0{:}15$
Transportzeit	t^{dh}	$= \begin{cases} t/2 & 0 \leq t \leq 4{:}00 \\ 2+t-4 & t > 4{:}00 \\ t & \text{Selbstfahrer} \\ 0 & \text{Schulungen} \\ 0 & \text{vor/nach Ruhezeit} \end{cases}$
Flugdienstzeit	t^f	$= t^{cin} + t^b + t^{cout} + t^{dh} \leq 10{:}00 + t^s$
Verlängerung	t^s	$\leq 4 \begin{cases} -1 & 2{:}00 \leq t^n \leq 4{:}00 \\ -2 & 4{:}00 < t^n \end{cases} \begin{cases} -1 & 4 \leq n^e \leq 5 \\ -2 & 6 \leq n^e \end{cases}$ $\sum_{7\,\text{Tage}} t_s \leq 8$
Nachtflugzeit	t^n	$= t^f \cap [01{:}00, 07{:}00]_{\text{loc. start airport}}$
Landungen	n^e	
Ruhezeit	t^r	$\geq 10{:}00$



i) Pricing-Problem ohne Verlängerung (Kürzeste Wege-Problem mit lin. Nebenbed.)

min $\sum \tilde{c}_a x_a$

$$x(\delta^+(s)) = x(\delta^-(t)) = 1$$

$$x(\delta^+(v)) = x(\delta^-(v)) \quad \forall\, v \neq s,t$$

$$\sum_{a \in A^i} t^f_a x_a \le 10{:}00 \quad \forall \text{ Duty periods } i$$

$$-\sum_{a \in A_{i,i+1}} t^v_a x_a \le -10{:}00 \quad \forall \text{ '' } i$$

$$x_a \in \{0,1\} \quad \forall\, a$$

$$c_a = \begin{cases} 1, & a = sv \\ 0, & \text{sonst} \end{cases}$$ min # Pairings ($c = c^{\#}$)

$c_a = t^f_a$ min Flugdienstzeit

$$c_a = \begin{cases} 1, & a = sv,\ a \in A_{i,i+1} \\ 0 \end{cases}$$ min # Duty periods (Produktionstage)

ii) Pricing-Problem mit Verlängerung (Kürzeste Wege-Problem mit komplexen Nebenbed.)

min $\sum \tilde{c}_a x_a$

$$x(\delta^+(s)) = x(\delta^-(t)) = 1$$

$$x(\delta^+(v)) = x(\delta^-(v)) \quad \forall\, v \neq s,t$$

$$-\sum_{a \in A_{i,i+1}} t^v_a x_a \le -10{:}00 \quad \forall\, i$$

$$\sum_{a \in A_i} t^f_a x_a \le 10{:}00 + t^s_i \le 14{:}00 \quad \forall\, i$$

$$\sum_{i=j}^{j+6} t^s_i \le 8{:}00 \quad \forall\, j$$

$$t_i^s \le 4{:}00 - x_i^n \cdot 1{:}00 - y_i^n \cdot 1{:}00 - y_i^u \, 1{:}00 \quad \forall i$$

$$\sum_{a \in A_i} t_a^n x_a - 2{:}00 \le 2{:}00 \, x_i^n \quad \forall i$$

$$\sum_{a \in A_i} n_a^e x_a - 4 \le y_i^n + M y_i^m \quad \forall i$$

$$y_i^n \ge y_i^m$$

$$x_a \in \{0,1\} \quad \forall a$$

$$t_i^s \ge 0 \quad \forall i$$

$$x_i^y, y_i^n, y_i^u \in \{0,1\} \quad \forall i$$

M. Das Azyklische Ressourcenbeschränkte Kürzeste-Wege-Problem

Def. (Azyklisches Ressourcenbeschr. Kürzeste-Wege-Problem (ARCSP)):

V: $D = (V, A)$ Digraph (azyklisch)

$s, t \in V$ Quelle, Senke

R Menge von Ressourcen

$w_a^r \in \mathbb{R}$ Verbrauch von Ressource r auf Bogen a

$u^r \in \mathbb{R}$ Limit für den Verbrauch von Ressource $r \in R$

$c_a \in \mathbb{R}$ Kosten

Def.: p st-Pfad in D

$w_p^r = \sum_{a \in p} w_a^r$ Verbrauch von Ressource r auf p

$w_p^r \le u^r \quad \forall r$ p zulässig

O.: min c_p, $w_p^r \le u^r \; \forall r \in R$

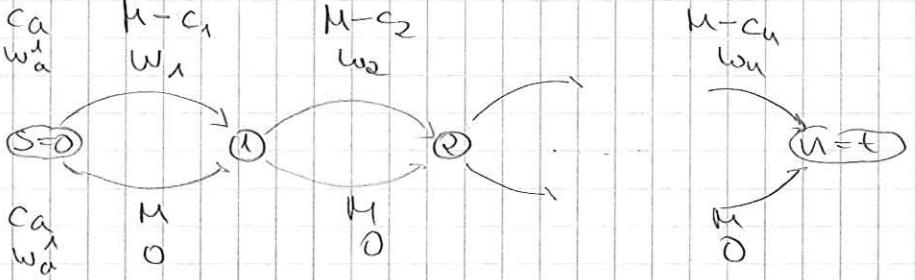
Prop.: ARCSP $\in$ NPH.

Bew.: Transformation des Rucksackproblems (KP) auf (ARCSP).

(KP) max $\sum a_i x_i$

$\sum w_i x_i \le u$

$x_i \in \{0,1\} \quad i = 1, \dots, n$



$\Rightarrow c_p = n \cdot M - \sum_{v \in \cdot} c_v$

□

Def. (IP-Formulierung des ARCSP): Mit $x_a \in \{0,1\}\ \forall a \in A$ ergibt sich folgende IP-Formulierung des (ARCSP):

(ARCSP) min $\sum_{a \in A} c_a x_a$

$$x(\delta^+(v)) - x(\delta^-(v)) = s_{sv} - s_{tv} \quad \forall v \in V$$

$$w^v(x) \leq u^v \quad \forall v \in Z$$

$$x_a \in \{0,1\} \quad \forall a \in A.$$

Bemerke: In nicht azyklischen Fall müssen zu obige Formulierung noch Zusammenhangsbedingungen wie beim TSP hinzugefügt werden!

Bsp.: Lösung eines ARCSPs mit Lagrange-Relaxierung

S. altes Skript S. 17 – 19.

min $c^T x$

$$x(\delta^+(v)) - x(\delta^-(v)) = s_{sv} - s_{tv}$$

$$w^v(x) + s^v = u^v$$

$$x_a \in \{0,1\} \quad \forall a$$

$$s^v \geq 0 \quad \forall v$$

$= \max_{\lambda} \min$ $c^T x - \sum \lambda_v \left(w^v(x) + s^v - u^v \right)$ $\quad \lambda > 0 \Rightarrow \min = -\infty$, $\lambda < 0 \Rightarrow s_v = 0$

$$x(\delta^+(v)) - x(\delta^-(v)) = s_{sv} - s_{tv}$$

$$x_a \in \{0,1\} \quad \forall a$$

$$s^v \geq 0 \quad \forall v$$

$= \max_{\lambda \geq 0} \min$ $c^T x + \sum \lambda_v \left(w^v(x) - u^v \right)$

$$x(\delta^+(v)) - x(\delta^-(v)) = s_{sv} - s_{tv}$$

$$x_a \in \{0,1\} \quad \forall a$$

$$s_i \geq 0 \quad \forall v$$

Algorithmus: Dyn. Prog. für das 1-RCSP mit nichtneg. Gewichten und Kosten

Voraussetzung: 1-RCSP, d.h. $k=1$, $w := w^1 \in \mathbb{N}_0^A$, $d = d^1$, $c \in \mathbb{N}_0^A$

Def.: Sei $D=(V,A)$ ein Digraph. Eine Folge von Kanten $(j_1 j_2, j_2 j_3, \dots, j_{k-1} j_k)$ aus A mit nicht notwendigerweise verschiedenen Knoten j_i heißt eine $j_1 j_k$-Kette.

Bem.: Eine Kette ist ein Pfad genau dann wenn $j_i \neq j_k$ $\forall i,k$ gilt.

Def.: Eine ij-Kette K mit $w(K) \le r$ heißt r-Kette.

(*) [illegible] Lemma:

Alg. von Joksch (modifiziert) → ($w \in \mathbb{N}^A$)

Inp.: 1-RCSP mit $w, c \in \mathbb{N}_0^A$, $d \in \mathbb{N}_0$, Knotenmenge T

Dat.: Tabelle $c_j(r)$, $j \in V$, $r = 0, \dots, d$

Out.: Länge $c_t(d)$ einer kürzesten d-Kette von s nach t

for $r = 0, \dots, d$ $\{ c_s(r) \leftarrow 0 \}$ (1)

forall $j \in V \setminus \{s\}$ $\{ c_j(0) \leftarrow \infty \}$ (2)

for $r = 1, \dots, d$ $\{$

forall $j \in V \setminus \{s\}$ $\{ c_j(r) \leftarrow \min\{ c_j(r-1), c_i(r - w_{ij}) + c_{ij}, ij \in A, 1 \le w_{ij} \le r \} \}$ (3)

for $(H \leftarrow V;\ i \leftarrow \operatorname{argmin}_{j \in H} c_j(r);\ H \leftarrow H \setminus T)$ $\{$ (4)

$T \leftarrow \{ j \in V : \exists\, 0\text{-Pfad von } i \text{ nach } j \}$ (5)

forall $j \in T$ $\{ c_j(r) \leftarrow c_i(r) + \min_{P_{ij} \in T} c(P_{ij}) \}$ (6)

$\}$

gib $c_t(d)$ aus

$r=d$	0					
	0		$c_j(r)$			
	0					
	0					
$r=0$	0	∞	∞	∞	∞	∞
	$v=s$					$v=t$

Satz: Der mod. Alg. von Joksch hat eine Laufzeit von $O(dm + dn \log n)$ und einen Speicherplatzbedarf von $O(nd)$

Bew.: (1)-(3): $O(d) + O(n) + O(dm)$

(4): $O(dn \log n)$ } auf einem Level (H_r wird nur einmal sortiert)

(5): $O(dm)$ } auf einem Level (H_r wird jeweils bfs durchlaufen)

(6): $O(dn)$

Folg.: Der Alg. von Joksch berechnet auch die Länge eines kürzesten d-Pfades von s nach t.

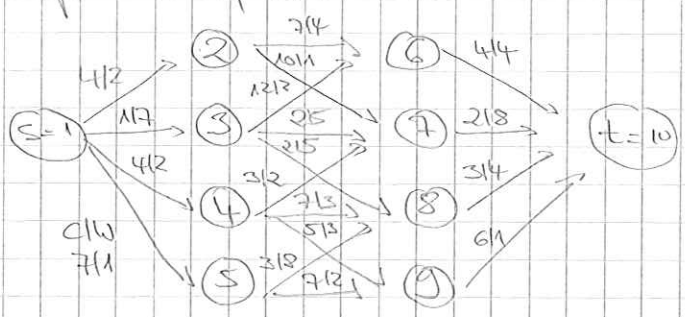
Bew.: Eliminiere ggf. Kreise mit Gewicht 0.

(*) Lemma: Sei $c_j(v)$ die Länge einer kürzesten v-Kante von s nach j. Dann gilt die Rekursion

$$c_s(v) = 0, \quad v = 0, \dots, d$$
$$c_j(0) = \infty, \quad j \neq s$$
$$c_j(v) = \min \{ c_j(v-1), c_i(v-w_{ij}) + c_{ij}, ij \in A, w_{ij} \leq v \}, v = 1, \dots, d, j \neq s.$$

Idee: Bellmansches Optimalitätsprinzip. □

Bsp.: $k=1$, $\omega^T x \leq 10$



a) Alg. von Jobson (Tuchi-Strategie)

$r=$	10	0	↑	↑	↑	↑	↑	↑	↑	↑	
	9	0	↑	↑	↑	↑	↑	10	↑	~~15~~ 17	
	8	0	↑	↑	↑	↑	↑	↑	↑	↑	
	7	0	↑	1	↑	↑	↑	↑	↑	↑	
	6	0	9	-	↑	↑	↑	↑	↑	15	
	5	0	↑	-	↑	↑	14	↑	(11)	9	↑
	4	0	↑	-	↑	↑	-	7	-	↑	20
	3	0	↑	-	↑	↑	-	14	-	14	-
	2	0	4	-	(4)	↑	-	-	-	-	-
	1	0	-	-	-	7	-	-	-	-	-
$v=0$	0	(0)	-	-	-	-	-	-	-	-	$=\infty$
		1	2	3	4	5	6	7	8	9	10
		‖									
		v									

b) Alg von Hassin

$c=$	20	0									
	19	0									
	18	0									
	17	0									
	16	0									
	15	0								6	
	14	0					3		3	(9)	
	13	0	↑	↑	↑	↑	96	↑	↑	↑	-
	12	0	↑	↑	↑	↑	↑	↑	↑	↑	-
	11	0	↑	↑	↑	↑	6	↑	(5)	↑	-
	10	0	↑	↑	↑	↑	-	↑	9	↑	-
	9	0	↑	↑	↑	↑	-	↑	-	5	-
	8	0	↑	↑	↑	↑	-	↑	-	-	-
	7	0	↑	↑	↑	1	-	4	-	-	-
	6	0	↑	↑	↑	-	-	-	-	-	-
	5	0	↑	↑	↑	-	-	-	-	-	-
	4	0	2	↑	(2)	-	-	-	-	-	-
	3	0	-	↑	-	-	-	-	-	-	-
	2	0	-	↑	-	-	-	-	-	-	-
	1	0	-	7	-	-	-	-	-	-	-
$c=$	0	(0)	-	-	-	-	-	-	-	-	-
		1	2	3	4	5	6	7	8	9	10
		‖									
		v									

Bem.: Das Problem, in einem Digraph mit nicht neg. Kanten einen kürzesten Weg mit höchstens k Kanten zu finden, ist polynomial lösbar, genauer in $O(km) \leq O(nm)$

Beob.: $W = \mathbb{N}$, $d = k$.

Theorem: Das Problem, in einem Digraph mit nicht neg. Kanten einen kürzesten Weg mit genau k Kanten zu finden, ist nicht polynomial lösbar.

Folg.: $k = n$ liefert das ATSP.

Alg.: Dyn. Prog. für das 1-RCSP mit nicht negativen und positiven Kanten

Voraus.: 1-RCSP, d.h. $k = 1$, $w := w' \in \mathbb{N}_0^A$, $d \in \mathbb{N}_0$, $c \in \mathbb{N}^A$.

Lemma: Sei OPT die (zumindest unterschiedlichen) Kosten eines kürzesten d-Wegs von s nach t.

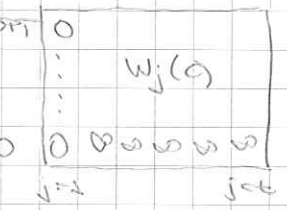
Sei $w_j(c)$ die min. Kosten eines sj-Wegs mit Kosten $\leq c$, $c = 0, \dots, OPT$, $j \in V$.

