ФАКУЛТЕТ ОРГАНИЗАЦИОНИХ НАУКА

Проф. др Мирко Вујошевић

Метода гранања и ограничавања

Методе оптимизације

САДРЖАЈ

- Проблем целобројног програмирања (ЦП)
- Методе претраживања – увод у гранање и ограничавање
- Релаксација проблема ЦП
- Гранање
- Ограничавање
- Рачунање граница
- Пример – илустрација

Проблем целобројног програмирања

- *Пример:* Задатак ранца

$$(\max) \qquad z = \sum_{j=1}^{n} c_j x_j$$

п.о.

$$\sum_{j=1}^{n} w_j x_j \le W$$

$$x_j \in N_0, \; j = 1,..., \; n$$

НП комплетни проблеми

- Проблем целобројног линеарног програмирања је НП-комплетан (Недертиминистички полиномно комплетан)
- Не постоји детерминистички алгоритам решавања чија сложеност полиномно расте са димензијом проблема

Претраживање

- Скуп допустивих решења је дискретан и коначан.
- Потпуно претраживање (експлицитна енумерација) је у пракси ограничено на мале проблеме.
- Посредно претраживање (имплицитна енумерација) је приступ којим се налажење оптималног решења настоји постићи испитивањем релативно малог дела допустивог скупа.

Увод - проблем

- Нека је D допустиви скуп решења за проблем целобројног (линераног) програмирања (ЦП) који треба да решимо
- $\max_{x \in D} z(x),\ z : D \to R$
- $\sup_{x \in D} z(x) = z^*$
- Задатак је наћи скуп оптималних решења
- $D^* = \{x^* \mid x^* \in D\ ,\ z(x^*) = z^*\}$

Посредно претраживање

- Гранање и ограничавање је општа метода претраживања за решавање оптимизационих проблема целобројног и мешовитог целобројног програмирања.
- Испитује се релативно мали део скупа допустивих решења.
- Остала решења се елиминишу на основу одређивања граница у којима се налази оптимално решење.
- Са рачунског аспекта, најважнији део у решавању ЦП је одређивање *доње* и *горње границе* за вредност критеријумске функције на разматраном скупу.

Основне операције

- Гранање и ограничавање се заснива на приступу **подели** (оригинални допустиви скуп на подскупове) и **овладај** (подскуповима) .
- **Гранање** – дељење допустивог скупа на колекцију подскупова
- **Ограничавање** – одређивање граница вредности критеријумске функције на разматраном подскупу решења
- **Релаксација** – ради лакшег одређивања граница на разматраном подскупу решавају се проблеми са мање рестриктивним претпоставкама (занемарују се нека ограничења, нпр. целобројности)

Релаксација

- Уводи се надскуп $T \supset D$ и функција $g : T \to R$ тако да је
  (a) $g(x) = f(x)$ за свако $x \in D$
  (b) постоји $x \in T$ тако да је $g(x) = f^*$
- $g(x)$ се бира тако да је релативно лако одредити $\sup_{x \in T} g(x)$
- Јасно да је
  $$\sup_{x \in T} g(x) \geq \sup_{x \in D} f(x)$$
- Од избора T и g зависи ефикасност рачунске процедуре

Гранање

- Посматрајмо поделу допустивог скупа D у подскупове $D_1, \ldots, D_k$.
- Тада је
  $$\max_{x \in D} z(x) = \max_{1 \leq i \leq k} \max_{x \in Di} z(x),$$
- Ово значи да се оптимизациони задатак може декомпоновати на k одвојених оптимизационих потпроблема.
- **Идеја:** Ако не можемо оригинални проблем да решимо директно, можда можемо да решавамо рекурзивно мање потпроблеме.
- Гранање је дељење оригиналног проблема у потпроблеме и одлучивање који од њих треба даље разматрати.

Ограничавање

- Означимо са $L(D)$ текућу најбољу вредност критеријумске функције z на до сада испитаном делу допустивог скупа. Ова вредност се назива доња граница јер оптимална вредност критеријумске функције не може бити мања од ње (решава се проблем максимума).
- Означимо са $U(D_i)$ горњу границу вредности критеријумске функције на подскупу D_i. Критеријумска функција на подскупу D_i не може имати вредност већу од $U(D_i)$.
- **Идеја:** Ако је $U(D_i) \leq L(D)$, није потребно даље разматрати потпроблем i.
- Лак начин за рачунање $U(D_i)$ је решавање релаксираног проблема на подскупу који укључује D_i.

Рачунање граница

- Доња граница је најбоља вредност критеријумске функције на испитаним решењима допустивог скупа.
- Горња граница се рачуна решавањем релаксираног проблема на скупу T или на његовом подскупу T_i.
- Код задатка ЦП релаксација се постиже изостављањем неког ограничења, најчешће изостављањем ограничења целобројности.
- Гранање, односно дељење области претраживања на подобласти, постиже се сукцесивним увођењем претходно релаксираних ограничења.

