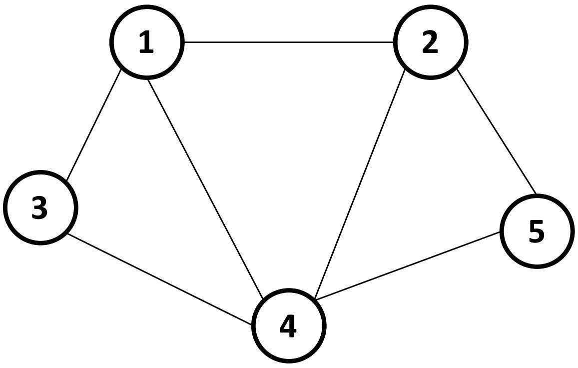
1 ОСНОВНИ ПОЈМОВИ ТРАНСПОРТНИХ МРЕЖА

1.1 Означавање транспортних мрежа

Општа ознака транспортне мреже је **G**$= (\boldsymbol{N}, \boldsymbol{A})$, где G означава мрежу у којој је скуп чворова означен са N , а скуп веза (грана) између појединих чворова са A. Грана између чворова i и j се означава са (i, j).

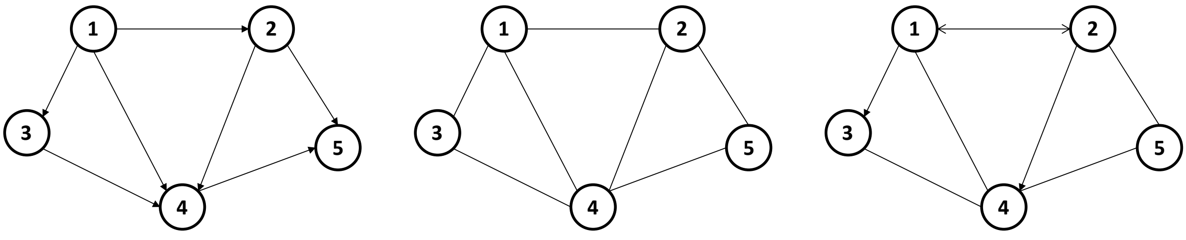


G = (N, A)
N = {1, 2, 3, 4, 5}
A = {(1,2), (1,3), (1,4), (2,1), (2,4), (2,5), (3,1), (3,4), (4,1), (4,2), (4,5), (5,2), (5,4)}

Свакој грани се може доделити једна или више нумеричких карактеристика. То су најчешће трошкови и капацитет.

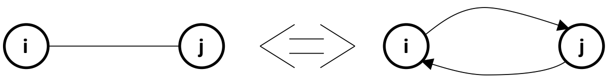
1.2 Оријентисаност мреже

Са аспекта оријентисаности (усмерености), транспортна мрежа може бити **оријентисана**, **неоријентисана** или **мешовита**. Оријентисана мрежа има гране дуж којих је кретање дозвољено само у једном смеру – оријентисане (усмерене) гране. Таква грана се означава стрелицом у допуштеном смеру. На пример, грана (2,3) у оријентисаној мрежи води од чвора 2 до чвора 3. Ако су неке гране у мрежи оријентисане, а неке неоријентисане, онда је таква мрежа мешовита.



Оријентисана мрежа　　Неоријентисана мрежа　　Мешовита мрежа

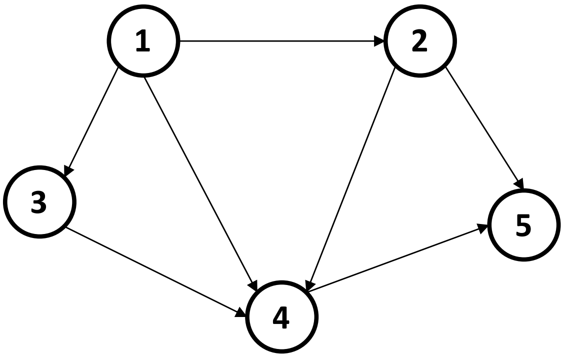
Ако је нека грана мреже неоријентисана, онда то значи да је кретање том граном дозвољено у оба смера, односно:



1.3 Степен чвора

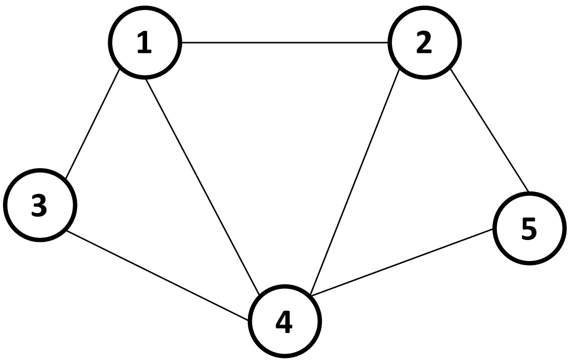
Степен чвора се разликује код оријентисаних и неоријентисаних мрежа.

Код оријентисаних мрежа постоји **улазни** и **излазни степен чвора**. Улазни степен чвора представља број грана које улазе у тај чвор. Излазни степен чвора представља број грана које излазе из тог чвора.



Улазни степен чвора 2 = 1
Излазни степен чвора 2 = 2
Улазни степен чвора 5 = 2
Излазни степен чвора 5 = 0

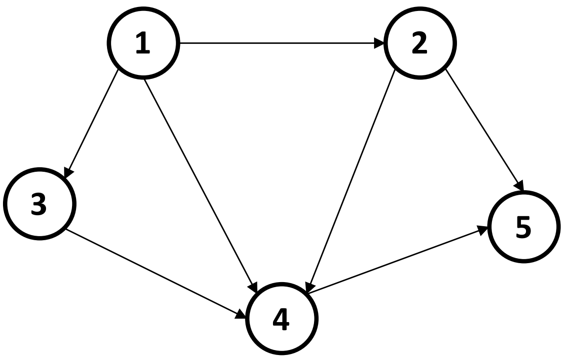
Код неоријентисаних мрежа, **степен чвора** представља број грана које повезују тај чвор са осталим гранама у мрежи.



Степен чвора 1 = 3
Степен чвора 2 = 3
Степен чвора 3 = 2
Степен чвора 4 = 4
Степен чвора 5 = 2

1.4 Пут у мрежи

Пут у мрежи од чвора i до чвора j представља низ грана и свих чворова кроз које се мора проћи при кретању од чвора i до чвора j. Пут се дефинише или набрајањем грана или набрајањем чворова кроз које пролази.



Пут од чвора 1 до чвора 5:
(1, 3, 4, 5)
или
((1,3), (3,4), (4,5))

Под **кружним путем (циклусом)** подразумева се пут који почиње и завршава се у истом чвору.

Почетни и завршни чвор називају се **изворним** и **циљним** чвором, респективно.

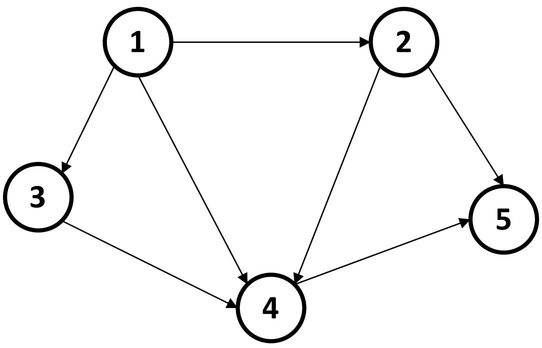
Уколико се поједине гране појављују у путу само једном, тај пут је **прост**. Уколико се сваки чвор појављује у путу само једном, тај пут је **елементарни**.

Два чвора i и j су **повезана** уколико постоји пут од чвора i до чвора j. За неоријентисану мрежу се каже да је повезана ако постоји одговарајући пут између свих парова чворова. За оријентисану мрежу се каже да је повезана ако је одговарајућа неоријентисана мрежа (добијена укидањем оријентације) повезана. Оријентисана мрежа у којој постоје путеви између свих парова чворова назива се јако повезаном оријентисаном мрежом.

