Dvofazni simplex

Program se pokrece sa argumentima komandne linije oblika arg1 arg2 arg3
arg1 je obavezan I on sadrzi putanju do fajla gde se nalazi linearni problem u odgovarajucem obliku
arg2 (opcioni) sadrzi uputstva kako ce program ispisivati svoj rad
arg3 (opcioni) je maksimalan broj iteracija koje ce simplex metoda raditi (da bi se zadao arg3 mora biti zadat I arg2)

Forma linearnog problema koju prima algoritam je u obliku:

tip funkcija_cilja
A znak b

primer:
(min) f=-x -2y
x+y<=3
-x+3y<=5
x,y>=0

se zadaje u obliku:

min -1 -2
1 1 <= 3
-1 3 <= 5

arg2 moze da sadrzi slova: ‘k’ , ‘t’ , ‘i’ , ‘b’ , ‘l’
‘k’ → vrsi se ispis svakog od koraka u programu
‘t’ → vrsi se ispis tabela pri radu simpleks metode (mora biti I ‘k’ stavljeno)
‘i’ → vrsi se ispis koliko iteracija je bilo potrebno simpleksu (zahteva I ‘k’ stavljeno)
‘b’ → oznacava da se pri izboru koristi blendovo pravilo
‘l’ (malo slovo L)→ ispis ce biti u obliku pajton lista za problem tj umesto -2*x1 + 3*x2 bice [-2 , 3]

primer argumenata za test5.txt gde su svi ispisi ukljuceni I maks iteracija za simpleks je 50

test5.txt ktibl 50

Program za resavanje simpleksa koristi obican tablicni simpleks metod

Kako on radi: (Podrazumeva da postoji podjedinacna matrica na ulazu I da su sve jednacine)

Cisti funkciju cilja od bazicnih (mnozi redove bazicne I dodaje na funkciju cilja da bi napravila 0)
Ciklus:
- Proverava da li postoji C<0
- Ako postoji:
  - Uzima jedno takvo (izbor moze po blendu ako blend nije zadat uzima nasumicno)
  - Trazi Aj u toj koloni da vazi , min bj/Aj za svako j gde je Aj>0
  - Pivotira (pravi 0 gore I dole od tog elementa) I stavlja taj element kao bazu u tom redu
- Inace :
- Kaze da je doslo do optimalnog resenja I vraca ga zaustavlja ciklus

Kako radi ceo program:

Ucitava linearni problem iz tekstualnog fajla
Prebacuje taj problem u standardni oblik (izbacuje nejednacine I sve b prebacuje u pozitivne ako su <0)
Trazi podjedinicnu matricu

Ako ona postoji:
    Nema potrebe za dvofaznim I radi obican simpleks
Inace:
    Dodaje vestacke promenljive u redove gde je potrebno napraviti bazu
    Radi obican simpleks nad novim problemom
    Ako je rezultat simpleksa != 0:
        Ispisi da pocetni problem nema resenje
    Inace:
        Uklanja vestacke promenljive iz dobijene tabele po algoritmu:
            Dok postoje vestacke radi:
            Ako je vestacka nebazisna ukloni samo celu kolonu
            Ako je bazisna I ostatak reda je 0 ukloni red I kolonu
            Ako je bazisna I postoji el!=0 u ostatku reda izaberi neki takav I pivotiraj oko njega (tada ta vestacka postaje nebazicna)

        Za tabelu sa uklonjenim vestackim stavlja da je poslednji red jednak onom iz pocetnog problema
        Radi simpleks nad tobom tabelom
        Ispisuje rezultat

Primeri izgleda programa:

```
Trenutna konfiguracija programa:
Koristi se blendovo pravilo: True
Maximalan broj iteracija u simpleksu: inf
Format ispisa kao pajton liste: True
Ipisuju se:
-koraci pri radu
-tabele pri iteracijama u simpleksu
-nakon koliko iteracija je zavrsen simpleks
-finalno resenje

Resavamo problem:
min f = [2.0, 0.0, 3.0, 1.0]

[0.0, -1.0, -1.0, 1.0] = 3.0
[2.0, 0.0, 2.0, 4.0] = 12.0
[1.0, 1.0, 2.0, 1.0] = 3.0

Prebacujemo u standardni oblik

min f = [2.0, 0.0, 3.0, 1.0]

[0.0, -1.0, -1.0, 1.0] = 3.0
[2.0, 0.0, 2.0, 4.0] = 12.0
[1.0, 1.0, 2.0, 1.0] = 3.0
```

```
Krecemo da primenjujemo simpleks

Pocetna tabela za simpleks:
[0.0, -1.0, -1.0, 1.0, 3.0]
[1.0, 2.0, 3.0, 0.0, 0.0]
[2.0, 0.0, 3.0, 1.0, 0]

Tabela za ociscenu ciljnu funkciju od bazicnih izgleda ovako:
[0.0, -1.0, -1.0, 1.0, 3.0]
[1.0, 2.0, 3.0, 0.0, 0.0]
[0.0, -3.0, -2.0, 0.0, -3.0]

Iteracija 0 , pivotirano po A[1][1]
[0.5, 0.0, 0.5, 1.0, 3.0]
[0.5, 1.0, 1.5, 0.0, 0.0]
[1.5, 0.0, 2.5, 0.0, -3.0]

Simpleks je zavrsio br iteracija = 1
optimalna vrednost funkcije je 3.0
x = [0 0.0 0 3.0 ]
```

Program bez prosledjenih argumenata ispisa (tj. Samo sa prosledjenom putanjom do fajla):

```
Trenutna konfiguracija programa:
Koristi se blendovo pravilo: False
Maximalan broj iteracija u simpleksu: inf
Format ispisa kao pajton liste: False
Ipisuju se:
-finalno resenje

optimalna vrednost funkcije je 3.0
x = [0 0.0 0 3.0 ]

Process finished with exit code 0
```

Dualni Simpleks

Prima 3 argumenta komandne linije arg1 arg2 arg3
arg1 – putanja do fajla

arg2 sadrzi:
‘i’ → Pisace indekse koje je uzeo (da bi se videlo da ne koristi blendovo pravilo ako nije receno)
‘b’ → Koristice se blendovo pravilo za indekse
‘l’ (malo L) → pisace koliko je iteracija bilo potrebno simpleksu

arg3 je max broj iteracija (mora biti arg2 naveden ako ocete I arg3)

format je isti kao za dvofazni:
primer:
(min) f = 2x + 4y +3z
x – y – z <= -2
2x + y >= 1

treba biti u formatu:

min 2 4 3
1 -1 -1 <= -2
2 1 0 >= 1

Kako radi:
Ucitava pocetni problem
Prebacuje problem u odgovarajuci oblik

Ciklus:
- Ako nema negativno b imamo optimalno resenje prekini ciklus I ispisi
- Inace izaberi neko b<0 I uzmi taj red
- Ako su u tom redu svi veci od 0 ne postoji dopustivo resenje
- Inace trazimo element Aj tako da vazi max Z[j] / Aj
- pivotiramo I vracamo se na ciklus