Univerzitet u Sarajevu
Elektrotehnički fakultet u Sarajevu
Odsjek za računarstvo i informatiku

Zadaća 5
Naziv zadaće

Osnove operacionih istraživanja

Ime i prezime: Elma Šeremet
Broj indexa: 18318
Grupa: 9
Datum: 15.01.2021.

Zadatak №

Podzadatak №

Lista:

- Prvi
- Drugi

Enumerirana lista [1]:

1. Prvi
2. Drugi

Slika 1:

Slika 1. Opis slike

[1] Fusnota

Matematske formule (primjer):

$$\begin{aligned} &\arg\max \mathrm{Z}(\mathbf{x}) = \mathbf{c}^{\mathbf{T}}\mathbf{x} \\ &\text{p.o.} \\ &\mathbf{A}\mathbf{x} \leq \mathbf{b} \\ &\mathbf{x} \geq 0 \end{aligned} \tag{1}$$

Primjer kôda (prilagođeno za Python) 1:

```
import numpy as np

def incmatrix(genl1,genl2):
        m = len(genl1)
        n = len(genl2)
        M = None
        VT = np.zeros((n*m,1), int)

        #compute the bitwise xor matrix
        M1 = bitxormatrix(genl1)
        M2 = np.triu(bitxormatrix(genl2),1)

        for i in range(m-1):
                for j in range(i+1, m):
                        [r,c] = np.where(M2 == M1[i,j])
                        for k in range(len(r)):
                                VT[(i)*n + r[k]] = 1;
                                VT[(i)*n + c[k]] = 1;
                                VT[(j)*n + r[k]] = 1;
                                VT[(j)*n + c[k]] = 1;

                                if M is None:
                                        M = np.copy(VT)
                                else:
                                        M = np.concatenate((M,
    VT), 1)

                                VT = np.zeros((n*m,1), int)

        return M
```

Listing 1. Python kôd

Pseudokôd/Algoritam 1:

Algoritam 1 Pseudokôd algoritma tabu pretraživanja

```
1: x ← x0
2: v ← f(x)
3: T ← ∅
4: repeat
5:    Ω′ ← ∅
6:    repeat
7:       Ñ(x, δ) = N(x, δ)\T
8:       izabrati x' ∈ Ñ(x, δ)
9:       v′ ← f(x')
10:      if v′ρv then
11:         uvrstiti v′ u Ω
12:      end if
13:   until ZavrsenoPretrazivanjeOkoline(x)
14:   x ← IzborNovogRjesenja(Ω′)
15:   T ← AzuriranjeTabuListe(x, Ω′)
16: until UslovZaustavljanja()
```

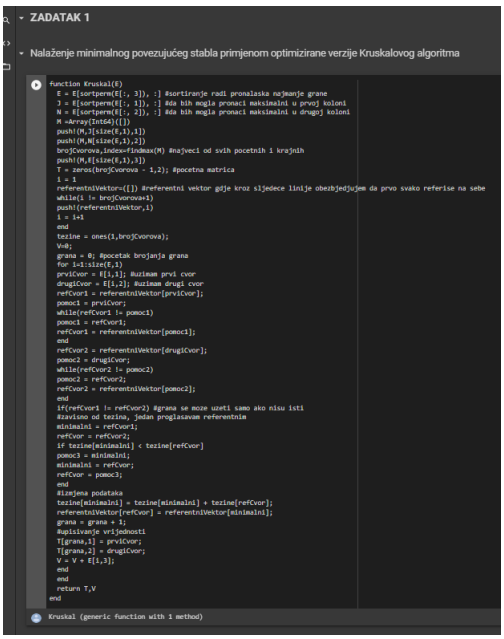
Tabela 1:

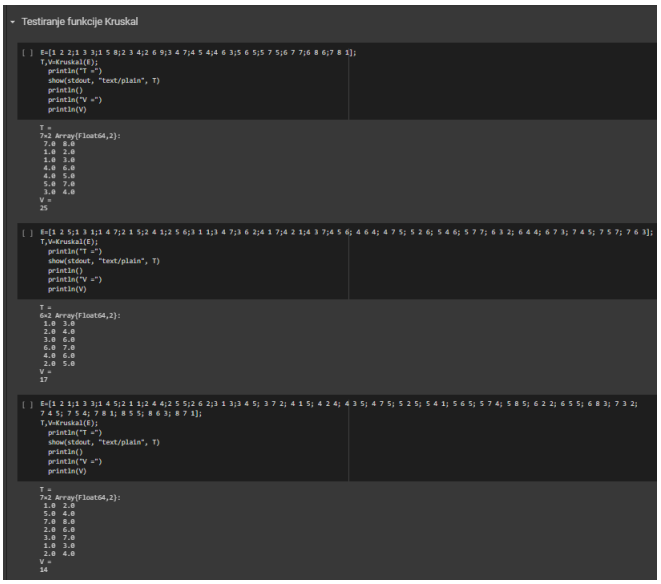
Baza	b_i	y_1	y_2	y_3	y_4	y_5	y_6
y_4	1	2	1	0	1	0	0
y_5	1	1	3	0	0	1	0
y_3	1/3	0	1/3	1	0	0	1/3
	–1/3	1	2/3	0	0	0	–1/3

Tabela 1. Primjer tabele

Zadatak 1

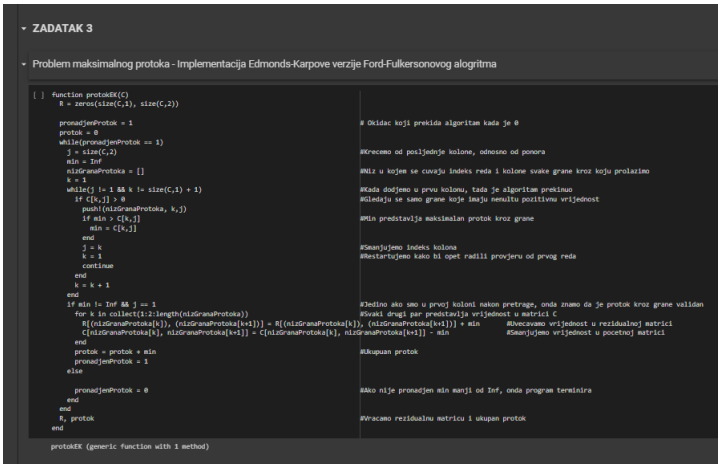
Vaš zadatak je da napravite funkciju za nalaženje minimalnog povezujućeg stabla primjenom optimizirane verzije Kruskalovog algoritma:





Zadatak 3

Vaš zadatak je da napravite funkciju za rješavanje problema maksimalnog protoka, ali ovaj put bez korištenja paketa JuMP i GLPK, nego ručnom implementacijom Edmonds-Karpove verzije Ford-Fulkersonovog algoritma, vršeći direktne manipulacije nad rezidualnom matricom (u skladu sa uputama datim na predavanjima). Funkcija treba da se zove protokEK, a imaće istu sintaksu kao i funkcija protokLP implementirana u prethodnom zadatku.



