

Zadaća 4

Student: Šejla Pljakić

Indeks: 17751

Prva četiri zadatka su rađena sa Venesom Šeremet

Zadatak 4

1)

Implementirana je funkcija za nalaženja minimalnog povezujućeg stabla primjenom optimizirane verzije Kruskalovog algoritma ($O(m \log_2 m)$). Kao parametar funkcija prima matricu E (matrica grana – elementi početni i krajni čvor te težina grane), a vraća matrice T (grane minimalnog povezujućeg stabla) i V (težina minimalnog povezujućeg stabla).

Prvo vršimo sortiranje grana sa funkcijom `sortrows` – grane su sortirane po trećem parametru matrice E u rastućem poretku.

Uvedena je pomoćna matrica koja u prvom redu čuva referentneč, a u drugom redu težine čvorova. Indeks kolone matrice predstavlja čvor čiji su referentni čvor i potencijal dati u matrici. Na početku je referentni čvor nula, a težine 1 za svaki čvor.

Formiramo stablo unutar petlje koja ide sve dok nije uzeto $n-1$ grana. Prvo što radimo jeste traženje korijenskih čvorova p i q . Nakon toga provjeravamo da li granu smijemo uzeti tj. da li su različiti korijenski čvorovi. Ako su korijenski čvorovi različiti razmatramo sljedeće slučajeve:

1. U slučaju da je $p==0$ čvoru q stavljamo p za referentni čvor, a težinu čvora p povećavamo za težinu čvora q .
2. Manju težinu ima čvor p – Čvoru p stavljamo q za referentni čvor, a težinu čvora q povećavamo za težinu čvora p .
3. Manju težinu ima čvor q – Čvoru q stavljamo p za referentni čvor, a težinu čvora p povećavamo za težinu čvora q .

Primjer 1:

```
>> E=[1,2,2;1,3,3;1,5,8;2,3,4;2,6,9;3,4,7;4,5,4;4,6,3;5,6,5;5,7,5;6,7,7;6,8,6;7,8,1];  
>> [T,V]=Kruskal(E);  
>>  
>> T  
  
T =  
  
    7    8  
    1    2  
    1    3  
    4    6  
    4    5  
    5    7  
    3    4  
  
>> V  
  
V =  
  
    25
```

Primjer 2:

```
>> E = [1,2,3; 1,3,7; 1,4,3; 1,5,2; 2,3,6; 2,4,8; 2,5,5; 3,4,10;4,5,4]
```

```
E =
```

1	2	3
1	3	7
1	4	3
1	5	2
2	3	6
2	4	8
2	5	5
3	4	10
4	5	4

```
>> [T,V]=Kruskal(E);
```

```
>> T
```

```
T =
```

1	5
1	2
1	4
2	3

```
>> V
```

```
V =
```

```
14
```

Primjer 3:

```
>> E = [1,2,5; 1,3,30; 2,3,20; 2,4,10;3,4,10;3,5,15; 4,5,5; 4,6,20; 5,6,15]
```

```
E =
```

1	2	5
1	3	30
2	3	20
2	4	10
3	4	10
3	5	15
4	5	5
4	6	20
5	6	15

```
>> [T,V]=Kruskal(E);
```

```
>> T
```

```
T =
```

1	2
4	5
2	4
3	4
5	6

```
>> V
```

```
V =
```

```
45
```

Primjer 4.

```
>> E = [1,2,3; 1,3,10; 1,4,2; 1,5,4; 2,5,7; 3,4,12; 3,6,15; 4,5,6; 4,6,4; 5,6,5; 5,7,2; 6,7,3]

E =

     1     2     3
     1     3    10
     1     4     2
     1     5     4
     2     5     7
     3     4    12
     3     6    15
     4     5     6
     4     6     4
     5     6     5
     5     7     2
     6     7     3

>> [T,V]=Kruskal(E);
>> T

T =

     1     4
     5     7
     1     2
     6     7
     1     5
     1     3

>> V

V =

    24
```

Primjer 2,3,4 au primjeri iz ZSR.

3)

Napravljena je funkcija za rješavanje problema maksimalnog protoka, ali ovaj put bez korištenja funkcije `linprog`, nego ručnom implementacijom Edmonds-Karpove verzije Ford-Fulkersonovog algoritma, vršeći direktne manipulacije nad rezidualnom matricom (u skladu sa uputama datim na predavanjima).

Funkcija `protokEK` prima matricu `C`. Na početku sačuvamo početnu matricu u promjenjivu `pocetnaMAtrica` jer ćemo dalje manipulacije raditi nad matricom `C`, a na kraju algoritma, biće nam potrebni ti podaci.

Implementirana je pomoćna funkcija `nadjiProtok` koja nam pomaže da nađemo povećavajući lanac. Unutar fukcije pravimo listu koja nam govori za svaki cvor koliko je udaljen od početka i sa kojeg cvora je upuceno na taj cvor. Te podatke smještamo u pomoćnu listu.

Takođe smo implementirali pomoćnu funkciju `promjenaMatrice` koja prima matricu `C` i pomocnu listu koju dobijamo koristeći funkciju `nadjiProtok`. Ova funkcija vraća parametre `min`, da i matrica, parametar `min` nam govori za koliko je povećan protok u toj iteraciji, parametar `da` vraća vrijednost

nula ukoliko nije pronađen povećavajući lanac, a 1 ukoliko jeste, dok parametar matrica vraća izmijenjenu matricu. Matricu mijenjamo tako što nađemo povećavajući lanac (i zapamtimo u listu polja koja se nalaze u tom lancu) te nakon toga smanjimo vrijednosti polja koja se nalaze u tom lancu za vrijednost najmanjeg elementa (tog lanca), a na simetrična polja dodamo tu vrijednost.

