RJEŠAVANJE PROBLEMA LINEARNOG PROGRAMIRANJA POMOĆU PROGRAMSKOG JEZIKA R I PAKETA LPSOLVEAPI

Fatka Kulenović[1], Amel Džanić[1], Damir Hodžić[1]
[1]Univerzitet u Bihaću - Tehnički fakultet Bihać, Irfana Ljubijankića bb,
kulen.a@bih.net.ba,amel.dzanic@gmail.com, hodzicdamir@yahoo.com

Ključne riječi: linearno programiranje, programski jezik R, funkcija cilja

SAŽETAK:

U ovome radu ćemo da prezentramo programski jezik R sa paketom lpSolveApi, i njegovu primjenu na probleme koji se rješavaju sa primjenama metoda linearnog programiranja. Prvo ćemo teoretski objasniti šta je to problem linearnog programiranja, kako ispravno modelirati sistem te onda kako taj model prevesti u programski jezik R da bi izvršili linerno programiranje na primjeru.

1. UVOD

Iz želje da se donošenje odluka temelji na naučnim osnovama, nikla je nova grana primenjene matematike - operaciona istraživanja. Naime, enciklopedija Britanica definiše pojam Operaciona istraživanja kao primjenu naučnih metoda za upravljanje i organizaciju vojnih, državnih, poslovnih i drugih procesa [1]. Hrvatsko društvo za operaciona istraživanja (HDOI) na svojoj web stranici definiše da su operaciona istraživanja interdisciplinarna znanstvena grana primjenjene matematike, koja pomoću metoda matematičkog modeliranja, statistike i različitih algoritama postiže optimalno ili približno optimalno rješenje složenih problema [2]. Istorijski gledano operaciona istraživanja se javljaju neposredno prije drugog svjetskog rata u Velikoj Britaniji u rješavanju problema vojne prirode, mada kako kaže enciklopedija Britanica, svako uključivanje nauke u proces upravljanja možemo nazvati prethodnikom operacionih istraživanja. Zbog svoje praktične primjene, operaciona istraživanja su se brzo proširila i na druga područja.
Operaciona istraživanja u praksi zahtjevaju timski rad i rješavanje problema koje se obavlja u više faza. Te faze rješavanja problema operacionog istraživanja su sljedeće [3]:

1. Formulaciju realnog problema koji treba riješiti pripkupljanje relevantnih podataka;
2. Formulacija odgovarajućeg matematičkog modela;
3. Rješenje modela;
4. Testiranje modela te njegova dorada po potrebi;
5. Priprema za primjenu modela;
6. Implementacija.

Linearno programiranje je posebna matematička metoda, tehnika, koja se koristi u operacionim istraživanjima. Kako literatura navodi, linearno programiranje je naročito uspješna metoda operacionih istraživanja. Uz pomoć računara, kao alata za preračunavanje algoritama linearnog

programiranja ova metoda se jako široko rašilirala skoro u sve segmente današnjeg života. Programski jezik R je i namjenjen kako stoji za naučna preračunavanja i računarsku statistiku. Moć programskog jezika R jeste u tome da je otvorenog koda te tako da ima jako puno korisnika koji doprinose, pišući nove biblioteke te unapređujući stare. Upravo u tome i leži moć programskog jezika R, koji se može primjeniti u rješavanju problema linearnog programiranja.