Dann gilt die Rekursion

$$w_s(c) := 0, \quad c = 0, \dots, OPT$$

$$w_j(0) := \infty, \quad j \in V \setminus \{s\}$$

$$w_j(c) := \min\{ w_j(c-1), w_i(c-c_{ij}) + w_{ij} \mid ij \in A, c_{ij} \leq c \}, \quad c = 1, \dots, OPT, \ j \in V \setminus \{s\}$$



Alg. von Hassin [1992]

Inp.: 1-RCSP mit $D = (V, A)$, $w \in \mathbb{N}_0^A$, $d \in \mathbb{N}_0$ und pos. Kosten

Dat.: Tabelle $w_j(c)$, $c = 0, \dots, OPT$ (wächst dyn.), $j \in V$

Out.: Länge OPT eines kürzesten d-Pfades von s nach t.

```
w_s(0) ← 0
forall j ∈ V\{s} { w_j(0) ← ∞ }
for c = 1,2,... {
    w_s(c) ← 0
    forall j ∈ V\{s} { w_j(c) ← min{ w_j(c-1), w_i(c-c_ij)+w_ij | ij ∈ A, c_ij ≤ c } }
    if w_t(c) < ∞ { OPT ← c; gib OPT aus; Ende }
}
```

Bedeutung: Der Algorithmus von Jaffe hat eine Laufzeit von $O(nmd)$ und Speicherplatzbedarf $O(nd)$.

Folgerung: RCSPs mit einer Ressource können in $O(nmd)$ und Speicherplatzbedarf $O(n^2 d)$ gelöst werden.

Folgerung: RCSP mit einer Ressource ist schwach NP-hart.

Algorithmus: Dynamische Programmierung für das RCSP mit einer Res. und pos. Kosten

(Hassin [1992])

Sei $c \in \mathbb{N}^A$ (zunächst nicht bekannt)

- Idee: Sei $g_j(c)$ die minimale Ressourcenverbrauch eines s-j-Pfades OPT Die Kosten eines kürzesten d-Wegs von s nach t.
mit Kosten $\leq c$. Dann gilt die Rekursion

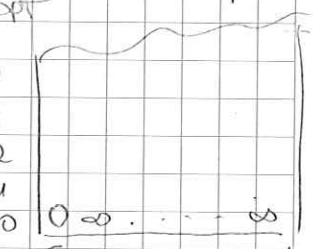
(W)
$$g_s(c) := 0, \quad c = 0, \dots, OPT$$
$$g_j(0) := \infty, \quad j \in V \setminus \{s\}$$
$$g_j(c) := \min \{ g_j(c-1), g_i(c-c_{ij}) + w_{ij} \mid ij \in A, c_{ij} \leq c \}$$
$$c = 1, \dots, OPT, \quad j \in V \setminus \{s\}.$$

Alg. von Hassin

Input: RCSP ($k=1$) mit $D=(V,A)$, $w \in \mathbb{N}_0^A$, pos. Kosten $c \in \mathbb{N}^A$, $d \in \mathbb{N}_0$.

Datenstr. Tabelle $g_j(c)$, $j \in V$, $c = 0, \dots, OPT$

$p_j(c)$, $j \in V$, $c = 0, \dots, OPT$



Output: kürzester d-Weg von s nach t, Länge OPT

Init: $g_s(0) \leftarrow 0$, $p_s(0) \leftarrow \emptyset$

forall $j \in V \setminus \{s\}$ { $g_j(0) \leftarrow \infty$, $p_j(0) \leftarrow \emptyset$ }

Schleife: for $c = 1, \dots$ {

~~$g_s(c) \leftarrow 0$ $p_s(c) \leftarrow \emptyset$~~

forall $j \in V \setminus \{s\}$ {

$g_j(c) \leftarrow \min \{ g_j(c-1), g_i(c-c_{ij}) + w_{ij} \mid ij \in A, c_{ij} \leq c \}$

$p_j(c) \leftarrow$ argmin $g_j(c)$

} if $g_t(c) < \infty$ { $OPT \leftarrow g_t(c)$; break }

Ausgabe: für OPT aus

$j \leftarrow t$, $i \leftarrow p_j(c)$, gib j, i aus

while $i \neq s$ { $c \leftarrow c - c_{ij}$, $j \leftarrow i$, $i \leftarrow p_j(c)$, gib i aus }

Lemma: Der Alg. von Hassin hat eine Laufzeit von $O(n \cdot OPT)$ und einen Speicherplatzbedarf von $O(n \cdot OPT)$.

Folg.: Der Alg von Hassin löst alg. RCSPs in $O(n \cdot OPT)$ und einen Speicherplatzbedarf von $O(n^2 \cdot OPT)$

Vollpolynomielles Approximationsschema für RCSPs und einen Pol

- Idee 1: Nutzen poly. Test für $OPT \geq V$ und bestimme V mit binärer Suche in $\{0, \dots, UB\}$, z.B. $UB = n \max c_{ij}$. Dann ist das Optimum in $O(\log UB)$ Aufrufen des Tests gefunden.

  RCSP hat viel $\Rightarrow$ sowieder approx. Test gut

  $OPT \geq V$ oder $OPT < V(1+\varepsilon)$ für $0 < \varepsilon < 1$.

- Def.: Ein Alg. A ist ein ε-approximativer Alg. für ein Maximierungsproblem und Opt. Lösung c^*, wenn A für jede Instanz in poly. Zeit und $\varepsilon > 0$ des Opt.p. eine Lösung mit Wert c^A findet, so dass

  $$c^A \geq (1+\varepsilon) c^*.$$

  (FPAS)

  Ein Alg ist ein poly. Approx schema, wenn A für jedes $\varepsilon > 0$ ein ε-approx. Alg. ist.

  Ein Alg A ist ein vollpoly. Approx schema, wenn A ein FPAS ist für jedes $\varepsilon > 0$ dessen Laufzeit poly. in $\frac{1}{\varepsilon}$ ist.

- Idee 2: Runde Kosten zu

  $$\tilde{c}_{ij} \leftarrow \left\lfloor \frac{c_{ij}}{V\varepsilon/(n-1)} \right\rfloor \cdot V\varepsilon/(n-1)$$

  $\Rightarrow \quad c_{ij} \geq \tilde{c}_{ij} \geq c_{ij} - V\varepsilon/(n-1)$

  $\Rightarrow c(P) \geq \tilde{c}(P) \geq c(P) - V\varepsilon$

  $\Rightarrow$ $\tilde{c}(P) \geq V \Rightarrow c(P) \geq V$!

  $\tilde{c}(P) < V \Rightarrow c(P) \leq \tilde{c}(P) + V\varepsilon \leq V(1+\varepsilon)$

  $\Rightarrow$ Runde Alg von Hassin mit Kosten $\bar{c}_{ij} := \left\lfloor \frac{c_{ij}}{V\varepsilon/(n-1)} \right\rfloor$ ...

Lemma: Der Alg. von Hassin hat eine Laufzeit von $O(OPT \cdot n)$ und einen Speicherplatzbedarf von $O(OPT \cdot n)$.

Beob.: Analog zum Alg. von Jakob kann der Alg von Hassin so modifiziert werden, dass er allg. λ-RCSTs mit Knotenkg. gewichten und Kanten in $O(OPTm + OPT n \log n)$ löst.

Alg. ε-Approximation für das λ-RCSP

Inp.: λ-RCST mit $D=(V,A)$, $w, c \in \mathbb{N}_0^A$, $d \in \mathbb{N}_0$

Schranken $L, U \in \mathbb{N}$ mit $L \le OPT \le U \le 2L$

Approx güte $0 < \varepsilon < 1$

Def.: gerundete Kosten $\bar{c}_{ij}$, $ij \in A$

Def.: Länge V eines λ-Pfades von s nach t mit $OPT \le V \le (1+\varepsilon) OPT$

$\forall ij \in A$ { $\bar{c}_{ij} \leftarrow \left\lfloor \frac{c_{ij}}{\varepsilon L/(n-1)} \right\rfloor$ } (7)

$\overline{OPT} \leftarrow$ Alg von Hassin für λ-RCST mit $D, w, \bar{c}, d$ (8)

$\bar{P} \leftarrow$ Pfad mit $\bar{c}(\bar{P}) = \overline{OPT}$ (9)

$V \leftarrow c(\bar{P})$, gib V aus

Lemma: Das obige ε-Approx schema ist korrekt.

Bew.: Sei P der bzgl. c kürzeste λ-Pfad von s nach t

$$\tilde{c}_{ij} := \bar{c}_{ij} \cdot \frac{L\varepsilon}{n-1} \quad \forall ij \in A$$

$$\Rightarrow c_{ij} \ge \tilde{c}_{ij} > c_{ij} - \frac{L\varepsilon}{n-1} \quad \forall ij$$

$$\Rightarrow c(P) \ge \tilde{c}(P) \ge \tilde{c}(\bar{P}) \ge c(\bar{P}) - L\varepsilon \ge c(P) - L\varepsilon$$

$$\Rightarrow \underbrace{c(P)}_{=OPT} + \underbrace{L\varepsilon}_{\le OPT\varepsilon} \ge \underbrace{c(\bar{P})}_{=V} \ge \underbrace{c(P)}_{=OPT}$$

$$\rightsquigarrow OPT(1+\varepsilon) \ge V \ge OPT$$

Lemma: Die Laufzeit des obigen ε-Approx. schemas ist $O\left(\frac{mn^2}{\varepsilon} + \frac{n^3}{\varepsilon}\log n\right)$.

Begr.: (7): $O(m \log \frac{n}{\varepsilon})$ dann

$\frac{c_{ij}(n-1)}{\varepsilon L} \le \frac{U(n-1)}{L \cdot \varepsilon} \le 2(n-1)\cdot\frac{1}{\varepsilon}$

$\Rightarrow$ $\bar{c}_{ij}$ (ganz Zahl eines Rests $\le 2(n-1)\frac{1}{\varepsilon}$) kann in $O(\log \frac{n}{\varepsilon})$ berechnet werden

(8): $O\left(\underbrace{n \max \bar{c}_{ij}}_{\le 2(n-1)\frac{1}{\varepsilon}} \cdot m + \underbrace{n^2 \max \bar{c}_{ij} \log n}_{\le 2(n-1)\frac{1}{\varepsilon}}\right) = O\left(\frac{mn^2}{\varepsilon} + \frac{n^3}{\varepsilon}\log n\right)$

(9): $O(n)$

Lösung Relaxierung für das RCSP in azyklischen Digraphen

Def. $D=(V,A)$ azyklisch $:\Leftrightarrow D$ enthält keinen gerichteten Kreis.

Lemma: $D=(V,A)$, $V=\{1,\dots,n\}$

D azyklisch $\Leftrightarrow \exists$ Permutation $\pi:\{1,\dots,n\}\to\{1,\dots,n\}: A\subseteq\{\pi_i\pi_j : i<j\}$

Def.: π wie im Lemma heißt topologische Ordnung von V.

Alg. Top. Ordnung

Inp. Digraph $D=(V,A)$

Out. Top. Ordnung π oder gerader Kreis $C\subseteq D$

```
for (i ← 1; V ≠ ∅; i ← i+1) {
    if (∃ v : δ⁺(v) = ∅) { π_i ← v }
    else { starte in irgendeinem Knoten, wandere Kante um Kante, bis
           sich ein Knoten v wiederholt, C ← Knotenfolge v→v, gib C aus
           , Ende } }
```

Lemma: Alg. Top. Ordnung ist korrekt

Bew. Induktion.

Def. Azyklisches RCSP (ARCSP)

RCSP = ARCSP $:\Leftrightarrow D=(V,A)$ azyklisch.

O.B.d.A. $s=1$, $t=n$.

Bem.: Viele Probleme im Umfeld von ARCSPs (regulär), z.B. VSP, DSP, ...

Vl.: Sei ein ARCSP mit $D=(V,A)$, $|V|=n$, $|A|=m$, $|R|=k$,

$W=\begin{pmatrix} w^1 \\ \vdots \\ w^k \end{pmatrix}\in\mathbb{N}_0^{k\times m}$, $d=\begin{pmatrix} d^1 \\ \vdots \\ d^k \end{pmatrix}\in\mathbb{N}^k$, $c\in\mathbb{Z}^m$

(ARCSP) $\min\ c^T x$

$$x(\delta^+(v)) - x(\delta^-(v)) = b_{1,n}(v) \quad \forall v\in V \qquad (i)$$

$$w^r x \le d^r \quad \forall r\in R \qquad (ii)$$

$$x \ge 0 \qquad (iii)$$

$$x\in\mathbb{Z}^m \qquad (iv)$$

ist eine IP-Formulierung von ARCSP, wobei

$$b_{1,n}(v) := \begin{cases} 1, & v=1 \\ 0, & v\neq 1,n \\ -1, & v=n \end{cases}.$$

(ARCSP) $\min c^T x,\ x(\delta^+(v)) - x(\delta^-(v)) = b_{1,n}(v),\ Wx\le d,\ x\ge 0$

die LP-Relax. von ARCSP

Lemma: (ARCSP) ist eine IP-Form. von ARCSP.

Beweis: (i), (iii), (iv) $\Rightarrow x$ ist ein gerader 1-n-Pfad $\Rightarrow x\in\{0,1\}^m$

Bd. ARCSP nichttriv. :$\Leftrightarrow$ ARCSP zul.

Lemma: ARCSP nichttrivial $\Leftrightarrow$ $\min w^v x$ $\le d^v \quad \forall v \in R$.
$x(\delta^+(u)) - x(\delta^-(u)) = \delta_{in}(u)$
$x \ge 0$

(klass.) Kürzeste-Wege-Problem

Def. Lagrange-Relaxierung. Zgg. ein nichttriv. ARCSP. Für $\lambda \in \mathbb{R}^k_+$ setze

$$\bar{c}^T := \bar{c}^T(\lambda) := c^T + \lambda^T W$$

$$f(\lambda) = \min \underbrace{c^T x + \lambda^T \underbrace{(Wx - d)}_{\le 0}}_{\bar{c}^T x}, \quad x(\delta^+(v)) - x(\delta^-(v)) = \delta_{in}(v), \; x \ge 0$$

$$x^*(\lambda) := \arg\min f(\lambda) \quad (\text{Mehrdeut. beliebig aufgelöst})$$

$$L := \max_{\lambda \ge 0} f(\lambda) = \max_{\lambda \ge 0} \bar{c}^T x - \lambda^T d, \quad x(\delta^+(v)) - x(\delta^-(v)) = \delta_{in}(v), \; x \ge 0$$

Lagrange-Relax. von (ARCSP).

Lemma:

(i) $f(\lambda)$, $x^*(\lambda)$ sind wohldef. ✓

(ii) L ist wohldef.

Bew.: (a) x Lsg von ARCSP $\to Wx - d \le 0 \Rightarrow \bar{c}(\lambda)^T x = c^T x + \lambda^T \underbrace{(Wx - d)}_{\le 0} \le c^T x < \infty$.

(b) $f(\lambda) = \min_{P \text{ zul. } 1n\text{-Pfad}} \bar{c}^T \chi(P) - \lambda^T d \Rightarrow f$ Min. über eine endl. Menge aff. Funkt.

$\overset{(a),(b)}{\Rightarrow}$ max ex.