Vrijednost najmanjeg elementa u povećavajućem lancu dodjeljujemo promjenjivoj min koja će nam služiti da izračunamo optimalnu vrijednost ukupnog protoka.

Unutar funkcije protokEK u petlji pozivamo promjenaMatrice (i nadjiProtok koja se šalje f-ji promjenaMatrice kao parametar), petlja se prekida kad se ne nađe povećavajući lanac. Usput se u petlji računa optimalna vrijednost ukupnog protoka tako što se sabiraju sve vrijednosti za koje se vrši povećanje lanca (nalazi se u prom. min).

Primjer 1:

(Prvi primjer sa predavanja)

```
>> 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)
```

X =

0	2	0	3	0	0	0	0
0	0	2	0	0	0	0	0
0	0	0	1	1	0	0	0
0	0	0	0	0	4	0	0
0	0	0	0	0	0	0	1
0	0	0	0	0	0	4	0
0	0	0	0	0	0	0	4
0	0	0	0	0	0	0	0

V =

5

Primjer 2:

(Prvi primjer sa ZSR)

```
>> C=[0 10 10 0 0 0; 0 0 2 4 5 0; 0 0 0 0 9 0; 0 0 0 0 0 10;  
0 0 0 6 0 10; 0 0 0 0 0 0];  
>> [X,V]=protokEK(C)
```

X =

0	9	9	0	0	0
0	0	0	4	5	0
0	0	0	0	9	0
0	0	0	0	0	8
0	0	0	4	0	10
0	0	0	0	0	0

V =

18

Activate

Primjer 3:

(Drugi primjer sa ZSR)

```
>> 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)
```

X =

0	4	2	0	0	0
0	0	0	2	2	0
0	0	0	0	2	0
0	0	0	0	0	2
0	0	0	0	0	4
0	0	0	0	0	0

V =

6

6 11

Primjer 4:

(Treći primjer sa ZSR)

```
>> 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)
```

X =

```
0    3    1    0    0    0  
0    0    5    3    0    0  
0    3    0    0    1    0  
0    0    0    0    0    3  
0    0    0    0    0    1  
0    0    0    0    0    0
```

V =

```
4
```

```
A = 1 2 3 4 5 6
```

Zadatak 5

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

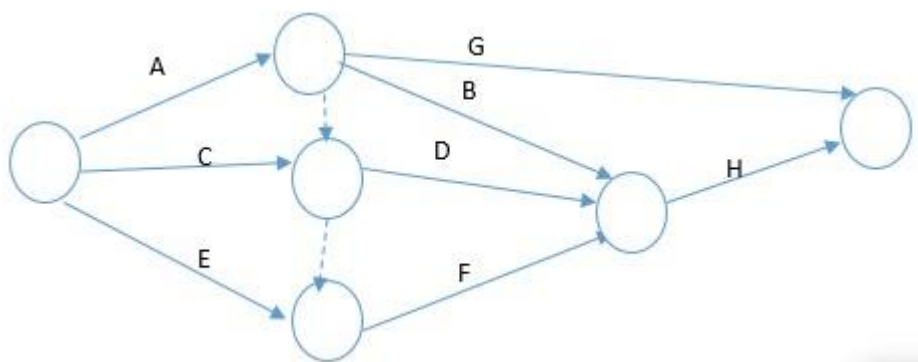
Aktivnost	Preduvjeti	Optimističko trajanje	Najvjerovatnije trajanje	Pesimističko trajanje
A	-	27	32	34
B	A	22	29	32
C	-	24	30	33
D	A, C	45	45	49
E	-	38	46	48
F	C, E	49	61	61
G	A	34	35	38
H	B, D, F	32	43	44

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.

Izrada:

Formirajmo mrežni dijagram. Crtanje uvijek započinje sa čvorom koji označava početak projekta, odnosno početak onih aktivnosti koje nisu uvjetovane ni jednom prethodnom aktivnošću. To su u

našem slučaju aktivnosti A, C i E. Aktivnosti ćemo predstaviti granama grafa. Svakoj grani pridružiti ćemo jednu i samo jednu usmjerenu granu. Svaka grana se završava čvorom. Pošto se ucrtaju grane koje polaze iz početnog čvora, u tabeli se traži aktivnost čiji je početak uvjetovan završetkom samo onih aktivnosti koje su već predstavljene na grafu a ne i aktivnosti koje još nisu nacrtane. Takve su aktivnosti B, D, F i G. Pri dodavanju ovih aktivnosti grafu može se desiti da je potrebno uvesti fiktivne čvorove. U slučaju kada neka aktivnost ovisi od završetka više drugih aktivnosti, završetke svih tih aktivnosti možemo objediniti u jedan događaj samo ukoliko nema aktivnosti koje ovise od završetka samo nekih a ne svih aktivnosti iz te skupine. U suprotnom će nam biti potrebne fiktivne aktivnosti. Postupak se nastavlja dok se ne nacrtaju sve aktivnosti projekta. Tako se za projekat iz razmatranog primjera dobija dijagram kao na sljedećoj slici:



Odredimo sada pomoću PERT/TIME metode očekivano vrijeme početka i završetka svakog od događaja.