2. LINEARNO PROGRAMIRANJE

Razvoj linearnog programiranja po nekim autorima je rangiran kao jedan od najznačajnijih napredaka nauke u polovini 20. vijeka. Utjecaj linarnog programiranja od 1950. godine pa do danas je zapanjujuć. Danas, linearno programiranje, jeste standardna alatka koja je sačuvala velike količine novca mnogim firmama širom svijeta. Najveći dio svih naučnih proračuna na računarima baš koristi linearno programiranje.Istorijski gledano, problem rješavanja linearnih nejednačina datira od Fouriera koji je 1827. godine objavio metod za rješavanje linearnih nejednačina po kojem je taj metod i nazvan metod eliminacije Fourier-Motzkin. Pa ipak, kao prvu pravu formulaciju problema linearnog programiranja dao je Sovjetski matematičar Leonid Kantorović 1939. godine, u rješavanju problema optimalne potrošnje resursa. Kasnije će posthumno 1975. godine on, zajedno sa američkim ekonomistom Tjalling Koopmans, dobiti i nobelovu nagredu za doprinos teoriji optimalne alokacije resursa, u kojoj linearno programiranje igra ključnu ulogu[10]. Nakon drugog svjetskog rata razvoj linearnog programiranja postaje rapidan jer mnoge branše su smatrale veoma korisnim alatom. Mnogi autori daju najvažniju ulogu radu Georga Dantziga iz 1947 godine [4], te njega smatraju osnivačem metoda linearnog programiranja. On je i izumio poznati simplex metod koji je vrlo lako implementirati u računarstvu, i koji rješava veliki broj problema linearnog programiranja.

2.1. Formulacija standardnog problema linearnog programiranja

Formulacija standardnog problema linearnog programiranja glasi [11][3] :
naći ono nenegativno rješenje (x_1 , x_2 , … , x_n) , ($x_i \geq 0$, $i = 1, 2, \dots, n$) sistema linearnih nejednačina (ograničenja , uvjeta)

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &\leq r_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &\leq r_2 \\ &\dots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &\leq r_2 \end{aligned} \tag{1}$$

za koje funkcija cilja ili funkcija kriterija

$$F = F(x_1, x_2, \dots, x_3) = c_1x_1 + c_2x_2 + \cdots + c_nx_n \tag{2}$$

Dostiže optimalnu (maksimalnu ili minimalnu) vrijednost.

Rješenje$(x_1, x_2, \dots, x_3)$ u primjenama, najčešće ima značenje *plana* ili *programa* (proizvodnje , transporta), pa je otuda ovaj problem dobio naziv *"programiranje"*, a naziv *"linearno programiranje"* potječe od toga što su ograničenja varijabli, kao i funkcija cilja linearni.Proizvoljno rješenje sistema nejednačina predstavlja tačku n-dimenzionalnog prostora , to jest $(x_1, x_2, \dots, x_3) \in R^n$.

Svako nenegativno rješenje sistema nejednačina naziva se dopustivim ili mogućim rješenjem. Svaki problem linearnog programiranja spada u jednu od triju kategorija:

1. ima optimalno rješenje
2. neizvediv je, ima višeznačna rješenja
3. skup mogućih rješenja neograničen

Može se reći : problem linearnog programiranja ima rješenje ako veličina F_{max} (F_{min}) ima konačnu vrijednost na skupu S dopustivih rješenja. Problem linearnog programiranja nema rješenja ako je skup S prazan skup ili ako veličina F_{max} (F_{min}) nema konačnu vrijednost.

Pri određivanju optimalnog rješenja pojavljuju se dva kriterija optimizacije : maksimizacija ili minimizacija vrijednosti funkcije cilja. Iako postoje dva kriterija optimizacije, problem izbora optimalnog rješenja može se smatrati jedinstvenim, jer se jedan kriterij optimizacije može zamjeniti drugim, kako to utvrđuje slijedeći teorem :

Teorem :
Dani kriterij optimizacije može se zamijeniti suprotnim, pri čemu zamjena ne utječe na optimalno rješenje, to jest, ako je za $X^* \in S$ *ispunjeno*

$$F(X^*) = \max F(X) \quad X \in S \tag{3}$$

Onda je

$$-F(X^*) = \min[-F(X)] \text{ i obrnuto } \quad X \in S \tag{4}$$

Dokaz:
Prema pretpostavci teorema ispunjeno je $F(X^*) \geq F(X)$ *za svako* $X \in S$. *Ako se ova nejednadžba pomnoži sa s-1, dobije se* $-F(X^*) \geq -F(X)$ *za svako* $X \in S$ *čime je teorem dokazan.*

