Lp – Lineáris programozás

Library for Efficient Models and Optimization in Networks

Balázs Dezső

ELTE, Operációkutatási tanszék

Lineáris program(LP)

A lineáris programozási feladatban adott $A : I \times J \to \mathrm{R}$ mátrix, $b : I \to \mathrm{R}$ és $c : J \to \mathrm{R}$ vektor. Az $I^{\leq}$ és $I^{=}$ az I, és $J = J^{\geq}$ és $J^{=}$ a J diszjunkt felbontása.

$$\begin{aligned} \sum_{j \in J} A_{ij} x_j &\leq b_i \qquad i \in I^{\leq} \\ \sum_{j \in J} A_{ij} x_j &= b_i \qquad i \in I^{=} \\ x_j &\geq 0 \qquad j \in J^{\geq} \\ \max & \sum_{j \in J} c_j x_j \end{aligned}$$

Megoldások halmaza P, az optimális megoldás x^*, az optimális megoldás érték opt_x.

Lineáris program duális

$$\begin{aligned}
\sum_{i \in I} A_{ij} y_i &\geq c_j & j \in J^{\leq} \\
\sum_{i \in I} A_{ij} y_i &= c_j & j \in J^{=} \\
0 &\leq y_i & i \in I^{\leq} \\
\min & \sum_{i \in I} b_i y_i
\end{aligned}$$

Megoldások halmaza D, az optimális megoldás y^*, az optimális megoldás érték opt_y.

Dualitás tétel

A dualitás tétel alapján a primál és a duál feladatra a következő esetek közül az egyik érvényes:

- $opt_x = opt_y \in \mathrm{R}$
- $P = \emptyset$ és $opt_y = -\infty$
- $D = \emptyset$ és $opt_x = \infty$
- $P = \emptyset$ és $P = \emptyset$

Kiegészítő többlet (complementary slackness) tétele, létezik olyan primál-duál megoldás pár:

- $\forall i \in I^{\leq}$-re $b_i - \sum_{j \in J} A_{ij} x_j$ és y_i közül csak az egyik lehet nem nulla.
- $\forall j \in J^{\geq}$-re $\sum_{i \in I} A_{ij} y_i - c_j$ és x_j közül csak az egyik lehet nem nulla.

Első példa

Adott egy gyár, ami alkatrészekből állít elő termékeket, majd azokat adott áron eladja. A modellben feltételezzük, hogy a gyárnak adott alkatrész készlete, amiből gyártani szeretne, de nem szeretne újabb alkatrészt venni. Feltételezzük, hogy a gyár minden terméket el tud adni, és gyártás mennyisége nincs hatással az eladási árra. Továbbá feltételezzük, hogy az alkatrész és a termék is szabadon felosztható, tört mennyiségeket is el lehet készíteni, eladni.

Első példa . . .

A feladatban az i. alkatrészből a j. termék előállításához pontosan A_{ij} mennyiség kell. Az i. alkatrészből pontosan b_i áll rendelkezésre, míg a j. árut pontosan c_j áron adjuk el. A j. termékből x_j mennyiséget termeljük, ezáltal a profit maximalizálás feladata a következő módon írható fel:

$$\sum_{j=1}^{m} A_{ij} x_j \quad \leq \quad b_i \qquad \forall 1 \leq i \leq n$$

$$\max \quad \sum_{j=1}^{m} c_j x_j$$

Első példa . . .

$$\sum_{i=1}^{m} A_{ij} y_i \quad \geq \quad c_i \qquad \forall 1 \leq j \leq m$$

$$\min \quad \sum_{i=1}^{m} b_i y_i$$

A feladat úgy értemlezhető, hogy rendeljünk az alkatrészekhez árat, hogy az a következő értelemben reális legyen. Egy terméket alkatrész áron ne értékeljünk kevesebbre, mint az eladási ára, és minimálisra értékeljük a gyár vagyonát. Mivel a dualitás tétel teljesül, ezért így a gyár vagyonát ugyanannyira értékeljük, mint a megszerezhető profitot.

