https://doi.org/10.37528/FTTE/9788673954455/POSTEL.2021.025

PRIMENA DEKLARATIVNOG PROGRAMIRANJA PRI RAZVOJU APLIKACIJE ZA MINIMIZIRANJE CENE PRENOSA PODATAKA

Snežana Mladenović[1], Ivana Stefanović[2], Stefan Zdravković[1], Slađana Janković[1]
[1]Univerzitet u Beogradu - Saobraćajni fakultet,
[2]Akademija tehničko-umetničkih strukovnih studija Beograd – Odsek Visoka škola elektrotehnike i računarstva, ivanas@viser.edu.rs
snezanam@sf.bg.ac.rs, s.zdravkovic@sf.bg.ac.rs, s.jankovic@sf.bg.ac.rs

Rezime: *Sa rapidnim razvojem bežičnih komunikacionih sistema javlja se potreba za razvojem softverskih sistema koji minimiziraju troškove, kašnjenje, gubitak ili pak maksimiziraju brzinu prenosa i slično. Rad se bavi problemom minimizacije cene prenosa podataka u funkciji kapaciteta telekomunikacionih linkova. U cilju rešavanja ovog problema, prvo je formulisan matematički model, a zatim dizajnirana i implementirana odgovarajuća aplikacija u deklarativnom programskom jeziku. Poznato je da je deklarativo programiranje dobar izbor za optimizacione probleme, jer je dovoljno specificirati relacije koje moraju biti zadovoljene, bez specifikacije efektivne procedure za pronalaženje vrednosti promenljivih odlučivanja. Za potrebe testiranja aplikacije, razvijen je pomoćni softver koji nasumično generiše telekomunikacionu mrežu sa zadatim osobinama. Ovo daje mogućnost da aplikaciju testiramo na proizvoljnom broju različitih telekomunikacionih mreža, sa različitim brojem čvorova i linkova. U radu su predstavljeni i detaljno analizarani rezultati optimizacije za nekoliko nasumično generisanih telekomunikacionih mreža.*

Ključne reči: *deklarativno programiranje, optimizacija, cena prenosa podataka, kapacitet telekomunikacionih linkova, protok mreže*

1. Uvod

Optimizacija telekomunikacione mreže se često bazira na primeni teorije grafova. Graf $G=(N,A)$ je uređeni par koji se sastoji od skupa čvorova N i skupa grana A. Učesnici između kojih se vrši razmena podataka unutar telekomunikacione mreže predstavljaju se kao elementi skupa N. Zavisno od konkretne telekomunikacione mreže učesnici između kojih se vrši razmena podataka mogu biti bazne stanice, računari, serveri, ruteri, senzori, različiti pametni uređaji poput telefona, tableta, satova, IoT (*Internet of Things*) uređaja itd. Cilj optimizacije telekomunikacione mreže je najčešće minimizacija troškova, kašnjenja i gubitaka kao i maksimizacija brzine prenosa, iskorišćenosti kapaciteta [1]. U zavisnosti od konkretnog optimizacionog problema, skup grana može predstavljati rastojanje između

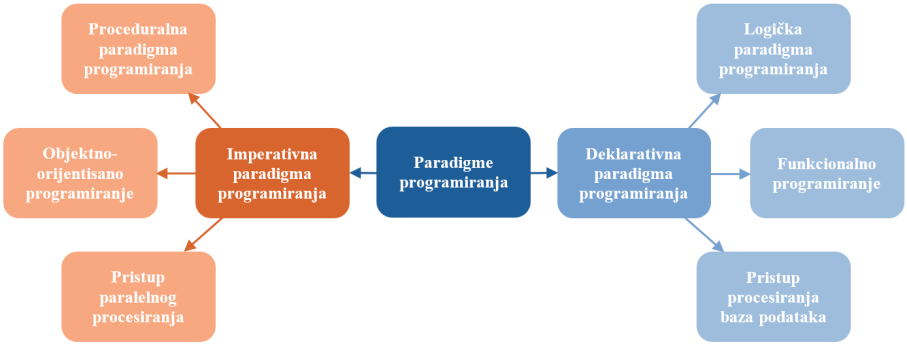
predajnika i prijemnika, slabljenje na linku, snagu predajnika, kapacitet linka, brzinu prenosa podataka, cenu prenosa podataka i slično [2].