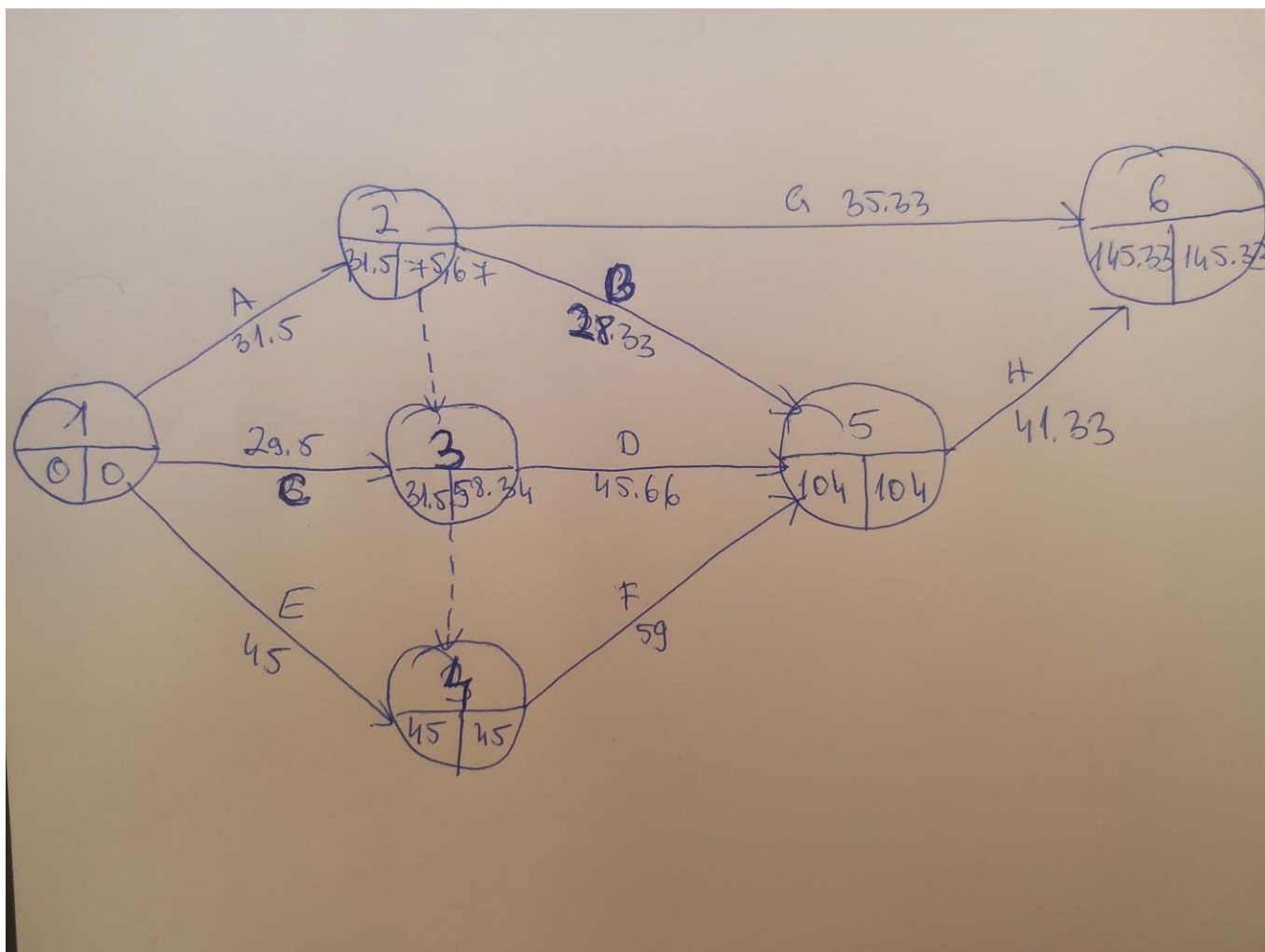
Za svaku aktivnost odredimo očekivano vrijeme i varijansu (urađeno u sljedećoj tabeli) po sljedećim formulama:

$$(te)_{i,j} = \frac{a_{i,j} + 4m_{i,j} + b_{i,j}}{6}$$

$$\sigma_{i,j}^2 = \left(\frac{a_{i,j} - b_{i,j}}{6} \right)^2$$

Aktivnost	Preduvjeti	Očekivano vrijeme	Varijansa
A	-	31.5	1.36
B	A	28.33	2.777
C	-	29.5	2.25
D	A, C	45.66	0.444
E	-	45	2.777
F	C, E	59	4
G	A	35.33	0.444
H	B, D, F	41.33	4

Sada ćemo upisati na dijagram očekivano vrijeme te izračunati najraniji početak i najkasniji početak za sve događaje.



Kritični put je: E-F-H. Kritični događaji su događaji 1,4,5 i 6.

Rezultati analize vremena nastupanja događaja zajedno sa odgovarajućim rezervama prikazani su u sljedećoj tabeli:

Događaj(i)	Najraniji početak (T_E) _i	Najkasniji početak (T_L) _i	Vremenska rezerva R_i
1	0	0	0
2	31.5	75.67	44.17
3	31.5	58.34	26.84
4	45	45	0
5	104	104	0
6	145.33	145.33	0

Izračunajmo sada vrijeme u kojem se projekat može završiti sa vjerovatnoćama od 25%, zatim 75% i skoro sigurno.

Na osnovu provedene analize vidimo da je očekivano trajanje projekta 145.33 vremenskih jedinica. Procjenu ćemo izvršiti na najprostiji način, ignorirajući moguće subkritične puteve koji bi mogli postati kritični.

