• Seminar za teorijsko računarstvo •
Poliedarske tehnike u kombinatornoj optimizaciji i cutting-planes algoritmi

Matko Botinčan

11. svibnja 2004.

1 Sadržaj seminara

- Matematičke osnove
- Primjer
- Cutting-planes algoritmi
- Osnovno o implementaciji branch-and-cut algoritama
- Postojeći softver

2 Matematičke osnove

Definicija 2.1 *Skup $P \subseteq \mathbb{R}^n$ naziva se poliedar ako vrijedi:*

$$P = \{x \in \mathbb{R}^n \mid Ax \le b\},$$

za neku matricu A i vektor b, tj. ukoliko je P presjek konačno mnogo afinih polu-prostora.
Ograničeni poliedar zovemo politopom.

Definicija 2.2 *Konveksna ljuska skupa X (u oznaci* $\mathrm{conv}(X)$*) je najmanji konveksni skup koji sadrži skup X i dan je s:*

$$\begin{aligned}\mathrm{conv}(X) &= \{\lambda_1 x_1 + \ldots + \lambda_t x_t \mid t \ge 1, x_1, \ldots x_t \in X, \lambda_1, \ldots, \lambda_t \ge 0,\\ &\quad \lambda_1 + \ldots + \lambda_t = 1\}\end{aligned}$$

za neku matricu A i vektor b, tj. $\mathrm{conv}(X)$ *je presjek konačno mnogo afinih polu-prostora.*

Definicija 2.3 *Dimenzija poliedra P (u oznaci* $\dim(P)$*) je dimenzija najmanjeg afinog prostora koji ga sadržava.*

Definicija 2.4 *Za nejednakost $\pi^\tau x \leq \pi_0$ kažemo da je valjana za poliedar P ukoliko svaka točka iz P zadovoljava nejednakost.*
Skup $F = \{x \in P \mid \pi^\tau x = \pi_0\}$ naziva se stranicom od P, a za valjanu nejednakost $\pi^\tau x \leq \pi_0$ kažemo da definira stranicu od F.
Stranica je prava ukoliko je $\emptyset \neq F \neq P$, tj. $0 < \dim(F) < \dim(P)$.
Ukoliko je $\dim(F) = \dim(P) - 1$ (tj. F je maksimalna), tada F nazivamo hiperstranom (engl. facet*).*

Napomena 2.5 *Zanimat će nas nejednakosti koje definiraju hiperstrane poliedra, tj.* facet-defining *nejednakosti, budući su to upravo one nejednakosti koje su nužne za opis konveksne ljuske.*

Definicija 2.6 *Za poliedar P definiramo cjelobrojnu ljusku P_I od P sa:*

$$P_I = \text{conv}\left\{x \in \mathbb{Z}^n \mid x \in P\right\}.$$

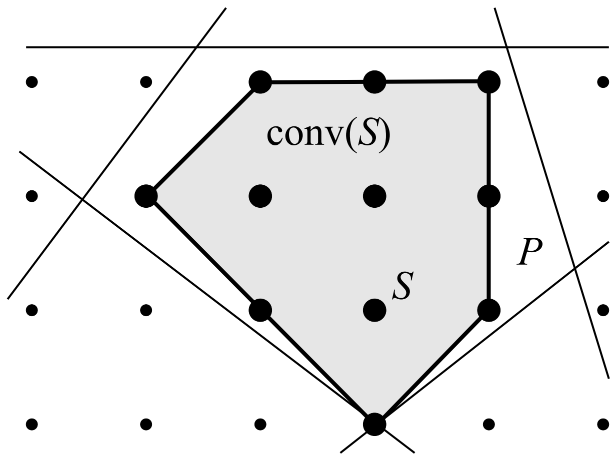
Teorem 2.7 ([14]) *Za svaku racionalnu matricu A postoji cjelobrojna matrica M takva da za svaki vektor b postoji vektor d tako da vrijedi:*

$$\left\{x | Ax \leq b\right\}_I = \left\{x \mid Mx \leq d\right\}.$$

Nažalost, ne postoji ograničenje na broj redaka u matrici M.

Teorem 2.8 ([14]) *Ne postoji polinom φ takav da za svaki racionalni poliedar $P = \{x \mid Ax \leq b\}$ cjelobrojna ljuska P_I od P ima najviše $\varphi(\text{size}(A, b))$ hiperstrana.*

Definicija 2.9 *P' se naziva relaksacija od P ako je P' poliedar i $P' \supseteq P$.*



Slika 1: Primjer poliedra

Definicija 2.10 *Problem cjelobrojnog linearnog programiranja* (engl. Integer linear programming problem) *definiran je s:*

$$\text{(ILP)} \qquad \min\{c^\tau x \mid x \in S\},$$

gdje je $S = P \cap \mathbb{Z}^n$, *a* $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$.