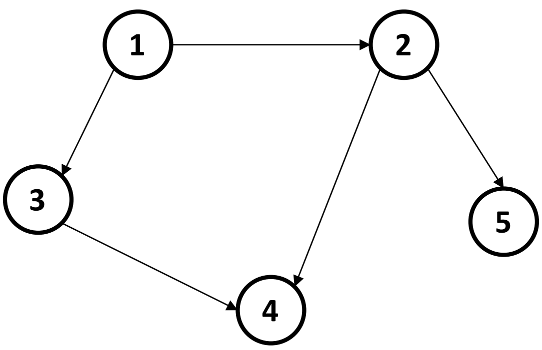
1.5 Парцијални граф и подграф

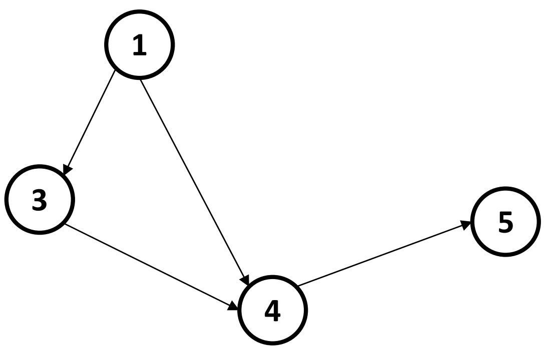
Парцијални граф графа G= (N, A) је граф $\mathrm{G}_1 = (N, A_1)$, при чему је A_1 подскуп скупа A. Дакле, парцијални граф се добија укидањем појединих грана оригиналног графа, док број чворова остаје исти.

Подграф графа G= (N, A) је оријентисани граф $\mathrm{G}_2 = (N_2, A_2)$, при чему је N_2 подскуп скупа N, а A_2 подскуп скупа A, али такав да скупу A_2 припадају само гране које повезују чворове скупа N_2. Дакле, подграф се добија укидањем појединих чворова оригиналног графа и грана које улазе/излазе из тог чвора.



Оригинални граф



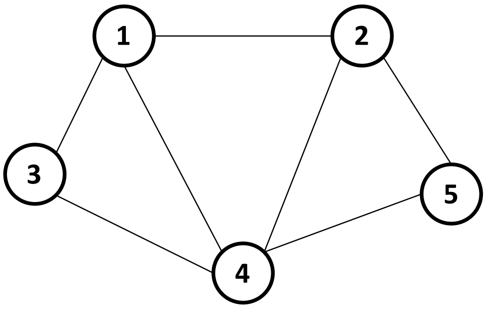


Парцијални граф

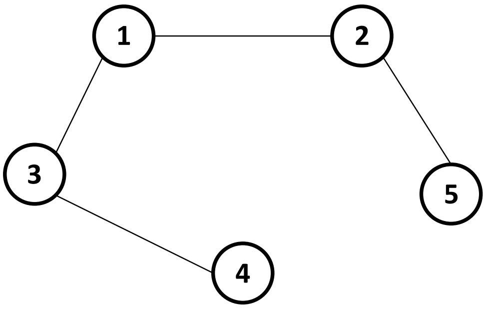
Подграф

Дрво је повезани граф са $n(n > 1)$ чворова и $n - 1$ грана. Под **разапињућим дрветом** неке мреже подразумева се дрво које садржи све чворове почетне мреже. Карактеристике разапињућег дрвета су следеће:

- између свих парова чворова у дрвету постоје одговарајући путеви;
- не садржи циклусе.



Оригинална мрежа



Разапињуће дрво
5 чворова, 4 гране

1.6 Степен повезаности мреже

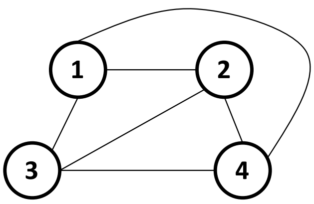
Квантитативни показатељи повезаности транспортне мреже омогућавају међусобно поређење различитих мрежа и праћење развоја мреже током времена.

Повезаност транспортне мреже се односи на степен повезаности чворова мреже. Као основни квантитативни показатељи повезаности мреже, користе се $\boldsymbol{\gamma}$ **индекс** и $\boldsymbol{\alpha}$ **индекс**. Они се примењују на мреже које се могу представити планарним графом. **Планарни граф** подразумева да се гране мреже укрштају само у чворовима, па стога планарни граф одликује максимални број грана.

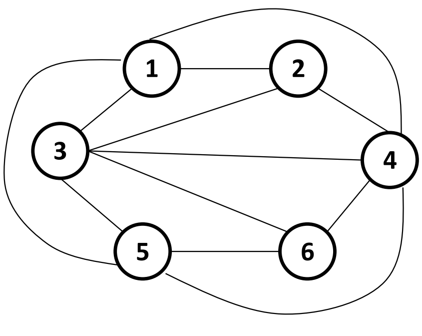
Нека је број чворова у планарном графу означен са v, а максимални број грана са e_{max}. Веза између ова два параметра је следећа:

$$e_{max} = 3(v - 2).$$

Максимални број грана у планарном графу представља максимални број грана којима су директно повезани чворови мреже, при чему се све гране укрштају само у чворовима. Мрежа са максималним бројем грана има максимално могућу повезаност.



v = 4
e_{max} = 3(4-2) = 6



v = 6
e_{max} = 3(6-2) = 12

За мрежу која има ν чворова и е грана, γ индекс се дефинише као однос броја грана у мрежи е и максималног броја грана e_{max}:

$$\gamma = \frac{e}{e_{max}} = \frac{e}{3(\nu-2)}$$

Вредности γ индекса налазе се у интервалу [0, 1].

У минимално повезаној мрежи број грана је једнак $\nu - 1$. Разапињуће дрво представља минимално повезану мрежу, у којој нема циклуса. Свака додатна грана доводи до стварања циклуса. Број створених фундаменталних циклуса c (циклуси који садрже само једну грану која не припада дрвету) је стога једнак разлици између броја грана у мрежи (е) и броја грана минимално повезане мреже ($\nu - 1$):

$$c = e - (\nu - 1) = e - \nu + 1$$

Максимални број циклуса, c_{max}, једнак је разлици између броја грана максимално и минимално повезане мреже:

$$c_{max} = e_{max} - e_{min} = 3(\nu - 2) - (\nu - 1) = 2\nu - 5$$

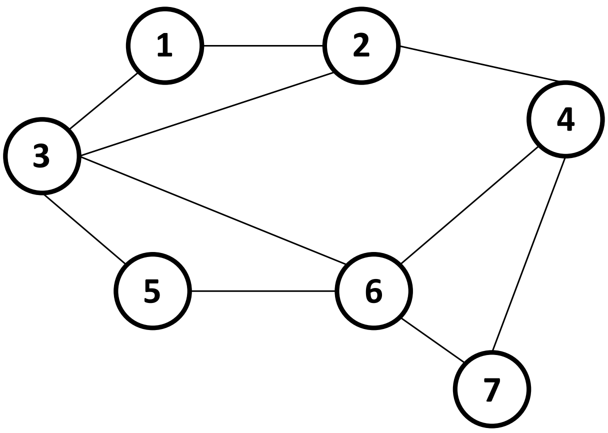
α индекс се дефинише као однос броја циклуса c и максималног броја циклуса c_{max}:

$$\alpha = \frac{c}{c_{max}} = \frac{e - e_{min}}{e_{max} - e_{min}} = \frac{e - \nu + 1}{2\nu - 5}$$

Вредности α индекса налазе се у интервалу [0, 1]. Вредност 0 одговара минимално повезаној мрежи, а вредност 1 максимално повезаној мрежи.

Пример 1-1

Одредити γ индекс и α индекс повезаности дате транспортне мреже.



Решење:

$\nu = 7$
$e = 10$

$\gamma = \frac{e}{e_{max}} = \frac{e}{3(\nu-2)} = \frac{10}{3(7-2)} = \frac{10}{15} = 0,67$ → дата мрежа има 67% повезаности у односу на максимално могућу повезаност.

$\alpha = \frac{c}{c_{max}} == \frac{e-\nu+1}{2\nu-5} = \frac{10-7+1}{2\cdot 7-5} = \frac{4}{9} = 0,44$

1.7 Приказивање мрежа у матричној форми

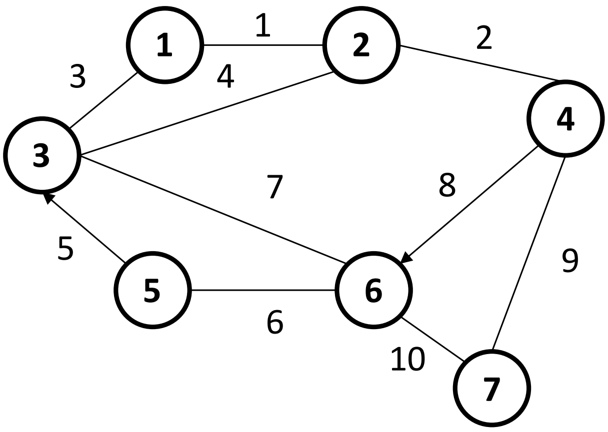
Постоје два начина за приказивање мрежа у матричној форми.