(iii) $\forall$ Lsg. x von (ARCSP), $\forall u \in \mathbb{R}^k_+$ gilt: $c^T x \ge \bar{c}^T x - \lambda^T d$

d.h. $\bar{c}^T x - \lambda^T d$ ist immer eine untere Schranke für $c^T x$

(iv) $L = \max_{\lambda \ge 0} f(\lambda) = \min c^T x$
$x(\delta^+(v)) - x(\delta^-(v)) = \delta_{in}(v)$, $Wx \le d$, $x \ge 0$

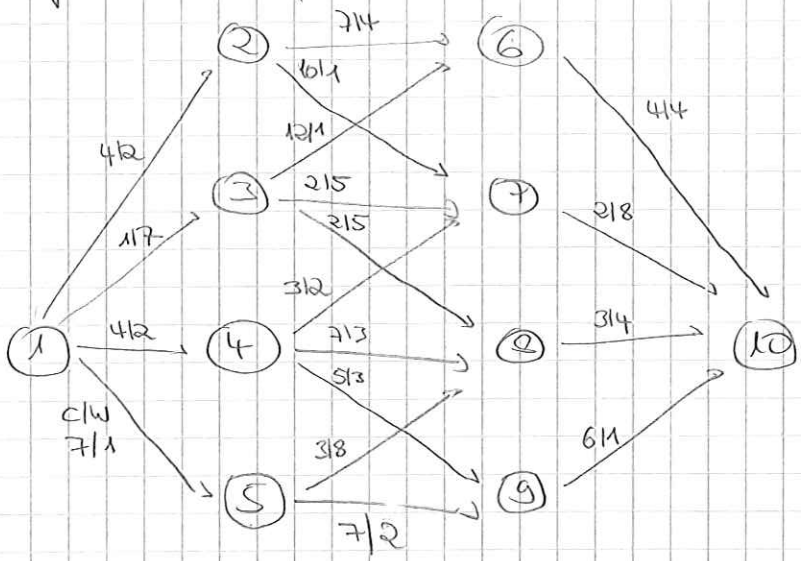
Bew.: $\min c^T x$
$x(\delta^+(v)) - x(\delta^-(v)) = \delta_{in}(v)$
$-Wx \ge -d$
$x \ge 0$
$\Leftrightarrow \max \mu^T \delta_{in} - \lambda^T d$
$\mu^T A - \lambda^T W \le c^T$
$\lambda \ge 0$

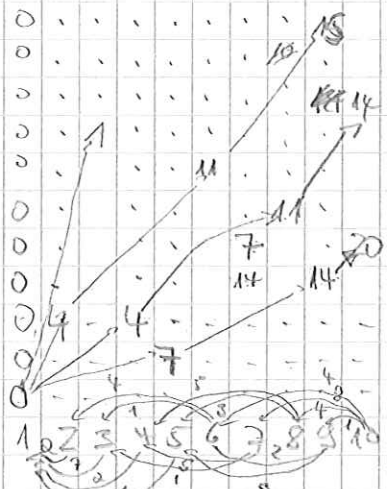
$\Leftrightarrow \max_{\lambda \ge 0} \min (c^T + \lambda^T W) x - \lambda^T d$, $x(\delta^+(v)) - x(\delta^-(v)) = \delta_{in}(v)$, $x \ge 0$. $\Leftrightarrow \max_{\lambda \ge 0} -\lambda^T d + \max \mu^T \delta_{in}$, $\mu^T A \le c^T + \lambda^T W$

z.B.

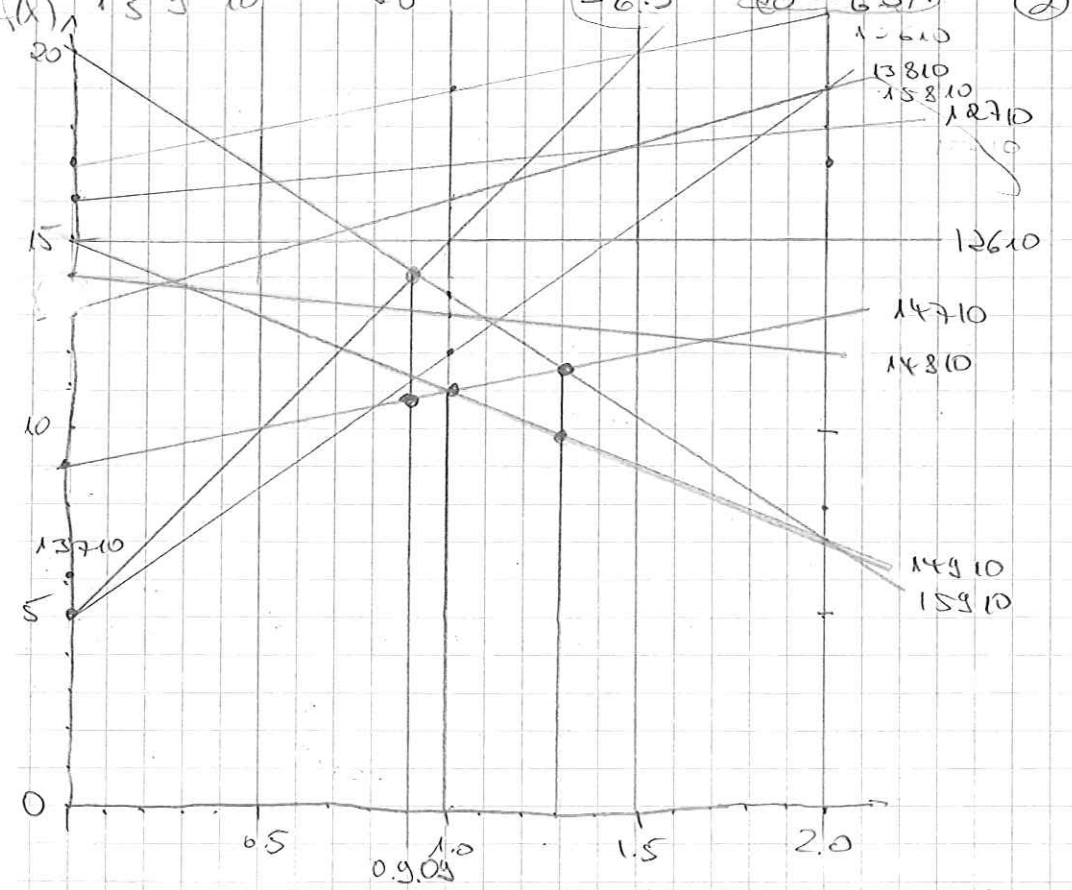
Bem.: Die Lagrange-Rel. kann mit einem Subgradientenverf. gelöst werden.

Bsp.: $k=1$, $w^T x \le 10 \Leftrightarrow -w^T x \ge -10 \Leftrightarrow 10 - w^T x \ge 0$





i	X	$c^T x$	$w^T x - 10$	$f_i(\lambda) = c^T x + \lambda(w^T x - 10) = c^T x - \lambda(10 - w^T x)$	
1	1 2 6 10	15	0	15	
2	1 2 7 10	16	1	$16 + \lambda$	
3	1 3 6 10	17	2	$17 + 2\lambda$	
4	1 3 7 10	5	10	$5 + 10\lambda$	(1)
5	1 3 8 10	6	16	$6 + 6\lambda$	
6	1 4 7 10	9	2	$9 + 2\lambda$ ← wahrsch.	(3)
7	1 4 8 10	14 *	−1 *	$14 - \lambda$ ← LP-optimal	
8	1 4 9 10	15	−4	$15 - 4\lambda$ ← wahrsch. opt.	(4)
9	1 5 8 10	13	3	$13 + 3\lambda$	
10	1 5 9 10	0 0	−6.5	$20 - 6.5\lambda$	(2)



$5 + 10\lambda = 20 - 6.5\lambda \Leftrightarrow 15 = 16.5\lambda = \frac{33}{2}\lambda \Leftrightarrow \lambda = \frac{15 \cdot 2}{33} = 0.\overline{909}$

$9 + 2\lambda = 20 - 6.5\lambda \Leftrightarrow 11 = 8.5\lambda = \frac{17}{2}\lambda \Leftrightarrow \lambda = \frac{22}{17} = 1.2941...$

$15 - 4\lambda = 9 + 2\lambda \Leftrightarrow 6 = 6\lambda \Leftrightarrow \lambda = 1.0$

Lemma: Sei $\lambda \geq 0$, $P_{ij} = \{P : P_{ij}\text{-Pfad in } D\}$

$\tilde{c}_{ij} = \min_{P \in P_{ij}} c(P)$

$\tilde{w}_{ij} = \min_{P \in P_{ij}} w(P)$

$\tilde{\bar{c}}_{ij} = \min_{P \in P_{ij}} \bar{c}(P)$

Sei $j \in V$, $P \in P_{ij}$, $Q \in P_{jn}$.

$c(P) + \tilde{c}_{jn} \leq c(P) + c(Q)$

$\bar{c}(P) + \tilde{\bar{c}}_{jn} \leq \bar{c}(P) + \bar{c}(Q) \leq c(P) + c(Q)$

$w(P) + \tilde{w}_{jn} \leq w(P) + w(Q)$.

Alg. Branch & Bound.

P: $A\lambda$-RCSP.

Out: Länge eines kürzesten $1n$-d-Wegs.

if ($\min_{P \in P_{1n}} w(P) > d$) { gib ∞ aus; Ende }

$\lambda^* \leftarrow \operatorname{argmax}_{\lambda \geq 0} f(\lambda)$

$\bar{c} \leftarrow c + \lambda^* w$

forall $j \in V$ { $\tilde{c}_{jn} \leftarrow \min_{P \in P_{jn}} c(P)$; $\tilde{w}_{jn} \leftarrow \min_{P \in P_{jn}} w(P)$; $\tilde{\bar{c}}_{jn} \leftarrow \min_{P \in P_{jn}} \bar{c}(P)$ }

$L \leftarrow \{1\}$; $U \leftarrow c(\operatorname{argmin} w(P))$

while($L \neq \emptyset$){

wähle $P \in L$

$L \leftarrow L \setminus \{P\}$

$i \leftarrow$ letzter Knoten in P

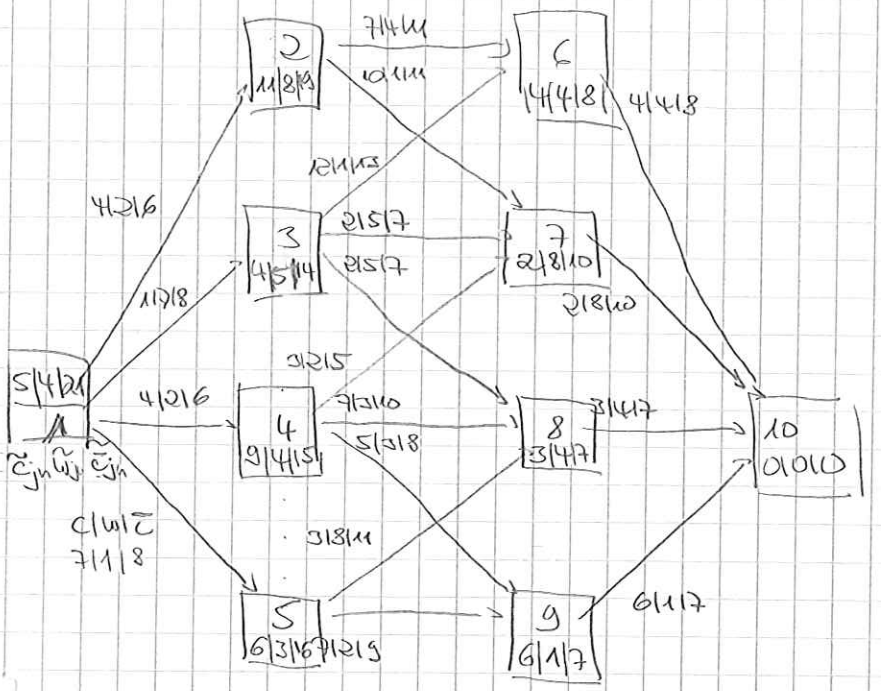
if($c(P) + \tilde{c}_{in} \geq U \vee w(P) + \tilde{w}_{in} > d \vee \bar{c}(P) + \tilde{\bar{c}}_{in} \geq U$){next i}

if $i = n$ { $U \leftarrow \min\{U, c(P)\}$; break i }

forall $ij \in A$ { $L \leftarrow L \cup \{P_j\}$ }

}

Isp. (Fortsetzung)



$\tilde{c}_{jn}$	$\tilde{w}_{jn}$	$\tilde{c}_{jn}$
1	5	4
2	11	8
3	4	5
4	9	4
5	6	3
6	4	4
7	2	8
8	3	4
9	6	1
10	0	0

	$\tilde{c}$
	21
	19
	14
	15
	16
	8
	10
	7
	7
	0

T	C(T)	w(T)	$\bar{c}(T) - \lambda d$	$u = 15$
1	0	0	-10	
12	4	2	-4	
13	1	7	-2	
14	4	2	-4	
147	7	4	1	
148	11	5	6	
1 4 8 10	14	9	11	$u = 14$
149	9	5	4	
15	7	1	-2	

Dynamische Programmierung für das RCSP (Multilabel-Algorithmen)

Voraus.: RCSP mit Digraph $D=(V,A)$, $|V|=n$, $|A|=m$, Start- und Zielknoten $s,t \in V$, Ressourcen R, Ressourcenverbräuchen $w^r \in \mathbb{N}_0^A$, Ressourcenlimits $d \in \mathbb{N}^R$, Kosten $c \in \mathbb{Z}^A$.

Bez.: a) Ein Tupel $(w_v, c_v) \in \bigtimes_{r \in R} \{0,1,\dots,d^r\} \times \mathbb{Z}$ heißt Label von $v \in V$.

b) (w_v, c_v) zulässiges bzw. primales Label von $v \in V$

$:\Leftrightarrow \exists$ sv-T-Pfad P : $w_v = w(P)$, $c_v = c(P)$

c) (w_v, c_v) duales Label von $v \in V$

$:\Leftrightarrow \nexists$ sv-T-Pfad P : $(w,c)(P) \leqq (w_v, c_v)$.

Bem.: RCSP ist äquivalent zu einem Kürzeste-Wege-Problem in einem „ressourcenexpandierten" Digraphen mit $n \cdot \Delta$ Knoten, wobei $\Delta = \prod_{r \in R} d^r$

Beg.: Ersetze jeden Knoten v durch $\prod_{r \in R} d^r$ Kopien, die die möglichen Ressourcenverbräuche bis v einnehmen und kodiere entsprechend.

Bem.: Wenn $w_{ij} > 0$ $\forall ij \in A$ gilt, ist dieser res. exp. Digraph azyklisch und ein kürzester st-Weg kann in $O(\Delta^2)$ berechnet werden.

Idee: Erzeuge nicht alle Ressourcenverbräuche explizit, sondern behandle „dominierte" implizit.

c) Seien (w'_v, c'_v), (w''_v, c''_v) irgendwelche Label (d.h. weder notw. prim. oder dual)

(w'_v, c'_v) dominiert (w''_v, c''_v) (voneinander) $:\Leftrightarrow (w'_v, c'_v) \leqq (w''_v, c''_v)$

d) Sei P eine Menge von Labeln von $v \in V$.