Definicija 2.11 *Linearna relaksacija problema (ILP) je problem definiran s:*

$$\text{(LP)} \qquad \min\{c^\tau x \mid x \in P\},$$

gdje je $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$.

Definicija 2.12 Facet-defining *nejednakosti su one nejednakosti koje su nužne za opis* $\operatorname{conv}(S)$, *tj. u nekom smislu to su "najjače moguće" nejednakosti.*

Napomena 2.13 *Ukoliko imamo poznat egzaktan opis* $\operatorname{conv}(S)$ *možemo jednostavno riješiti problem linearnog programiranja* $\min\{c^\tau x \mid x \in \operatorname{conv}(S)\}$ *i na taj način dobiti i rješenje polaznog problema (ILP).*

Napomena 2.14 *Za bilo koji ograničeni poliedarski skup moguće je algoritamski pronaći* $\operatorname{conv}(S)$ *u konačnom broju koraka, no općenito ne postoji gornja ograda na broj koraka (obzirom na dimenziju skupa* S*).*

3 Primjer

Veliki proboj u metodama za egzaktno rješavanje problema trgovačkog putnika zbio se 1954. godine kada su George Dantzig, Ray Fulkerson i Selmer Johnson objavili rad s opisom metode za rješavanje TSP-a ilustrirajući snagu predložene metode egzaktnim rješavanjem instance problema s 49 gradova (što je za to doba bilo impresivno). Njihova metoda u osnovi je predstavljala ono što danas nazivamo *cutting-plane* algoritmom.

Definicija 3.1 (TSP) *Problem trgovačkog putnika: u danom potpunom neusmjerenom grafu* $G = (V, E)$ *u kojem je svakom bridu* $e \in E$ *pridružena cijena* c_e *treba pronaći Hamiltonov ciklus mimimalne cijene.*

Definicija 3.2 (IP formulacija TSP-a)

$$\begin{cases} \min & \sum_{e\in E} c_e x_e \\ s.t. & \sum_{e\in\delta(v)} x_e = 2, \quad \forall v \in V \qquad\qquad (\triangle) \\ & \sum_{e\in\delta(U)} x_e \geq 2, \quad \forall U \subseteq V \;\; (U \neq \emptyset) \quad (\square) \\ & x \in \{0,1\}, \end{cases}$$

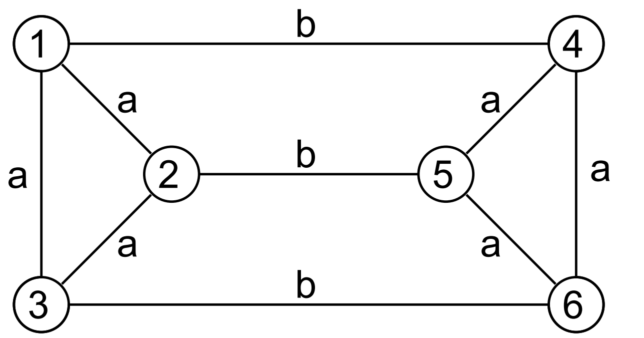
gdje $\delta(v)$ *označava skup svih bridova incidentnih s vrhom* v, *odnosno* $\delta(U)$ *skup svih bridova s točno jednim krajem u* U.

Napomena 3.3 *Nejednakosti* ($\triangle$) *nazivaju se* degree constraints*, a nejednakosti* ($\square$) *nazivaju se* subtour elimination constraints.
Problem s ograničenjima ($\square$) *je u tome što ih ima eksponencijalno mnogo (obzirom na broj vrhova), a za dobivanje rješenja TSP problema potrebno ih je samo manji broj.*

Definicija 3.4 (LP relaksacija TSP-a)

$$\begin{cases} \min & \sum_{e \in E} c_e x_e \\ s.t. & \sum_{e \in \delta(v)} x_e = 2, \quad \forall v \in V \\ & \sum_{e \in \delta(U)} x_e \geq 2, \quad \forall U \subseteq V \ (U \neq \emptyset) \\ & 0 \leq x \leq 1 \end{cases}$$

Napomena 3.5 *Gotovo uvijek se u LP relaksaciji TSP-a izbacuje i ograničenje* ($\square$) *radi redukcije veličine LP problema.*