Први начин се користи за приказивање мреже у облику **квадратне матрице**, у којој сваком чвору мреже одговара једна врста и једна колона. Нека је дата мрежа $G = (N, A)$ која има ν чворова и e грана. Одговарајућа матрица X ће имати ν врста и ν колона. Елементи x_{ij} матрице X дефинишу се на следећи начин:

$$x_{ij} = \begin{cases} 1, \text{ако грана } (i,j) \text{ постоји у мрежи } G \\ 0, \text{у осталим случајевима} \end{cases}$$

Пример 1-2

Дату транспортну мрежу представити у облику квадратне матрице X.



Решење:

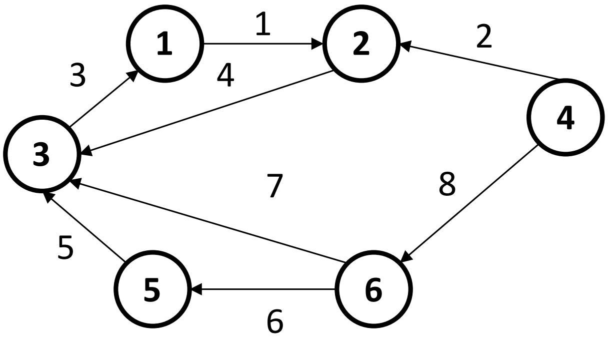
$$X = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array} \begin{array}{c} \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 7 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

Други начин се користи за приказивање мреже у облику матрице Y, чији број врста одговара броју чворова, а број колона одговара броју грана дате мреже; односно сваком чвору одговара једна врста, а свакој грани једна колона матрице. Елементи y_{ij} матрице Y дефинишу се на следећи начин:

$$y_{ij} = \begin{cases} +1, \text{ако } j - \text{та грана излази из чвора } i \\ -1, \text{ако } j - \text{та грана улази у чвор } i \\ 0, \text{у осталим случајевима} \end{cases}$$

Пример 1-3

Дату транспортну мрежу представити у облику матрице Y.



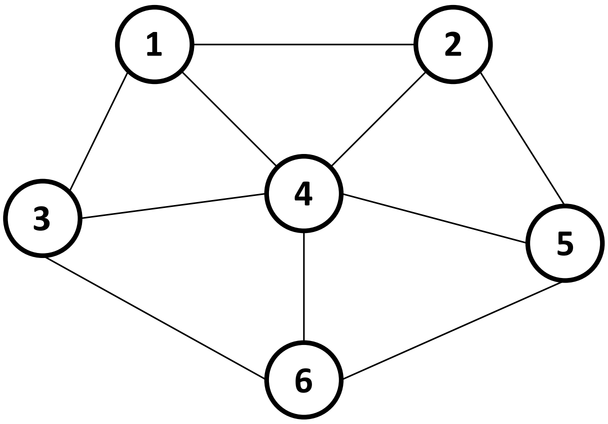
Решење:

$$Y = \begin{array}{cc} & \begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} & \begin{bmatrix} +1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & +1 & 0 & 0 & 0 & 0 \\ 0 & 0 & +1 & -1 & -1 & 0 & -1 & 0 \\ 0 & +1 & 0 & 0 & 0 & 0 & 0 & +1 \\ 0 & 0 & 0 & 0 & +1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & +1 & +1 & -1 \end{bmatrix} \end{array}$$

2 АЛГОРИТАМ ЗА КОНСТРУКЦИЈУ РАЗАПИЊУЋЕГ ДРВЕТА

Под разапињућим дрветом подразумева се дрво које садржи све чворове почетне мреже.

Нека чворови на слици испод представљају аеродроме у различитим градовима. Сви аеродроми су повезани летовима – неки директно, неки индиректно. Сада се поставља питање како се број летова може смањити на најмањи могући број тако да се и даље може стићи од сваког до сваког другог аеродрома. Мрежа коју би добили на тај начин заправо представља разапињуће дрво, чији је број грана (тј. директних летова) за један мањи од броја чворова (тј. аеродрома). Овај проблем има више решења.



Помоћу алгоритма за конструкцију разапињућег дрвета за сваку грану оригиналне мреже се испитује да ли она треба да буде укључена у разапињуће дрво или не. Редослед испитивања гране је **произвољан**, а испитивање гране се врши **бојењем гране**. Гране које се не укључују у разапињуће дрво боје се **црвеном бојом**, а оне које се укључују боје се **зеленом бојом**. Грана се укључује у дрво **ако њено укључење у дрво не ствара циклус**.

Приликом конструисања дрвета, гране које су већ укључене у дрво чине једну или више повезаних компоненти. На почетку су све компоненте празне, а све гране необојене. **Алгоритам за конструкцију разапињућег дрвета** састоји се од следећих корака:

Корак 1: Произвољно изабрати грану која није „петља“ (петља је грана које излази и улази у један исти чвор). Обојити ову грану у зелено и оба њена чвора сместити у компоненту.

Корак 2: Изабрати било коју необојену грану која није петља. Затим су могуће следеће четири ситуације:
1) Оба чвора ове гране припадају истој компоненти → обојити грану у црвено и вратити се на почетак корака 2;
2) Један чвор ове гране припада компоненти, а други чвор гране не припада ниједној компоненти → чвор који не припада ниједној компоненти треба укључити у компоненту којој припада чвор на другом крају гране;

3) Ни један ни други чвор ове гране не припадају ниједној компоненти → грану треба обојити у зелено, тј. укључити у дрво;
4) Чворови ове гране припадају различитим компонентама → обојити грану у зелено, тј. укључити је у дрво, а ове компоненте спојити у једну.

Корак 3: Уколико сви чворови припадају једној компоненти, завршити са алгоритмом; зелене гране чине разапињуће дрво. У супротном се вратити на корак 2.

Редослед испитивања грана утиче на облик разапињућег дрвета, али то је одраз чињенице да је могуће да за једну транспортну мрежу има више решења, тј. да се за исту транспортну мрежу може конструисати више различитих разапињућих дрвета.

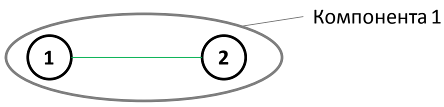
Пример 2-1

Конструисати разапињуће дрво за транспортну мрежу на горњој слици.

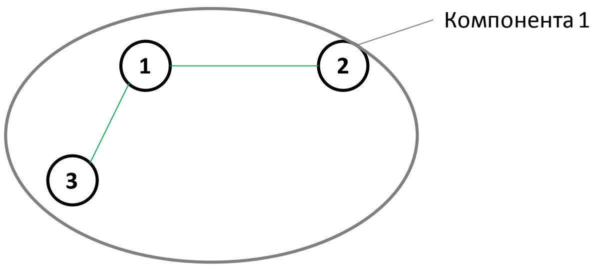
Решење:

Произвољан редослед испитивања грана: (1,2), (1,3), (2,4), (2,5), (3,4), (3,6), (4,5), (4,6), (5,6), (4,1)

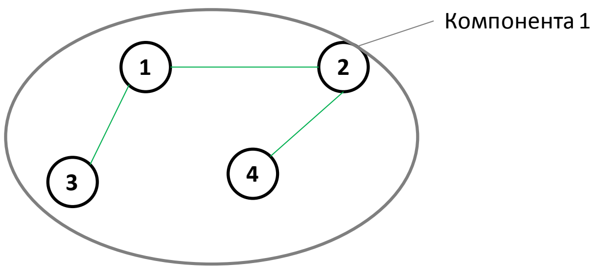
Грана (1,2): случај 3) → боји се у зелено и смешта у компоненту 1:



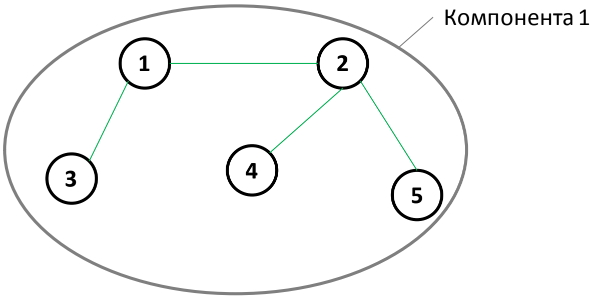
Грана (1,3): случај 2) → боји се у зелено и прикључује постојећој компоненти 1:



Грана (2,4): случај 2) → боји се у зелено и прикључује постојећој компоненти 1:

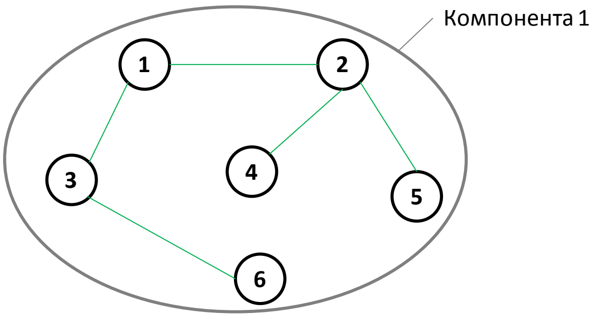


Грана (2,5): случај 2) → боји се у зелено и прикључује постојећој компоненти 1:

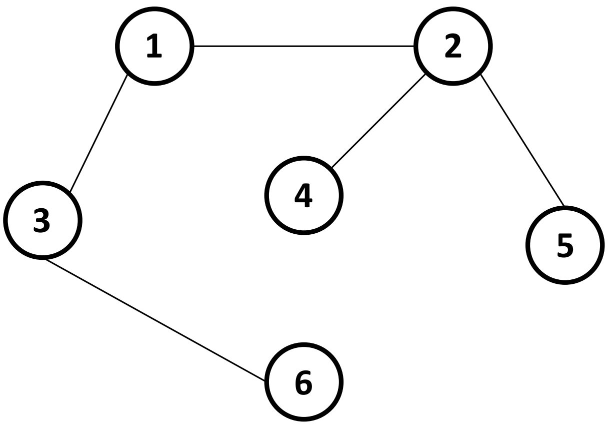


Грана (3,4): случај 1) → боји се у црвено, тј. не укључује се у дрво.

