Fachbereich Mathematik
Prof. Dr. M. Joswig
Katja Kulas

TECHNISCHE UNIVERSITÄT DARMSTADT

Sommersemester 2007
30.05.2007

Diskrete Optimierung II

7. Übung mit Lösungshinweisen

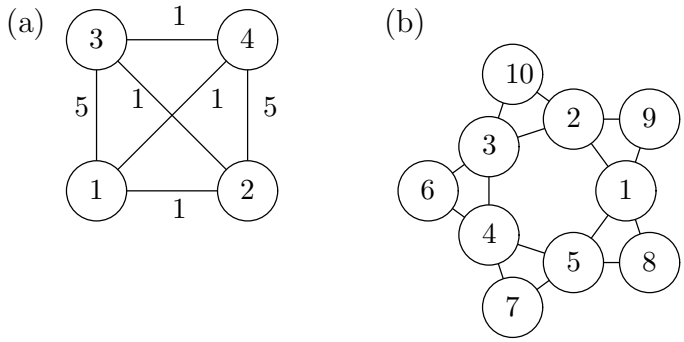
Gruppenübungen

Aufgabe G1

(i) Bestimmen Sie für die folgenden Graphen (z. B. mit dem Algorithmus von Kruskal) jeweils einen minimalen aufspannenden Baum, und ermitteln Sie dessen Durchmesser.

Hinweis: Der Durchmesser eines Baumes T ist definiert durch die Länge des längsten Weges zwischen zwei Knoten in T. Dabei ist mit der Länge eines Weges die Anzahl seiner Kanten gemeint.

(ii) Können Sie in einem (oder mehreren) der Fälle einen weiteren minimalen aufspannenden Baum angeben, welcher einen geringeren Durchmesser hat als der in Teil (i) bestimmte?



Alle Kantengewichte 1

LÖSUNG: (i) Wir betrachten den Graphen $G = (V, E)$. Gesucht ist eine Kantenmenge $T^* \subseteq E$, welche zusammenhängend und kreisfrei ist, mit minimalem Gewicht $w(T^*) = \min_{T \subseteq E} w(T)$, wobei $w(T) = \sum_{t \in T} w(e_t)$. Wir sortieren die Kanten aufsteigend nach ihren Gewichten: $E = (e_1, ..., e_m)$ mit $w(e_1) \leq w(e_2) \leq ... \leq w(e_n)$.

Beginnend mit $T^{(0)} = \emptyset$, führen wir für $i = 1, ..., m$ folgendes durch: Falls $G_i := (V, T^{(i-1)} \cup \{e_i\})$ kreisfrei ist, fügen wir die Kante e_i zu T hinzu: $T^{(i)} = T^{(i-1)} \cup \{e_i\}$; ansonsten setzen wir $T^{(i)} = T^{(i-1)}$. Dann ist $(V, T^{(n)})$ ein minimaler aufspannender Baum (zum Beweis dieser Aussage siehe z. B. *D. Jungnickel: Graphs, Networks and Algorithms; ACM Vol 5; Springer*).

Somit ist beispielsweise in Fall (a) ein minimaler aufspanndender Baum durch die Kantenmenge $T^* = \{\{1, 2\}, \{2, 3\}, \{3, 4\}\}$ gegeben. Der Durchmesser dieses Baumes beträgt $d = 3$, das Gewicht $w(T^*) = 3$.

Für Fall (b) ist jeder aufspannende Baum minimal, da alle Kantengewichte gleich sind und jeder aufspannende Baum die gleiche Kantenanzahl hat. Beispielsweise ist hier durch $T^* = \{\{1, 8\}, \{1, 9\}, \{2, 9\}, \{2, 10\}, \{3, 10\}, \{3, 6\}, \{4, 6\}, \{4, 7\}, \{5, 7\}\}$ ein minimaler aufspannender Baum mit Durchmesser $d = 9$ und Gewicht $w(T^*) = 9$ gegeben.

(ii) Für Fall (a) müssten wir einen aufspannenden Baum (V, T) finden mit $w(T) = 3$ und $d \leq 2$. Jeder aufspannende Baum mit Durchmesser $d = 2$ enthält allerdings eine Kante mit Gewicht 5, so dass das Gewicht eines solchen Baumes $w(T) = 7$ beträgt.

In Fall (b) können wir einen aufspannenden Baum mit Durchmesser $d < 9$ finden, z. B. (V, T) mit $T = \{\{1, 2\}, \{1, 8\}, \{2, 3\}, \{2, 9\}, \{3, 4\}, \{3, 10\}, \{4, 6\}, \{4, 7\}, \{5, 7\}\}$, welcher Durchmesser $d = 6$ besitzt.