Slika 2: Primjer TSP-a

Pogledajmo primjer na slici (2), pri čemu neka je $a = 1$, $b = 2$, a svi neprikazani bridovi neka imaju duljinu 10. Vektor određen s

$$x_{12} = x_{23} = x_{13} = x_{45} = x_{46} = x_{56} = 0.5$$
$$x_{14} = x_{25} = x_{36} = 1$$

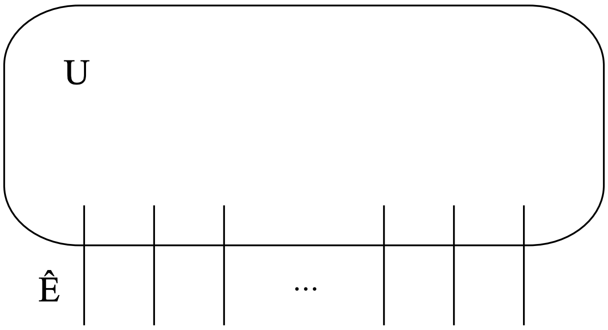
očito zadovoljava sve nejednakosti u LP relaksaciji TSP-a, no ne predstavlja incidencijski vektor nekog Hamiltonovog ciklusa.

Označimo s U skup vrhova koji čine lijevi trokut, tj. $U = \{1, 2, 3\}$. Neka je $E(U)$ skup svih bridova s oba kraja u U, dakle $E(U) = \{\{1,2\}, \{2,3\}, \{1,3\}\}$, te neka je $\hat{E} = \{\{1,4\}, \{2,5\}, \{3,6\}\}$, tj. svaki brid iz $\hat{E}$ ima točno jedan kraj u U.
Očito, iz skupa bridova $E(U) \cup \hat{E}$ najviše četvero može tvoriti bridove nekog Hamiltonovog ciklusa, jer inače će biti narušena nejednakost ($\triangle$), što znači:

$$x_{12} + x_{23} + x_{13} + x_{14} + x_{25} + x_{36} \leq 4,$$

odnosno:

$$\sum_{e \in E(U)} x_e + \sum_{e \in \hat{E}} x_e \leq 4 = |U| + \frac{|\hat{E}| - 1}{2}$$



Slika 3: *2-matching* nejednakosti

Teorem 3.6 (*2-matching* nejednakosti) *Neka je $U \subseteq V$ podskup skupa vrhova, te $\hat{E} \subseteq E$ neparan skup disjunktnih bridova tako da svaki ima točno jedan kraj u U. Označimo li s $E(U)$ skup bridova koji imaju oba svoja kraja u U, vrijedi slijedeća nejednakost:*

$$\sum_{e \in E(U)} x_e + \sum_{e \in \hat{E}} x_e \leq |U| + \frac{|\hat{E}| - 1}{2}$$

4 Cutting-planes algoritmi

- Cutting-planes algoritmi "opće namjene":
  - Gomoryjev cutting-plane algoritam
  - Chvátalova procedura zaokruživanja
  - Schrijverova procedura zaokruživanja
- Cutting-planes algoritmi bazirani na poliedarskim nejednakostima

4.1 Gomoryjev cutting-plane algoritam

Napomena 4.1 *Gomoryjev algoritam pretpostavlja da su u definiciji (ILP), a onda i u definiciji linearne relaksacije (LP) matrica A i vektor b cjelobrojni ili racionalni.*

Primjer:
Pretpostavimo da smo riješili LP relaksaciju nekog ILP-a simpleks metodom, te da jedan od redaka u optimalnoj tabeli glasi:

$$x_1 - \frac{1}{11}x_3 + \frac{2}{11}x_4 = \frac{36}{11},$$

gdje je x_1 osnovna varijabla s vrijednošću $x_1 = 36/11$, dok x_3 i x_4 imaju vrijednosti $x_3 = x_4 = 0$.
Razdvajanjem svakog koeficijenta na cjelobrojni i racionalni dio, te prebacivanjem cjelobrojnih članova na lijevu, a racionalnih članova na desnu stranu dobivamo:

$$x_1 - x_3 - 3 = -\frac{10}{11}x_3 - \frac{2}{11}x_4 + \frac{3}{11}.$$

Za svako valjano rješenje ILP-a lijeva strana jednakosti treba biti cjelobrojna, pa onda to mora vrijediti i za desnu stranu. Zbog pretpostavke o nenegativnosti svih varijabli u ILP-u zaključujemo:

$$\frac{3}{11} - \frac{10}{11}x_3 - \frac{2}{11}x_4 \leq 0.$$

Ovo je očito valjana nejednakost koja eliminirava pretpostavljeno racionalno rješenje budući je $x_3 = x_4 = 0$.