Општи поступак

- **Корак 1.** Релаксирати простор решења брисањем ограничења (нпр. ограничења целобројности; у том случају резултат релаксације је регуларни проблем континуалне оптимизације).
- **Корак 2.** Решити релаксирани проблем, идентификовати оптимално решење и одговарајућу вредност критеријума.
- **Корак 3.** Полазећи од оптималног решења релаксираног проблема, додавати ограничења која модификују простор решења на начин који ће на крају довести до екстремне тачке која задовољава сва ограничења.

Општи поступак

- Најпре се (решавањем релаксираног проблема) израчуна горња граница на читавом скупу $U(D)$.
- Ако се решавањем неког потпроблема добије допустиво решење које даје вредност критеријума $L(\mathrm{D})$ једнаку горњој граници $U(D)$, онда је то решење оптимално.
- Из практичних разлога, поступак се може зауставити и када се разлика између доње и горње границе сведе на прихватљиво малу вредност $U(D) - L(\mathrm{D}) \leq \varepsilon$.

Савладавање потпроблема

- **Корак 1**. Изабрати потпроблем *i* који треба решити и покушати га савладати на један од следећих начина:

(а) Оптимална вредност критеријумске функције не може бити боља од постојеће доње границе

(б) Решавањем потпроблема добија се боље решење од текуће доње границе

(в) Потпроблем нема допустиво решење.

Могу настати два случаја:

(а) Ако је потпроблем *i* савладан, ажурирати доњу границу ако је пронађено боље решење; ако су сви потпроблеми савладани, онда се поступак завршава: пронађено је оптимално решење. У супротном, ставља се $i=i+1$ и понавља корак 1.

(б) Ако потпроблем *i* није савладан, ићи на корак 2

Савладавање потпроблема

- **Корак 2.** Изабрати једну од целобројних променљивих x_j чија оптимална вредност x_j^* у релаксираном проблему није целобројна.
  Елиминисати област $[x_j^*]<x_j<[x_j^*]+1$ (где $[v]$ дефинише највеће цело $\le v$) креирањем два нова потпроблема која одговарају додатним ограничењима

$$x_j \le [x_j^*]$$
$$x_j \ge [x_j^*]+1$$

- Ставити $i=i+1$ и вратити се на корак 1.

Алгоритам гранања и ограничавања заснован на ЛП

$$(\max)\ z = \Sigma c_j x_j$$

п.о.

$$\Sigma a_{ij} x_j \le b_i \qquad i = 1,\ldots,m$$
$$x_j \ge 0 \qquad j = 1,\ldots, n$$
$$L_j \le x_j \le U_j \qquad j = 1,\ldots, n$$
$$x_j \in N_0 \qquad j = 1,\ldots, n$$

Алгоритам заснован на ЛП-релаксацији

Корак 0: *Иницијализација*. Нека на главној листи за почетак буде само оригинални ЛП проблем и нека је t=1, и $z_1 = -\infty$.

Корак 1: *Гранање*. Крај ако је главна листа празна. У супротном изабрати проблем са главне листе.

Алгоритам (наставак)

- Корак 2: *Релаксација.* Решити проблем узет са главне листе. Ако проблем нема допустиво решење или ако је вредност критеријумске функције z мања од z_t (овај потпроблем је савладан), ставити $z_{t+1}=z_t$ и ићи на Корак 1. У супротном ићи на Корак 3.
- Корак 3. Ако решење проблема ЛП задовољава ограничења целобројности,запамтити ово решење и одговарајућу вредност критеријумске функције. Пошто је овај потпроблем савладан, ићи на Корак 1. Ако нису задовољена ограничења целобројности, ићи на корак 4.

Алгоритам (наставак)

- **Корак 4:** *Раздвајање*. Изабрати неку променљиву x_j чија вредност b_j у текућем решењу не задовољава ограничење целобројности. Додати два проблема на главну листу; ови проблеми су идентични претходно решаваном осим што се у једном додаје ограничење

  $x_j \geq [b_j]+1$

  а у другом ограничење:

  $x_j \leq [b_j]$

  Ставити $z_{t+1} = z_t$ и вратити се на Корак 1.

Пример

Max $Z = 21x_1+11x_2$

п.о.