3. Programski jezik R

R je programski jezik i okruženje za statističke izračune i vizualizaciju. R je slobodan programski jezik (engl. *free*) što znači da se može slobodno koristiti i distribuirati te da je otvorenog kôda (engl. *open-source*). R je implementacija jezika S kojeg je razvio John Chambers s kolegama u Bell Laboratories te Robert Gentleman sa sveučilišta u Aucklandu, Novi Zeland. Ista grupa razvila je i jezik C i UNIX. Trenutno jezik R razvija tzv. „jezgra“, osnovna grupa za razvoj jezika R (engl. *R Core Team*). Postoje neke važne razlike, ali mnogi programi (kôd) napisani u jeziku S rade nepromijenjeni i u jeziku R.[10]

R pruža širok izbor statističkih metoda za linearno i nelinearno modeliranje, klasične statističke testove, analize vremenskih serija, klasteriranje. Lako je proširiv s velikim izborom grafičkih tehnika. Statističari su razvili stotine specijaliziranih statističkih procedura za širok raspon upotrebe putem tzv. pridodanih paketa (engl. *contributed packages*) koji su slobodno dostupni i integrirani direktno sa sistemom R. Donedavno je postojalo uvriježeno mišljenje da sistem R koristi uglavnom akademska zajednica dok je za analize podataka u tvrtkama standard sistem SAS. No to se danas promijenilo. Sistem R je npr. Federalna agencija za lijekove (*Federal Drug Administration - FDA*) identificirala je dijelove R-a pogodnima za tumačenje podataka iz kliničkih istraživanja. Također, velik je broj kompanija koje prelaze na sistem R za analizu i prezentaciju svojih podataka[4].

3.1. Rješavanje linearnog programiranja pomoću programskog jezika R

Da bi probleme linearnog programiranje rješili u programskom jeziku R potrebno je učitati paket sa pomoću koje ćemo rješavati problem. U programskom jeziku R postoji više paketa za rješavanje.Svakipaket ima svoju pomoćnu dokumentciju i to se dobija pomoću naredbe help(package=„*ime_paketa*“). Postoji više paketa koji omogućavaju rješavanje problema linearnog programiranja . Među njma su: lpSolveAPI, linprog, lpsymphony, Rcplex. Mi ćemo u našem radu proći paket lpSolveAPI.Prvo ćemo da instaliramo paket lpSolveAPI sa naredbom install.packages(„lpSolveAPI“). Poslje instalacije korištenje biblioteke se postiže sa naredbom

library(lpSolveAPI) .Kada smo instalirali paket sada možemo da s pomoću naredbe help(package="lpSolveAPI") pregeledamo sadržaj paketa. Nakon toga možemo da pristupimo pisanju programskog koda koji će da izvrši optimizaciju pomoću lineranog programiranja. No prije programskog koda moramo da napravimo model koji ćemo moći prikazati računaru. Paket omogućava rješavanje svih vrsta linearnog programiranja, tako da je u osnovi neophodno da se dobije skup linearnih nejednačina i funkcija cilja kako je to prikazano sa formulama 1 i 2. Da bi smo to pokazali uzmimo problem alokacije sa sljedećim tekstom:

Teretna kompanija treba da nađe način kako da maksimizira profit po transportu dobara. Kompanija ima 3 vlaka (V_1-V_3) i 4 vrste tereta (T_1-T_4). Koliko svakog tereta treba u svaki vagon ukrcati da bi se maksimizirala zarada. Treba voditi pažnje na ograničenja: zapremina svakog vlaka, i njegova nosivost, zapremina tereta i dostupnost tereta.Podaci su dati u tabelama:

Tabela 1: Podaci o vagonima

VAGONI	NOSIVOST(t)	KAPACITET(m3)
V_1	10	7000
V_2	12	8000
V_3	10	6000

Tabela 2: Podaci o teretu

TERET	DOSTUPNO (t)	ZAPREMINA(t/m3)	PROFIT (KM/t)
T_1	20	400	2500
T_2	15	350	3500
T_3	10	300	4000
T_4	20	500	3500

Kada imamo ovako zadat zadatak sada treba da napravimo model koji odgovara formulama (1) i (2). Prvo ćemo da modeliramo funkciju cilja, ako u svaki vagon mogu se rasporediti sve vrste tereta tada dobijamo funkciju:

$$\begin{aligned} Fmax = {} & (2500V_1T_1 + 3500V_1T_2 + 4000V_1T_3 + 3500V_1T_4) + \\ & (2500V_2T_1 + 3500V_2T_2 + 4000V_2T_3 + 3500V_2T_4) + \\ & (2500V_3T_1 + 3500V_3T_2 + 4000V_3T_3 + 3500V_3T_4) \end{aligned} \qquad (5)$$

Sada sljede funkcije ograničenja:

$$\begin{aligned} & V_1T_1 + V_1T_2 + V_1T_3 + V_1T_4 \leq 10 \\ & V_2T_1 + V_2T_2 + V_2T_3 + V_2T_4 \leq 12 \\ & V_3T_1 + V_3T_2 + V_3T_3 + V_3T_4 \leq 10 \\ & 400V_1T_1 + 350V_1T_2 + 300V_1T_3 + 500V_1T_4 \leq 6000 \\ & 400V_2T_1 + 350V_2T_2 + 300V_2T_3 + 500V_2T_4 \leq 5000 \\ & 400V_3T_1 + 350V_3T_2 + 300V_3T_3 + 500V_3T_4 \leq 8000 \\ & V_1T_1 + V_2T_1 + V_3T_1 \leq 20 \\ & V_1T_2 + V_2T_2 + V_3T_2 \leq 15 \\ & V_1T_3 + V_2T_3 + V_3T_3 \leq 10 \\ & V_1T_4 + V_2T_4 + V_3T_4 \leq 20 \end{aligned} \qquad (6)$$

Da bi bilo preglednije možemo koeficjente iz (6) staviti u tabelu:

Tabel 3. Koeficjenti nejednačina ograničenja

V_1T_1	V_1T_2	V_1T_3	V_1T_4	V_2T_1	V_2T_2	V_2T_3	V_2T_4	V_3T_1	V_3T_2	V_3T_3	V_3T_4	**Const**
1	1	1	1	0	0	0	0	0	0	0	0	10
0	0	0	0	1	1	1	1	0	0	0	0	12
0	0	0	0	0	0	0	0	1	1	1	1	10
400	350	300	500	0	0	0	0	0	0	0	0	6000
0	0	0	0	400	350	300	500	0	0	0	0	5000
0	0	0	0	0	0	0	0	400	350	300	500	8000
1	0	0	0	1	0	0	0	1	0	0	0	20
0	1	0	0	0	1	0	0	0	1	0	0	15
0	0	1	0	0	0	1	0	0	0	1	0	10
0	0	0	1	0	0	0	1	0	0	0	1	20

Sada je jednostavno napisati i programski kod:

```
#ucitavanje biblioteke
library("lpSolveAPI")
#Postavljanje modela
lprec <-make.lp(10,12)
lp.control (lprec,sense="max")
#Parametri funkicje cilja
set.objfn(lprec,c(2500,3500,4000,3500,2500,3500,4000,3500,2500,3500,4000,3500))
#Postavljanje funkcija ogranicenja
add.constraint(lprec,c(1,1,1,1,0,0,0,0,0,0,0,0),"<=",10)
add.constraint(lprec,c(0,0,0,0,1,1,1,1,0,0,0,0),"<=",12)
add.constraint(lprec,c(0,0,0,0,0,0,0,0,1,1,1,1),"<=",10)
add.constraint(lprec,c(400,350,300,500,0,0,0,0,0,0,0,0),"<=",6000)
add.constraint(lprec,c(0,0,0,0,400,350,300,500,0,0,0,0),"<=",5000)
add.constraint(lprec,c(0,0,0,0,0,0,0,0,400,350,300,500),"<=",8000)
add.constraint(lprec,c(1,0,0,0,1,0,0,0,1,0,0,0),"<=",20)
add.constraint(lprec,c(0,1,0,0,0,1,0,0,0,1,0,0),"<=",15)
add.constraint(lprec,c(0,0,1,0,0,0,1,0,0,0,1,0),"<=",10)
add.constraint(lprec,c(0,0,0,1,0,0,0,1,0,0,0,1),"<=",20)
#kraj modela
print (lprec)
print (solve(lprec))
print (get.objective(lprec))
rjesenja<-get.varijables(lprec)
names(rjesenja)<-c("T1V1","T2V1","T3V1","T4V1","T1V2","T2V2","T3V2","T4V2",
                   "T1V3","T2V3","T3V3","T4V3")
print (rjesenja)
#obriši prostor
delete.lp(lprec)
```