Algoritam 1 Gomoryjev cutting-plane algoritam

$k \leftarrow$ broj varijabli;
$x^* \leftarrow$ rješenje za (LP);
while x^* nije cjelobrojan **do**
  odaberi redak i_0 u optimalnoj tabeli koji pripada osnovnoj varijabli s racionalnom vrijednošću;
  (redak i_0 glasi: $\overline{a}_{i_0,1}x_1 + \ldots + \overline{a}_{i_0,k}x_k = \overline{b}_{i_0}$)
  $a'_{i_0,j} \leftarrow \overline{a}_{i_0,j} - \lfloor \overline{a}_{i_0,j} \rfloor$;
  $b'_{i_0} \leftarrow \overline{b}_{i_0} - \lfloor \overline{b}_{i_0} \rfloor$;
  (LP) $\leftarrow$ (LP) $\cup \left\{ -a'_{i_0,1}x_1 - \ldots - a'_{i_0,k}x_k + x_{k+1} = -b'_{i_0} \right\}$;
  (x_{k+1} je nova slack varijabla)
  $k \leftarrow k + 1$;
  $x^* \leftarrow$ rješenje za (LP);
end while
return x^*;

Teorem 4.2 (Gomory, 1963.) *Postoji implementacija Gomoryjevog cutting-plane algoritma takva da nakon konačnog broja iteracija nalazi optimalno cjelobrojno rješenje ili dokazuje da je skup mogućih rješenja $S = P \cap \mathbb{Z}^n$ prazan.*

4.2 Chvátalova procedura zaokruživanja

Napomena 4.3 *Chvátal je proučavao općenitiju varijantu (ILP) gdje je skup S ograničen, a elementi matrice A i vektora b realni brojevi.*

Definicija 4.4 *Za nejednakost $\sum_{j=1}^{n} a_j x_j \leq b$ (a_j cjelobrojni) kaže se da pripada elementarnom zatvorenju skupa linearnih nejednakosti P (u oznaci $e^1(P)$) ukoliko postoje nejednakosti $\sum_{j=1}^{n} a_{i,j} x_j \leq b_i,\ i = 1, \ldots, m$ koje definiraju P i $\lambda_1, \ldots, \lambda_m \geq 0$ takvi da:*

$$\sum_{i=1}^{m} \lambda_i a_{i,j} = a_j, \quad j = 1, \ldots, n, \quad \text{i} \quad \left\lfloor \sum_{i=1}^{m} \lambda_i b_i \right\rfloor \leq b.$$

Za $k > 1$ $e^k(P)$ definirano je induktivno s:

$$e^k(P) = e(P \cup e^{(k-1)}(P)).$$

Zatvorenje od P definira se kao:

$$c(P) = \cup_{k=1}^{\infty} e^k(P).$$

Teorem 4.5 (Chvátal, 1973.) *Ukoliko je S ograničeni poliedar, tada je* $\operatorname{conv}(S)$ *moguće dobiti nakon konačnog broja operacija zatvorenja.*

4.3 Schrijverova procedura zaokruživanja

Napomena 4.6 *Schrijver je proučavao varijantu (ILP) gdje je skup S neograničen, a elementi matrice A i vektora b racionalni brojevi.*

Definicija 4.7 *Sustav racionalnih nejednadžbi $Ax \leq b$ je TDI (totalno dualno integralan) ukoliko za sve cjelobrojne vektore c takve da je* $\max\{c^\tau x \mid Ax \leq b\}$ *konačan, dualni problem* $\min\{y^\tau b \mid y^\tau A = c,\ y \geq 0\}$ *ima cjelobrojno optimalno rješenje.*

Napomena 4.8 *Ukoliko je $Ax \leq b$ TDI i b je cjelobrojan, tada je $P = \{x \mid Ax \leq b\}$ cjelobrojni poliedar (tj. sve ekstremne točke od P su cjelobrojne).*

Svaka iteracija Schrijverove procedure sastoji se od slijedeća dva koraka:

- Za dani racionalni poliedar P nalaženje TDI sustava $Ax \leq b$ s A cjelobrojnom matricom koji definira P.
- Zaokruživanje naniže vektora desne strane b.