Aufgabe G2

(i) Entwickeln Sie ein IP-Modell, welches zur Bestimmung eines minimalen aufspannenden Baumes eines ungerichteten Graphen $G = (V, E)$ mit Durchmesser von höchstens D (D wird vorgegeben) benutzt werden kann.

Hinweis: Betrachten Sie hierfür Flüsse zwischen allen Paaren von Knoten des Graphen.

(ii) Bestimmen Sie mit diesem IP-Modell und CPLEX jeweils eine optimale Lösung für die Graphen aus Aufgabe G1.

Wählen Sie für Beispiel (a) den maximalen Durchmesser $D = 2$, und rechnen Sie Beispiel (b) mit $D = 4$ und $D = 5$.

Eine Dokumentation für `CPLEX` finden Sie beispielsweise im Verzeichnis `/opt/cplex/cplex81/doc/userman/onlinedoc/`.

Falls Sie das Programm ZIMPL bereits kennen, können Sie das IP-Modell auch zunächst in ZIMPL umsetzen und anschließend mit CPLEX eine optimale Lösung bestimmen. Eine Dokumentation von ZIMPL finden Sie online unter http://www.zib.de/koch/zimpl/download/zimpl.pdf.

LÖSUNG: (i) Um einen minimalen aufspannenden Baum (V, T) mit beschränktem Durchmesser zu bestimmen, muss das Modell aus Aufgabe G1 insofern erweitert werden, als wir eine Möglichkeit zur Bestimmung von Weglängen innerhalb eines vorliegenden aufspannenden Baumes benötigen.

Dieses realisieren wir durch ein sogenanntes *Multi-Commodity-Flow (MCF)-Modell*. Hierbei werden Flüsse zwischen allen Paaren von Knoten des Graphen betrachtet.

Wir benutzen folgende Mengen:

- Graph: $G = (V, E)$
- Commodities: $K = \{\{v, w\} \mid v, w \in V\}$

Als Parameter sind die Kantengewichte $c(e)$ für $e \in E$ gegeben.

Wir führen folgende Variablen ein:

- Binärvariablen x_e für $e \in E$ mit $x_e = 1$, falls $e \in T$, 0, sonst
- Binärvariablen $y_{ij}^{v,w}$ für $\{i, j\} \in E$, $\{v, w\} \in K$ mit $y_{ij}^{v,w} = 1$, falls Kante $\{i, j\}$ auf dem (eindeutig bestimmten) Weg von v nach w in Richtung ij durchlaufen wird, 0, sonst.

Dann lautet die Zielfunktion

$$\min \sum_{e \in E} c(e) x_e.$$

Als Nebenbedingungen sind einzuhalten:

- T enthält genau $|V| - 1$ Kanten:

$$\sum_{e \in E} x_e = |V| - 1$$

- Flusserhaltungsbedingung für den Fluss $y^{v,w}$ für alle Knoten außer Startpunkt v und Endpunkt w; Ausfließen von 1 Flusseinheit aus v und Einfließen von 1 Flusseinheit in w:

$$\forall i \in V,\ \{v,w\} \in K: \sum_{j \in V} y_{ij}^{v,w} - \sum_{j \in V} y_{ji}^{v,w} = \begin{cases} 1, & \text{falls } i = v \\ 0, & \text{falls } i \notin \{v,w\} \\ -1, & \text{falls } i = w \end{cases}$$

- Eine Kante kann nur dann Fluss enthalten, wenn sie in T ist:

$$\forall \{i,j\} \in E,\ \{v,w\} \in K: y_{ij}^{v,w} + y_{ji}^{v,w} \leq x_e$$

- Der Durchmesser des aufspannenden Baumes ist durch D beschränkt:

$$\forall \{v,w\} \in K: \sum_{\{i,j\} \in E} (y_{ij}^{v,w} + y_{ji}^{v,w}) \leq D$$

- 0/1-Bedingungen:

$$\begin{aligned} \forall e \in E: x_e &\in \{0,1\} \\ \forall \{i,j\} \in E,\ \{v,w\} \in K: y_{ij}^{v,w} &\in \{0,1\} \end{aligned}$$