Na kritičnom putu leže aktivnosti E, F i H pa varijansa kriričnog puta iznosi:

$$\sigma^2 = \sigma_E^2 + \sigma_F^2 + \sigma_H^2 = 2.777 + 4 + 4 = 10.777$$

Odgovarajuća standardna devijacija je:

$$\sigma = \sqrt{10.777} = 3.283$$

Za vjerovatnoću 25% dobijamo procjenu trajanja projekta:

$$(T_S)_6 = (T_E)_6 + \sigma \Phi^{-1}(0.25) \approx 145.33 - 3.283 \cdot 0.6745 \approx 143.115 \text{ (oko 143 vremenske jedinice)}$$

Za vjerovatnoću 75% dobijamo procjenu trajanja projekta:

$$(T_S)_6 = (T_E)_6 + \sigma \Phi^{-1}(0.75) \approx 145.33 + 3.283 \cdot 0.6745 \approx 147.544 \text{ (oko 147 vremenskih jedinica)}$$

Za skoro stoprocentnu vjerovatnoću 99.87% dobijamo procjenu trajanja projekta:

$$(T_S)_6 = (T_E)_6 + \sigma \Phi^{-1}(0.9987) \approx 145.33 + 3.283 \cdot 3 \approx 155.179 \text{ (oko 155 vremenskih jedinica)}$$

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).

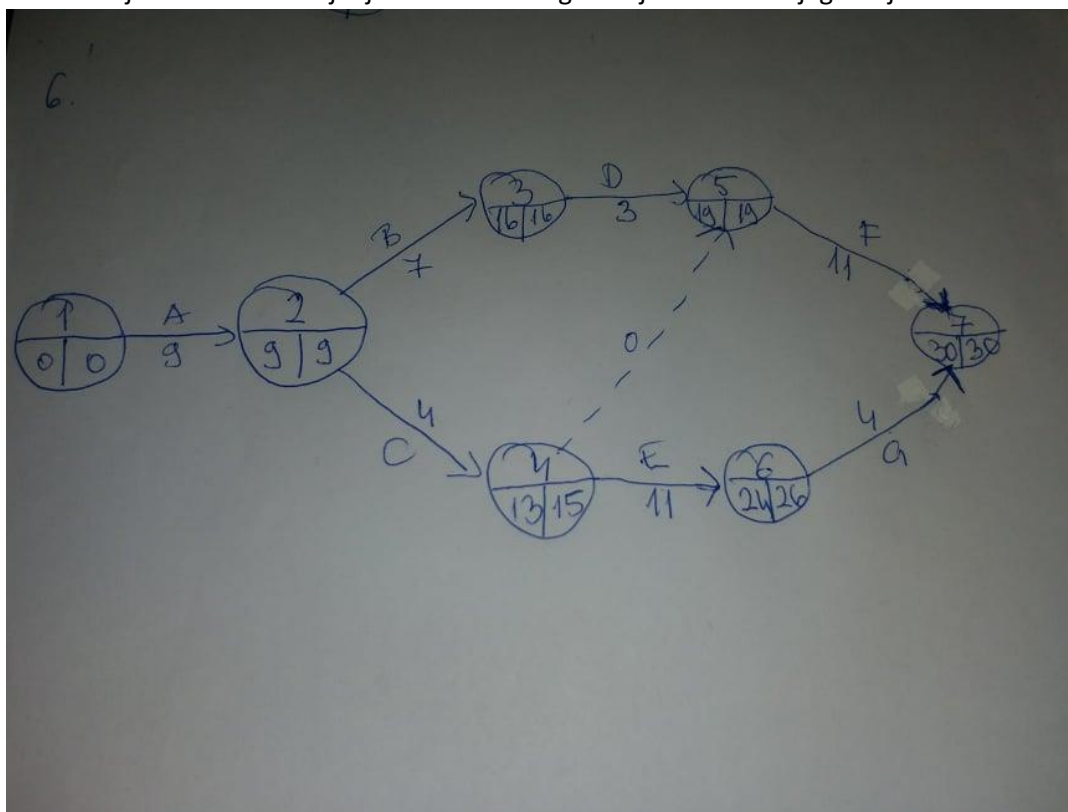
Aktivnost	Preuvjeti	$t^{(n)}$	$t^{(u)}$	$c^{(n)}$	$c^{(u)}$
A	-	9	6	5	17
B	A	7	7	14	14
C	A	4	3	15	21
D	B	3	2	4	8
E	C	11	7	3	15
F	C, D	11	6	7	37
G	E	4	3	7	9

Vaš zadatak je da:

- 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.
- Odredite plan realizacije projekta sa minimalnim trajanjem, uz najmanje moguće poskupljenje projekta. Obavezno naglasite koliko će iznositi novi troškovi projekta.
- Provjerite rezultat pod b. tako što ćete problem modelirati koristeći modele linearnog programiranja i rješavanjem odgovarajućih problema uz pomoć linprog funkcije u MATLAB-u. Obavezno naznačite šta su bili ulazni a šta izlazni podaci prilikom korištenja ove funkcije.

Izrada:

- Za nalaženje najekonomičnije realizacije, dovoljno je izvršiti klasičnu analizu strukture i vremena uzimajući normalna trajanja aktivnosti. Odgovarajući mrežni dijagram je:



Kritični put: A-B-D-F

Sa dobijenog mrežnog dijagrama možemo očitati da je normalno trajanje projekta $T = 30$, dok su normalni (minimalni) troškovi projekta $C_{\min} = 55$ (suma normalnih cijena).