Teorem 4.9 (Gilles, Pulleyblank (1979.); Schrijver (1981.)) *TDI sustav naveden u prvom koraku Schrijverove procedure postoji za svaki racionalni poliedar P i on je jedinstven ukoliko je P pune dimenzije. Nalaženje takvog TDI sustava moguće je napraviti u konačnom vremenu.*

Teorem 4.10 (Schrijver, 1980.) *Za svaki racionalni poliedar P postoji prirodan broj k takav da se* $\operatorname{conv}(S)$ *dobije nakon k iteracija Schrijverove procedure.*

4.4 Rezultati o složenosti

Definicija 4.11 ((P1) *lower-bound feasibility problem*) *Instanca problema dana je cijelim brojevima m, n, $m \times n$ matricom A, vektorima b i c, te skalarom δ.*
Pitanje: Postoji li $x \in \mathbb{Z}^n$ t.d. $Ax \leq b$ i $c^\tau x > \delta$?

Definicija 4.12 ((P2) *facet validity problem*) *Instanca problema dana je cijelim brojevima m, n, $m \times n$ matricom A, vektorima b i c, te skalarom δ.*
Pitanje: Da li $c^\tau x \leq \delta$ definira hiperstranu od $\mathrm{conv}(\{x \in \mathbb{Z}^n \mid Ax \leq b\})$

Lema 4.13 *Ako neki NP-potpun problem pripada klasi co-NP, onda je NP=co-NP.*

Teorem 4.14 (Karp, Papadimitriou (1980.)) *Ako je (P1) NP-potpun, a (P2) se nalazi u NP, onda je NP=co-NP.*

Korolar 4.15 *Ukoliko prihvatimo da je NP$\neq$co-NP i ako je problem $\min\{c^\tau x \mid x \in S\}$ NP-težak, tada postoje klase hiperstrana od $\mathrm{conv}(S)$ za koje ne postoji polinomijalan "dokaz" da su one doista hiperstrane.*

Definicija 4.16 (Problem separacije za familiju poliedara) *Za danu familiju poliedara FP i poliedar $P \in FP$ i vektor x^* treba naći nejednakost $c^\tau x \leq \delta$ (valjanu za P) t.d. je $c^\tau x^* > \delta$ ili dokazati da je $x^* \in P$.*

Definicija 4.17 (Problem optimizacije za familiju poliedara) *Neka je dana familija poliedara FP i polieadar $P \in FP$ takav da je $P \neq \emptyset$ i P je ograničen. Za dani vektor $c \in \mathbb{R}^n$ treba naći optimalno rješenje x^0 t.d. $c^\tau x^0 \leq c^\tau x$ za sve $x \in P$.*

Teorem 4.18 (Grötschel, Lovász i Schrijver, 1981.) *Postoji polinomni algoritam za problem separacije za FP ako i samo ako postoji polinomni algoritam za problem optimizacije za FP.*

Iako teorem kaže da je problem separacije generalno jednako težak kao i problem optimizacije, moguće je upotrijebiti specifične familije valjanih nejednakosti koje su svojstvene pojedinim problemima.

Definicija 4.19 (Problem separacije) *Za dan vektor x^* treba naći nejednakost $c^\tau x \leq \delta$ koja se nalazi u familiji valjanih nejednakosti FI t.d. je $c^\tau x^* > \delta$ ili dokazati da takva nejednakost u FI ne postoji.*

Separacijski problem baziran na familiji valjanih nejednakosti može biti polinomijalno rješiv čak i ako je pripadajući problem optimizacije NP-težak.

4.5 Poliedarski cutting-plane algoritam

Definicija 4.20 *Prema teoremu 2.7 svaki cjelobrojni program moguće je zapisati u obliku:*

$$P \begin{cases} \min & c^T x \\ s.t. & Ax = b \\ & lx \leq l_0, \quad \forall\ (l, l_0) \in \mathcal{L} \\ & 0 \leq x \leq 1 \end{cases}$$

Pri tome $\mathcal{L}$ predstavlja familiju hiperstrana koje definiraju konveksnu ljusku svih mogućih rješenja za cjelobrojni program.

Napomena 4.21 (Problem separacije) *Za danu točku $x \in \mathbb{R}^n$ i familiju $\mathcal{L}$ nejednakosti u $\mathbb{R}^n$ potrebno je identificirati jednu ili više nejednakosti u $\mathcal{L}$ koje x ne zadovoljava, odnosno dokazati da niti jedna takva nejednakost ne postoji.*

Definicija 4.22 *Za skup nejednakosti $L \subseteq \mathcal{L}$ definiramo relaksaciju originalnog problema:*

$$P'(L) \begin{cases} \min & c^T x \\ s.t. & Ax = b \\ & lx \leq l_0, \quad \forall\ (l, l_0) \in L \\ & 0 \leq x \leq 1 \end{cases}$$

Algoritam 2 Poliedarski-cutting-plane($\mathcal{L}$)