(ii) ZIMPL-Datei DMST.zpl für Graph (a):

```
# Max. Durchmesser
param D := 2;

# Anzahl Knoten
param n := 4;

# Knoten
set V := { 1 to n };

# Kanten
set E := { <1,2>,<1,3>,<1,4>,
           <2,3>,<2,4>,
           <3,4> };

# Kanten (mit Richtung)
set EY := { <1,2>,<1,3>,<1,4>,
           <2,1>,<2,3>,<2,4>,
           <3,1>,<3,2>,<3,4>,
           <4,1>,<4,2>,<4,3> };

# Commodities
set K := { <1,2>,<1,3>,<1,4>,
           <2,3>,<2,4>,
           <3,4> };

# Kosten
param C[E] :=
 <1,2> 1,
 <1,3> 5,
 <1,4> 1,
```

```
 <2,3> 1,
 <2,4> 5,
 <3,4> 1;

# Variablen
var x[E] binary;
var y[K*EY] binary;

# Zielfunktion
minimize cost: sum <i,j> in E do C[i,j]*x[i,j];

# Nebenbedingungen

# n-1 Kanten im Baum
subto numedges: sum <i,j> in E do x[i,j]  == n-1;

# Zusammenhangsbedingungen
subto connect1: forall <i,v,w> in V*K with i == v do
      sum <i,j> in EY do y[v,w,i,j] - sum <j,i> in EY do y[v,w,j,i] == 1;
subto connect2: forall <i,v,w> in V*K with i != v and i != w do
      sum <i,j> in EY do y[v,w,i,j] - sum <j,i> in EY do y[v,w,j,i] == 0;
subto connect3: forall <i,v,w> in V*K with i == w do
      sum <i,j> in EY do y[v,w,i,j] - sum <j,i> in EY do y[v,w,j,i] == -1;

# Kopplung x-y
subto xtoy: forall <v,w,i,j> in K*E do
      y[v,w,i,j] + y[v,w,j,i] <= x[i,j] ;

# Durchmesser
subto diameter: forall <v,w> in K do
      sum <i,j> in E do (y[v,w,i,j] + y[v,w,j,i]) <= D;
```