(w_v, c_v) effizient (bzgl. P) $:\Leftrightarrow \nexists (w'_v, c'_v) \in P : (w'_v, c'_v)$ dom. (w_v, c_v)

$\mathrm{eff}(P) := \{(w_v, c_v) \in P : (w_v, c_v)$ effizient $\}$

$= \{(w^1_v, c^1_v), \dots, (w^q_v, c^q_v)\}$.

f) Sei P eine Menge von Labeln von v

$$f_P : \begin{cases} \bigtimes_{r \in R} \{0,1,\dots,d^r\} \to \mathbb{Z} \cup \{+\infty\} \\ w \mapsto \min\{c_j,\ w_j \leqq w,\ (w_j, c_j) \in P\} \end{cases}$$

Kostenfunktion (zu P)

Bem.: $f_P = f_{\mathrm{eff}(P)}$, f_P ist stückweise konstant.

g) Seien P und Q Mengen von primalen bzw. dualen Labeln von $v \in V$.

$x^*(MLP) = \left(1, \frac{2}{3}, 0, \frac{1}{3}\right)$, $v(IP) = 3+2+1 = 6$

$$X = \left\{ \begin{pmatrix}0\\0\\0\\0\end{pmatrix}, \begin{pmatrix}1\\0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\0\\1\end{pmatrix}, \begin{pmatrix}1\\0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\1\\0\end{pmatrix} \right\}$$

$$Y_1 = \left\{ \begin{pmatrix}0\\0\\0\\0\end{pmatrix}, \begin{pmatrix}1\\0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\\0\end{pmatrix} \right\}, \quad Y_2 = \left\{ \begin{pmatrix}0\\0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\\0\end{pmatrix}, \begin{pmatrix}0\\0\\0\\1\end{pmatrix} \right\}$$

$$X \cap \{ x_{11} + x_{21} \le 1,\ x_{13} + x_{23} \le 1 \} = (Y_1 + Y_2) \cap \{ y_{12} + y_{22} \le 1,\ y_{13} + y_{23} \le 1 \}$$

$$\begin{aligned}
(MIP)\quad \max\ & 2\lambda_{12} + 3\lambda_{13} + 1\lambda_{22} + 1\lambda_{23} \\
& \lambda_{12} + \lambda_{22} \le 1 \\
& \lambda_{13} + \lambda_{23} \le 1 \\
& \lambda_{12} + \lambda_{13} \le 1 \\
& \lambda_{22} + \lambda_{23} \le 1 \\
& \lambda \ge 0 \\
& \lambda \in \{0,1\}^{2\times 3}
\end{aligned}$$

$x^*(MLP) = (0,1,0,0,0,1)$, $v(MIP) = 4 < 6 = v(IP)$

Spaltenerzeugung und Dantzig & Price

Def. Sei $A \in \mathbb{Q}^{m\times n}$, $c \in \mathbb{Q}^n$, $b \in \mathbb{Q}^m$.

a) (MLP) $\max c^T x,\ Ax \le b$ — Master-LP
(MIP) $\max c^T x,\ Ax \le b,\ x \in \mathbb{Z}^n$ — Master-IP

b) Sei $J \subseteq \{1,\dots,n\}$ eine Teilmenge der Spalten von MLP.
$(RMLP)(J)$ $\max c_J^T x_J,\ A_{\cdot J} x_J \le b$ — Reduziertes Master-LP
$(RMIP)(J)$ $\max c_J^T x_J,\ A_{\cdot J} x_J \le b,\ x \in \mathbb{Z}^{|J|}$ — Reduziertes Master-IP

c) Sei π (ein dual optimale) Dualvar. von RLP
$(PRICE)(\pi)$ $\max_{j \in N} c_j - \pi^T A_{\cdot j}$ oder Subproblem bzw. Pricing-Problem

Alg.: Spaltenerzeugung (Gilmore & Gomory [1961])

Input: (RLP) $\max c_J^T x_J,\ A_{\cdot J} x_J \le b$ (bzw.: zulässig)
$(PRICE)$ Modell zur Lösung von $(PRICE)(\pi)$

Output: Opt lg x^* von (MLP) $\max c^T x,\ Ax \le b$

Bel.: $c_J,\ A_{\cdot J},\ x_J,\ J,\ \pi^*,\ j^*,\ x^*$

Lemma (Jansen, Roos, Terlaky [2001]): Sei $\varepsilon > 0$ und ε-PRICE ein ε-approximatives Verfahren zur Lösung von (PRICE), d.h. für $j_\varepsilon = \arg\max \varepsilon\text{-PRICE}$ gilt: $\max_{j \in N} (1-\varepsilon) c_j - \pi^T A_{\cdot j} > 0 \Rightarrow c_{j_\varepsilon} - \pi^T A_{\cdot j_\varepsilon} > 0$.

Sei ferner $x_\varepsilon = \text{colgen}(\text{RMLP}, \varepsilon\text{-PRICE})$, $\pi_\varepsilon = \text{dual}(\text{colgen}(\text{RMLP}, \varepsilon\text{-PRICE}))$ und $x^* = \text{colgen}(\text{RMLP}, \text{PRICE})$. Dann gilt:

$$c^T x_\varepsilon \geq (1-\varepsilon)\, c^T x^*.$$

Bew.: Es gilt $(1-\varepsilon) c_j - \pi_\varepsilon^T A_{\cdot j} \leq 0 \quad \forall j \in N$, d.h. π_ε ist dual zulässig für $\max (1-\varepsilon) c^T x, \; Ax \leq b$. Es folgt $\Leftrightarrow \pi^T A \geq (1-\varepsilon) c^T$

$$\begin{aligned} c^T x_\varepsilon = \pi_\varepsilon^T b &\geq \min \pi^T b, \; (1-\varepsilon) c^T - \pi^T A \leq 0, \; \pi \geq 0 \\ &= \max (1-\varepsilon) c^T x, \; Ax \leq b \\ &= (1-\varepsilon) \max c^T x, \; Ax \leq b \\ &= (1-\varepsilon)\, c^T x^*. \end{aligned}$$

Alg Spaltenerzeugung 2 (Grötschel, Jünger, Reinelt [1984]): Sei PRICE ein polynomiales Verfahren zur Lösung von (PRICE)(π). Dann kann (MLP) in polynomialer Zeit gelöst werden.

Bew.: Anwendung des Ellipsoidverfahrens auf das duale MLP

$$\max \pi^T b, \; \pi^T A \geq c^T, \; \pi \geq 0.$$

Alg.: Branch & Bound

Input: MIP, Branching-Regel β

Output: $x^* \in \arg\max \text{MIP}$

1. $Q \leftarrow \{\text{MLP}\}$, $v^* \leftarrow -\infty$;
2. while $Q \neq \emptyset$ {
3. wähle $\text{MLP}' \in Q$
4. if $Q \leftarrow Q \setminus \text{MLP}'$,
5. $x' \leftarrow \arg\max \text{MLP}'$;
6. if $(x' \in Z^n)$ $\{ Q \leftarrow Q \cup \beta(\text{MLP}', x') \}$
7. else if $(c^T x' > v^*)$ $\{ v^* \leftarrow c^T x', \; x^* \leftarrow x' \}$
8. }

Prop.: Der Alg. B&B ist korrekt und endlich.

Bew.: $|(\text{MIP})^{-1}| < \infty$ und $|(\text{MIP}'')^{-1}| < |\text{MIP}'^{-1}| \; \forall \text{MLP}'' \in \beta(\text{MLP}', x')$

Def: Der Graph $G = (V, E)$ mit $V = \{\text{MLP}' \in Q\}$, $E = \{(v, w) : v \in \beta(w, \cdot)\}$ heißt Branch & Bound-Baum.

Spalten_erzeugung (RMLP, PRICE)
do {
  $\pi^* \leftarrow$ optimale $\pi^T b$, $\pi^T A_{\cdot j} \geq c_j$, $\pi \geq 0$ (RLP)
  $j^* \leftarrow \operatorname{argmax}_{1 \leq j \leq n} \{ c_j - \pi^{*T} A_{\cdot j} = \bar{c}_j \}$ (PRICE) (π)
  if $\bar{c}_{j^*} > 0$ then $J \leftarrow J \cup \{j^*\}$
  return (π^*, j^*);
}
do { Colgen (RMLP, PRICE) } while $\bar{c}_{j^*} > 0$;
$x^* \leftarrow$ optimale RMLP
return x^*;

Lemma (Gil. Book): Sei $\varepsilon > 0$ und π_ε eine ε-PRICE-approx. Lösung zur Lösung von (PRICE)(π), d.h. (d.h. das Optimum j_ε von PRICE gibt: $\max_{1 \leq j \leq n} (1-\varepsilon) c_j - \pi^T A_{\cdot j} > 0$ $\Rightarrow c_{j_\varepsilon} - \pi_\varepsilon^T A_{\cdot j_\varepsilon} > 0$
Dann gilt für die Lsg x_ε = colgen(RMLP, ε-PRICE) im Vergleich zur $x^* = \operatorname{argmax}$ RMLP: $c^T x_\varepsilon \geq (1-\varepsilon) c^T x^*$.
Bew.: $c^T x_\varepsilon = \pi_\varepsilon^T b$, π_ε dz. zul. ... x_ε gib. Zielfkt.
es gilt $(1-\varepsilon) c_j - \pi_\varepsilon^T A_{\cdot j} \leq 0 \quad \forall j \in \mathbb{N}$,
d.h. π_ε ist dual zul. für $\max (1-\varepsilon) c^T x \; Ax \leq b$, argmax_x $\min \pi_\varepsilon^T b \geq (1-\varepsilon) c^T x^*$.

Lemma: Alg. Spaltenerzeugung ist korrekt und endlich

Bew.: Die Anzahl der Spalten des kanon. Problems ist endlich

Def.: Sei
(MLP) $\max c^T x \; Ax \leq b, \; x \in \mathbb{Z}^n$; (MLP) $\max c^T x \; Ax \leq b$ LSG.
$MLP^{-1} = \{x : Ax \leq b, x \in \mathbb{Z}^n\}$; $MLP^{-1} = \{x : Ax \leq b\}$ LSG.
$Ax \leq b$ eine beschreibende Ungleichung.
$(\beta, b)(MLP) = \{MLP, \exists' x \leq b' : \exists' x \leq b' \subset \exists x \leq b\}$
a) Eine Abb. $\beta : (\beta, b)(MLP) \times MLP^{-1} \to 2^{(\beta, b)(MLP)}$

Branching-Regel für MLP, wenn gilt:
i) $(MLP')^{-1} \subseteq (MLP)^{-1} \quad \forall MLP' \in (\beta, b)(MLP), MLP'' \in \beta(MLP', x'), x' \in MLP'^{-1}$
ii) $\bigcup_{MLP'' \in \beta(MLP', x')} (MLP'')^{-1} = (MLP')^{-1} \quad \forall MLP' \in (\beta, b)(MLP), x' \in MLP'^{-1}$
iii) $\bigcup_{MLP'' \in \beta(MLP', x')} (MLP'')^{-1} \not\ni x' \quad \forall MLP' \in (\beta, b)(MLP), x' \notin MLP'^{-1}$
iv) $\exists \Sigma : (MLP'')^{-1} = (MLP)^{-1} \quad \forall MLP'' \in \beta^k(MLP, \cdot)$

b) Pricing-
Def.: Ein Alg. PRICE heißt kompatibel zu einer Branching-Regel b für MLP, falls PRICE das Pricing-Problem zu RMLP' für jedes $MLP' \in b^k(MLP)$, $k \in \mathbb{N}$, lösen kann.

Alg. Branch & Price (Gilmore & Gomory [1961], Ryan/Foster, Johnson, Nemhauser, Savelsbergh, Vance [1994])
Inp.: RMLP
b Branching-Regel
PRICE mit b kompatibles Pricing-Problem
Out: argmax MIP

b) Wir schreiben

$$P(MLP', x') = \{ MLP'' : \forall' \exists'' x \leq b'' \setminus \exists k \leq b' \}$$

$$MLP'' \in P(MLP', x')$$

Def: $Q \subseteq 2^{(B,b)(RMLP)}$

$Q \leftarrow \{RMLP\}$

while $Q \neq \emptyset$ {

  wähle $RMLP \in Q$

  $Q \leftarrow Q \setminus RMLP$

  $\lambda \leftarrow$ Lösen $(RMLP, PRICE)$

  if $(\lambda \in \mathbb{Z}^u \wedge \bar{c}^T\lambda > \bar{c}^T\lambda^*)$ { $\lambda^* \leftarrow \lambda$; }

  elseif $(\lambda \notin \mathbb{Z}^u \wedge \bar{c}^T\lambda > \bar{c}^T\lambda^*)$ { $Q \leftarrow Q \cup b(RMLP, \lambda^*)$; }

}

Bsp. a) MMP

(IP) $\max c^T x \;|\; \sum_{k \in M} x_{ki} \le 1 \;\forall i \;|\; \sum_i w_i x_i \le d_k \;\forall k \;|\; x \in \{0,1\}^{M \times N}$

(MP) $\max \sum_k c_{pk} \lambda_{pk} \;|\; \sum_{p \in P_k \ni pi} \lambda_{pk} \le 1 \;\forall i \;|\; \sum_{p \in P_k} \lambda_{pk} = 1 \;\forall k \;|\; \lambda_{pk} \in \{0,1\}$

$(RMLP)$ $\max \sum_k c_{pk} \lambda_{pk} \;|\; \sum_{P_k \ni p \ni i} \lambda_{pk} \le 1 \;\forall i \;|\; \sum_{p \in P_k} \lambda_{pk} = 1 \;\forall k \;|\; \lambda_{pk} \ge 0$

$b(RMLP, \lambda^*)$: $\lambda_{pk} = 0 \;\vee\; \lambda_{pk} = 1$ für $\lambda_{pk} \in (0,1)$ „Branching auf neue Item"

$PRICE(\pi, \mu)$: $\max_{p \in P_k} c_{pk} - \sum_{p \ni i} \pi_i - \mu_k \quad \forall k$ (kompatibel (Fixierung des Items i in k))

$= \max_{p \in P_k} \sum_{p \ni i} \underbrace{(c_{ki} - \pi_i)}_{=: \bar{c}_{ki}} - \mu_k \quad \forall k$ Knapsack-Problem

$= \max_{p \in P_k} \sum_{p \ni i} \bar{c}_{ki} - \mu_k, \sum_{p \ni i} w_i \le d \quad \forall k$ (Knapsack k)

$\max \bar{c}^T x \;|\; Ax \le b \;|\; x \ge 0 \;\Leftrightarrow\; \max \gamma^T b \;|\; \gamma^T A \le \bar{c}^T \;|\; \bar{c}^T - \gamma^T A \ge 0$

b) w-ATSP

$(RMLP)$ $\min \sum_{i=1}^{w} \sum_{j \in E_i} \lambda_{ij} c(\gamma_{ij}) \;|\; \sum_{i=1}^{w} \sum_{j \in E_i} \lambda_{ij} \gamma_{ij}(\delta^+(v)) \ge 1 \;\forall v \neq 0 \;|\; \sum_{j \in E_i} \lambda_{ij} = 1 \;\forall i \;|\; \lambda \ge 0$

$(PRICE)$ $\min c(\gamma_{ij}) - \sum_{j \ni v \neq 0} \pi_v - \underbrace{\mu_i}_{= \text{const } \forall i} \quad \forall i$

$\pi_0 := 0$ $= \min \sum_{j \in E_i} c_a + \sum_{j \ni v} \pi_v - \mu_i \quad \forall i$

$\pi_A := \pi_a, a = uv$ $= \min \sum_{j \in E_i} \underbrace{(c_a + \pi_a)}_{= \bar{c}_a} - \mu_i \quad \forall i$

$= \min \sum_{j \in E_i} \bar{c}_a - \mu_i \quad \forall i$ Longest Simple Problem

$b(RMLP, \lambda^*)$: $\lambda_{ij} = 1 \;\forall e_i \ni j \ni a \;\vee\; \lambda_{ij} = 0 \;\forall i' \neq i, j \ni a, \forall e_i \ni j \not\ni a$ „Branching auf eine Kante"

c) CSP

$(RMLP)$ $\min \lambda_p c_p \;|\; -\sum \lambda_p w_p \ge -d \;|\; \sum \lambda_p = 1 \;|\; \lambda \ge 0$

$(PRICE)$ $\min c_p - (-\pi w_p) - \mu = \min c_p + \pi w_p - \mu$

$= \min \sum_{p \ni a} \underbrace{(c_a + \pi w_a)}_{= \bar{c}_a \ge 0} - \mu$ Kürzeste-Wege-Problem

$b(\text{EMLP}, \lambda^*)$: $\lambda_p = 1 \ \forall p \ni a \ \vee \ \lambda_p = 0 \ \forall p \not\ni a$ „Branching auf Kante a" (kompatibel)

Beob.: Beim MVT und m-ATSP bei EMLP die Form

$\min c^T\lambda, \ A\lambda \leqq \mathbb{1}, \ \lambda \geq 0, \ A \in \{0,1\}^{u \times v}$.