$L \leftarrow \emptyset$;
repeat
  $x' \leftarrow$ optimalno rješenje za LP $P'(L)$;
  $L \leftarrow L \cup \mathrm{SSEP}(\mathcal{L}, x')$;
until $\mathrm{SSEP}(\mathcal{L}, x') = \emptyset$
return x';

Rutina SSEP rješava problem separacije vraćajući jednu ili više nejednakosti iz $\mathcal{L}$ koje su narušene s x', odnosno prazan skup ukoliko je x' valjano rješenje. Algoritam očito staje u konačno mnogo koraka budući je skup $\mathcal{L}$ konačan, a svaka nejednakost iz $\mathcal{L}$ dodaje se u L najviše jednom.

Napomena 4.23 *Kao što znamo, za većinu teških problema nije poznat egzaktan opis konveksne ljuske mogućih rješenja (odnosno svih hiperstrana), pa su poznati samo parcijalni opisi familije $\mathcal{L}$. Još i gore, obzirom da je problem separacije generalno jednako težak kao i problem optimizacije, niti egzaktni separacijski algoritmi općenito nisu poznati.*

Općenito možemo očekivati da za danu podfamiliju $\overline{\mathcal{L}} \subseteq \mathcal{L}$ dobiveni rezultat x' neće biti valjano rješenje originalnog problema.

Dva načina kako je moguće iskoristiti x' za nalaženje optimalnog rješenja:

1. x' upotrijebimo za inicijalizaciju branch-and-bound algoritma
2. kompletni cutting-plane algoritam upotrijebimo pri svakom koraku branch-and-bound procedure $\rightarrow$ branch-and-cut algoritam.

4.6 Branch-and-bound algoritam

Definicija 4.24 *Za skup nejednakosti $L \subseteq \mathcal{L}$ i disjunktne skupove $F_0, F_1 \subseteq \{1, \ldots, n\}$ definiramo linearni program:*

$$
P'(L, F_0, F_1) \begin{cases} \min & c^\tau x \\ s.t. & Ax = b \\ & lx \leq l_0, \quad \forall\ (l, l_0) \in L \\ & x_i = 0, \quad \forall\ i \in F_0 \\ & x_i = 1, \quad \forall\ i \in F_1 \\ & 0 \leq x \leq 1 \end{cases}
$$

Algoritam 3 Branch-and-bound(L, x')

$S \leftarrow \{(\emptyset, \emptyset)\}$;
$x^* \leftarrow x'$;
while $S \neq \emptyset$ **do**
    $(F_0, F_1) \leftarrow get(S)$;
    riješi linearni program $P'(L, F_0, F_1)$;
    if rješenje nije moguće **then**
        continue;
    end if
    $\overline{x} \leftarrow$ optimalno rješenje za $P'(L, F_0, F_1)$;
    if $c^\tau \overline{x} \geq c^\tau x^*$ **then**
        continue;
    end if
    if $\overline{x}$ je cjelobrojan **then**
        $x^* \leftarrow \overline{x}$;
        continue;
    end if
    odaberi varijablu $\overline{x}_i$ t.d. je $0 < \overline{x}_i < 1$;
    $put(S, (F_0 \cup \{i\}, F_1))$;
    $put(S, (F_0, F_1 \cup \{i\}))$;
end while
return x^*;

4.7 Branch-and-cut algoritam

Algoritam 4 Branch-and-cut(x')

1: $S \leftarrow \{(\emptyset, \emptyset)\}$;
2: $x^* \leftarrow x'$;
3: $L \leftarrow \emptyset$;
4: **while** $S \neq \emptyset$ **do**
5:     $(F_0, F_1) \leftarrow get(S)$;
6:     **repeat**
7:         riješi linearni program $P'(L, F_0, F_1)$;
8:         **if** rješenje nije moguće **then**
9:             **go to 4**
10:         **end if**
11:         $\overline{x} \leftarrow$ optimalno rješenje;
12:         **if** $c^\tau \overline{x} \geq c^\tau x^*$ **then**
13:             **go to 4**
14:         **end if**
15:         $L \leftarrow L \cup \mathrm{SSEP}(\overline{\mathcal{L}}, \overline{x})$;
16:     **until** $\mathrm{SSEP}(\overline{\mathcal{L}}, \overline{x}) = \emptyset$
17:     **if** $\overline{x}$ je cjelobrojan **then**
18:         $x^* \leftarrow \overline{x}$;
19:         **continue**;
20:     **end if**
21:     odaberi varijablu $\overline{x}_i$ t.d. je $0 < \overline{x}_i < 1$;
22:     $put(S, (F_0 \cup \{i\}, F_1))$;
23:     $put(S, (F_0, F_1 \cup \{i\}))$;
24: **end while**
25: **return** x^*;