U radu se razmatra problem minimizacije cene prenosa podataka u funkciji kapaciteta telekomunikacionih linkova. Pri tome, razmena informacija između čvorova mreže je jednosmerna, odnosno telekomunikaciona mreža se predstavlja pomoću orjentisanog grafa. Ovaj problem je veoma zastupljen kod telekomunikacionih i računarskih mreža prilikom procesa rutiranja saobraćaja u funkciji raspoloživih resursa, odnosno kapaciteta telekomunikacionih linkova u ovom slučaju [3].

Naredne sekcije rada su organizovane na sledeći način: u sekciji 2 je dat kratak osvrt na deklarativno programiranje. Sekcija 3 formuliše matematički model za posmatrani optimizacioni problem, dok sekcija 4 ukratko opisuje njegovu implementaciju u optimizacionom paketu *IBM ILOG CPLEX Optimization Studio*. U sekciji 5 predstavljeni su i analizirani najznačajniji rezultati istraživanja. Na kraju rada, dati su zaključci o mogućnostima korišćenja deklarativnog programiranja za optimizacione probleme koji se javljaju u telekomunikacionom saobraćaju.

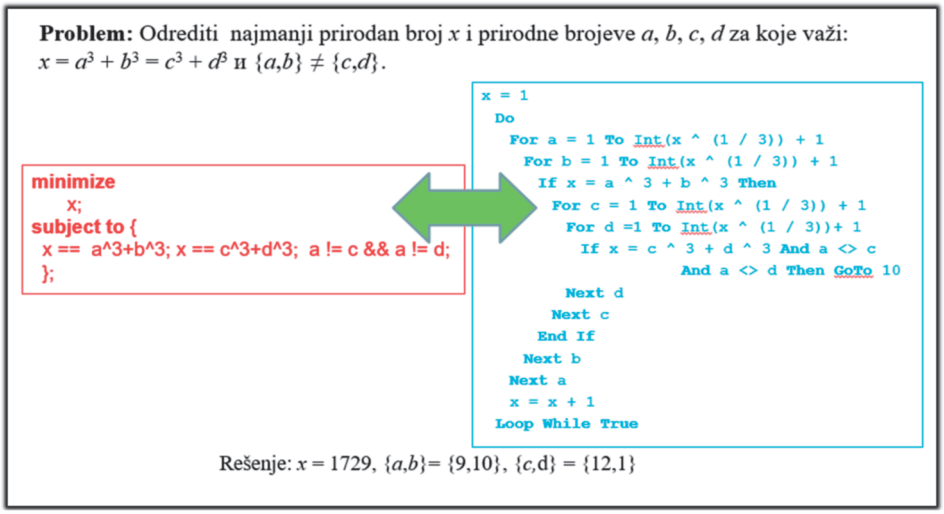
2. Deklarativno programiranje

Paradigma programiranja je pristup programiranju zasnovan na određenoj matematičkoj teoriji ili koherentnom skupu principa. Svaka paradigma programiranja podržava skup koncepata koji je čini posebno podesnom za određenu klasu problema. Postoje različite paradigme programiranja (neki autori su prebrojali čak 30! [4]), ali se generalno razlikuju dve osnovne: imperativna i deklarativna, kao što je prikazano na Slici 1 [5]).



Slika 1. Paradigme programiranja

Imperativno programiranje je fokusirano na to da opiše kako se problem rešava, tj. da implementira već poznat algoritam, pa se stoga često naziva proceduralnim. S druge strane, deklarativano programiranje specificira relacije koje moraju biti zadovoljene, bez specifikacije same efektivne procedure. Dakle, opisuje se ono **šta** se rešava umesto **kako** se rešava. Moderni programski jezici često podržavaju obe paradigme, npr. *JavaScript* objektno orijentisanu i funkcionalnu [5]. Najćešće spominjana prednost deklarativnog programiranja je lako i kratko kodiranje, kao što je prikazano na Slici 2, a takođe i jednostavnost verifikacije, testiranja, održavanja, nadogradnje i distribuirane implementacije softverskog sistema [6].



Slika 2. Deklarativno vs. imperativno programiranje – kod u OPL-u i Visual Basic-u za isti problem

Deklarativo programiranje je dobar izbor za optimizacione probleme i publikovani su brojni radovi koji se time bave [7, 8, 9]. Takođe, kroz istoriju se pojavio veliki broj deklarativnih programskih jezika: *Prolog*, *Oz*, *Elipce*, *OPL*, *ECLiPSe*, itd. Iako deklarativno programiranje nudi niz pogodnosti, ono ima i jedan ozbiljan nedostatak: ne može direktno da stupi u interakciju sa stvarnim svetom. Kako većina optimizacionih aplikacija, zahteva više od samog rešavanja optimizacionog zadatka, često je neophodno koristiti i neki imperativniscript jezik (*JavaScript*, *Python*, *VBScprit*, *OPL Script*, itd.) za komponovanje i upravljanje optimizacionim modelima.