Ovo je najekonomičnija (najjeftinija) realizacija projekta.

Znamo da se vremenske rezerve računaju po formulama:

$$(Rt)_{i,j} = t_j(1) - t_i(0) - t_{i,j}$$

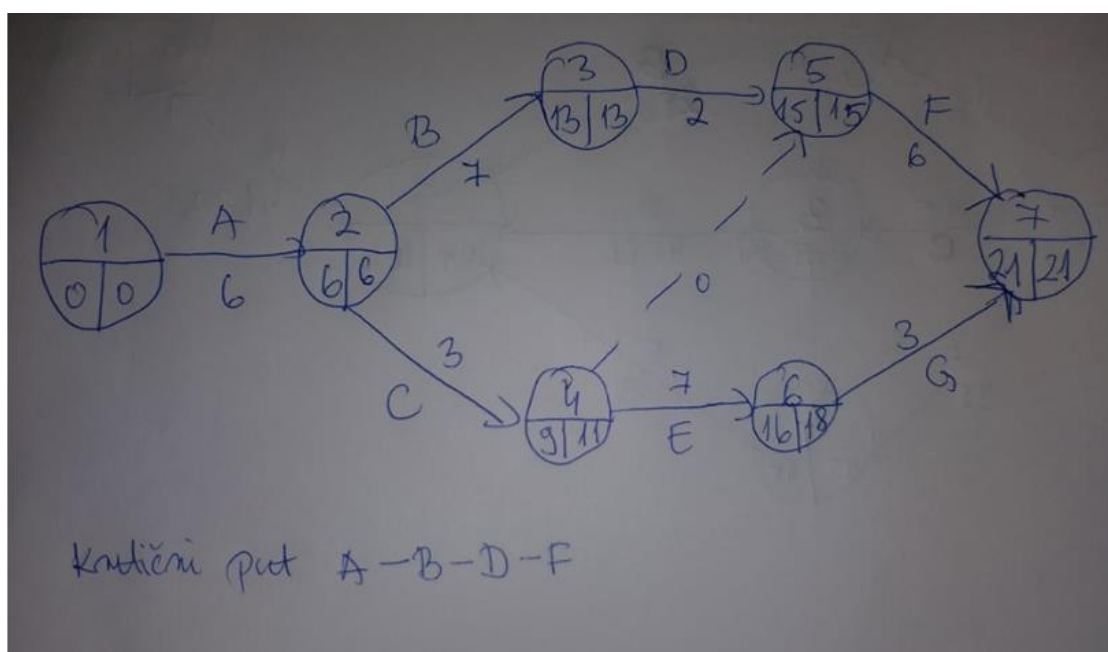
$$(Rs)_{i,j} = t_j(0) - t_i(0) - t_{i,j}$$

$$(R_n)_{ij} = t_{j(0)} - t_{i(1)} - t_{ij}$$

Izračunajmo sada vremenske rezerve tabelarno:

i-j	t_{ij}	$t_{i(0)}$	$t_{j(0)}$	$t_{i(1)}$	$t_{j(1)}$	$(Rt)_{ij}$	$(Rs)_{ij}$	$(Rn)_{ij}$
1-2	9	0	9	0	9	0	0	0
2-3	7	9	16	9	16	0	0	0
2-4	4	9	13	9	15	2	0	0
3-5	3	16	19	16	19	0	0	0
4-5	0	13	19	15	19	6	6	4
4-6	11	13	24	15	26	2	0	-2
5-7	11	19	30	19	30	0	0	0
6-7	4	24	30	26	30	2	2	0

Sada ćemo ponoviti isti postupak sa usiljenim trajanjima aktivnosti. Kao rezultat dobijamo:



Kritični put: A-B-D-F.

Najkraće trajanje projekta je $T_{min} = 21$ i odgovaraju mu troškovi $C_{max} = 121$ (suma usiljenih cijena).

Izračunajmo sada vremenske rezerve tabelarno koristeći odgovarajuće formule:

i-j	$t_{i,j}$	$t_{(i)0}$	$t_{(j)0}$	$t_{(i)1}$	$t_{(j)1}$	$(Rt)_{i,j}$	$(Rs)_{i,j}$	$(Rn)_{i,j}$
1-2	6	0	6	0	6	0	0	0
2-3	7	6	13	6	13	0	0	0
2-4	3	6	9	6	11	2	0	0
3-5	2	13	15	13	15	0	0	0
4-5	0	9	15	11	15	6	6	4
4-6	7	9	16	11	18	2	0	-2
5-7	6	15	21	15	21	0	0	0
6-7	3	16	21	18	21	0	2	0