Második példa

A két személyes zéróösszegű játékok egy egyszerű modellt nyújtanak sok típusú problémára. A két szerplő választhat egymástól függetlenül alternatívák közül. A két szereplő választásától függően az egyik valamennyire jól jár míg a másik ugyanennyire jár rosszul. A feladatot egy $M \in \mathrm{R}^{n \times m}$ mátrixszal szokták jellemezni, az A játékos választ egy i sort, B egy j oszlopot, és ezáltal A nyer M_{ij} értéket, míg B veszít ugyanennyit.

Második példa . . .

Első lépésként nézzük meg, hogy hogyan határozhatjuk meg az optimális stratégiát. Vegyük B kevert stratégiáját, ezt $x_1, x_2, \ldots, x_m$ változókkal írhatjuk le, ahol x_j az j. oszlophoz tartozó valószínűség, természetesen a valószínűségek összege egy, $\sum_{j=1}^{m} x_j = 1$. Ha A ismerné B kevert stratégiáját, akkor könnyen tudna stratégiát választani, ugyanis az i. sorban várható nyereménye $\sum_{j=1}^{m} M_{ij} x_j$, ezáltal azon sorokhoz, ahol a várható érték nem maximális, nulla valószínűséget rendelve, a lehető legjobb döntést hozta. Nyilván B-nek az a célja, hogy A minél kevesebbet tudjon nyerni, így A nyereményének a sorok szerint várható maximumát szeretné minimalizálni.

Második példa . . .

A feladat formalizálásához egy segédváltozót vezetünk be:

$$\begin{aligned}
\sum_{j=1}^{m} M_{ij} x_j - z &\leq 0 \qquad \forall 1 \leq i \leq n \\
\sum_{j=1}^{m} x_j &= 1 \\
x_j &\geq 0 \qquad \forall 1 \leq j \leq n \\
\max \quad & -z
\end{aligned}$$

A feladat egyik legszebb tulajdonsága az, hogy ha A-stratégiájára felírjuk a lineáris programozási feladatot, akkor -1-gyel való szorzás után pontosan megkapjuk a B stratégiájához felírt duális feladatot.

Harmadik példa

A legrövidebb utak problémája az egyik legismertebb gráfelméleti feladat. Legyen adott egy $G(V, A)$ gráf, egy s kezdő csúcs, egy t cél csúcs és egy $A \to \mathrm{R}$ költségfüggvény az éleken. A feladat több polinomiális algoritmus ismert, nem negatív élsúlyok esetén a Dijkstra alkoritmus, vagy általános esetben a Bellman-Ford algoritmus, de most megmutatjuk, hogy hogyan lehet formalizálni a feladatot lineáris programozás segítségével.

Harmadik példa . . .

Minden élhez rendeljünk egy f_a változót, ami 0 vagy 1 értéket vehet fel, attól függően, hogy az úton van-e vagy nincs. A következő állítást fogalmazzuk meg a program segítségével: az összes csúcsból, kivéve s-et és t-t ugyanannyi út él jön ki, mint amennyi bemegy. Ugyanez az érték t-ben plusz egy:

$$\sum_{a \in \delta^+(u)} f_a - \sum_{a \in \delta^-(u)} f_a = 0 \qquad u \in V \setminus s, t$$

$$\sum_{a \in \delta^+(t)} f_a - \sum_{a \in \delta^-(t)} f_a = 1$$

$$0 \leq f_a$$

$$\min \sum_{a \in A} c_a f_a$$

Harmadik példa . . .

Tekintsük a feladat duálisát:

$$\begin{aligned} \pi_v - \pi_u &\leq c_{uv} \qquad uv \in A, u \neq s \\ \pi_v &\geq c_{sv} \qquad sv \in A \\ \max \pi_t, & \end{aligned}$$

vagy kisebb átfogalmazással:

$$\begin{aligned} \pi_v - \pi_u &\leq c_{uv} \qquad uv \in A \\ \pi_s &= 0 \\ \max \pi_t & \end{aligned}$$

A π_u lényegében egy alsó becslést ad minden u csúcs távolságára, míg a célfüggvény miatt a legnagyobb ilyen alsó becslést fogjuk megkapni az a t csúcsra. A komplementáris többlet tétele alapján, csak azokat az éleket választhatjuk be az útba, amire feszes a π függvény, azaz $\pi_v - \pi_u = c_{uv}$.