3. Matematički model

Posmatrajmo mrežu sa m čvorova. Ako pretpostavimo da je svaki čvor numerisan prirodnim brojem od 1 do m, možemo smatrati da je skup čvorova N={1,2, …, m}. Neka je b_i, i=1, …, n, protok i-tog čvora. Slično, ukoliko mreža ima n grana, skup grana možemo označiti sa A={a_1, a_2, …, a_n}, pri čemu je a_j, j=1, ..n, uređena petorka sledećeg oblika (l_j, u_j, c_j, s_j, d_j), gde je: l_j – donja granica kapaciteta grane a_j, u_j– gornja granica kapacitet grane a_j, c_j - cena prenosa podataka za granu a_j, s_j – izvorni čvor grane a_j, d_j – odredišni čvor grane a_j.

Neka je dalje S={$s_j|a_j \in$A} skup svih čvorova iz kojih polazi bar jedna grana, a D={$d_j|a_j \in$A} skup svih čvorova u kojima se završava bar jedna grana. Matematička formulacija problema minimizacije cene prenosa podataka u funkciji kapaciteta telekomunikacionih linkova kao zadatka linearnog programiranja može se predstaviti na sledeći način:

$$\min F(x) = \sum_{j=1}^{n} c_j x_j \tag{1}$$

pri ograničenjima:

$$l_j \le x_j \le u_j, \text{ za } j=1,\ldots,n \tag{2}$$

$$\sum_{i|i\in S\wedge j\in D} x_j - \sum_{k|k\in S\wedge j\in D} x_j = b_i, \text{ za } i=1,\ldots,m \quad (3)$$

$$\sum_{i\in N} b_i = 0 \quad (4)$$

gde je:

$F(x)$ - ciljna funkcija koju treba minimizirati i

x_j - promenljive odlučivanja (kapaciteti grana) koje treba odrediti.

4. Razvoj aplikacije

Formulisani problem minimizacije cene prenosa podataka u funkciji kapaciteta telekomunikacionih linkova pripada klasi problema najkraćeg puta za koji postoji više algoritama [10]. Jedan od najpoznatih je svakako Dijkstrin algoritam, decenijama popularan u telekomunkacijama, a i generalno u računarskim naukama i operacionim istraživanjima. Dijkstrin algoritam se relativno lako implementira imperativnim jezicima [11], ali ima i određena ograničenja koja nisu predmet ovog rada.

Naš cilj u ovom istraživanju je korišćenje deklarativnog programiranja za formulisani optimizacioni problem za nasumično generisanu telekomunikacionu mrežu. Pod nasumično generisanom telekomunikacionom mrežom podrazumeva se mreža kod koje se broj čvorova i grana, veze između čvorova i grana, gornja granica kapaciteta grana, protok čvorova i cene prenosa podataka generišu slučajno. Ovo je bilo neophodno jer cilj kreirane aplikacije nije optimizacija konkretne telekomunikacione mreže, već analiza samog pristupa u rešavanju opisanog optimizacionog problema na velikom broju različitih telekomunikacionih mreža.

Za razvoj aplikacije korišćen je optimizacioni paket *IBM ILOG CPLEX Optimization Studio* [12]. Ovo integrisano razvojno okruženje (*Integrated Development Environment* – IDE) omogućava kreiranje i modifikovanje modela koristeći deklarativni jezik OPL (*Optimization Programming Language)*, izvršavanje modela uz pomoć prisutnih optimizacionih mašina – solvera, kao i kombinovanje i upravljanje modelima koristeći proceduralni jezik *OPL Script*. Takođe, okruženje pruža podršku prikazivanju rezultata u tabelarnom i grafičkom obliku.

Razvoj aplikacije obuhvatao je sledeće korake:

- Generisanje slučajnog celog broja čvorova u intervalu od 5 do10.
- Generisanje slučajnog celog broja grana na intervalu između 10 i 25.
- Kreiranje niza petorki Grane sa poljima opisanim u Sekciji 3.
- Inicijalizacija prva 3 polja u nizu torki Grane, pri čemu se svakoj grani dodeljuje fiksna donja granica kapaciteta jednaka 0, promenljiva, slučajno generisana, gornja granica kapaciteta na intervalu od 1 do 20 i promenljiva, slučajno generisana, cena prenosa podataka na intervalu od 1 do 10.
- Formiranje i inicijalizacija matrice incidencije dimenzija brojCvorova x brojGrana čiji su svi elementi jednaki 0.
- Orjentacija grana dodavanjem vrednosti -1 i +1 u svaku kolonu matrice incidencije na slučajno odabranim pozicijama.
- Provera da li u matrici incidencije ne postoje petlje.