Грана (3,6): случај 2) → боји се у зелено и прикључује постојећој компоненти 1:



Пошто сви чворови припадају истој компоненти (компоненти 1), алгоритам се завршава. На доњој слици је приказано добијено разапињуће дрво.



2.1 Алгоритми за изналажење разапињућег дрвета најмање дужине

2.1.1 Примов алгоритам

Овај алгоритам се састоји из следећих корака:

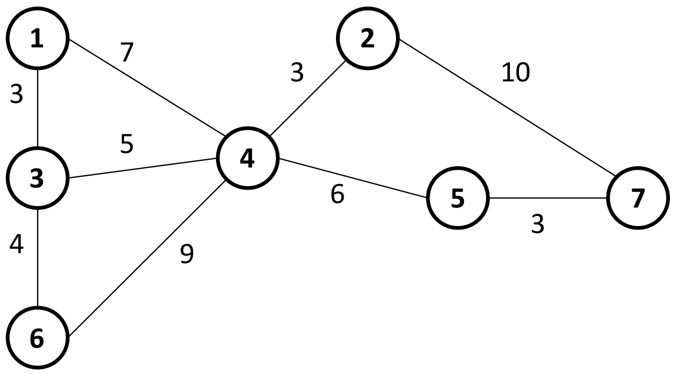
Корак 1: Почети конструкцију разапињућег дрвета у произвољном чвору *i*. Затим пронаћи чвор најближи чвору *i*. Нека је то неки чвор *j*. Чвор *j* и грана (*i,j*) се укључују у најкраће разапињуће дрво. Затим се укидају (уколико постоје) гране које би онемогућиле конструисање дрвета.

Корак 2: Уколико су сви чворови повезани, најкраће разапињуће дрво је конструисано. Уколико нису, прелази се на корак 3.

Корак 3: Изоловани чвор који је најближи до сада формираном разапињућем дрвету укључује се заједно са одговарајућом граном у састав најкраћег разапињућег дрвета. Укинути гране које онемогућавају конструкцију разапињућег дрвета. Затим се вратити на корак 2.

Пример 2-2

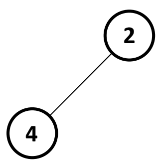
Конструисати разапињуће дрво за транспортну мрежу на слици користећи Примов алгоритам. Бројеви изнад грана означавају дужину гране.



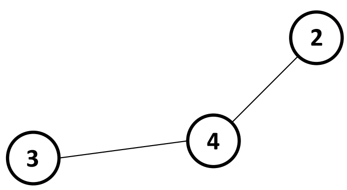
Решење:

Пошто се чвор од кога се почиње конструкција разапињућег дрвета произвољно бира, нека то буде чвор 2.

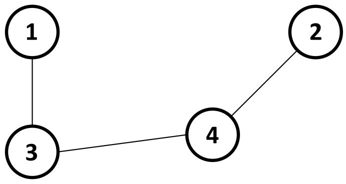
Имајући у виду дужине грана, чвору 2 је најближи чвор 4, па се чвор 4 укључује у разапињуће дрво преко гране (2,4).



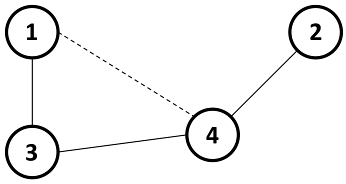
Овако конструисаном дрвету је најближи чвор 3, који се укључује у дрво преко гране (3,4).



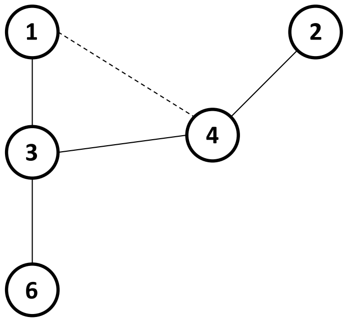
Овако конструисаном дрвету је најближи чвор 1, који се укључује у дрво преко гране (3,1).



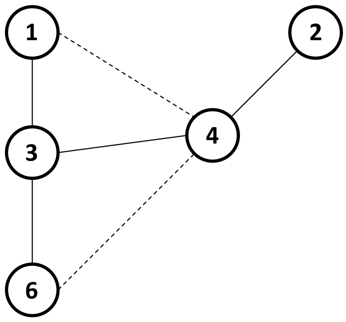
Пошто би грана (1,4) створила циклус, она се укида.



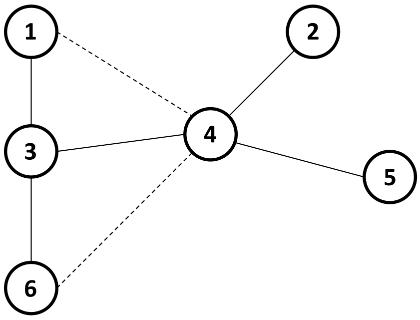
Овако конструисаном дрвету је најближи чвор 6, који се укључује у дрво преко гране (3,6).



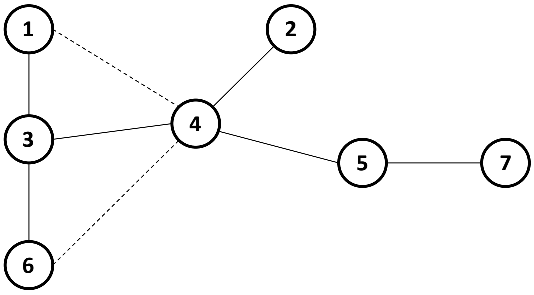
Пошто би грана (4,6) створила циклус, она се укида.



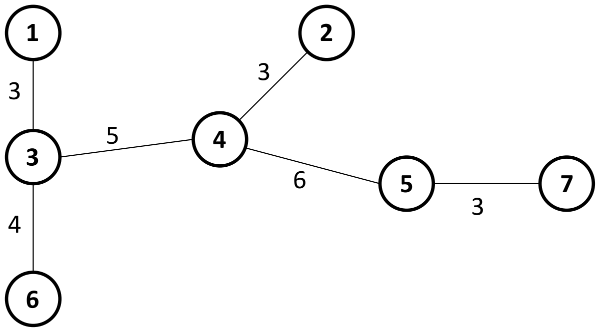
Овако конструисаном дрвету је најближи чвор 5, који се укључује у дрво преко гране (4,5).



Овако конструисаном дрвету је најближи чвор 7, који се укључује у дрво преко гране (5,7).