$\Leftrightarrow \min c^T\lambda, \ A\lambda + IS = \mathbb{1}, \ \lambda, S \geq 0$

$\Leftrightarrow \min c'^T\lambda', \ A'\lambda' = \mathbb{1}, \ \lambda' \geq 0$,

d.h. EMLP ist äquivalent zu einem Set-Partitioning-LP.

Satz (Ryan & Foster [1981]): Sei $A \in \{0,1\}^{u \times v}$ mit $A_{\cdot i} \neq A_{\cdot j} \ \forall i,j$ (keine doppelten Spalten) und $A_{ij} \neq 0 \ \forall i$ und $0 \leq x \leq \mathbb{1}, \ x \notin \mathbb{Z}^u$ mit $Ax = \mathbb{1}$. Dann gibt es $i,j \in \{1,\dots,u\}$ und $r,s \in \{1,\dots,v\}$ mit $0 < x_i, x_j < 1$ und

$$\begin{pmatrix} a_{ri} & a_{rj} \\ a_{si} & a_{sj} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

Bew.: Wähle $0 < x_i < 1$.

1. Fall: $A_{\cdot i} = \mathbb{1}$. $\exists j: 0 < x_j < 1, \ A_{\cdot j} \neq \mathbb{1}$ ✓

2. Fall: $A_{\cdot i} \neq \mathbb{1}$. O.B.d.A. sei $A_{\cdot i} = \begin{pmatrix} 1 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$ (1 bis r), dann $A_{ri} = 1$. $\exists j: 0 < x_j < 1, \ a_{rj} = 1$.

$$\begin{pmatrix} a_{ri} & a_{rj} \\ a_{si} & a_{sj} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \ \forall \ 1 \leq s \leq r$$

$$\begin{pmatrix} a_{ri} & a_{rj} \\ a_{si} & a_{sj} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \ \forall \ r+1 \leq s \leq m$$

$\Rightarrow A_{\cdot i} = A_{\cdot j}$ ↯.

Def.: Constraint-Branching-Regel für Set-Partitioning-Probleme (Ryan & Foster [1981]):

$b(c^Tx, Ax = \mathbb{1}, x \geq 0; x^*) : \sum_{j \in A_{i\cdot} \cap A_{j\cdot}} x_j = 1 \ \vee \sum_{j \in A_{i\cdot} \cap A_{j\cdot}} x_j = 0$, $\begin{pmatrix} a_{ri} & a_{rj} \\ a_{si} & a_{sj} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$, $0 < x_{ij} < 1$

Bem.: Die Ryan & Foster-Regel ist wohldefiniert, weil das Einzelziehen der Branching-Constraints nicht aus der Klasse der Set-Partitioning-Probleme herausführt.

Bsp. (Fahrzeugdisposition): Sei S eine Menge von Touren in einem Digraphen $D = (V, A)$ und (MLP) mit $\sum_{p \ni a} c_p x_p$, $\sum_{p \ni i} x_p = 1 \ \forall i \in V$, $x \geq 0$.

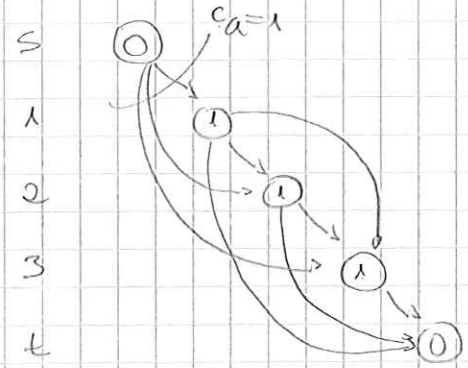
$b(MLP, x^*) : \sum_{p \ni ij} x_p = 1 \ \vee \sum_{p \ni ij} x_p = 0$ „Branching auf einem Bogen ij"

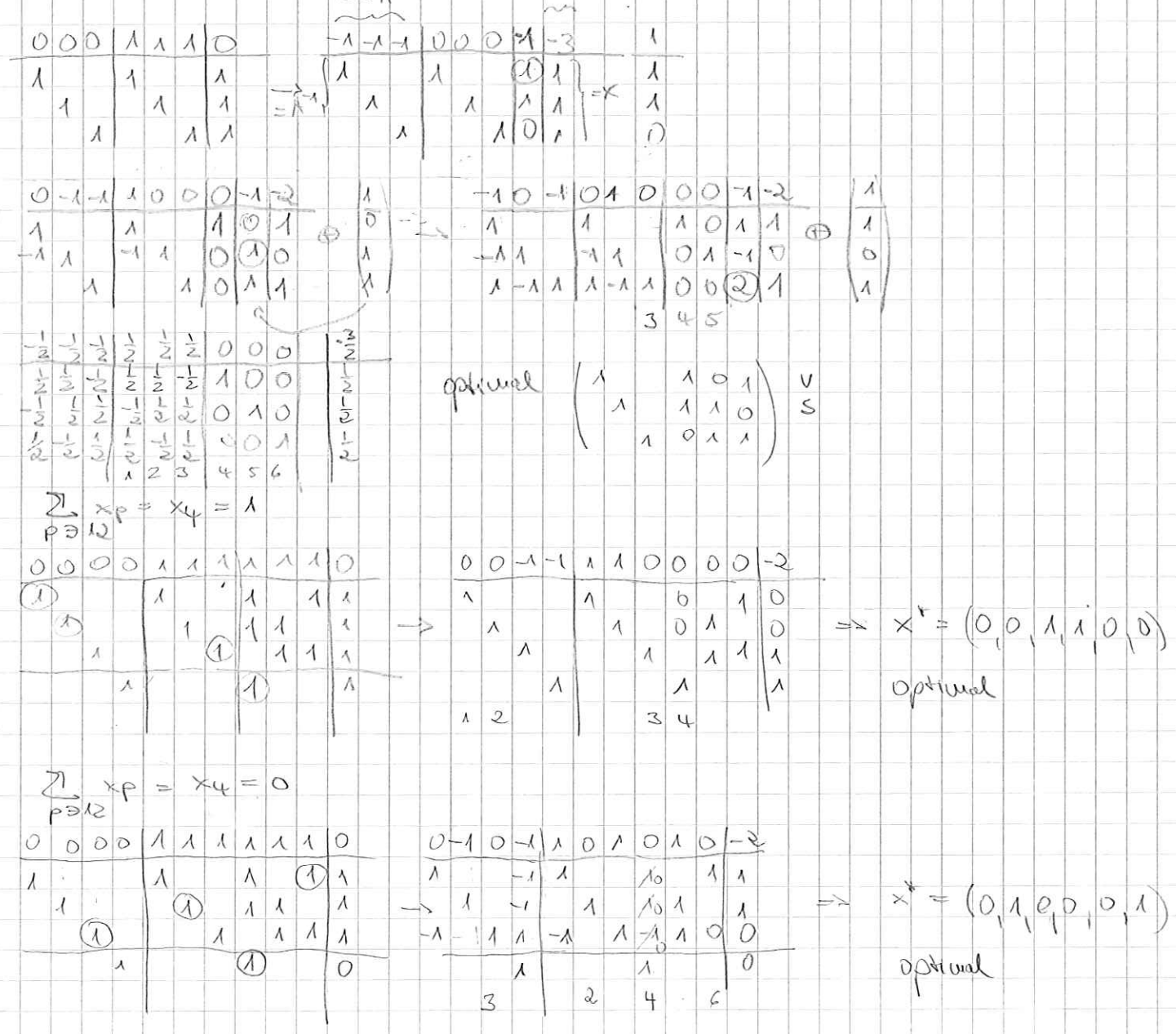
ist eine spezielle Version der Ryan & Foster-Regel.

Bem.: $\sum_{j \in A_{i\cdot} \cap A_{j\cdot}} x_j = 1 \Leftrightarrow \sum_{j \in A_{i\cdot} \triangle A_{j\cdot}} x_j = 0$. Die Ryan & Foster-Regel kann damit über Elimination von Spalten implementiert werden.

1p. (Pfadüberdeckung) (MLP) min $\sum_{P:w_P \le 2} c_P x_P$, $\sum_{P \ni v} x_P = 1 \ \forall v$, $x \ge 0$

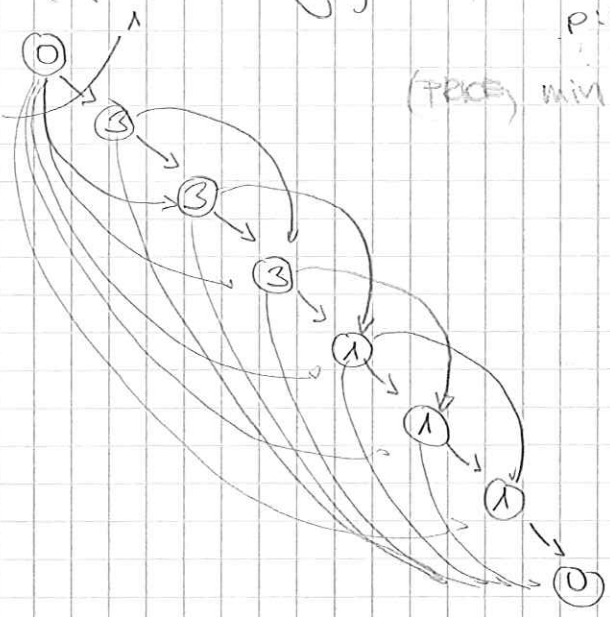
(DICE) min $c_P - \sum_{p \ni v} \pi_v = c_P - \pi_P \ge 0$

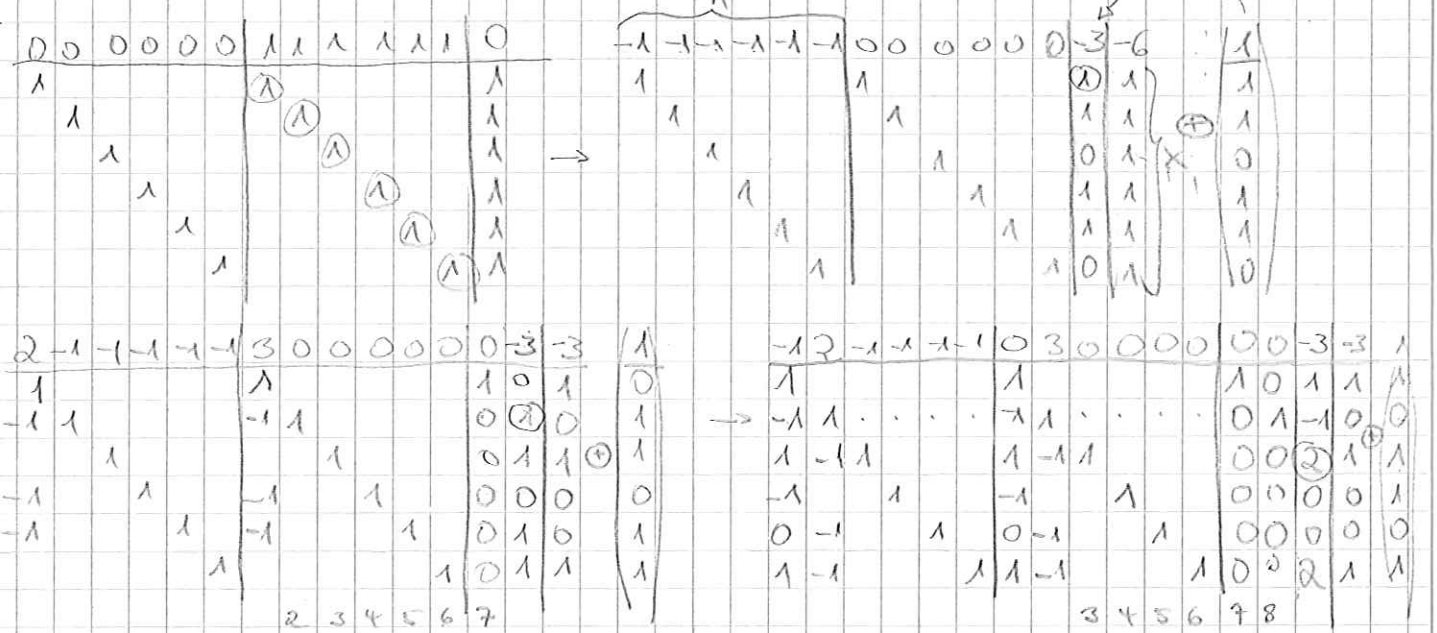




Bsp. (Rucksackbelegung): (KLT) $\min \sum_{p:w_p \le 8} c_p x_p \;,\; \sum_{p \in \{1,\dots,6\}} x_p = 1 \;,\; x \ge 0$

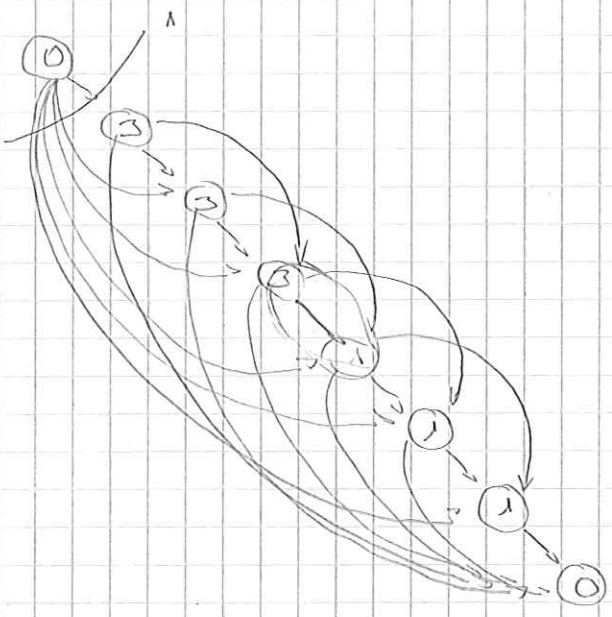
(DUAL) $\min c_p - \sum_{p \ge v} \pi_v = \min 1 - \pi_p \overset{!}{\le} 0$