- Pronalazak čvorova koji isključivo šalju podatke, odnosno isključivo primaju podatke, vrši se pronalskom vrsta u matrici incidencije koje sadrže isključivo elemente 0 i +1, odnosno 0 i -1.
- Čvorovima koji isključivo šalju podatke dodeliti slučajno generisanu vrednost za protok u intervalu od 1 do 20, a čvorovima koji isključivo primaju podatke slučajno generisanu vrednost u intervalu od -20 do -1.
- Odrediti zbir protoka svih čvorova u mreži.
- Ukoliko je zbir protoka čvorova u mreži različit od nule, slučajno odabranom čvoru dodeliti protok takav da zbir protoka kroz sve čvorove mreže bude jednak nuli.
- Na osnovu matrice incidencije, dopuniti nedostajuća polja u nizu torki Grane, koja odgovaraju izvornom i odredišnom čvoru.
- Definisati funkciju cilja u vidu sume proizvoda kapaciteta grana i cene prenosa podataka.
- Postaviti ograničenja za svaki čvor u mreži na osnovu protoka mrežnog saobraćaja kroz čvorove.
- Postaviti ograničenja za kapacitet svake grane na osnovu donje i gornje granice kapaciteta.
- Kako matematički model, formulisan u sekciji 3, pripada linearnom programiranju, pokrenuti *IBM ILOG CPLEX solver* za izvršavanje kreiranog modela.
- Dobijene rezultate, minimalnu cenu prenosa i kapacitete grana, sačuvati u *Excel* radnoj svesci radi tabelarne i grafičke prezentacije.

Za nasumično generisanje parametara mreže korišćena je funkcija *srand*. Funkcija *srand* koristi kao "seme" za generator pseudoslučajnih brojeva trenutak pokretanja modela. Na ovoj način, omogućeno je dobjanje različitih slučajnih vrednosti prilikom svakog pokretanja softvera za generisanje mreže.

5. Rezultati i analiza rezultata

U cilju boljeg razumevanja, prvo ćemo posmatrati jednu „manju" nasumično generisanu telekomunikacionu mrežu koja se sastoji od 5 čvorova i 11 grana. U prozoru *Scripting log* integrisanog razvojnog okruženja *IBM ILOG CPLEX Optimization Studio* mogu se videti: matrica incidencije, petorke dodeljene svakoj od grana i informacije o protoku za svaki čvor u mreži. Dimenzije matrice incidencije su 5x11. U svakoj koloni matrice incidencije postoji tačno jedan element čija je vrednost +1 i jedan element čija je vrednost -1, dok ostali elementi u koloni imaju vrednost 0. Na osnovu matrice incidencije moguće je za svaku granu odrediti izvorni i odredišni čvor. Npr. ako se posmatra prva kolona matrice incidencije, koja odgovara grani a_1, vidi se da je izvorni čvor grane a_1 čvor N_5, dok je N_4 odredišni čvor. Svakoj od torki Grane dodeljeno je pet vrednosti, ranije opisanih u Sekciji 3. U *Scripting log* prozoru, kao što je prikazano na Slici 3, prikazuju se i poruke da čvorovi 1 i 3 imaju pozitivne protoke. Kao što se vidi iz matrice incidencije, u prvoj i trećoj vrsti, koje odgovaraju prvom i trećem čvoru, pojavljuju se samo članovi 0 i +1. Dakle, iz prvog čvora izlazi samo grana a_9, dok iz trećeg čvora izlaze grane a_5, a_8 i a_{10}. Kako nema ulaznih grana, očekuje se pozitivan protok kroz prvi i treći čvor.