4.8 Napomene o terminologiji

- Poliedarski algoritmi za separaciju $\rightarrow$ cutting-planes
  Cutting-planes + branch-and-bound $\rightarrow$ branch-and-cut
- Dantzig-Wolfe dekompozicija $\rightarrow$ price-and-cut
  Price-and-cut + branch-and-bound $\rightarrow$ branch-price-and-cut
- Lagrangeova relaksacija $\rightarrow$ relax-and-cut
  relax-and-cut + branch-and-bound $\rightarrow$ branch-relax-and-cut

5 Osnovno o implementaciji branch-and-cut algoritama

Implementacija branch-and-cut algoritma u osnovi sastoji se od slijedećih komponenata:

- nadzorni dio

- upravljač stablom pretraživanja (*tree manager*)
- cutting-planes generator
- skladište za čuvanje ograničenja/nejednakosti (*cutting pool*)
- LP solver

5.1 Uloga LP solvera

Uočimo da će efikasnost branch-and-cut algoritma izrazito ovisiti o veličini (broju redaka i stupaca) LP relaksacija i o kvaliteti dobivenih donjih/gornjih ograda.

Obično je potrebno napraviti trade-off:

- manje LP relaksacije → brže rješenje LP-a
- veće LP relaksacije → bolje ograde

Cilj: držati relaksacije što je moguće manjima ne žrtvujući pritom kvalitetu ograda.

5.2 Napredne tehnike

- preprocesiranje ILPa
- postprocesiranje ILPa
- upotreba heuristika za dobivanje dobrih donjih/gornjih ograda
- fiksiranje varijabli obzirom na reduciranu cijenu (*fixing by reduced cost*)
- column generation (→ branch-cut-and-price)
- rad s globalnim i lokalnim cut-ovima (→ *lifting*)
- pažljiva organizacija skladišta za čuvanje ograničenja/nejednakosti (*cutting pool*)

5.3 Paralelizacija

Dobra mogućnost paralelizacije s osnovnim ciljem ublažavanja 4 uska grla:

- veličina stabla pretraživanja
- računanje mogućih rješenja pomoću heuristika
- generiranje narušenih nejednakosti (*cutting-planes*)
- rješavanje LP-a

6 Postojeći softver

6.1 Branch-and-cut softver

- COIN/BCP
  - http://www.coin-or.org/
  - paralelni branch-cut-and-price framework
  - *autori:* Laszlo Ladanyi, Ted Ralphs
- SYMPHONY (Single- or Multi-Process Optimization over Networks)
  - http://branchandcut.org/SYMPHONY/
  - paralelni branch-cut-and-price framework
  - *autori:* Ted Ralphs, Laszlo Ladányi, Márta Esö, Les Trotter, Menal Guzelsoy
- CONCORDE
  - http://www.math.princeton.edu/tsp/concorde.html
  - sekvencijalni solver za TSP
  - *autori:* David Applegate, Robert Bixby, Vašek Chvátal, William Cook
- ABACUS (A Branch-And-CUt System)
  - http://www.informatik.uni-koeln.de/old-ls_juenger/projects/abacus.html
  - sekvencijalni branch-and-cut solver
  - *autori:* Böhm, Christof, Jünger, Reinelt, Thienel
- MINTO (Mixed INTeger Optimizer)
  - http://www.isye.gatech.edu/faculty/Martin_Savelsbergh/software/
  - sekvencijalni branch-and-bound solver
  - *autori:* George Nemhauser, Martin Savelsbergh, Gabriele Sigismondi

6.2 LP solveri

- COIN/CLP
  - http://www.coin-or.org/
  - COIN-OR LP simpleks solver
  - *autor:* John J Forrest
- COIN/OSI (Open Solver Interface)
  - http://www.coin-or.org/

  - API za pozivanje rutina iz raznog softvera za matematičko programiranje
  - podržava slijedeće LP solvere: CLP, CPLEX, dylp, GLPK, OSL, SOPLEX, VOL, XPRESS-MP

- QSopt
  - `http://www.isye.gatech.edu/ wcook/qsopt/`
  - *autori:* David Applegate, William Cook, Sanjeeb Dash, Monika Mevenkamp
- GLPK (GNU Linear Programming Kit)
  - `http://www.gnu.org/software/glpk/`
- OSL (Optimization Solutions and Library)
  - `http://www-306.ibm.com/software/data/bi/osl/`
- AMPL/CPLEX 8.0 Student Edition
  - `http://www.ampl.com/DOWNLOADS/cplex80.html`