Пошто су сви чворови укључени у дрво, алгоритам се завршава, а најкраће разапињуће дрво је конструисано. Дужина најкраћег разапињућег дрвета, L, израчунава се сабирањем дужина грана које чине разапињуће дрво:



L = 3+4+5+3+6+3 = 24

Поступак се може записати и у следећој форми:

Чворови који су тренутно укључени у дрво	Чворови који тренутно нису укључени у дрво						Грана која се укида
2	1	3	**4** **$d_{2-4} = 3$**	5	6	7 $d_{2-7} = 10$	/
2, 4	1 $d_{4-1} = 7$	**3** **$d_{4-3} = 5$**		5 $d_{4-5} = 6$	6 $d_{4-6} = 9$	7 $d_{2-7} = 10$	/
2, 4, 3	**1** $d_{4-1} = 7$ **$d_{3-1} = 3$**			5 $d_{4-5} = 6$	6 $d_{4-6} = 9$ $d_{3-6} = 4$	7 $d_{2-7} = 10$	(4,1)
2, 4, 3, 1				5 $d_{4-5} = 6$	**6** $d_{4-6} = 9$ **$d_{3-6} = 4$**	7 $d_{2-7} = 10$	(4,6)
2, 4, 3, 1, 6				**5** **$d_{4-5} = 6$**		7 $d_{2-7} = 10$	/
2, 4, 3, 1, 6, 5						**7** $d_{2-7} = 10$ **$d_{5-7} = 3$**	(2,7)
2, 4, 3, 1, 6, 5, 7							

2.1.2 Крускалов алгоритам

Помоћу **Крускаловог алгоритма** може се конструисати **разапињуће дрво најмање или највеће дужине**. Састоји се од следећих корака:

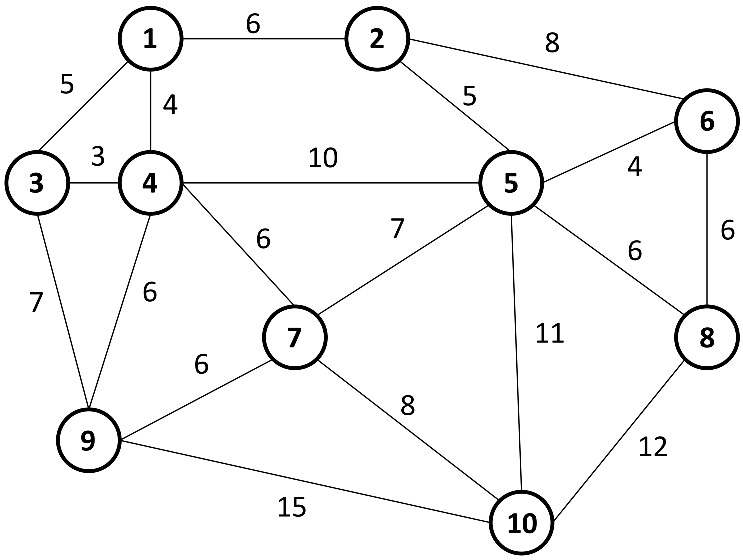
Корак 1: Направити листу грана оригиналне мреже, сортираних у растућем поретку њихових дужина (на првом месту је најкраћа грана, а на последњем најдужа). У случају када две или више грана имају исту дужину, произвољним редоследом их укључити у листу једну иза друге.

Корак 2: Применити алгоритам за конструкцију разапињућег дрвета испитујући гране редоследом по коме се налазе на листи.

У случају када треба конструисати разапуњуће дрво највеће дужине, гране мреже се у кораку 1 сортирају у опадајућем поретку њихових дужина (на првом месту је најдужа грана, а на последњем најкраћа).

Пример 2-3

За дату мрежу помоћу Крускаловог алгоритма конструисати разапињуће дрво:
a) најмање дужине
b) највеће дужине.



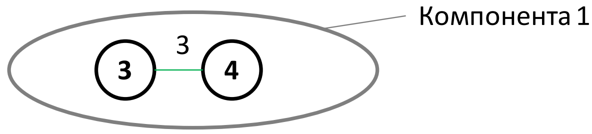
Решење:

a) Листа грана сортираних у растућем поретку:

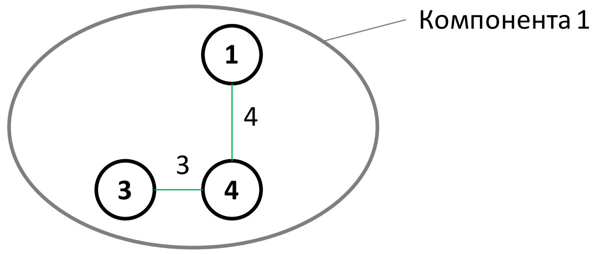
(3,4), (1,4), (5,6), (1,3), (2,5), (1,2), (4,9), (4,7), (6,8), (5,8), (7,9), (3,9), (5,7), (2,6), (7,10), (4,5), (5,10), (8,10), (9,10)

Примена алгоритма за конструкцију разапињућег дрвета:

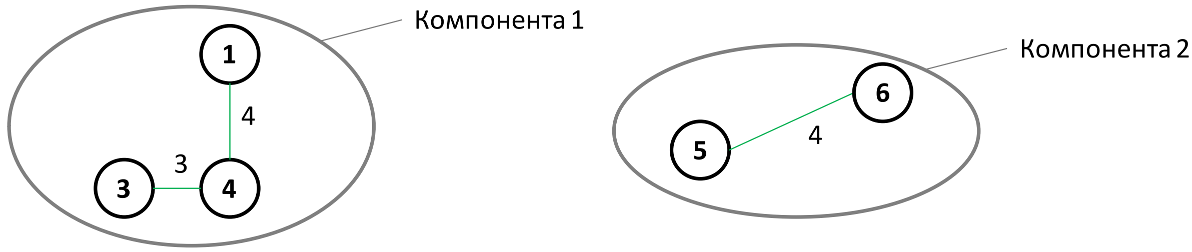
Грана (3,4): случај 3) → боји се у зелено и смешта у компоненту 1:



Грана (1,4): случај 2) → боји се у зелено и прикључује компоненти 1:

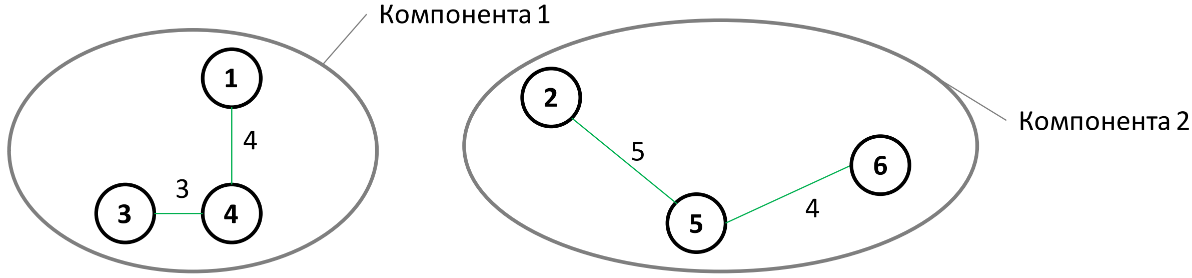


Грана (5,6): случај 3) → боји се у зелено и смешта у компоненту 2:

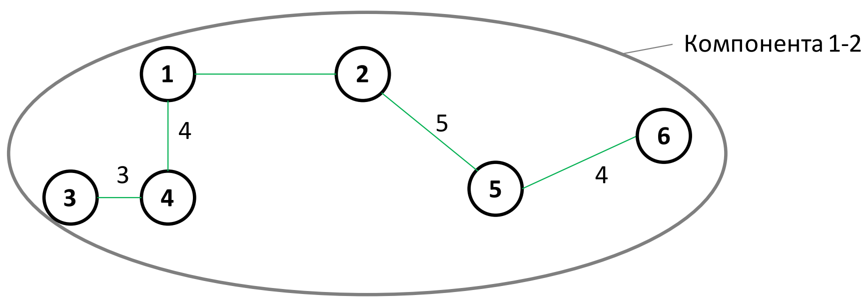


Грана (1,3): случај 1) → боји се у црвено, тј. не укључује се у дрво.

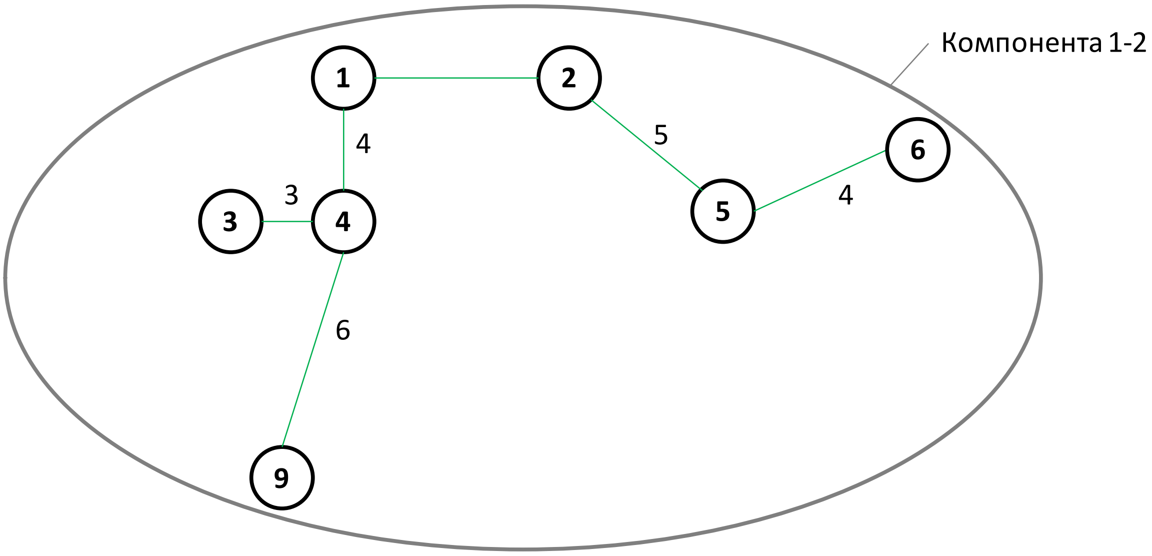
Грана (2,5): случај 2) → боји се у зелено и прикључује компоненти 2:



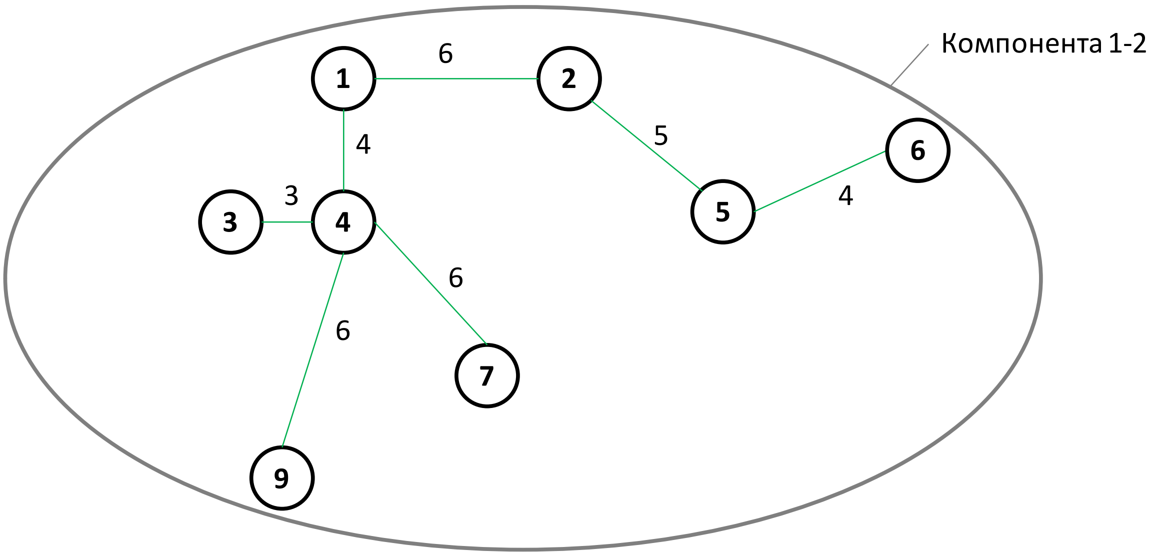
Грана (1,2): случај 4) → боји се у зелено, а компоненте 1 и 2 се спајају:



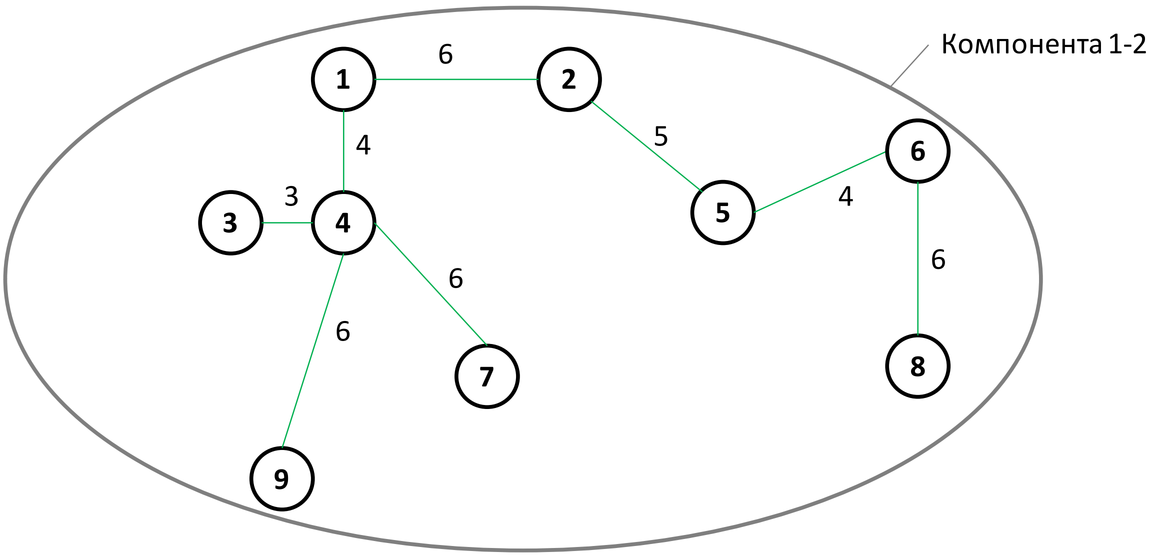
Грана (4,9): случај 2) → боји се у зелено и прикључује компоненти 1-2:



Грана (4,7): случај 2) → боји се у зелено и прикључује компоненти 1-2:



Грана (6,8): случај 2) → боји се у зелено и прикључује компоненти 1-2:



Грана (5,8): случај 1) → боји се у црвено, тј. не укључује се у дрво.

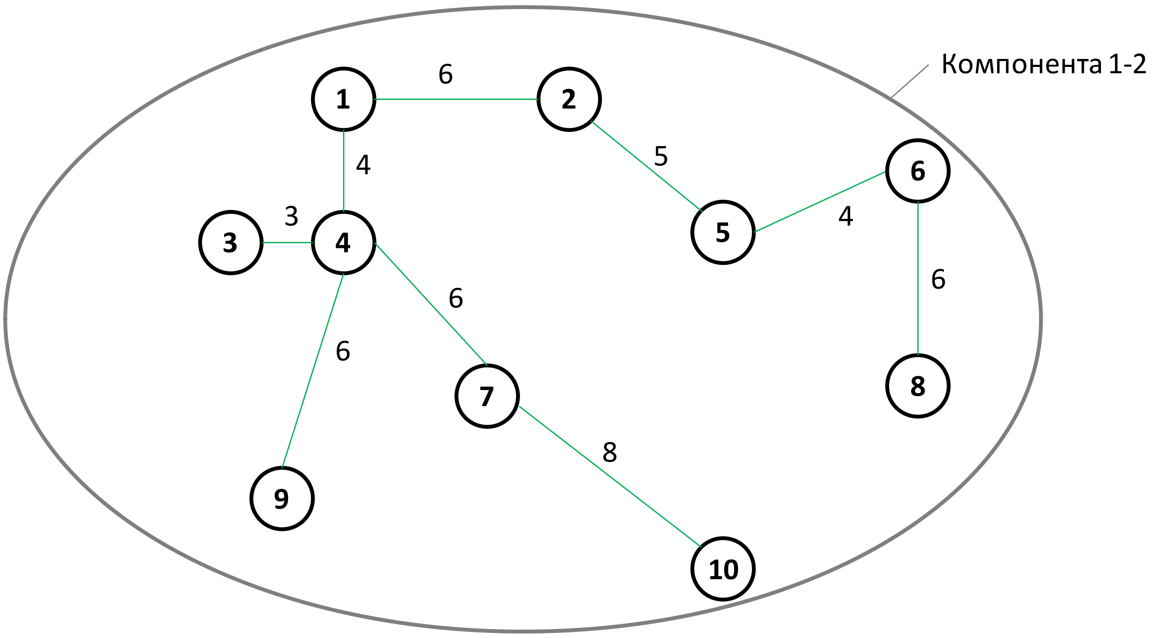
Грана (7,9): случај 1) → боји се у црвено, тј. не укључује се у дрво.

Грана (3,9): случај 1) → боји се у црвено, тј. не укључује се у дрво.

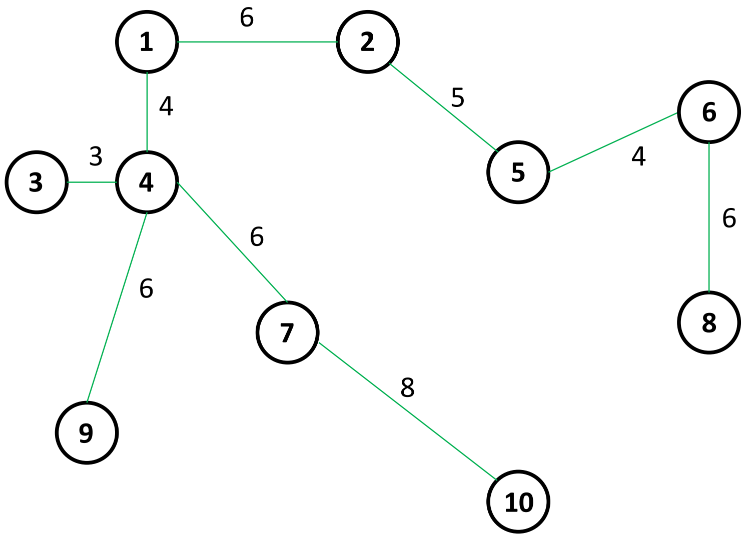
Грана (5,7): случај 1) → боји се у црвено, тј. не укључује се у дрво.