Negyedik példa

A maximális folyam probléma informálisan a következő: Van egy csőhálózat egyirányú(ezt nehéz elképzelni) csövekkel, és ezek a csövek csomópontokban találkoznak. Minden csőnek van egy kapacitása, ami azt mondja meg, hogy adott időegység alatt mennyi vizet tudunk átjuttatni rajta. És adott egy forrás és egy cél csúcs. Mennyi vizet tudunk összesen átjuttani a rendszeren, és az egyes csöveken mekkora folyam folyik át?

Negyedik példa . . .

Formalizáljuk a feladatot gráfelmélet segítségével. Legyen $G = (V, A)$ gráf, a csomópontok felelnek meg a gráf csúcsainak, a csövek pedig az éleknek. A csövek kapacitását adja meg a $c : A \to \mathrm{R}$ függvényt, és s a forrást, t a cél csúcsot. Az éleken áthaladó folyam, $f : A \to \mathrm{R}$ lesz a lineáris progrmozási feladat változói:

$$\sum_{a \in \delta^-(u)} f_a - \sum_{a \in \delta^+(u)} f_a = 0 \qquad u \in V \setminus s, t$$

$$0 \leq f_a \leq c_a \qquad a \in A$$

$$\max - \sum_{a \in \delta^-(t)} f_a + \sum_{a \in \delta^+(t)} f_a$$

Negyedik példa . . .

Most vegyük a duális feladatot:

$$\begin{aligned} \pi_u - \pi_v + y_{uv} &\geq 0 \qquad \forall uv \in A, uv \notin \{s,t\} \\ \pi_u + y_{ut} &\geq 1 \qquad \forall ut \in A \\ -\pi_v + y_{tv} &\geq -1 \qquad \forall tv \in A \\ y_{uv} &\geq 0 \qquad \forall uv \in A \\ \min &\sum_{a \in A} c_a y_a \end{aligned}$$

Szintén egyszerűsíthetjük kissé a felírást:

$$\begin{aligned} \pi_u - \pi_v + y_{uv} &\geq 0 \qquad \forall uv \in A \\ \pi_s &= 0 \\ \min &\sum_{a \in A} c_a y_a \end{aligned}$$

Negyedik példa . . .

A duális a minimális vágás problémáját fogalmazza a gráfban, avagy bontsuk X és és $V \setminus X$ részre a csúcshalmazt, hogy $s \in X$ és $t \in V \setminus X$, és az X és $V \setminus X$ között menő élek összsúlya minimális. A dualitás tétel alapján bizonyítható a maximális folyam–minimális vágás tétele, míg a komplementáris többlet alapján kapjuk, hogy a vágáson átmenő élek telítettek.

Az `lgf` file

```
@nodeset
coords          label
(-260,-77)      1
(-29,-146)      2
(434,99)        8
@edgeset
                        capacity        label
1               3       7               1
1               2       5               2
1               6       4               3
2               8       7               15
@nodes
source 1
target 8
@end
```

Gráf betöltése `lgf`-ből

```
#include <lemon/graph_reader.h>

ListGraph graph;
ListGraph::EdgeMap<double> cap(graph);
ListGraph::NodeMap<dim2::Point<double> > coords(g
ListGraph::Node s, t;
GraphReader<ListGraph> gr("file.lgf", graph);
gr.readEdgeMap("capacity", cap);
gr.readNodeMap("coords", coords);
gr.readNode("source", s);
gr.readNode("target", t);
gr.run();
```

Gráf kirajzolása

```
#include <lemon/graph_to_eps.h>

graphToEps(graph, "file.eps").
  coods(coords).run();
```

Sok paraméter megváltoztatható:

- nodeScale()
- nodeShape()
- edgeWidths()
- edgeWidthScale()
- drawArrows()
- arrowWidth()
- arrowLength()
- ...

Feladatok

- Minimális költségű s, t út
- Minimális szélességű s, t út
- Maximális s, t folyam
- Maximális s, t folyam minimális megtett úttal

Variánsok

- Felírás primál feladat alapján
- Felírás duál feladat alapján