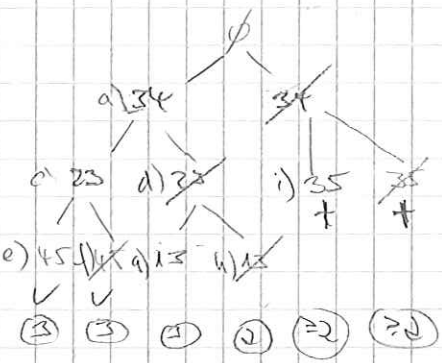


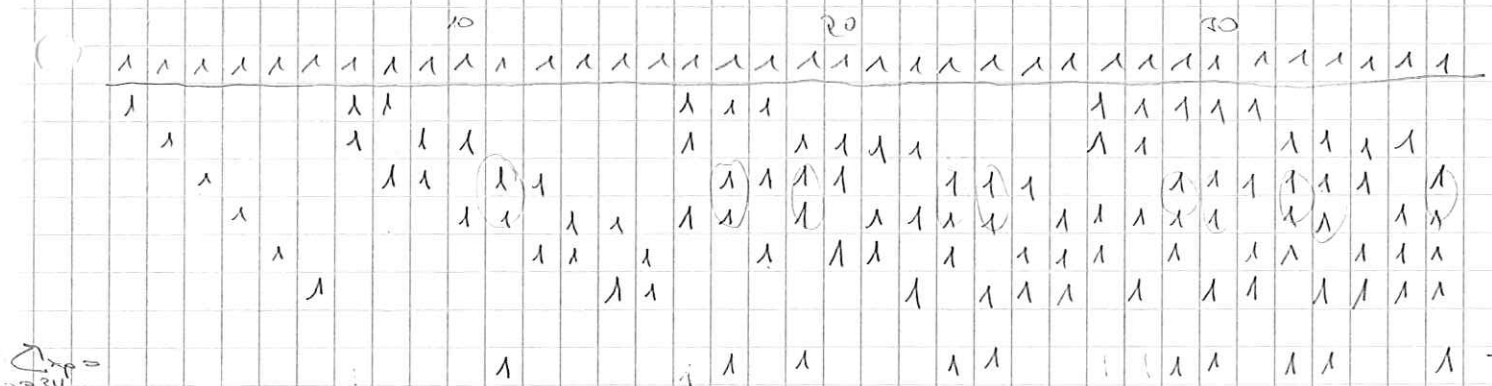


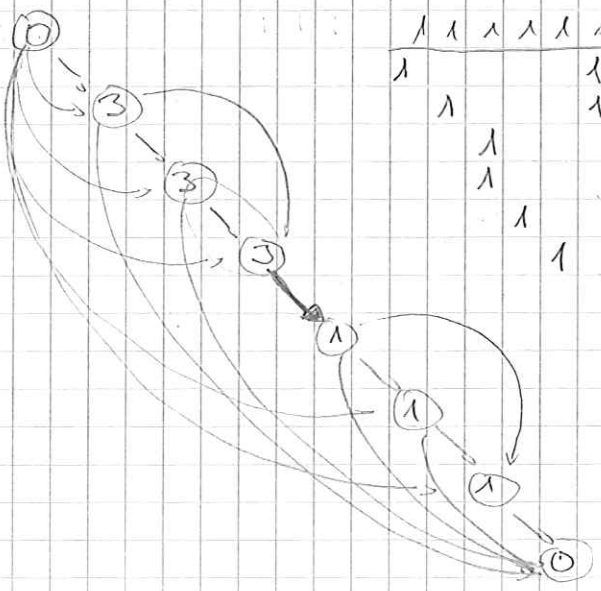
usw.

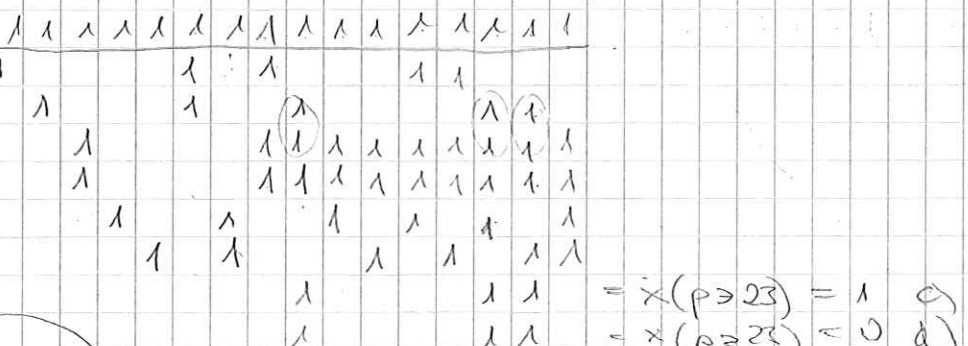
Bsp. (Pfad überdeckung) (MLP) $\min \sum_{p} \lambda_p \cdot x_p$, $\sum_{p \ni v} x_p = 1 \;\forall v$, $x \geq 0$
$p: \sum_{v \in p} w_v \leq B$

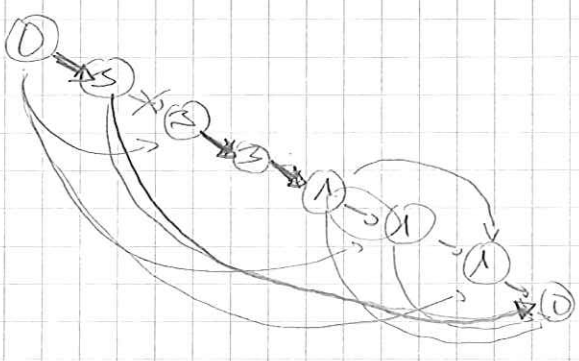


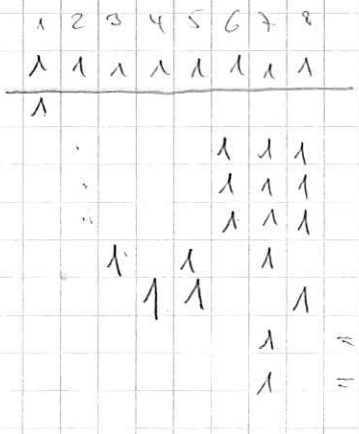


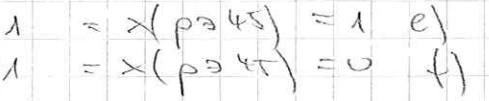




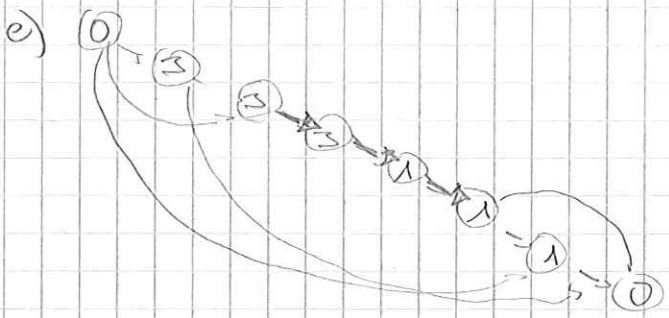




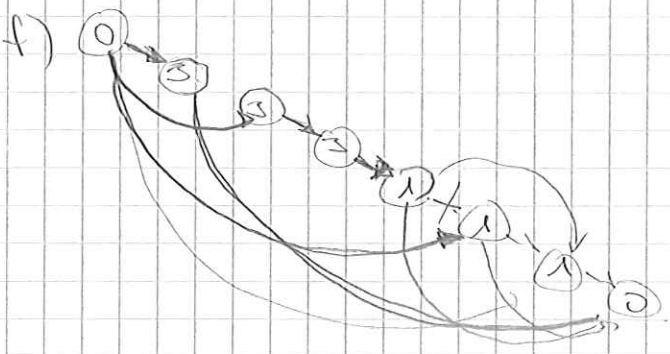


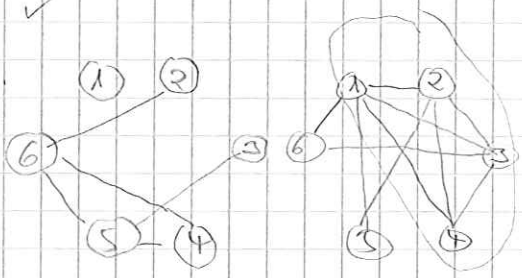


e)

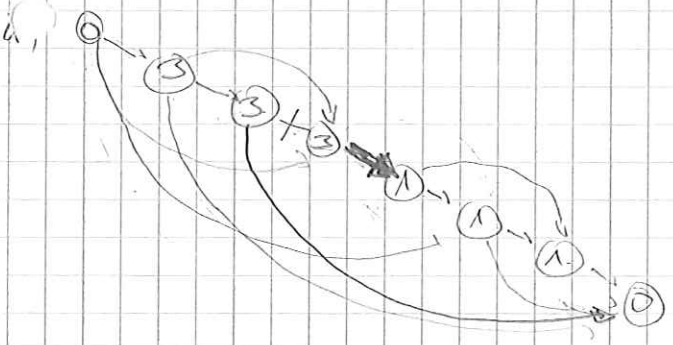


f)

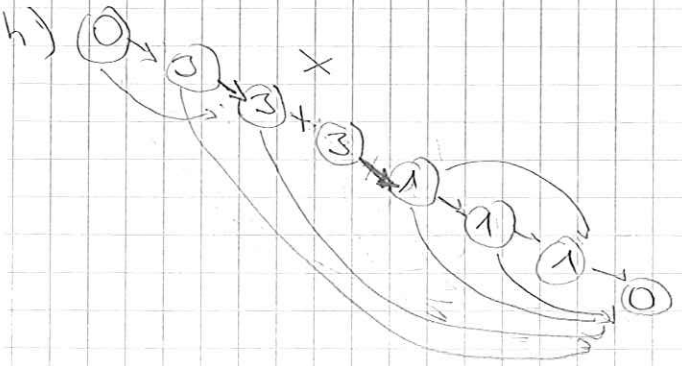


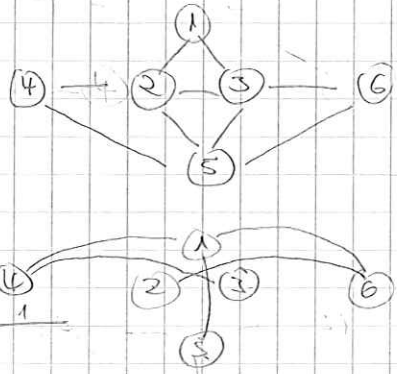


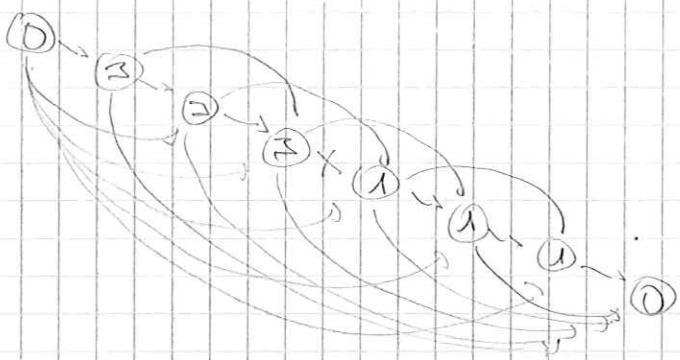
g)



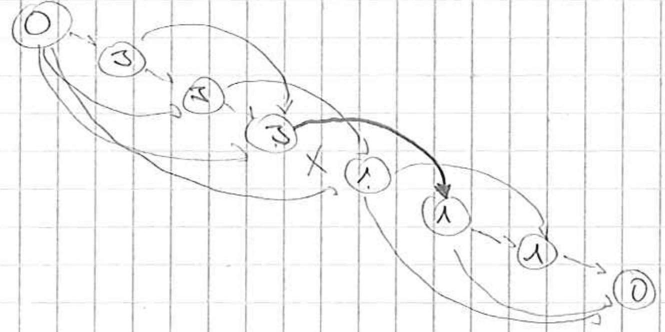
h)



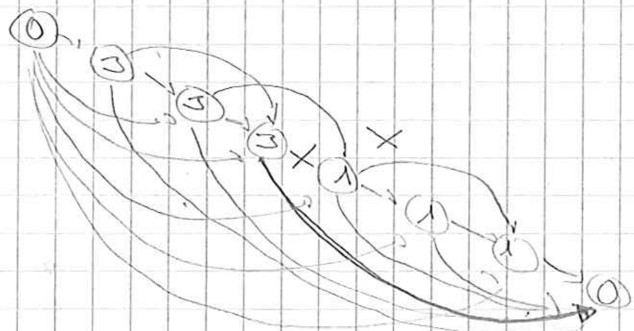


b) 

$x(p \ni 35) = 1$ i)
$x(p \ni 34) = 0$ j)

i) 

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
1	1	1	1	1	1											
	1	1	1			1	1	1	1	1						
				1	1		1	1			1	1				
		1	1						1	1			1	1		≥ 2
				1	1		1	1			1	1				
			1		1			1		1		1		1	1	

j) 

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
1	1	1	1	1	1																
	1	1	1	1		1	1	1	1	1	1										
					1		1					1									
		1	1	1				1	1	1	1		1	1	1	1					≥ 2
			1						1		1			1		1	1	1			
			1							1	1				1	1		1	1		

Def. Strong Branching (Bixby [?]) : Sei (MLP) $\min c^T x$, $Ax \geq b$, $x \in \mathbb{Z}^n$ und $b_1, \dots, b_k$ Branching-Regeln für MLP mit $b_i(MLP, x^*) = \{ MLP_i^1(x^*), MLP_i^2(x^*) \}$.
Sei $v_i^1 = \min MLP_i^1(x^*)$, $v_i^2 = \min MLP_i^2(x^*)$, $v_i^3 = \min\{v_i^1, v_i^2\}$, $i = 1, \dots, k$
v_i^1, v_i^2 heißen Probing-Werte bzgl. b_i.
$b(MLP, x^*) = b_{\operatorname{argmax} v_i^3}(MLP, x^*)$ Strong Branching bzgl. $b_1, \dots, b_k$

Bem. : In der Praxis verwendet man häufig auch Abschätzungen für v_i^1 und v_i^2. In Spaltengen. verfahren generiert man z.B. nur einige Spalten neu, bei statischen Problemen macht man nur einige Pivotoperationen.

Bem. : In der Praxis ist es wichtig, Branching durch Preprocessing zu unterstützen.