6.3 COIN-OR

COIN-OR (COIN) - COmputational INfrastructure for Operations Research
`http://www.coin-or.org/`

Trenutno aktivni projekti:

- BCP: a parallel branch-cut-price framework,
- CGL: a cut generation library,
- CLP: COIN L P, a native simplex solver,
- DFO: a package for solving general nonlinear optimization problems when derivatives are unavailable,
- IPOPT: an interior point algorithm for general large-scale nonlinear optimization.
- Multifario: a continuation method for computing implicitly defined manifolds.
- NLPAPI: a subroutine interface for defining and solving nonlinear programming problems.
- OSI: an open solver interface layer,
- OTS: an open framework for tabu search,

- SBB: Simple Branch and Bound, a branch and cut code,
- SMI: Stochastic Modeling Interface, for optimization under uncertainty,
- VOL: the Volume Algorithm,

6.4 SYMPHONY

SYMPHONY (Single- or Multi-Process Optimization over Networks)

- paralelna generička implementacija branch-cut-and-price algoritma za rješavanje ILP i MILP problema
- framework je zamišljen tako da omogućava istovremeni razvoj sekvencijalne, shared memory i distribuirane (PVM-based) verzije iste aplikacije
- može se koristiti i kao generički MILP solver podržavajući:
  - MPS format (kroz COIN/OR MPS modul)
  - AMPL format (kroz GLPK parser)
- zahtjeva vanjski LP solver za rješavanje LP relaksacija koji se može koristiti putem COIN/OSI-a.

Primjeri problema za koje postoji source kod za download:

- Generički mješoviti cjelobrojni linearni programi
- TSP (Travelling Salesman Problem)
- VRP (Vehicle Routing Problem)
- SPP (Set Partitioning Problem)
- MPP (Mixed Postman Problem)
- Problemi sparivanja

Literatura

[1] K. Aardal, C. van Hoesel. Polyhedral Techniques in Combinatorial Optimization I: Theory, Statistica Neerlandica 50 (1996), 3.

[2] K. Aardal, C. van Hoesel. Polyhedral Techniques in Combinatorial Optimization II: Applications and Computations, Statistica Neerlandica 50 (1996), 3.

[3] D. Applegate, R. Bixby, W. Cook. On the solution of traveling salesman problems, Documenta Mathematica Journal der Deutschen Mathematiker-Vereinigung, International Congress of Mathematicians (1998), 645-656.

[4] D. Applegate, R. Bixby, V. Chvátal, and W. Cook. CONCORDE TSP Solver
`http://www.math.princeton.edu/tsp/concorde.html`

[5] G. B. Dantzig, R. Fulkerson, and S. M. Johnson. Solution of a large-scale traveling salesman problem, Operations Research 2 (1954), 393-410.

[6] M. Jünger, G. Reinelt, and S. Thienel, Practical Problem Solving with Cutting Plane Algorithms in Combinatorial Optimization, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society (1995), 111.

[7] J. E. Mitchell: Branch-and-Cut Algorithms for Combinatorial Optimization Problems. Handbook of Applied Optimization, Oxford University Press, 2002.

[8] G.L. Nemhauser, M.W.P. Savelsbergh, G.S. Sigismondi. MINTO, a Mixed INTeger Optimizer, Operations Research Letters 15 (1994), 47-58.
`http://www.isye.gatech.edu/faculty/Martin_Savelsbergh/software/`

[9] C.H. Papadimitriou, K. Steiglitz. Combinatorial Optimization: Algorithms and Complexity. Prentice Hall, 1982.

[10] T. Ralphs, M. Galati. Decomposition and Dynamic Cut Generation in Integer Programming. Lehigh University Technical Report, July 2003.

[11] T. Ralphs, L. Ladányi. SYMPHONY Version 4.0 User's Guide
`http://www.branchandcut.org/SYMPHONY`

[12] K. S. Ruland. Polyhedral solution to the pickup and delivery problem. Dissertation. Washington University, Sever Institute of Technology, Department of Systems Science and Mathematics.

[13] M.W.P. Savelsbergh, G.L. Nemhauser (1995). A MINTO short course. Report COC-95-xx, Georgia Institute of Technology.

[14] A. Schrijver, Theory of Linear and Integer Programming, John Wiley & Sons (1986).

[15] P. Toth, D. Vigo. The Vehicle Routing Problem. SIAM monographs on discrete mathematics and applications, 2002.