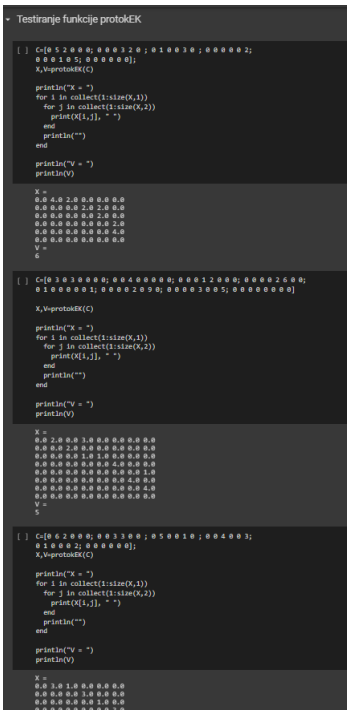
```
ZADATAK 3
Problem maksimalnog protoka - Implementacija Edmonds-Karpove verzije Ford-Fulkersonovog algoritma

[ ]  function protokEK(C)
       R = zeros(size(C,1), size(C,2))

       pronadjenProtok = 1                                  # Okidac koji prekida algoritam kada je 0
       protok = 0
       while(pronadjenProtok == 1)
         j = size(C,2)                                      #Krecemo od posljednje kolone, odnosno od ponora
         min = Inf
         nizGranaProtoka = []                               #Niz u kojem se cuvaju indeks reda i kolone svake grane kroz koju prolazimo
         k = 1
         while(j != 1 && k != size(C,1) + 1)                #Kada dođemo u prvu kolonu, tada je algoritam prekinuo
           if C[k,j] > 0                                    #Gledaju se samo grane koje imaju nenultu pozitivnu vrijednost
             push!(nizGranaProtoka, k,j)
             if min > C[k,j]                                #Min predstavlja maksimalan protok kroz grane
               min = C[k,j]
             end
             j = k                                          #Smanjujemo indeks kolona
             k = 1                                          #Restartujemo kako bi opet radili provjeru od prvog reda
             continue
           end
           k = k + 1
         end
         if min != Inf && j == 1                            #Jedino ako smo u prvoj koloni nakon pretrage, onda znamo da je protok kroz grane validan
           for k in collect(1:2:length(nizGranaProtoka))    #Svaki drugi par predstavlja vrijednost u matrici C
             R[(nizGranaProtoka[k]), (nizGranaProtoka[k+1])] = R[(nizGranaProtoka[k]), (nizGranaProtoka[k+1])] + min      #Uvecavamo vrijednost u rezidualnoj matrici
             C[nizGranaProtoka[k], nizGranaProtoka[k+1]] = C[nizGranaProtoka[k], nizGranaProtoka[k+1]] - min            #Smanjujemo vrijednost u pocetnoj matrici
           end
           protok = protok + min                            #Ukupuan protok
           pronadjenProtok = 1
         else

           pronadjenProtok = 0                              #Ako nije pronadjen min manji od Inf, onda program terminira
         end
       end
       R, protok                                            #Vracamo rezidualnu matricu i ukupan protok
     end

     protokEK (generic function with 1 method)
```



```
Testiranje funkcije protokEK

[ ]  C=[0 5 2 0 0 0; 0 0 0 3 2 0 ; 0 1 0 0 3 0 ; 0 0 0 0 0 2;
       0 0 0 1 0 5; 0 0 0 0 0 0];
     X,V=protokEK(C)

     println("X = ")
     for i in collect(1:size(X,1))
       for j in collect(1:size(X,2))
         print(X[i,j], " ")
       end
       println("")
     end

     println("V = ")
     println(V)

     X =
     0.0 4.0 2.0 0.0 0.0 0.0
     0.0 0.0 0.0 2.0 2.0 0.0
     0.0 0.0 0.0 0.0 2.0 0.0
     0.0 0.0 0.0 0.0 0.0 2.0
     0.0 0.0 0.0 0.0 0.0 4.0
     0.0 0.0 0.0 0.0 0.0 0.0
     V =
     6

[ ]  C=[0 3 0 3 0 0 0 0; 0 0 4 0 0 0 0 0; 0 0 0 1 2 0 0 0; 0 0 0 0 2 6 0 0;
       0 1 0 0 0 0 0 1; 0 0 0 0 2 0 9 0; 0 0 0 0 3 0 0 5; 0 0 0 0 0 0 0 0]

     X,V=protokEK(C)

     println("X = ")
     for i in collect(1:size(X,1))
       for j in collect(1:size(X,2))
         print(X[i,j], " ")
       end
       println("")
     end

     println("V = ")
     println(V)

     X =
     0.0 2.0 0.0 3.0 0.0 0.0 0.0 0.0
     0.0 0.0 2.0 0.0 0.0 0.0 0.0 0.0
     0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0
     0.0 0.0 0.0 0.0 0.0 4.0 0.0 0.0
     0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0
     0.0 0.0 0.0 0.0 0.0 0.0 4.0 0.0
     0.0 0.0 0.0 0.0 0.0 0.0 0.0 4.0
     0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
     V =
     5

[ ]  C=[0 6 2 0 0 0; 0 0 3 3 0 0 ; 0 5 0 0 1 0 ; 0 0 4 0 0 3;
       0 1 0 0 0 2; 0 0 0 0 0 0];
     X,V=protokEK(C)

     println("X = ")
     for i in collect(1:size(X,1))
       for j in collect(1:size(X,2))
         print(X[i,j], " ")
       end
       println("")
     end

     println("V = ")
     println(V)

     X =
     0.0 3.0 1.0 0.0 0.0 0.0
     0.0 0.0 0.0 3.0 0.0 0.0
     0.0 0.0 0.0 0.0 1.0 0.0
```

Zadatak 4

Napravite izvještaj u kojem ćete opisati kako ste implementirali sve one funkcije koje ste uradili u prethodna 3 zadatka (naravno, za one funkcije koje su urađene), kao i rezultate testiranja ovih funkcija na bar tri problema (po svakoj funkciji) za koje znate rješenje. Izvještaj treba biti u .pdf formatu.