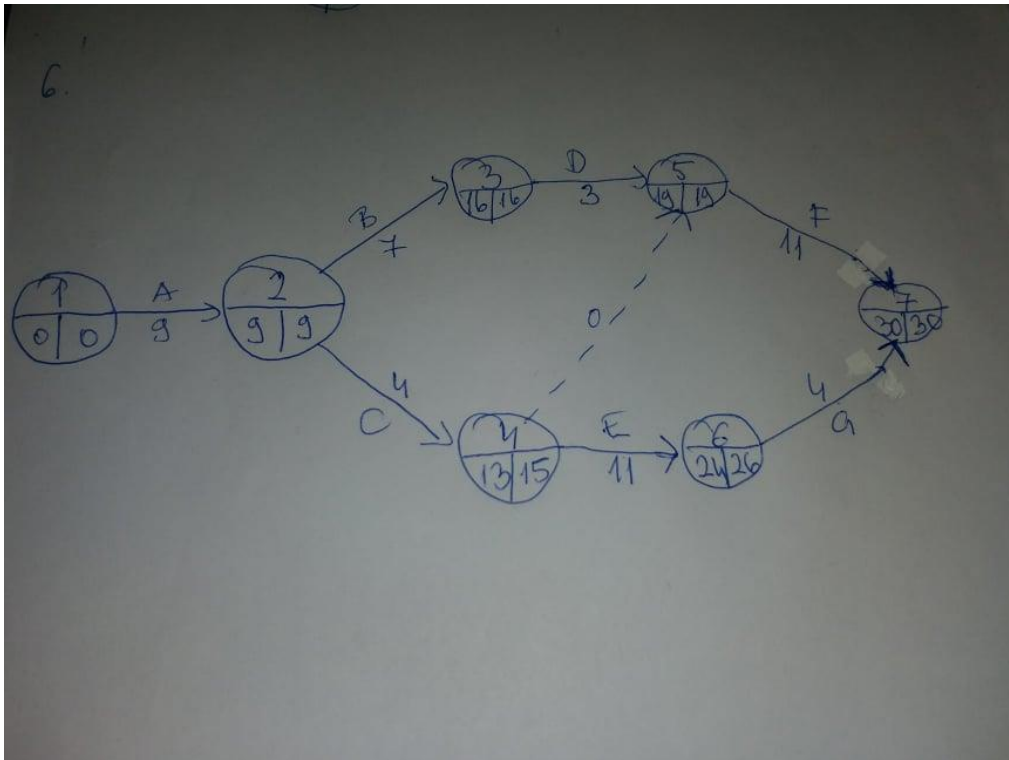
- b) Odredimo sada plan realizacije projekta sa minimalnim trajanjem uz najmanje moguće poskupljenje projekta. Koristićemo PERT/TIME metod te ćemo postupak raditi tabelarno (dvije tabele). Prva tabela prati trajanja svih aktivnosti u svakoj od iteracija, dok druga prikazuje trajanja svih puteva od početka do kraja u svakoj od iteracija. Pored toga, obje tabele sadrže i neke pomoćne informacije korisne za lakše praćenje postupka. U prvoj tabeli, osjenčena polja odgovaraju aktivnostima koje u odgovarajućoj iteraciji ima smisla kratiti (tj. koje se uopće mogu kratiti i koje se pri tome nalaze na barem jednom kritičnom putu), dok tamno osjenčena polja prikazuju aktivnosti koje smo zaista odabrali za kraćenje. U drugoj tabeli, osjenčena polja

prikazuju puteve na kojima se nalaze aktivnosti odabrane za kraćenje, pri čemu tamno osjenčena polja odgovaraju kritičnim putevima.

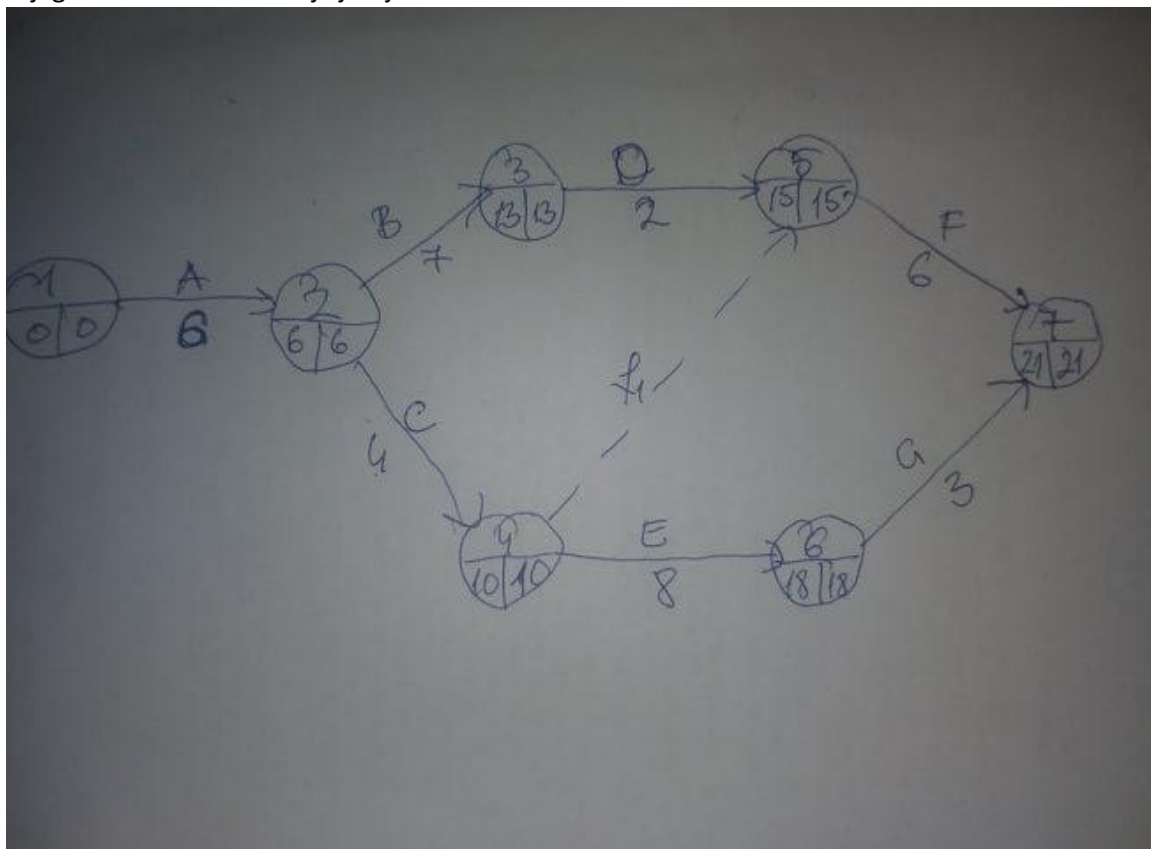
Izračunajmo prvo jedinični priraštaj troška $\alpha_{i,j}$.