Грана (2,6): случај 1) → боји се у црвено, тј. не укључује се у дрво.

Грана (7,10): случај 2) → боји се у зелено и прикључује компоненти 1-2:



Пошто су сви чворови део исте компоненте, алгоритам се завршава, а разапињуће дрво најмање дужине има облик:

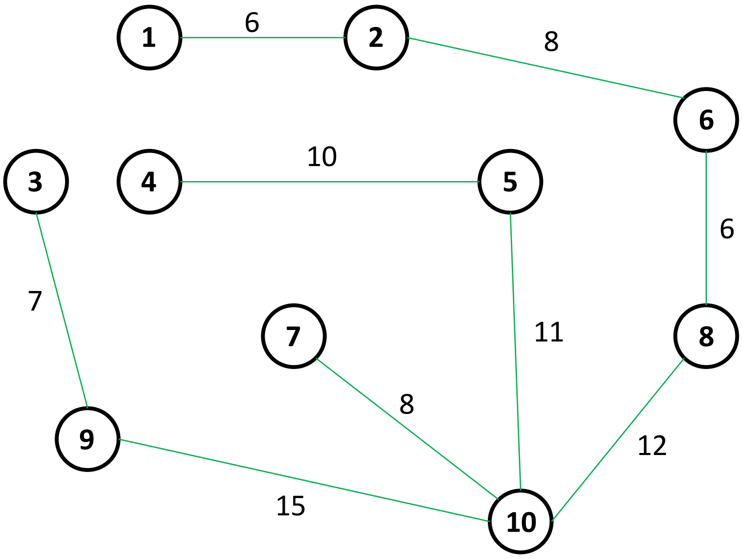


Његова дужина је 6+3+4+6+6+8+5+4+6 = 48.

b) Листа грана сортираних у растућем поретку:

(9,10), (8,10), (5,10), (4,5), (7,10), (2,6), (5,7), (3,9), (7,9), (5,8), (6,8), (4,7), (4,9), (1,2), (2,5), (1,3), (1,4), (5,6), (3,4)

Применом алгоритма за конструкцију разапињућег дрвета добија се разапињуће дрво највеће дужине, које има следећи облик:



Његова дужина је 6+8+6+12+11+8+15+7+10 = 83.

3 ОДРЕЂИВАЊЕ НАЈКРАЋИХ ПУТЕВА ИЗМЕЂУ ПАРОВА ЧВОРОВА У МРЕЖИ

При кретању између два чвора у мрежи често се сусрећемо са проблемом изналажења оптималног пута којим треба да се крећемо. У зависности од контекста, оптимални пут може да буде најкраћи пут, најдужи пут, најбржи пут, пут са највећим капацитетом, најјефтинији пут, најпоузданији пут итд. Гране у мрежи су окарактерисане „дужином“. Дужину гране треба схватити у најопштијем смислу. Под „дужином“ гране може да се подразумева дужина, време путовања, транспортни трошкови, поузданост итд. У неким ситуацијама је потребно одредити пут на основу два или више критеријума (на пример, водећи рачуна и о трошковима и о времену путовања) .

У следећим поглављима ће се оптимални пут називати најкраћим путем, без обзира да ли се ради о најкраћем, најјефтинијем или најбржем путу, јер је у питању минимизација пута по неком параметру (дужина, трошкови или време).

3.1 Алгоритам Dijkstra-е за одређивање најкраћих путева од једног чвора до свих осталих чворова у мрежи

Dijkstra (1959) је развио један од најефикаснијих и највише коришћених алгоритама за одређивање накраћих путева од једног чвора до свих осталих чворова у мрежи.

Означимо са *a* чвор за који ћемо истраживати најкраће путеве до свих осталих чворова у мрежи. Сваки чвор у мрежи може током овог процеса да се налази у 2 могућа стања и то у тзв. **отвореном стању**, уколико је чвор означен привременим ознакама, или у **затвореном стању**, уколико је означен сталним ознакама.

Почетна растојања између било која два чвора на мрежи дефинишу се на следећи начин:

- Растојање од било ког чвора до њега самог једнако је нули;
- Уколико чворови нису међусобно повезани граном, растојање је бесконачно велико;
- Уколико су чворови међусобно повезани граном, растојање између њих је једнако дужини гране;
- Уколико су два чвора међусобно повезана већим бројем грана, растојање између њих је једнако дужини накраће гране која их повезује.

Сваки чвор у мрежи *i* означава се са две ознаке:

- d_{ai} - дужина најкраћег пута од чвора *a* до чвора *i*, откривена до тренутка у коме посматрамо транспортну мрежу,
- q_i – чвор који се налази испред чвора *i* на најкраћем путу од чвора *a* до чвора *i*, који је је откривен до тренутка у коме посматрамо транспортну мрежу (чвор-претходник чвору *i*).

Нека је *c* последњи чвор за кога смо у процесу налажења најкраћих путева утврдили да се налази у затвореном стању. Чвор-претходник чвора *a* означавамо симболом +.

Алгоритам Dijkstra-е се састоји од следећих корака:

Корак 1: Процес започињемо од чвора *a*. Пошто је дужина најкраћег пута од чвора *a* до чвора *a* једнака нули, онда је $d_{aa} = 0$. Чвор-претходник чвора *a* се означава симболом +, па је $q_a = +$. Пошто су дужине накраћих путева од чвора *a* до свих осталих чворова $i \neq a$ за сада неистражене, ставимо привремено $d_{ai} = \infty$ за $i \neq a$. Пошто су и чворови претходници чворовима $i \neq a$ на најкраћим путевима непознати, ставимо $q_i = -$ за све $i \neq a$. Једини чвор који је тренутно у затвореном стању је чвор *a*. Због тога је $c = a$.

Корак 2: Да би неке од привремених ознака трансформисали у сталне, испитајмо све гране (c, i) које излазе из последњег чвора који је у затвореном стању (чвор *c*). Уколико је и чвор *i* у затвореном стању, пређимо на испитивање следећег чвора. Уколико је чвор *i* у отвореном стању, његову прву ознаку d_{ai} добијамо на основу релације:

$$d_{ai} = min\{d_{ai}, d_{ac} + d(c, i)\}$$

при чему је на левој страни ове релације нова ознака чвора *i*. Ознака d_{ai} са десне стране релације представља стару ознаку чвора *i*.

Корак 3: Да би одредили који је следећи чвор који ће из отвореног прећи у затворено стање, упоредимо величине d_{ai} свих чворова који се налазе у отвореном стању. Изаберимо чвор са најмањом вредношћу величине d_{ai}. Нека је то неки чвор *j*. Чвор *j* прелази из отвореног у затворено стање, с обзиром да не постоји пут од *a* до *j* краћи од d_{aj}. Пут преко било ког другог чвора био би дужи.

Корак 4: Одредимо чвор-претходник чвору *j* на најкраћем путу који води од чвора од *a* до чвора *j*. Испитајмо дужине свих грана (i, j) које воде од чворова који се налазе у затвореном стању до чвора *j* све док не утврдимо да је испуњена релација:

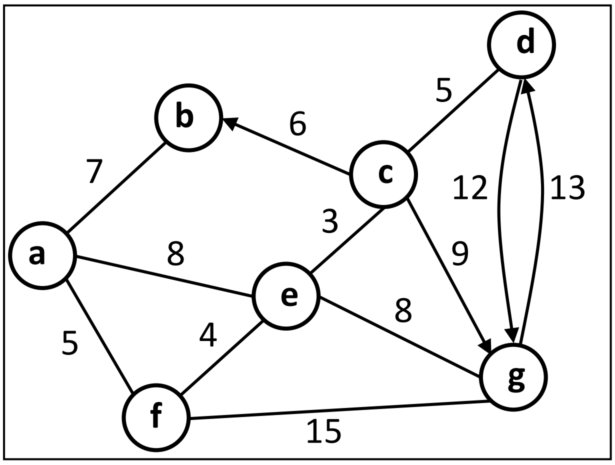
$$d_{aj} - d(i, j) = d_{ai}.$$

Нека је ова релација испуњена за неки чвор t. Ово значи да је чвор t чвор-претходник чвору *j* на најкраћем путу који води од чвора *a* до чвора *j*. Дакле, можемо написати да је $q_j = t$.

Корак 5: Чвор *j* се налази у затвореном стању. Уколико се сви чворови у мрежи налазе у затвореном стању, тада је поступак завршен. Уколико постоје још неки чворови у отвпреном стању, онда се враћамо на корак 2.