Niz protokCvorova predstavlja slučajno generisane vrednosti za protoke čvorova. Prvom i trećem čvoru su dodeljene slučajno generisane pozitivne vrednosti za protok. Kako je zbir protoka prvog i trećeg čvora 24, potrebno je da se nekom od preostalih čvorova

dodeli protok čija je vrednost -24, kako bi ukupan protok kroz čvorove u mreži bio jednak 0. Iz niza protokCvorova se vidi da je to čvor broj 5.

```
Matrica incidencije:
    [[0 0 0 0 0 0 0 0 1 0 0]
     [0 0 0 1 0 1 0 0 -1 -1 0]
     [0 0 0 0 1 0 0 1 0 1 1]
     [-1 -1 1 -1 -1 -1 1 0 0 0 -1]
     [1 1 -1 0 0 0 -1 -1 0 0 0]]
Grane:
{<0 5 3 5 4> <0 5 2 5 4> <0 9 10 4 5>
 <0 14 9 4 2> <0 12 1 3 4> <0 18 3 2 4>
 <0 20 1 4 5> <0 5 8 3 5 > <0 8 8 1 2>
 < 0 1 9 3 2> < 0 15 8 3 4>}
Cvorovi sa pozitivanim protokom: 1 3
Protok cvorova: [5 0 19 0 -24]
```

Slika3. Sadržaj prozora Scripting log

```
// solution (optimal) with objective 142
//
x = [0 0 0 0 12 5 19 5 5 0 2];
```

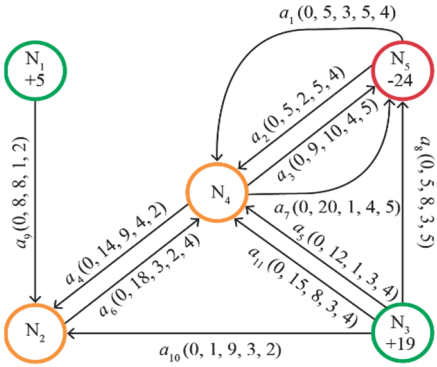
Slika 4. Sadržaj prozora Solutions

U okviru prozora *Solutions,* koji je prikazan na Slici 4, prikazano je optimalno rešenje, odnosno minimalna cena prenosa koja iznosi 142. Takođe, prikazan je i niz x čiji elementi predstavljaju izračunate vrednosti kapaciteta grana koje odgovaraju optimalnom rešenju, tj. to su nađene vrednosti promenljivih odlučivanja.

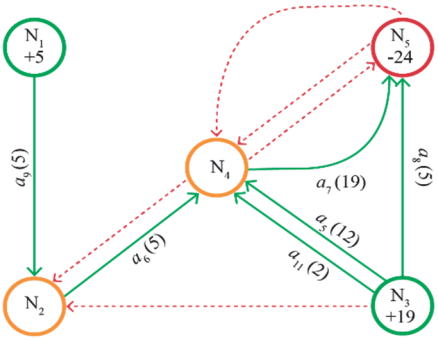
Na osnovu niza torki Grane, matrice incidencije i promenljive protokCvorova moguće je grafički predstaviti dobijenu telekomunikacionu mrežu koja je prikazana na Slici 5. Sa slike se može uočiti da postoji više grana sa istim izvornim i odredišnim čvorom, poput grana a_1 i a_2. Pri tome grane a_1 i a_2 imaju istu donju i gornju granicu kapaciteta ali različitu cenu. Prilikom kreiranja optimizacionog modela nije postavljeno ograničenje u vidu broja grana koje mogu imati isti izvorni i odredišni čvor. Kod telekomunikacionih mreža podaci između čvorova mogu da se prenose putem različitih linkova. Npr. za prenos podataka od čvora N_5 ka čvoru N_4 mogu se koristiti Wi-Fi link i LTE link. Koji link će biti korišćen za prenos zavisi od dostupnosti linka, kapaciteta linka, cene prenosa i sl. LWA (*LTE Wi-Fi Aggregation*) i LWIP (*LTE Wi-Fi radio-level integration with IP security tunnel*) su primeri tehnologija koje podržavaju deljeni spektar što se postiže agregacijom LTE i Wi-Fi nosioca, odnosno prebacivanjem dela saobraćaja na slobodan Wi-Fi link [13]. Ovo je razlog zbog koga u optimizacioni model nije uključeno pomenuto ograničenje.

Provera validnosti minimalne cene prenosa može se izvršiti zamenom dobijenih vrednosti kapaciteta grana u funkciju cilja: $F(x)=3x_1+2x_2+10x_3+9x_4+x_5+3x_6+x_7+8x_8+8x_9+9x_{10}+8x_{11}=3*0+2*0+10*0+9*0+1*12+3*5+1*19+8*5+8*5+9*0+8*2=12+15+19+40+40+16=142$. Iz niza x se vidi da je nađeni kapacitet grana a_1, a_2, a_3, a_4 i a_{10} jednak 0, odnosno da se za prenos podataka koriste grane a_5, a_6, a_7, a_8, a_9 i a_{11}.