$t^{(n)}$	$t^{(u)}$	$c^{(n)}$	$c^{(u)}$	$\alpha_{i,j}$
A 9	6	5	17	4
B 7	7	14	14	-
C 4	3	15	21	6
D 3	2	4	8	4
E 11	7	3	15	3
F 11	6	7	37	6
G 4	3	7	9	2

Krećemo od dijagrama :



Dijagram nakon skraćivanja je sljedeći:



Kritični put: A-B-D-F I A-C-E-G.

Aktivnost	$t_{i,j}(n)$	$t_{i,j}(u)$	$\alpha_{i,j}$	Iteracija					
				I	II	III	IV	V	VI
A	9	6	4	6	6	6	6	6	6
B	7	7	-	7	7	7	7	7	7
C	4	3	6	4	4	4	4	4	4
D	3	2	4	3	2	2	2	2	2
E	11	7	3	11	11	11	11	8	8
F	11	6	6	11	11	10	9	6	6
G	4	3	2	4	4	4	3	3	3

Put	Iteracija					
	I	II	III	IV	V	VI
A-B-D-F	30	27	26	25	24	21
A-C-E-G	28	25	25	25	24	21
A-C-f1-F	24	21	21	20	19	16
Troškovi:	55	55+4*3=67	67+4=71	71+6=77	77+6+2=85	85+18+9=112
Krati se:	A	D	F	F,G	F,E	-

I iteracija:

Samo jedan od tri moguća puta je kritičan i to je put A-B-D-F. Najmanji jedinični prirast troškova (prirast 4) imaju aktivnosti A i D, aktivnost A kratimo jer možemo skratiti više vremenski jedinica od 1.

Aktivnost A možemo skratiti za 3 vremenske jedinice zbog usiljene vrijednosti. Ovim kraćenjem skraćujemo sve moguće puteve.

II iteracija:

Ponovo imamo jedan kritični put A-B-D-F te aktivnosti koje možemo skratiti su D i F, odlučujemo se za α_D koje je 4 jer je vrijednost manja od $\alpha_F=6$. Smanjujemo aktivnost D za 1 vremensku jedinicu zbog usiljene vrijednosti.

III iteracija:

Ponovo imamo jedan kritični put A-B-D-F i jedina aktivnost koju možemo skratiti je F i to je možemo skratiti za 5 vremenskih jedinica, ali kako je drugi najduži put dužine 25, a kritični put je inače dužine 5, to ćemo aktivnost F smanjiti za vrijednost $26-25=1$ vremenska jedinica. Ovim skraćujemo samo kritični put.

IV iteracija:

Sada imamo 2 kritična puta A-B-D-F i A-C-E-G, kako je njihova zajednička aktivnost maksimalno skraćena do usiljene vrijednosti, ostaje da skraćujemo pojedinačne aktivnosti na svakom putu respektivno. Što se tiče prvog kritičnog puta tu imamo aktivnost F koju možemo skratiti za 4 vremenske jedinice, a što se tiče drugog puta možemo aktivnost sa najmanjim prirastom je G i možemo je skratiti za jednu vremensku jedinicu zbog usiljene vrijednosti. Pa skraćujemo aktivnosti F i G za jednu vremensku jedinicu.

V iteracija:

Ponovo posmatramo dva kritična puta A-C-E-G i A-B-D-F. Sada aktivnost F možemo skratiti za 3 vremenske jedinice, a na drugom putu aktivnost najmanjeg prirasta je E $\alpha_E=3$ i tu aktivnost možemo smanjiti za 4 vremenske jedinice. Tako da na kraju aktivnost F i E smanjujemo za 3 vremenske jedinice

VI iteracija:

Dalja skraćivanja nisu moguća pošto na kritičnom putu A-B-D-F imamo sve usiljene vrijednosti, tako da algoritam završava. Minimalno trajanje projekta je 21(vremenska jedinica), a novi troškovi projekta su 115 (novčanih jedinica).

c) Provjerimo dobijeni rezultat pod b tako što ćemo modelirati problem koristeći modele linearnog programiranja i rješavanjem odgovarajućih problema uz pomoć linprog funkcije u MATLAB-u.

Uvesti ćemo promjenljive $t_1, t_2, t_3, t_4, t_5, t_6, t_7$. Ograničenja:

$t_2 - t_1 \geq t_{1,2}$	$6 \leq t_{1,2} \leq 9$	$t_2 - t_1 \geq 9$
$t_3 - t_2 \geq t_{2,3}$	$7 \leq t_{2,3} \leq 7$	$t_3 - t_2 \geq 7$
$t_4 - t_2 \geq t_{2,4}$	$3 \leq t_{2,4} \leq 4$	$t_4 - t_2 \geq 4$
$t_5 - t_3 \geq t_{3,5}$	$2 \leq t_{3,5} \leq 3$	$t_5 - t_3 \geq 3$
$t_5 - t_4 \geq t_{4,5}$	$7 \leq t_{4,5} \leq 11$	$t_5 - t_4 \geq 0$
$t_6 - t_4 \geq t_{4,6}$	$6 \leq t_{4,6} \leq 11$	$t_6 - t_4 \geq 11$
$t_7 - t_5 \geq t_{5,7}$	$3 \leq t_{5,7} \leq 4$	$t_7 - t_5 \geq 11$
$t_7 - t_6 \geq t_{6,7}$		$t_7 - t_6 \geq 4$

Trajanje projekta: $T = t_7 - t_1$.