Der Aufruf `zimpl DMST.zpl` erzeugt u.a. die Datei `DMST.lp`, die Cplex lesen kann.

```
Minimize
 cost:  + x12 +5 x13 + x14 + x23 +5 x24 + x34

Subject to
 numedges_1:  + x34 + x24 + x23 + x14 + x13 + x12 = 3
 connect1_1:  - y1241 - y1231 - y1221 + y1214 + y1213 + y1212  = 1
 connect1_2:  - y1341 - y1331 - y1321 + y1314 + y1313 + y1312  = 1
 connect1_3:  - y1441 - y1431 - y1421 + y1414 + y1413 + y1412  = 1
 connect1_4:  - y2342 - y2332 - y2312 + y2321 + y2324 + y2323  = 1
 connect1_5:  - y2442 - y2432 - y2412 + y2421 + y2424 + y2423  = 1
 connect1_6:  - y3443 - y3423 - y3413 + y3431 + y3434 + y3432  = 1
 connect2_1:  - y2341 - y2331 - y2321 + y2314 + y2313 + y2312  = 0
 connect2_2:  - y2441 - y2431 - y2421 + y2414 + y2413 + y2412  = 0
 connect2_3:  - y3441 - y3431 - y3421 + y3414 + y3413 + y3412  = 0
 connect2_4:  - y1342 - y1332 - y1312 + y1321 + y1324 + y1323  = 0
 connect2_5:  - y1442 - y1432 - y1412 + y1421 + y1424 + y1423  = 0
 connect2_6:  - y3442 - y3432 - y3412 + y3421 + y3424 + y3423  = 0
 connect2_7:  - y1243 - y1223 - y1213 + y1231 + y1234 + y1232  = 0
 connect2_8:  - y1443 - y1423 - y1413 + y1431 + y1434 + y1432  = 0
 connect2_9:  - y2443 - y2423 - y2413 + y2431 + y2434 + y2432  = 0
 connect2_10:  - y1234 - y1224 - y1214 + y1241 + y1243 + y1242  = 0
 connect2_11:  - y1334 - y1324 - y1314 + y1341 + y1343 + y1342  = 0
 connect2_12:  - y2334 - y2324 - y2314 + y2341 + y2343 + y2342  = 0
 connect3_1:  - y1242 - y1232 - y1212 + y1221 + y1224 + y1223  = -1
 connect3_2:  - y1343 - y1323 - y1313 + y1331 + y1334 + y1332  = -1
 connect3_3:  - y2343 - y2323 - y2313 + y2331 + y2334 + y2332  = -1
 connect3_4:  - y1434 - y1424 - y1414 + y1441 + y1443 + y1442  = -1
 connect3_5:  - y2434 - y2424 - y2414 + y2441 + y2443 + y2442  = -1
 connect3_6:  - y3434 - y3424 - y3414 + y3441 + y3443 + y3442  = -1
 xtoy_1:  - x12 + y1221 + y1212 <= 0
 xtoy_2:  - x13 + y1231 + y1213 <= 0
 xtoy_3:  - x14 + y1241 + y1214 <= 0
 xtoy_4:  - x23 + y1232 + y1223 <= 0
 xtoy_5:  - x24 + y1242 + y1224 <= 0
 xtoy_6:  - x34 + y1243 + y1234 <= 0
 xtoy_7:  - x12 + y1321 + y1312 <= 0
 xtoy_8:  - x13 + y1331 + y1313 <= 0
 xtoy_9:  - x14 + y1341 + y1314 <= 0
 xtoy_10:  - x23 + y1332 + y1323 <= 0
 xtoy_11:  - x24 + y1342 + y1324 <= 0
 xtoy_12:  - x34 + y1343 + y1334 <= 0
 xtoy_13:  - x12 + y1421 + y1412 <= 0
 xtoy_14:  - x13 + y1431 + y1413 <= 0
 xtoy_15:  - x14 + y1441 + y1414 <= 0
 xtoy_16:  - x23 + y1432 + y1423 <= 0
 xtoy_17:  - x24 + y1442 + y1424 <= 0
 xtoy_18:  - x34 + y1443 + y1434 <= 0
 xtoy_19:  - x12 + y2321 + y2312 <= 0
 xtoy_20:  - x13 + y2331 + y2313 <= 0
```

```
 xtoy_21:  - x14 + y2341 + y2314 <= 0
 xtoy_22:  - x23 + y2332 + y2323 <= 0
 xtoy_23:  - x24 + y2342 + y2324 <= 0
 xtoy_24:  - x34 + y2343 + y2334 <= 0
 xtoy_25:  - x12 + y2421 + y2412 <= 0
 xtoy_26:  - x13 + y2431 + y2413 <= 0
 xtoy_27:  - x14 + y2441 + y2414 <= 0
 xtoy_28:  - x23 + y2432 + y2423 <= 0
 xtoy_29:  - x24 + y2442 + y2424 <= 0
 xtoy_30:  - x34 + y2443 + y2434 <= 0
 xtoy_31:  - x12 + y3421 + y3412 <= 0
 xtoy_32:  - x13 + y3431 + y3413 <= 0
 xtoy_33:  - x14 + y3441 + y3414 <= 0
 xtoy_34:  - x23 + y3432 + y3423 <= 0
 xtoy_35:  - x24 + y3442 + y3424 <= 0
 xtoy_36:  - x34 + y3443 + y3434 <= 0
 diameter_1:
  + y1243 + y1234 + y1242 + y1224 + y1232 + y1223
  + y1241 + y1214 + y1231 + y1213 + y1221 + y1212<= 2
 diameter_2:
  + y1343 + y1334 + y1342 + y1324 + y1332 + y1323
  + y1341 + y1314 + y1331 + y1313 + y1321 + y1312  <= 2
 diameter_3:
  + y1443 + y1434 + y1442 + y1424 + y1432 + y1423
  + y1441 + y1414 + y1431 + y1413 + y1421 + y1412  <= 2
 diameter_4:
  + y2343 + y2334 + y2342 + y2324 + y2332 + y2323
  + y2341 + y2314 + y2331 + y2313 + y2321 + y2312  <= 2
 diameter_5:
  + y2443 + y2434 + y2442 + y2424 + y2432 + y2423
  + y2441 + y2414 + y2431 + y2413 + y2421 + y2412  <= 2
 diameter_6:
  + y3443 + y3434 + y3442 + y3424 + y3432 + y3423
  + y3441 + y3414 + y3431 + y3413 + y3421 + y3412  <= 2
Binary
 x12 x13 x14 x23 x24 x34 y1212 y1213 y1214
 y1223 y1224 y1221 y1232 y1234 y1231 y1242 y1243
 y1241 y1312 y1313 y1314 y1323 y1324 y1321 y1332
 y1334 y1331 y1342 y1343 y1341 y1412 y1413 y1414
 y1423 y1424 y1421 y1432 y1434 y1431 y1442 y1443
 y1441 y2312 y2313 y2314 y2323 y2324 y2321 y2332
 y2334 y2331 y2342 y2343 y2341 y2412 y2413 y2414
 y2423 y2424 y2421 y2432 y2434 y2431 y2442 y2443
 y2441 y3412 y3413 y3414 y3423 y3424 y3421 y3432
 y3434 y3431 y3442 y3443 y3441
End
```