Na Slici 6 prikazane su dobijene vrednosti kapaciteta grana. Isprekidanim linijama su označene grane koje se ne koriste za prenos, a punim one koje se koriste. Može uočiti da je za prenos podataka od čvora N_4 do čvora N_5 izabrana grana a_7, a ne grana a_3 zbog niže cene prenosa podataka. Takođe, vidi se da se za prenos podataka od čvora N_3 ka čvoru N_4 istovremeno koriste grane a_5 i a_{11}, pri čemu se granom a_5 prenosi veća količina podataka zbog niže cene. Kapacitet grane a_5 odgovara gornjoj granici kapaciteta (9), dok se manja količina saobraćaja realizuje i granom a_{11} po skupljoj ceni prenosa da bi se ispunila postavljena ograničenja.



Slika 5. Nasumično generisana telekomunikaciona mreža za koju se vrši optimizacija



Slika 6. Kapaciteti grana za mrežu sa slike 5 koji odgovaraju optimalnom rešenju

Lako se može proveriti da su zadovoljena i ograničenja koja se tiču kapaciteta grana i protoka čvorova, definisana formulama (2), (3) i (4), u matematičkom modelu.

U okviru prozora *Engine log*, *Statistics* i *Profiler* dostupne su informacije o korišćenoj metodi za rešavanje optimizacionog problema, vremenu izvršavanja, iskorišćenju memorije i drugi statistički podaci. Ukupno vreme izvršavanja ovog modela iznosi 0.452*s* pri čemu je najviše vremena utrošeno na učitavanje konfiguracije modela. Sa druge strane, najviše memorije utrošeno je za pronalazak optimalnog rešenja. Ukupan broj ograničenja je 16, što odgovara zbiru broja čvorova i broja grana.

Kako se brojni parametri mreže generišu slučajno, realno je moguća situacija da su neka od postavljenih ograničenja u konfliktu i da softver ne može da pronađe optimalno rešenje. Značajna funkcionalnost softvera je identifikacija konfliktnih ograničenja i pronalaženje relaksacije koja bi omogućila pronalazak optimalnog rešenja. Procesuiranje mreže sa konfliktom je duže i memorijski zahtevnije u poređenju sa mrežom bez konflikta istih dimenzija, s obzirom da pronalazak relaksacije troši dodatne resurse.

Tokom eksperimenata sa realizovanim optimizacionim softverom, nađeno je optimalno rešenje za veći broj nasumično generisanih mreža, a sumarni rezultati za 20 odabranih prikazani su u Tabeli 1. Ineresantno je zapaziti da se optimizacija ukoliko nema konfliktnih ograničenja izvršava vrlo brzo: prosečno vreme utrošeno na dostizanje optimalnog rešenja iznosi 0.021*s*, odnosno 5.73% od ukupnog CPU vremena izvršenja koda. Sa druge strane prosečna memorija utrošena na dostizanje optimalnog rešenja iznosi 2.497MB što iznosi čak 84.65% ukupnog memorijskog prostora.

Mreža broj 9 ima maksimalan broj čvorova i grana koje model dopušta, ona je dimenzija 10x25, i zaslužuje malo dublju analizu. Na Slici 7 prikazane su gornje granice kapaciteta i dobijene optimalne vrednosti kapaciteta linkova za ovu mrežu. Minimalna cena prenosa podataka iznosi 108 (vrednost ciljne funkcije), pri čemu se za prenos podataka koriste linkovi a_7, a_8, a_9, a_{10}, a_{14}, a_{20} i a_{25}. Očigledno, najveća količina podataka prenosi se putem linka a_{20}. U slučaju da link a_{20} nije u funkciji (zbog zauzeća ili prekida), nije moguće pronaći optimalno rešenje. Ukoliko smanjimo opterećenje grane a_{20} sa 9 na 7, softver nalazi optimalno rešenje sa vrednošću 148, što odgovara povećanju cene od čak 37.04%! Na Slici 8 prikazane su gornje granice kapaciteta i dobijene optimalne vrednosti kapaciteta linkova za mrežu 9 nakon smanjenja opterećenja linka a_{20}. Može se zapaziti da nakon ove

modifikacije grana a_5 postaje opterećena (i to maksimalno), a grane a_7 i a_8 poprimaju povećano opterećenje.