Netzwerke

Bsp.: Airline Crew Scheduling

gegeben: A Menge von Flughäfen

$\mathcal{F} = \{1, \dots, n\}$ Menge von Flügen (Legs)

$o(i) / d(i) \in A$ Start- / Zielflughafen von Flug $i \in \mathcal{F}$

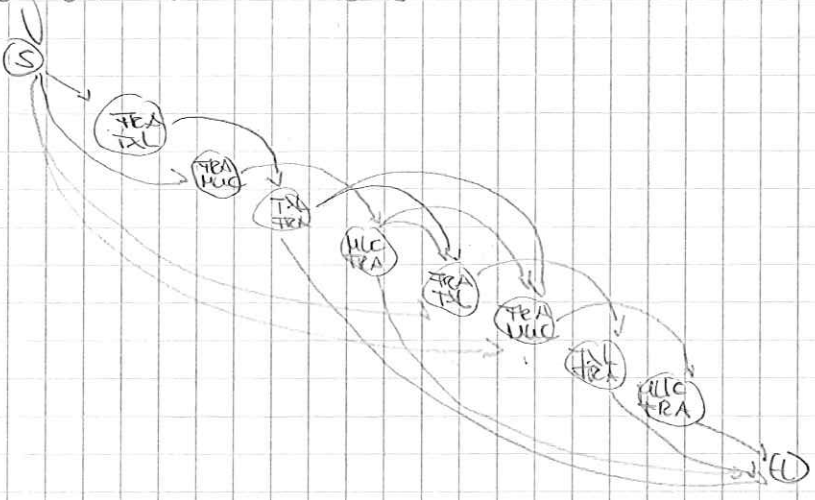
$s(i) / t(i)$ Abflug- / Ankunftszeit von Flug $i \in \mathcal{F}$

$\mathcal{R}$ Regeln zur Bildung von Flugdienten (Pairings)

ges.: Kostenminimale Menge von Pairings, die alle Flüge mind. einmal überdecken

Def.: $\mathcal{H} \in A$ Menge von Flughäfen, an denen ein Pairing beginnen kann (Homebase)

a) Day-to-Day-Netzwerk



$A = \{FRA, TXL, MUC\}$

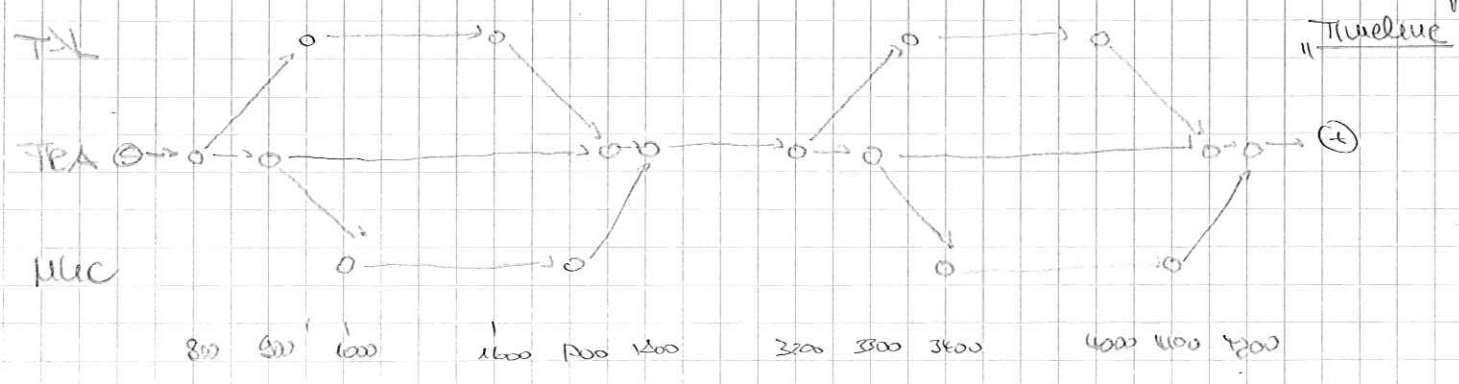
$\mathcal{H} = \{FRA\}$

$\mathcal{F} = \{1, \dots, 8\}$

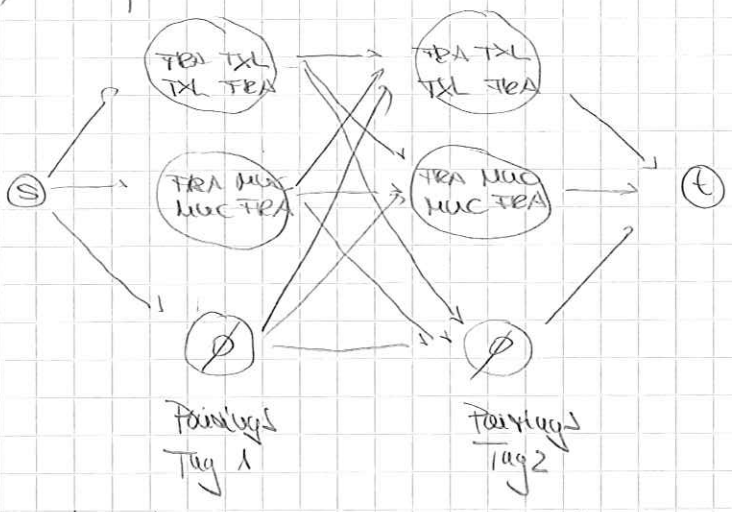
	o	d	s	t
1	FRA	TXL	800	950
2	FRA	MUC	900	1000
3	TXL	FRA	1600	1750
4	MUC	FRA	1500	1800
5	FRA	TXL	3200	3350
6	FRA	MUC	3300	3400
7	TXL	FRA	4000	4150
8	MUC	FRA	4100	4200

Viele zusätzliche Kanten!

b) Tag-zu-Tag-Netzwerk



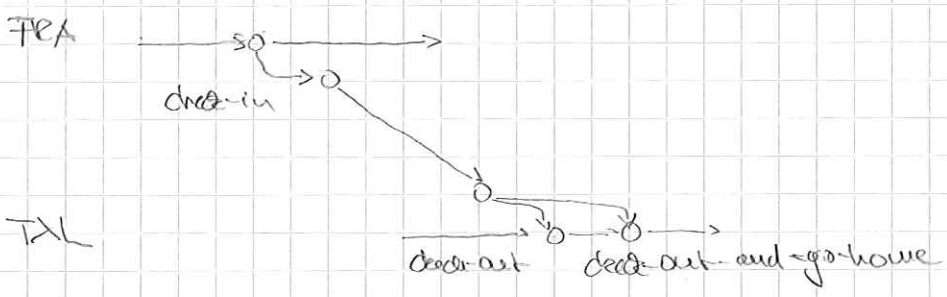
c) Time-Period-Network



Idee:	Tag-on-Node	Tag-on-Arc	Time-Period
$\|V\|$	$O(\|V\|\cdot\|T\|)$	$O(\|V\|\cdot\|T\|)$	$O(2^{\|T\|})$
$\|A\|$	$O(\|V\|\cdot\|T\|^2)$	$O(\|A\|\cdot\|T\|)$	$O((2^{\|T\|})^2)$
Vorteil	genaue Verfolgung über Folgetage	geringer Speicherplatzbedarf	Regeln zur Bildung von Fahrtagen häufig linear, sehr groß, dynamische Behandlung möglich
Nachteil	sehr viele Knoten, Kanten bei Umsetzung von Transportbedingungen	Verfolgung über Folgetage ungenau	

Exp. (Fortsetzung)

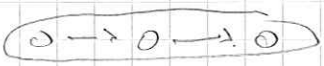
d) Expansionselemente



e) Idee: Flugtransport mit eigenen und Fremdflügen, Übernachtungen, ect. Periodizität

f) Techniken zur Behandlung großer Netzwerke:

Net-available (Ryan):  blocke nur 2-3 aussichtsreiche Nachfolger pro Knoten

Aggregation (Desrosiers et.al.):  temporäre Fixierung wiederauftretender Verbindungen

15. Linienplanung

15.1 Def. (Linienplanungsproblem):

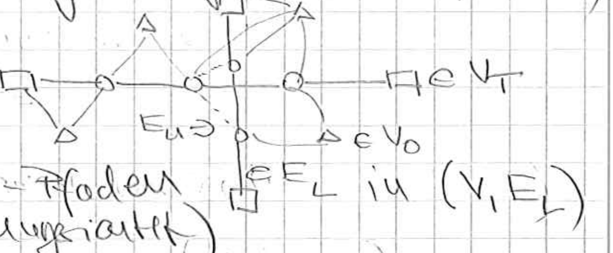
geg.: $N = (V, E)$, $E = E_L \dot\cup E_U$ — Infrastrukturnetz (ungerichtet)

$V_L \subseteq V$ — mögliche Endhaltestellen von Linien

$V_O \subseteq V$ — Quellen u. Ziele von Nachfrage (Origin-Destination-Knoten)

$E_L \subseteq E$ — Linienkanten

$E_U \subseteq E$ — Umsteigekanten



L — Menge von $(V_T \times V_T)$-Pfaden in (V, E_L) (mögliche Linien, ungerichtet)

$F \subseteq \mathbb{N}_0 / \mathbb{Q}_{\geq 0}$ — Menge von möglichen Linienfrequenzen

$\kappa_\ell \in \mathbb{Q}_{\geq 0}$ — Kapazität von Linie $\ell \in L$

$c_0, c_\ell \in \mathbb{Q}_{\geq 0}$ — Fixkosten und variable Kosten für Linie $\ell \in L$

$c_{f\ell} = c_0 + f_\ell c_\ell$ — Kosten für den Betrieb von Linie $\ell \in L$ mit Frequenz f_ℓ

$D \in \mathbb{Q}_{\geq 0}^{V_O \times V_O}$ — Quelle/Ziel-(Nachfrage-)Matrix (OD-Matrix)

$G = (V, A)$ — (Passagier-)Routingdigraph (gerichtet)

$A = A(E) \dot\cup \overline{A}(E) = \{a(e) := uv : e = uv \in E\} \dot\cup \{\overline{a}(e) := vu : e = uv \in E\}$, entsprechend $e(a) = e$, $\overline{a}(e) = e$, A_L usw.

P_{st} — Menge von st-(Passagier-)Routen in G, $d_{st} > 0$

$P := \dot\bigcup_{d_{st}>0} P_{st}$

$\tau_a \in \mathbb{Q}_{\geq 0}$ — Reisezeit auf Bogen $a \in A$

$\tau_p := \sum_{a \in p} \tau_a$ — " " st-Route $p \in P_{st}$

ges.: $x_p \in [0, d_{st}]$ — Passagierfluss auf st-Route $p \in P_{st}$

$\sum_{p \in P_{st}} x_p = d_{st} \;\forall st$ — Nachfragebedingungen (1)

$f_\ell \in F$ — Taktung / Linienplan

$L^* = \{\ell \in L : f_\ell > 0\}$

$\sum_{p \ni a} x_p \leq \sum_{\ell \ni e(a)} f_\ell \kappa_\ell \;\forall a \in A_L$ — Kapazitätsbedingungen (2)

$c_x := \sum_{p \in P} \tau_p x_p$ — Gesamtreisezeit

$c_f = \sum_{\ell \in L} c_{f\ell}$ — Gesamtkosten

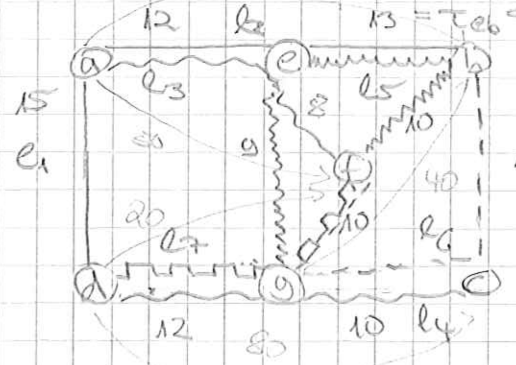
$\lambda c_f + (1-\lambda) c_x$ — Gesamtkosten

$\min \lambda c_f + (1-\lambda) c_x$ — Linienplanungsproblem

(f, x) erfüllt (1), (2)

f fix: … x: (1)(2) zulässiges Passagierrouting, zulässiger Linienplan

15.2 Bsp (Linienplanungsproblem (Karbstein [2012])).



st	d_{st}
af	50
ab	50
df	20
dc	80
gb	40

$V = V_L = V_0$, $L = \{l_1, \dots, l_7\}$

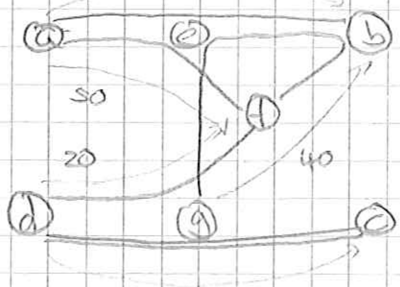
$A_T = \{vw : v \in V\}$

$\tau_a = 5 \quad \forall a \in A_T$

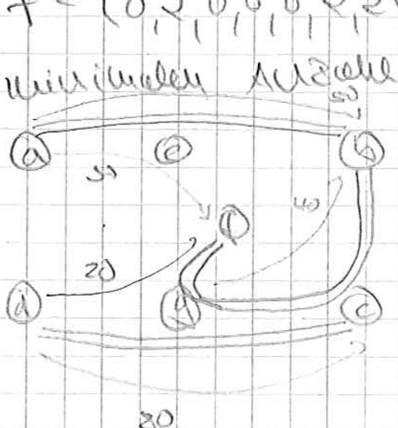
$F = \{0,1,2\}$, $\kappa = 50$

a) $f^0 = (2, \dots, 2)$ ist ein zulässiger Linienplan, denn (f^0, x^0) mit $x^0_{aef} = x^0_{aeb} = 50$, $x^0_{dgf} = 20$, $x^0_{dgc} = 80$, $x_{gfb} = 40$ ist zulässig.

b) $f^1_{50} = (0,1,1,2,1,0,1)$ ist ebenfalls ein zulässiger Linienplan, denn (f^1, x^0) erfüllt (2). x^0 realisiert minimale Reisezeiten und eine minimale Anzahl an Umstiegen.



c) $f^2 = (0,2,0,0,0,2,2)$ ist ein zulässiger Linienplan mit einer minimalen Anzahl an Linien. Beachte die Verwendung der Kapazität von Linie l_6 auf Kante gf in beiden Richtungen!



15.3 Def. (Steiner Connectivity Problem): Für

$V_0 =: T$ Terminals

$d_{st} = 1 \quad \forall st \in T \times T$

$\tau \equiv 0$

$\kappa \equiv \infty$

$F = \{0,1\}$

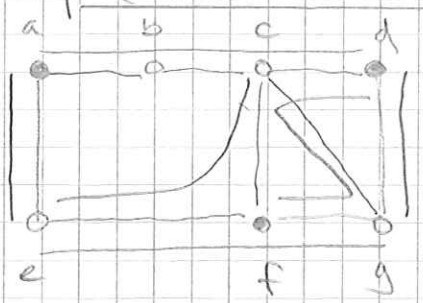
$c_0 = 0$

d.h. wenn wir Reisezeiten und Kapazitäten (und damit auch Frequenzen) ignorieren, müssen die Terminals kostenminimal untereinander verbunden werden. Diesen Fall nennt man Steiner Connectivity Problem (SCP), bei dessen weiter eingeschränktem Fall $L = A$ Steinerbaumproblem (STP).

Wir identifizieren in beiden Fällen Linienpläne $f \in \{0,1\}^L$ und supp $f \subseteq L$.