Način implementacije je objašnjen u komentarima, dok su u prethodnim zadacima također ostavljene slike testova.

Zadatak 5

U sljedećoj tablici dati su podaci o svim aktivnostima nekog projekta, njihovim logičkim međuzavisnostima, kao i procjene trajanja pojedinih aktivnosti:

Aktivnost	Preduvjeti	Optimističko trajanje	Najvjerovatnije trajanje	Pesimističko trajanje
A	-	26	27	27
B	A	29	32	34
C	-	29	38	39
D	A, C	28	29	43
E	-	38	40	43
F	C, E	33	37	39
G	A	23	23	27
H	B, D, F	28	31	33

Formirajte mrežni dijagram projekta i pomoću PERT /TIME metoda odredite očekivana vremena početka i završetka svakog od događaja, najranije i nakasnije početke i završetke za sve aktivnosti, očekivane vremenske rezerve za sve aktivnosti, te vrijeme u kojem se projekat može završiti sa vjerovatnoćama od 25 %, zatim 75 % i skoro sigurno. Rezultate provedene analize prikažite u tabelarnoj formi. U svim slučajevima pretpostavite da će očekivano kritični put zaista biti kritičan.

Prvi korak u analizi je da se za svaku aktivnost odredi očekivano vrijeme i varijansa, što je urađeno u tabeli datoj u nastavku, prije čega je dat način računanja:

$$\text{Očekivano vrijeme} = \frac{\text{OptimistickoTrajanje}+4\cdot\text{NajvjerovatnijeTrajanje}+\text{PesimistickoTrajanje}}{6}$$

$$\text{Varijansa}^2 = \left(\frac{\text{OptimistickoTrajanje}-\text{PesimistickoTrajanje}}{6}\right)^2$$

Aktivnost	Preduvjeti	Očekivano vrijeme	Varijansa
A	-	$\frac{161}{6}$	$\frac{1}{36}$
B	A	$\frac{191}{6}$	$\frac{25}{36}$
C	-	$\frac{110}{3}$	$\frac{25}{9}$
D	A, C	$\frac{187}{6}$	$\frac{25}{4}$
E	-	$\frac{241}{6}$	$\frac{25}{36}$
F	C, E	$\frac{110}{3}$	1
G	A	$\frac{71}{3}$	$\frac{4}{9}$
H	B, D, F	$\frac{185}{6}$	$\frac{25}{36}$

Na osnovu ove tabele, prvo vršimo analizu strukture, tj. sastavljamo mrežni dijagram:

1
0
0
A
161/6
C
110/3
E
241/6
2
164/6
230/6
3
110/3
234/6
4
241/6
241/6
5
503/6
546/6
6
461/6
461/6
7
546/6
546/6
B
191/6
D
187/6
F
110/3
G
71/3
H
185/6

Kritični put je 1 – 4 – 6 – 7 odnosno E – F – H jer je to put u kojem zbir težina grana daje najveći mogući rezultat.

Događaj	$(T_e)_i$	$(T_l)_i$	Rezerve
1	0	0	0
2	161/6	270/6	109/6
3	110/3	274/6	54/6
4	241/6	241/6	0
5	303/6	646/6	343/6
6	461/6	461/6	0
7	646/6	646/6	0

Iz provedene analize zaključujemo da očekivano trajanje projekta iznosi $(T_e)_7 = \frac{646}{6} = 107.66$ vremenskih jedinica.

Što se tiče varijanse kritičnog puta, ona iznosi:

$\sigma = E^2 + F^2 + H^2 = \frac{25}{4} + 1 + \frac{25}{36} = \frac{143}{18} = 7.944.$

Standardna devijacija:

$\sqrt{7.944} = 2.81851.$

$(T_s)_7 = (T_e)_7 + \sigma \text{ x } \varphi^{-1}(0.25) = 107.66 + 2.81851 \text{ x } (-0.6745) = 105.758915005$
$(T_s)_7 = (T_e)_7 + \sigma \text{ x } \varphi^{-1}(0.75) = 107.66 + 2.81851 \text{ x } 0.6745 = 109.561084995$
$(T_s)_7 = (T_e)_7 + \sigma \text{ x } \varphi^{-1}(0.9987) = 107.66 + 2.81851 \text{ x } 3 = 116.11553$

Ukoliko želimo biti skoro sigurni da će ovaj projekat biti završen, potrebno je dati rok od 116 vremenskih jedinica. Za vjerovatnoću od 25% je potrebno dati rok od 106, a za 75% rok od 110 vremenskih jedinica.