Tabela 1: Performanse odbranih mreža za koje je pronađeno optimalno rešenje

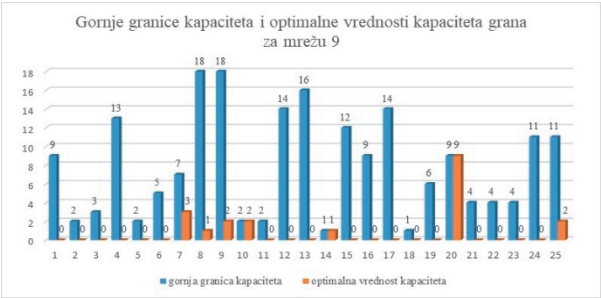
Mre-ža	Odre-đena min cena	Prosečni kapacitet	Prosečna gornja granica kapciteta	Vreme dostizanja optimalnog rešenja		Memorija utrošena na optimizaciju		Dimenzije mreže (*mxn*)
				(s)	(%)	(MB)	(%)	
1	96	2,74	10,74	0,014	6,06	2,475	97	6x22
2	16	1,12	12,24	0,045	10,47	2,945	93	7x16
3	218	4,85	14,69	0,076	15,90	2,428	84	10x12
4	51	1,36	12,43	0,030	9,17	3,042	95	7x13
5	211	3,06	9,33	0,035	10,03	3,109	83	5x15
6	78	1,60	10,36	0,037	8,53	2,585	98	10x24
7	81	2,17	10,21	0,006	2,24	1,845	72	7x11
8	269	3,38	12,50	0,015	4,36	2,265	86	7x15
9	108	1,73	8,54	0,019	6,11	2,808	97	10x25
10	33	1,42	9,58	0,012	5,24	1,940	77	6x11
11	79	2,55	9,36	0,011	4,14	2,859	92	5x10
12	77	2,46	9,62	0,008	2,20	1,975	73	9x12
13	36	1,47	9,53	0,004	1,02	2,720	86	9x18
14	100	2,11	9,42	0,015	3,88	2,472	78	6x18
15	180	2,93	10,20	0,014	3,52	2,296	76	10x14
16	204	4,42	12,25	0,013	4,44	2,269	82	8x11
17	87	1,67	10,52	0,014	4,86	2,358	74	7x20
18	48	2,67	9,83	0,015	4,00	2,183	77	8x11
19	44	1,95	9,50	0,017	4,70	2,767	89	8x18
20	207	3,35	7,80	0,016	3,71	2,605	84	7x19

6. Zaključak

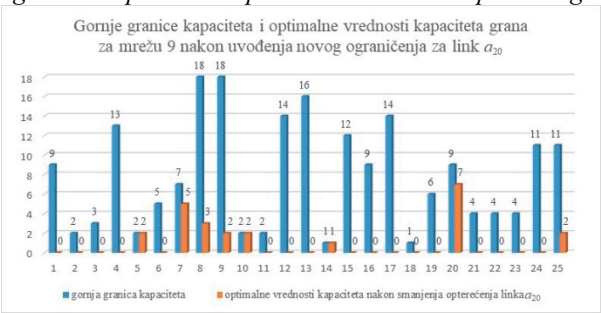
Deklarativno programiranje je odličan izbor za optimizacione probleme jer nudi niz pogodnosti: lako i kratko kodiranje, jednostavnost verifikacije, testiranja, održavanja, nadogradnje kao i distribuirane i inkrementalne implementacije softverskog sistema.

Kreirana aplikacija može se koristiti kao edukativni alat u cilju razumevanja problema minimizacije cene prenosa podataka u funkciji kapaciteta telekomunikacionih linkova. Naravno, možemo optimizirati i neku realnu telekomunikacionu mrežu.

U radu je prikazana prva verzija aplikacije. U daljem istraživanju moguće su brojne modifikacije i poboljšanja. Opsezi koji su korišćeni za inicijalizaciju parametara mreže, poput broja čvorova i grana, gornje granice kapaciteta telekomunikacionih linkova, odabrani su slučajno i moguća je njihova jednostavna promena. Skoro svi parametri mreže generišu se slučajno, na unapred definisanom intervalu, osim donje granice kapaciteta telekomunikacionih linkova koja je fiksna i jednaka 0. Fiksna vrednost donje granice kapaciteta je odabrana da bi se smanjio broj nasumično generisanih telekomunikacionih mreža bez optimalnog rešenja.