Slika 1. *Programski kod u jeziku R za rješavanje konkretnog problema*

Pokretanjem ovoga koda u R-u dobijamo sljedeće rezultate :

```
Model name:
  a linear program with 12 decision variables and 10 constraints
[1] 0
[1] 107000
T1V1 T2V1 T3V1 T4V1 T1V2 T2V2 T3V2 T4V2 T1V3 T2V3 T3V3 T4V3
   0    0   10    0    0   12    0    0    0    3    0    7
```

Slika 2. *Ispis rješenja u programskom jeziku R*

Prvi redak ispisa programskog jezika r jeste [1] 0, ako se pogleda u dokumentaciju paketa lpSolveAPI, to znači da je metoda solve() rješila problem linearnog programiranje i da su dobijeni optimalni rezultati. Zatim, vidljivo da je maksimalna dobit 107000KM u slučaju kada je sa 10 tona tereta3 natovaren vagon1, vagon 2 da je natovaren sa 12. tona tereta2 i vagon3 da je natovaren sa 3. tone tereta2 i 7. tona tereta4 .

4. ZAKLJUČAK

Linearno programiranje je dokazana tehnika, metoda koja rješava veliki skup problema koji značajno smanjuju troškove i povećavaju dobit u ekonomiji ili uopšte vrše optimizaciju tražeći optimalno rješenje, bilo to minimum ili maksimumtj. ono polazi od sistema i unaprijed utvrđenihkriterija koji mogu biti različiti u zavisnosti odproblema, ispituje uvjete optimalnosti, određujući ona rješenja koja su optimalna.

U konkretnom primjeru vidjeli smo kako jednostavno uz pomoć programskog jezika R i paketa lpSolveAPI možemo riješiti jedan složen problem alokacijsko transpornog tipa, gdje se maksimizira dobit, na osnovu pravilne alokacije tereta po vagonima.Da bi linearno programiranje bilo uspješno mora se dobro modelirati sistem tj. moraju se sva ograničenja predvidjeti i uvrstiti u model kao i sve varijable odlučivanja.

5. LITERATURA

[1] William K. Holstein, Russell L. Ackoff, Morris Tanenbaum, Samuel Eilon: *Operations research*, https://www.britannica.com/topic/operations-research#ref22348 (*pristupano 20.04.2017)*

[2] http://hdoi.hr/hr/about-us/sto-je-hdoi/ (*pristupano 20.04.2017*)

[3] Frederick S. Hillier, Gerald J. Lieberman: *INTRODUCTION TO OPERATIONS RESEARCH, Seventh Edition*,McGraw-Hill 2001,

[4] dr. sc. Andreja Radović:*Upoznavanje sa sintaksom jezika R i njegova primjena u osnovnoj statističkoj i grafičkoj analizi podataka*,www.srce.unizg.hr/files/srce/docs/edu/R/s720_polaznik.pdf*(pristupano 26.04.2017)*

[10]W.N. Venables, D.M. Smith: Uvod u korištenje R-a,https://cran.r-project.org/doc/contrib/Kasum+Legovic-UvodUr.pdf*(pristupano 26.04.2017)*

[11]Prof.sc.dr. Husein Pašagić: *MATEMATIČKE METODE U PROMETU*, Sveučilište u Zagrebu Fakultet prometnih znanosti, Zagreb 2003.