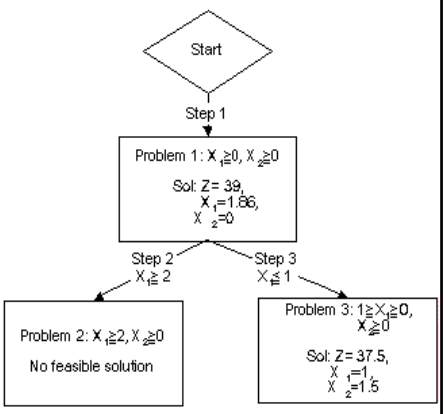
$7x_1+4x_2 \leq 13$

$x_1 \geqslant 0,\ x_2 \geqslant 0$

x_1, x_2 целобројно

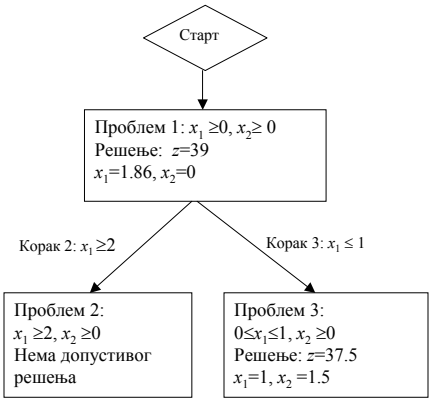
Пример (наставак)

- Корак 0: Ставити $Z_1 = -\infty$. Креирати Проблем 1.
- Корак 1: Уклонити Проблем 1 са главне листе.
- Корак 2: Решити Проблем 1.
- Корак 3: Гранати се на X_1, јер је X_1 нецелобројно.
- Корак 4: Креирати проблеме 2 & 3. Ставити их на главну листу.



Пример (наставак)

- Корак 0: Ставити $Z_1 = -\infty$. Креирати Проблем 1.
- Корак 1: Уклонити Проблем 1 са главне листе.
- Корак 2: Решити Проблем 1.
- Корак 3: Гранати се на x_1, јер је x_1 нецелобројно.
- Корак 4: Креирати проблеме 2 & 3. Ставити их на главну листу.



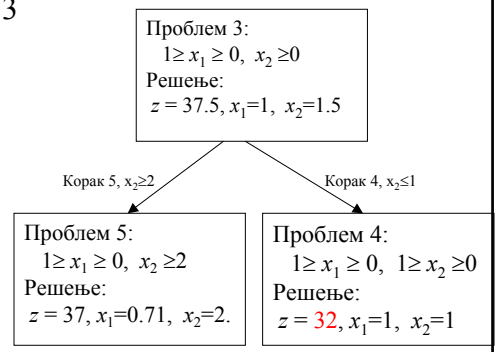
Пример (наставак)

- Корак 0: Ставити $Z_2 = -\infty$.
- Корак 1: Уклонити Проблем 2 са главне листе.
- Корак 2: Решити Проблем 2.
- Корак 3: Нема допустивог решења. Крај.

Проблем 2:
$x_1 \geq 2$, $x_1 \geq 0$
Нема допустивог решења

Пример (наставак)

- Корак 0: Ставити $Z_3 = -\infty$
- Корак 1: Уклонити Проблем 3 са главне листе.
- Корак 2: Решити Проблем 3
- Корак 3: Гранање на x_2, јер x_2 није целобројно
- Корак 4: Креирати проблеме 4 & 5 и ставити их на главну листу.



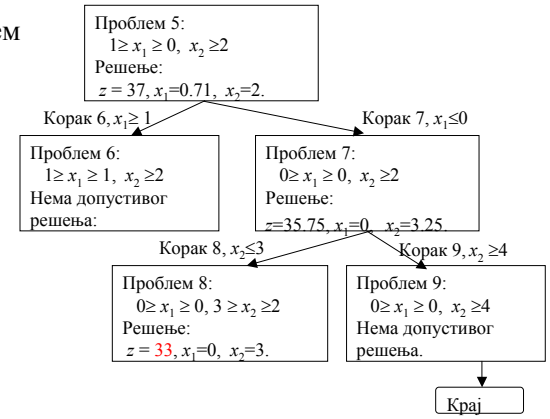
Пример (наставак)

- Корак 0: Ставити $Z_4 = -\infty$.
- Корак 1: Уклонити Проблем 4 са главне листе.
- Корак 2: Решити Проблем 4.
- Корак 3: Решење задовољава ограничења целобројности. Запамтити решење и крај!

Проблем 4:
$1 \geq x_1 \geq 0,\ 1 \geq x_2 \geq 0$
Решење:
$Z = 32,\ x_1 = 1,\ x_2 = 1$

Пример (наставак)

- Следећи исте кораке завршити са рачунањем када је главна листа празна.



Софтвер за методу гранања и ограничавања

- На Интернету се могу наћи различити програми за алгоритам гранања и ограничавања.
- Например, за Matlab је алгоритам BNB на сајту:

  ftp://ftp.mathworks.com/pub/contrib/v5/optim/

Закључак

- Иако постоје бројни алгоритми за решавање проблема ЦП, алгоритми гранања и ограничавања су се показали у пракси прилично ефикасним.
- Предност ових алгоритама за решавање линеарног ЦП је што решавају континуалне проблеме ЛП као потпроблеме оригиналног проблема.
- Метода гранања и ограничавања се користи и у многим софтверима за глобалну оптимизацију.