Slika 7. Gornje granice kapaciteta i optimalne vrednosti kapaciteta grana za mrežu 9



Slika 8. Gornje granice kapaciteta i optimalne vrednosti kapaciteta grana za mrežu 9 nakon smanjenja gornje granice opterećenja linka a_{20}

Trenutno, za oko 65% generisanih telekomunikacionih mreža softver nalazi optimalno rešenje, dok preostalih 35% zahteva relaksaciju zbog prisutnih konflikata. Kao što je prikazano u rezultatima, moguće je generisanje telekomunikacione mreže kod koje veći broj grana ima isti izvorni i odredišni čvor, ali je lako uvesti i drugačije ograničenje.

Zahvalnica

Ovaj rad delimično je podržan od strane Ministarstva prosvete, nauke i tehnološkog razvoja Republike Srbije, u okviru projekata 036012 i 032025.

Literatura

[1] H. Kim, *Desingn and Optimization for 5G Wireless Communications, Ney York, NY, USA,* John Wiley & Sons Inc., 2020.

[2] Y. Li, A. Pollastro, S. Napoli, "Simple network design and power allocation for 5G device-to-device communication", *2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, Athens, Greece, December 2014. DOI: 10.1109/CAMAD.2014.7033235

[3] M. Pioro, D. Medhi, *Routing, Flow, and Capacity Design in Communication and Computer Networks*, San Francisco, USA, Elsevier Inc, 2004.

[4] P. Van Roy and H. Seif, *Concepts, Techniques, and Models of Computer Programming*, MIT Press, Cambridge, MA, 2004.

[5] D. Austin, “What Are JavaScript Programming Paradigms?“ [Online]. Available at: https://javascript.plainenglish.io/what-are-javascript-programming-paradigms-3ef0f576dfdb.
[6] P. Van Roy, “A software system should be declarative except where it interacts with the real world“, *First Logic and Practice of Programming Workshop*, pp. 73-74, July, 2018. [Online]. Available at: https://arxiv.org/pdf/2008.07901.pdf#page=73
[7] J. Wikarek, P. Sitek, M. Jagodziński, “A declarative approach to shop orders optimization“, *Applied Computer Science*, vol. 15, no. 4, pp. 5–15, 2019. DOI: 10.23743/acs-2019-25
[8] S. Mladenović, S. Vesković, I. Branović, S. Janković and S. Aćimović, “Heuristic Based Real-Time Train Rescheduling System“, *Networks*, Vol. 67, Issue 1, pp. 32-48, 2016. DOI: 10.1002/net.21625
[9] A. Chi Zhou, B. He, X. Cheng, and C. Tong Lau, "A Declarative Optimization Engine for Resource Provisioning of Scientific Workflows in IaaS Clouds", *HPDC '15: Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing*, pp. 223–234, June 2015. DOI: 10.1145/2749246.2749251
[10] A. Sifaleras, “Minimum cost network flow: problems, algoritams and software”, *Yugoslav Journal of Operation Research*, 2013, 3-17, DOI:10.2298/YJOR121120001S
[11] “Dijkstra's Algorithm in C++ | Shortest Path Algorithm”, [Online]. Available at: https://favtutor.com/blogs/dijkstras-algorithm-cpp
[12] IBM ILOG CPLEX Optimization Studio CPLEX User’s Manual, Version 12, Release 8, 2017. [Online]. Available at: https://www.ibm.com/docs/en/SSSA5P_12.8.0/ilog.odms.studio.help/pdf/opl_languser.pdf
[13] C. B. Papadias, T. Ratnarajah, D. T.M. Slock, *Spectrum Sharing The Next Frontier in Wireless Networks*. New York, NY, USA, John Wiley & Sons, Inc., 2020.

Abstract: *With the rapid development of wireless communication systems, there is an increasing need to develop software systems to solve the problem of minimizing costs, delays, losses and maximizing speed, etc. This paper deals the minimume cost network flow problem. In order to solve this problem, first a mathematical model was formulated, and then an appropriate application was designed and implemented using declarative programming language. It is known that declarative programming is a good choice for optimization problems, because it is enough to specify the relations that must be satisfied, without specifying an effective procedure for finding the value of decision variables. For the needs of application testing, auxiliary software has been developed that randomly generates a telecommunication network with given characteristics. This allows us to test the application on arbitrary number of different telecommunications networks, with different numbers of nodes and links. This paper presents and analyzes in detail the results of optimization for several randomly generated telecommunication networks.*

Keywords: *declarative programming, optimization, data transmission cost, capacity of telecommunication links, network flow*

USING OF DECLARATIVE PROGRAMMING FOR DEVELOPMENT OF APPLICATION FOR MINIMUM COST NETWORK FLOW PROBLEM

Snežana Mladenović, Ivana Stefanović, Stefan Zdravković, Slađana Janković