Zadatak 6

U sljedećoj tablici dati su podaci o svim aktivnostima nekog projekta, njihovim logičkim međuzavisnostima, kao i procjene trajanja i troškova pojedinih aktivnosti (za svaku aktivnost data su normalne i usiljene vrijednosti trajanja i troškova).

Aktivnost	Preduvjeti	$t^{(n)}$	$t^{(u)}$	$c^{(n)}$	$c^{(n)}$
A	-	5	4	6	10
B	A	8	3	15	45
C	A	10	5	3	33
D	B	4	3	9	12
E	C	8	5	5	20
F	C, D	10	5	8	33
G	E	9	6	3	18

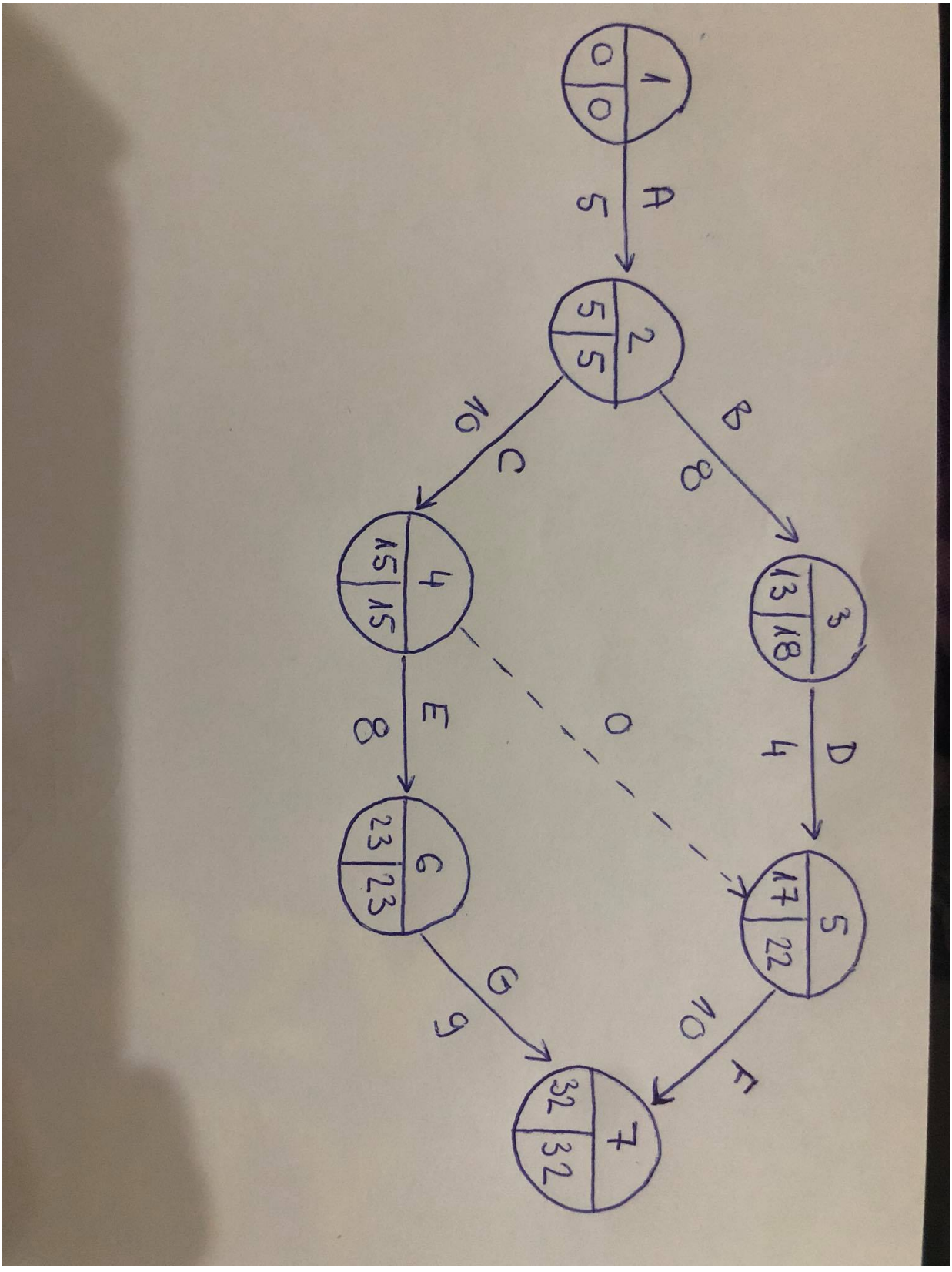
Vaš zadatak je da:

a. Izvršite analizu strukture i analizu vremena prema CPM metodi za ovaj projekat uz najekonomičniju realizaciju. Na mrežnom dijagramu treba da se vide najranija i najkasnija vremena započinjanja svih događaja, te kritični put. Potrebno je također izračunati i sve tipove vremenskih rezervi za sve aktivnosti (rezultate prikazati tabelarno), kao i cijenu projekta pri najekonomičnijoj realizaciji. [0.4 poena]

Cijena projekta pri najekonomičnijoj realizaciji je suma koja je jednaka sumi troškova prilikom normalnih trajanja aktivnosti. To predstavlja suma u našoj trećoj koloni u tabeli (pod pretpostavkom da je jedinica n.j.):

$6 + 15 + 3 + 9 + 5 + 8 + 3 = 49.$

Mrežni dijagram je dat u nastavku:


1
0
0
A
5
2
5
5
C
10
B
8
4
15
15
3
13
18
E
8
D
4
0
6
23
23
5
17
22
G
9
F
10
7
32
32

Kritični put je 1-2-4-6-7.
Normalno trajanje projekta izračunamo sabirajući vrijednosti iz grana koje su ušle u kritični put što iznosi 32 vremenske jedinice.

U tabeli su prikazane vremenske rezerve (R_t - ukupna vremenska rezerva, R_s - slobodna vremenska rezerva, R_n - nezavisna vremenska rezerva):

Aktivnost	i-j	$t_{i,j}$	t_i^0	t_j^0	t_i^1	t_j^1	$R_{ti,j}$	$R_{si,j}$	$R_{ni,j}$
A	1 - 2	5	0	5	0	5	0	0	0
B	2 - 3	8	5	13	5	18	5	0	0
C	2 - 4	10	5	15	5	15	0	0	0
D	3 - 5	4	13	17	18	22	5	0	-5
E	4 - 6	8	15	23	15	23	0	0	0
Fiktivna	4 - 5	0	15	17	15	22	7	2	2
F	5 - 7	10	17	32	22	32	5	5	0
G	6 - 7	9	23	32	23	32	0	0	0

b. Odredite plan realizacije projekta sa minimalnim trajanjem, uz najmanje moguće poskupljenje projekta. Obavezno naglasite koliko će iznositi novi troškovi projekta. [0.4 poena]

Odrediti ću plan realizacije projekta sa minimalnim trajanjem, uz najmanje moguće poskupljenje projekta koriteći tabele.

Aktivnost	$t_{i,j}(n)$	$t_{i,j}(u)$	$\alpha_{i,j}$	Iteracija	
				I	II
A	5	4	4	5	5
B	8	3	6	8	8
C	10	5	6	10	5
D	4	3	3	4	4
E	8	5	5	8	8
F	10	5	5	10	10
G	9	6	5	9	9

Vidimo da se više ne može poboljšati kritični put, shodno tome on ostaje i jedini kritični put. Također vidimo da smo dobili minimalno trajanje koje iznosi 27 vremenskih jedinica. Ovo predstavlja minimalno trajanje projekta.
c. Provjerite rezultat pod b. tako što ćete problem modelirati koristeći modele linearnog programiranja i rješavanjem odgovarajućih problema uz pomoć JuMP i GLPK paketa u Juliji. Obavezno naznačite šta su bili ulazni a šta izlazni podaci prilikom korištenja ove funkcije. [0.3 poena]

Što se tiče drugog zadatka, komentari i kod su na colabu.