Troškovi projekta(izraženi u funkciji trajanja onih aktivnosti čije trajanje nije fiksno) iznose:

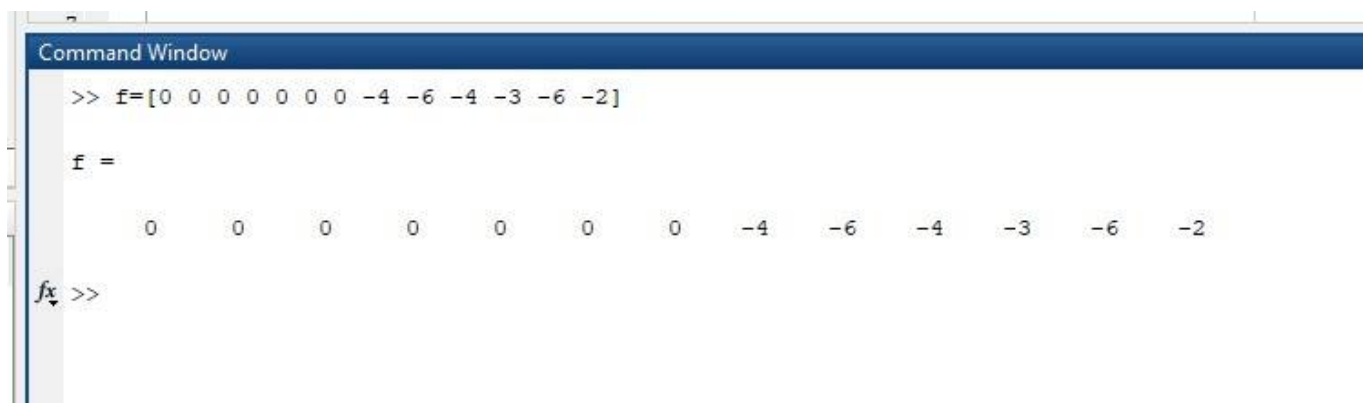
$$C = \sum_{(i,j)} c_{i,j}^{(n)} + \alpha_{i,j}^{(n)} t_{i,j}^{(n)} - \sum_{(i,j)} \alpha_i t_{i,j}^{(n)}$$

U našem slučaju imamo:

$$C = (5 + 4 \cdot (9 - t_{1,2})) + (14+0)+0+ (15+ 6 \cdot (4 - t_{2,4})) +(4 + 4 \cdot (3 - t_{3,5})) + (3 + 3 \cdot (11 - t_{4,6})) + (7+ 6 \cdot (11 - t_{5,7})) + (7+ 2 \cdot (4 - t_{6,7}))$$

UNUTAR ZAGRADE TN IZVAN ZAF+GRADE JE ALFA *()-

$$C = 234 - 4t_{1,2} - 6t_{2,4} - 4t_{3,5} - 3t_{4,6} - 6t_{5,7} - 2t_{6,7}$$



```

Command Window
>> f=[0 0 0 0 0 0 0 -4 -6 -4 -3 -6 -2]

f =

    0    0    0    0    0    0    0   -4   -6   -4   -3   -6   -2

fx >>
  
```

A1 =

1	-1	0	0	0	0	0
0	1	-1	0	0	0	0
0	1	0	-1	0	0	0
0	0	1	0	-1	0	0
0	0	0	1	-1	0	0
0	0	0	1	0	-1	0
0	0	0	0	1	0	-1
0	0	0	0	0	1	-1

A2 =

1	0	0	0	0	0
0	1	0	0	0	0
0	0	1	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	1	0
0	0	0	0	0	1

```
>> A3 = zeros(12,7);
>> A4 = eye(6);
>> A5 = -eye(6);
>> A = [A1 A2; A3 [A4;A5]]
```

A =

1	-1	0	0	0	0	0	1	0	0	0	0	0
0	1	-1	0	0	0	0	0	1	0	0	0	0
0	1	0	-1	0	0	0	0	0	1	0	0	0
0	0	1	0	-1	0	0	0	0	0	0	0	0
0	0	0	1	-1	0	0	0	0	0	0	0	0
0	0	0	1	0	-1	0	0	0	0	1	0	0
0	0	0	0	1	0	-1	0	0	0	0	1	0
0	0	0	0	0	1	-1	0	0	0	0	0	1
0	0	0	0	0	0	0	1	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	1	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	1	0
0	0	0	0	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	-1	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	-1	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	-1	0
0	0	0	0	0	0	0	0	0	0	0	0	-1

```
>> b=[0; 7; 0; 0; 0; 0; 0; 0; 0; 6; 3; 2; 7; 6; 3; -9; -4; -3; -11; -11; -4]
```

```
b =
```

```
0
7
0
0
0
0
0
0
0
6
3
2
7
6
3
-9
-4
-3
-11
-11
-4
```

```
>> x=linprog(f,A,b)
```

```
x =
```

```
112
```

```
>> |
```

Rješenje je isto kao i pod b).