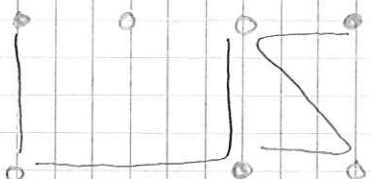
15.4 Bsp (Steiner Connectivity und Steinerbaumproblem (BKG [2013]):

a)



$V_0 = T = \{a, d, f\}$

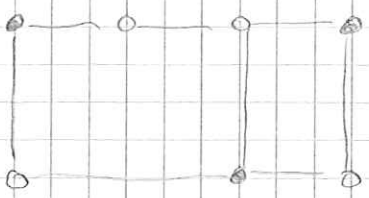
ℓ_1	abcd
ℓ_2	efg
ℓ_3	ae
ℓ_4	efc
ℓ_5	gd
ℓ_6	fgcd

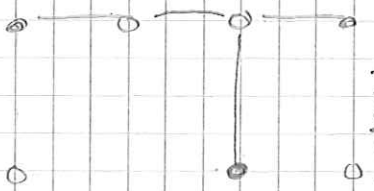


$\operatorname{supp} f = \{\ell_3, \ell_4, \ell_6\}$

Die Linien $(T\text{-Hadc})$ $\{\ell_3, \ell_4, \ell_6\}$ verbinden T.

b)





$\operatorname{supp} f = \{ab, bc, cd, cf\}$

Die Kanten $\{ab, bc, cd, cf\}$ verbinden T.

15.5 Def. (T-connecting Set, Steinerbaum):

a) Sei $G=(V,E)$, $T=V_0 \subseteq V$, L, $c_L \in \mathbb{Q}^L$ ein SCP

i) $L^* \subseteq L$ zulässig heißt T-connecting Set.

ii) $L \subseteq L$: $L \setminus L$ nicht zulässig heißt T-disconnecting Set.

iii) Sei $\emptyset \subsetneq W \subsetneq V$: $W \cap T \neq \emptyset$, $T \setminus W \neq \emptyset$

$S_L(W) = \{\ell \in L : \delta(W) \cap \ell \neq \emptyset\}$ T-Pfadschnitt

b) Sei $G=(V,E)$, $T=V_0 \subseteq V$, $L=E$ ein SPR

i) $E^* \subseteq E$ zulässig Steinerbaum

ii) $E \subseteq E$: $E \setminus E$ nicht zulässig Steiner-Schnitt (Steinercut)

15.6 Prop. (Connecting und Disconnecting Sets): Betrachte ein SCP (V, E, T, L, c):

a) Die minimalen T-disconnecting sets sind genau die minimalen T-Pfadschnitte.

b) Das SCP kann wie folgt als ganzzahliges Programm in vom Set-Covering-Typ formuliert werden:

$$\min \sum_{\ell \in L} c_\ell x_\ell$$

$$x(S_L(W)) \geq 1 \quad \forall \emptyset \neq T \cap W \neq T$$

$$x \in \{0,1\}^L$$

Bew.: Übung D

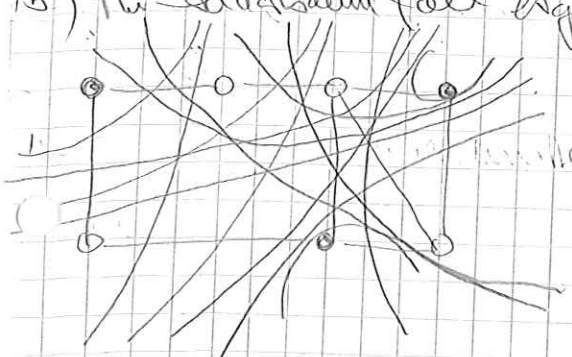
c) Für SPR kann wie folgt auch ganzz. Programm von Steinerschnitte $S(W)$ min $\sum_{\ell \in L} c_\ell x_\ell$ $x(S(W)) \geq 1$ $\forall$ $x \in \{0,1\}^L$.

15.4 Bsp. (Fortsetzung): a) Es ergeben sich folgende T-Überdeckungen:

L	U
$\{l_1, l_3\}$	$\{a\}$ oder $\{a,b\}$
$\{l_1, l_2, l_4\}$	$\{a,e\}$
$\{l_1, l_5, l_6\}$	$\{d\}$
$\{l_4, l_6, l_2\}$	$\{f\}$
$\{l_4, l_6, l_3, l_5\}$	$\{e,f,g\}$

$\min \sum c_e x_e$ mit $\begin{pmatrix} 1&0&1&0&0&0 \\ 1&1&0&1&0&0 \\ 1&0&0&0&1&1 \\ 0&1&0&1&0&1 \\ 0&0&1&1&1&1 \end{pmatrix} x \geq \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$, $x \in \{0,1\}^6$

b) Im Steinbaum-Fall ergeben sich folgende Steinerbäume:



15.7 Prop. (Komplexität):

a) Das SCP ist NP-hart, sogar für $T = V$.

b) Das STP ist NP-hart.

Bew.: a) Übung.

b) Folgt aus [Daus], □

15.8 Alg. (Approximationsalgorithmus für das SCP):

Vor.: $G = (V, E)$ zusammenhängend

L Linien

$c_L \in \mathbb{Q}_{\geq 0}^{L}$ Kosten

$T \subseteq V$ Terminals

A α-Approximationsalgorithmus für Steinerbaumprobleme

Beh.: T-überdeckende Menge L^*

1. Falls $e \subseteq E$ {
   $l \leftarrow \operatorname{argmin}\left\{ \frac{c_e}{|e \cap l|} \mid e \subseteq l \in L \right\}$
   $w_e \leftarrow \frac{c_e}{|e \cap l|}$
   $l_e \leftarrow l$
   }
2. $E^* \leftarrow A(G, T, c)$
3. $L^* \leftarrow L(E^*)$

15.9 Satz: Alg. 15.8 ist korrekt und liefert eine Lösung mit

$$c(L^*) \leq \alpha \cdot k \cdot c(L^{opt}),$$

wobei $L^{opt} = \text{argmin}$ SCP und $k = \max_{l \in L} |l|$.

Bew.: Sei $E^{opt} = \text{argmin}$ SteinerTree(G, T, ω). Dann gilt

$$c(L^{opt}) = \sum_{l \in L^{opt}} c_l = \sum_{l \in L^{opt}} \sum_{e \in l} \frac{c_e}{|l|} \geq \sum_{e \in E(L^{opt})} \omega_e = \omega(E^{opt})$$

($E(L^{opt})$: Steinerbaum)

$$\Rightarrow c(L^*) \leq \sum_{e \in E^*} c_{l_e} = \sum_{e \in E^*} \omega_e |l_e|$$

$$\leq \sum_{e \in E^*} k \omega_e \leq k \alpha \omega(E^{opt}) \leq k \alpha c(L^{opt}). \square$$

15.10 Alg. Minimum Spanning Tree Heuristik für Steinerbaumproblem (von Kou, Markowsky & Berman [1981]):

Input: $G = (V, E)$ zusammenhängender Graph

$c \in \mathbb{Q}_{\geq 0}^E$ Kantengewichte

$T \subseteq V$ Terminals

Output: Steinerbaum $Y \subseteq E$

1. $G^* \leftarrow (T, \binom{T}{2})$

   $p^*_{uv} \leftarrow$ argmin $c(p)$ $\forall u, v \in T$

   p_{uv} w.r.t. in G

   $c^*_{uv} \leftarrow c(p^*_{uv})$

2. $Y^* \leftarrow$ MST in (G^*, c^*)

   $H \leftarrow \bigcup_{e \in Y^*} E(p^*_e)$

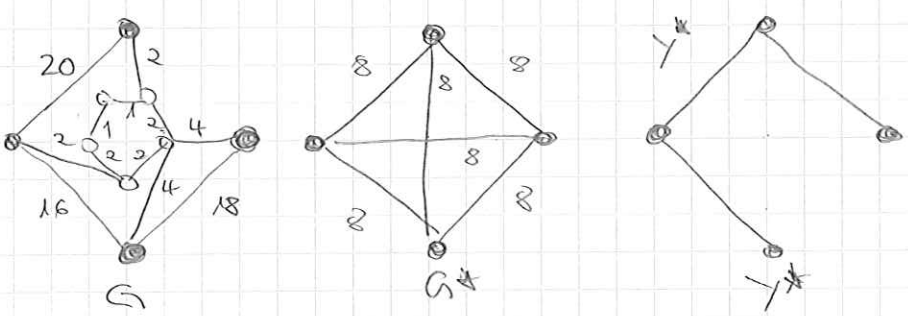
3. $Y \leftarrow$ MST in H

4. while ($\exists$ Blatt $v \in V(Y) \setminus T$) {

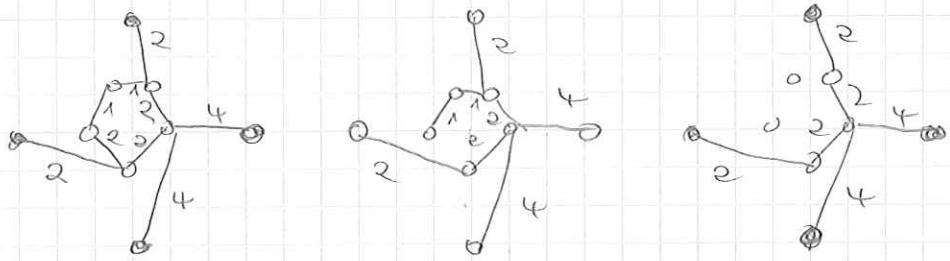
   lösche v aus Y

}

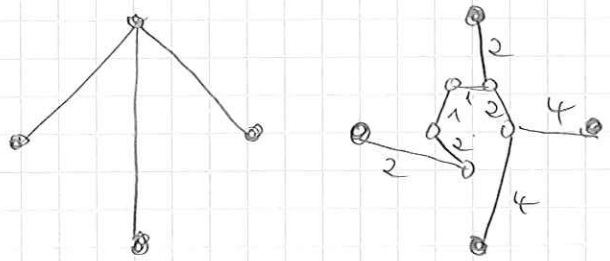
15.11 Bsp:



G $\quad$ G^*



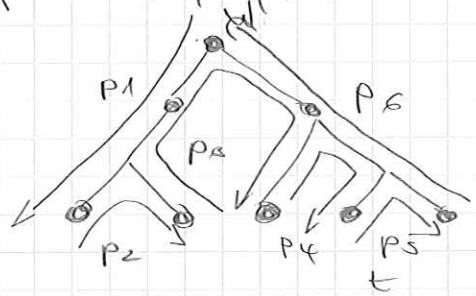
Alternativ:



15 R Satz: Sei (G, c, T) ein Steinbaumproblem. Sei t die Minimalanzahl an Blättern in einem optimalen Steinbaum (mit Gewicht c_{opt}). Dann gilt für den Output Y von Alg. 15.10:

$$c(Y) \leq 2 c_{opt} \frac{t-1}{t} \leq 2 c_{opt}.$$

Bew: Wähle irgendeinen Knoten $w \in T$ als Wurzel von Y^*. Wandere von w entgegend Y^* als sei eine Feuerkäfer entlang und untersuche Risse an den Blättern von Y^* in Pfade $p_1, \dots, p_t$:



O.B.d.A. sei $c(p_t) = \max_{i=1}^{t} c(p_i)$. Jeder entspricht jedem Pfad p_i mindestens eine $e_i \in E(G^*)$, $(e_1, \dots, e_t)$ ist ein Kreis in G^*, $(e_1, \dots, \hat{e}_i, \dots, e_t)$ ist ein Baum in G^* und $H = E(p_1) \cup \dots \cup E(p_i) \cup \dots \cup E(p_t)$ enthält einen Steinbaum Y. Deshalb gilt:

$$c(Y) \leq c(H) \leq \sum_{i=1}^{t} c(p_i) - c(p_t) \leq 2 c_{opt}\left(1 - \frac{1}{t}\right). \square$$

Alg. 15.13: Kürzeste-Reisezeiten-Algorithmus (J. Mehlhorn [2013]; Möllenberg [2014]).

Input: $N = (V, E)$ Infrastrukturnetz

L V-verbindende Menge von Pfaden in N (Linien)

$D = (V, A)$ Passagierrouting digraph

$A = A(E) \cup \bar{A}(E)$

$\tau_a = \tau_{e \in a} \geq 0$ Fahrzeit auf Bogen $a \in A$

$dist_e(v_i, v_j) = \sum_{k=i}^{j-1} \tau_{v_k v_{k+1}} \quad \forall v_i, v_j \in (v_1, \dots, v_{n_\ell}) = \ell \in L$

Fahrzeit von v_i nach v_j auf Linie ℓ

$\sigma \geq 0$ Umsteigezeit

$u(p) = (u_1, \dots, u_{k_p})(p)$ Ein-/Umstiege in p, d.h.

$p = (u_1, \dots, u_2, \dots, u_3, \dots, u_{k_p})$ für $\ell_1, \ell_2, \dots, \ell_{k_p - 1} \in L$, wobei $(u_1 \dots u_2) \in \ell_1$, $(u_2 \dots u_3) \in \ell_2$

$\sigma(p) = |u(p)| \sigma$ Anzahl Ein-/Umstiege auf p

$s, t \in V$ Start-/Zielknoten

Output: $p^* \in \operatorname{argmin}_{p \text{ st-Passagierpfad in } D} \tau(p) + \sigma(p)$

1. $d(s) \leftarrow 0$, $d(v) \leftarrow \infty \; \forall s \neq v \in V$, $p(v) \leftarrow nil \; \forall v \in V$
2. $Q \leftarrow \{s\}$ // markiere s
3. while $(Q \neq \emptyset)$ {
4. $\quad u \leftarrow \operatorname{argmin}_{v \in Q} d(v)$
5. $\quad$ forall $(u \in \ell \in L)$ { $\quad (*)$
6. $\quad\quad$ forall $(v \in \ell)$ {
7. $\quad\quad\quad$ if $(d(v) > d(u) + \sigma + dist_\ell(u, v))$ {
8. $\quad\quad\quad\quad d(v) \leftarrow d(u) + \sigma + dist_\ell(u, v)$
9. $\quad\quad\quad\quad p(v) \leftarrow u$
10. $\quad\quad\quad\quad Q \leftarrow v$
11. $\quad\quad\quad$ }
12. $\quad\quad$ }
13. $\quad$ }
14. $p^* = (t, p(t), p^2(t), \dots, s)$.

Satz 15.14: Alg 15.13 ist korrekt.

Bew.: Induktion über die Anzahl der Umfangsknoten $|\sigma(p^*)|$.

Ind.-Anf.: $|\sigma(p^*)| = 1$ Auf dem kürzesten st-Fluchtpfad wird nur einmal umgestiegen. Alg. 15.13 findet diesen Pfad beim Durchlaufen der entsprechenden Ecke in der Schleife (*)

Ind.-Vor.: Alg 15.13 findet kürzeste st-Fluchtpfade mit $|\sigma(p^*)| = k \geq 1$ Umstiegen

Ind.-Schritt: Vor. $|\sigma(p^*)| = k+1$, $u(p) = (u_1, \dots, u_{k+1})$

$p^* = (u_1, \dots, u_2, \dots, u_k, \dots, \underbrace{u_{k+1}, \dots, t}_{\ell_{k+1}})$

Nach Ind.-Vor. findet Alg 15.13 den kürzesten su_{k+1}-Fluchtpfad mit k Umstiegen, und dann beim nächsten Durchlauf von der Schleife (*) Ecke l_{k+1} und damit p^*. □