Der Aufruf `cplex` in der Shell öffnet CPLEX. Dort erhält man nach den Aufrufen `read DMST.lp, opt` und `dis sol var 1-` den optimalen Baum mit den Kanten $\{1,2\},\{2,3\},\{2,4\}$ mit Durchmesser 2 und Gewicht 7.

Hausübungen

Abgabe am 30.05.2007

Aufgabe H1 (9+2 Punkte)

In dieser Aufgabe betrachten wir folgendes kombinatorisches Optimierungsproblem

CARDINALITY–BIPARTITE–MATCHING–Problem	
Instanz:	Graph $G = (V, E)$ mit $X, Y \subset V$, so dass $V = X \cup Y$, $X \cap Y = \emptyset$ und $E = \{(x, y) : x \in X, y \in Y\}$.
Aufgabe:	Finde eine Kantenmenge $M \subseteq E$, so dass keine zwei Kanten aus M einen gemeinsamen Endknoten haben und $\|M\|$ maximal ist.

Eine Studentin ist eine Woche zu Besuch in Berlin, um sich in dieser Zeit auf der Berlinale folgende Filme anzuschauen, die jedoch nur an den ausgewiesenen Tagen vorgestellt werden. Sie möchte so viele Filme wie möglich, allerdings maximal einen pro Tag sehen.

Film	**Vorstellungstage**
Skagafjördur	Sonnabend, Montag
The Garden	Montag, Sonnabend, Sonntag
Forty Shades of Blue	Montag, Dienstag, Mittwoch
Gender X	Montag, Freitag
Abordage	Mittwoch, Donnerstag
Wer ist Helene Schwarz?	Dienstag
Heaven‘s Gate	Sonntag, Freitag

(i) Stellen Sie dieses MATCHING–Problem als bipartiten Graphen dar.

(ii) Formulieren Sie

- das ganzzahlige lineare Programm (ILP),
- das relaxierte lineare Programm (ohne die Ganzzahligkeitsbedingungen) (LP),
- das duale Programm zu (LP)

sowohl allgemein als auch für die obige Instanz.

(iii) Berechnen Sie z. B. mithilfe von `CPLEX` die Optima für die unter (ii) formulierten Programme.

(iv) Visualisieren Sie mithilfe von `polymake` die Matching–Polytope für zusammenhängende bipartite Graphen mit genau drei Kanten.

(v) **Zusatzaufgabe:** Wie läßt sich das duale Problem graphentheoretisch interpretieren?

Aufgabe H2 (5 Punkte)

Gegeben Sei folgendes Optimierungsproblem:

EQUALITY–KNAPSACK–Problem	
Instanz:	$a_i, w_i \in \mathbb{R}_+$ mit $i = 1, \ldots, n$ ($n \in \mathbb{N}$), $\lambda \in \mathbb{R}_+$.
Aufgabe:	Finde $x \in \mathbb{N}^n$, so dass $\sum_{i=1}^{n} a_i x_i$ maximal ist und $\sum_{i=1}^{n} w_i x_i = \lambda$ gilt.

(i) Berechnen Sie mithilfe von `CPLEX` ein Optimum für die Instanz I mit $a = (213, -1928, -11111, -2345, 9123)$, $w = (12223, 12224, 36674, 611, 85569)$ und $\lambda = 89643482$.

(ii) Betrachten Sie nun das entsprechende relaxierte UNBOUNDED–KNAPSACK–Problem, d. h. $\sum_{i=1}^{n} w_i x_i \leq \lambda$ und $x_i \in \mathbb{R}_+ \quad \forall i = 1, \ldots, n$. Stellen Sie für I das resultierende KNAPSACK–Polytop mithilfe von `polymake` graphisch dar. Nutzen Sie dazu die Funktionen `facet, proj` und `polymake VISUAL`, wobei `facet` eine gegebene Facette eines Polyeders als neues Polyeder speichert und `proj` eine orthogonale Projektion eines Polyeders liefert.

(iii) Berechnen Sie mithilfe von `polymake MAXIMAL_VALUE` ein Optimum für das relaxierte Problem.