Пример 3-1

Користећи алгоритам Dijkstra-е одредити најкраће путеве од чвора а до свих осталих чворова транспортне мреже, приказане на доњој слици.



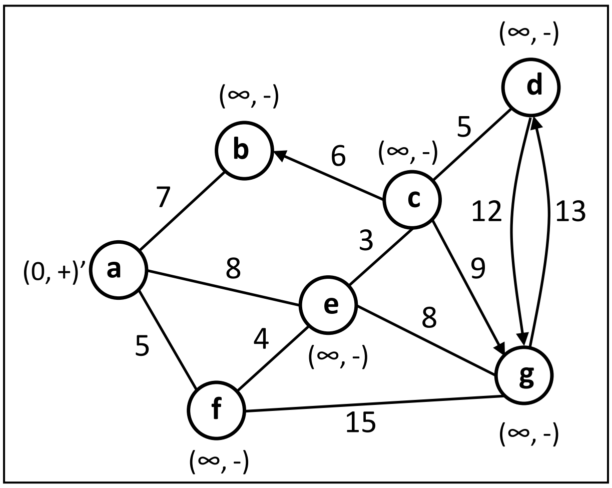
Решење:

1° $d_{aa} = 0 \Rightarrow \mathbf{c} = \mathbf{a}$

$\mathbf{q_a} = +$

$d_{ai} = \infty \, za \, \forall i \neq a$

$q_i = - \, za \, \forall i \neq a$



Čvorovi u **zatvorenom** stanju	Čvorovi u **otvorenom** stanju
a	

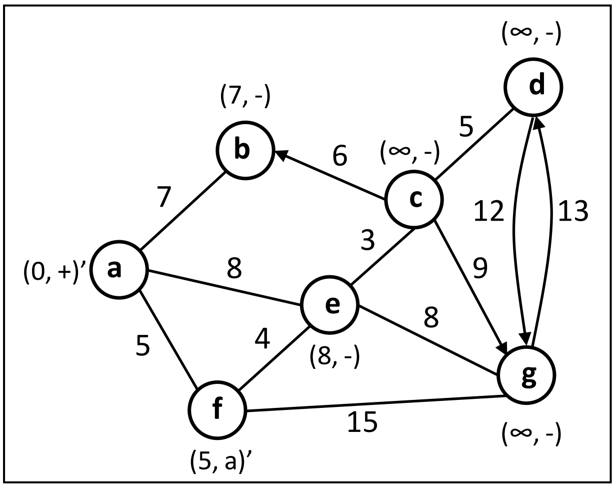
2° $d_{ab} = \min(\infty, d_{aa} + d(a,b)) = \min(\infty, 0 + 7) = 7$

$$d_{ae} = \min(\infty, d_{aa} + d(a,e)) = \min(\infty, 0+8) = 8$$

$$d_{af} = \min(\infty, d_{aa} + d(a,f)) = \min(\infty, 0+5) = 5$$

$$d_{\min} = \min(d_{ab}, d_{ae}, d_{af}) = \min(7,8,5) = 5 = d_{af} \Rightarrow \mathbf{c} = \mathbf{f}$$

$$d_{af} - d(a,f) = 5-5 = 0 = d_{aa} \Rightarrow \mathbf{q_f} = \mathbf{a}$$



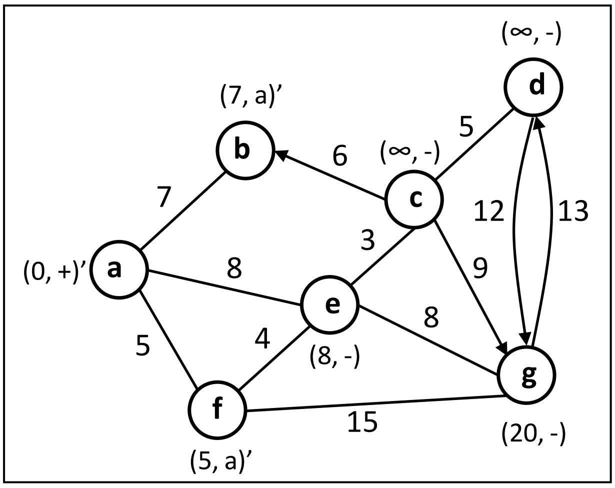
Čvorovi u **zatvorenom** stanju	Čvorovi u **otvorenom** stanju
a, f	b, e, ~~f~~

3° $d_{ae} = \min(8, d_{af} + d(f,e)) = \min(8, 5+4) = 8$

$$d_{ag} = \min(\infty, d_{af} + d(f,g)) = \min(\infty, 5+15) = 20$$

$$d_{\min} = \min(d_{ab}, d_{ae}, d_{ag}) = \min(7,8,20) = 7 = d_{ab} \Rightarrow \mathbf{c} = \mathbf{b}$$

$$d_{ab} - d(a,b) = 7-7 = 0 = d_{aa} \Rightarrow \mathbf{q_b} = \mathbf{a}$$



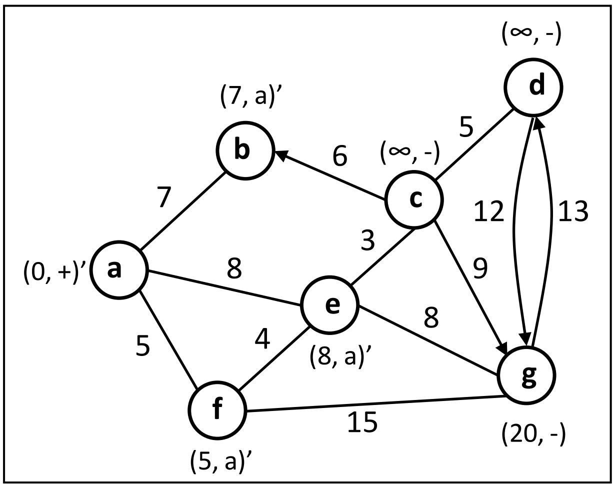
Čvorovi u **zatvorenom** stanju	Čvorovi u **otvorenom** stanju
a, f, b	~~b~~, e, g

4° Из чвора *b* не постоји веза ка другим не-затвореним чворовима, па се преко њега не могу рачунати дужине путева од чвора *a* до осталих чворова. Зато се тражи минимална путања само до оних чворова који су отворени у претходном кораку (тј. *e* i *g*):

$$d_{\min} = \min\left(d_{ae}, d_{ag}\right) = \min(8, 20) = 8 = d_{ae} \Rightarrow \mathbf{c} = \mathbf{e}$$

$$d_{ae} - d(a, e) = 8 - 8 = 0 = d_{aa} \Rightarrow \mathbf{q_e} = \mathbf{a}$$

$$d_{ae} - d(f, e) = 8 - 4 = 4 \neq d_{af}$$



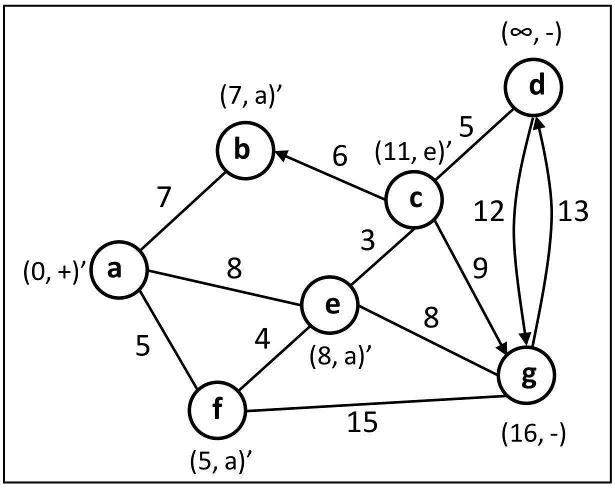
Čvorovi u **zatvorenom** stanju	Čvorovi u **otvorenom** stanju
a, f, b, e	~~e~~, g

5° $d_{ac} = \min(\infty, d_{ae} + d(e,c)) = \min(\infty, 8+3) = 11$

$d_{ag} = \min(20, d_{ae} + d(e,g)) = \min(20, 8+8) = 16$

$d_{\min} = \min(d_{ag}, d_{ac}) = \min(16, 11) = 11 = d_{ac} \Rightarrow \mathbf{c} = \mathbf{c}$

$d_{ac} - d(e,c) = 11 - 3 = 8 = d_{ae} \Rightarrow \mathbf{q_c} = \mathbf{e}$



Čvorovi u **zatvorenom** stanju	Čvorovi u **otvorenom** stanju
a, f, b, e, c	g, ~~e~~

6° $d_{ad} = \min(\infty, d_{ac} + d(c,d)) = \min(\infty, 11+5) = 16$

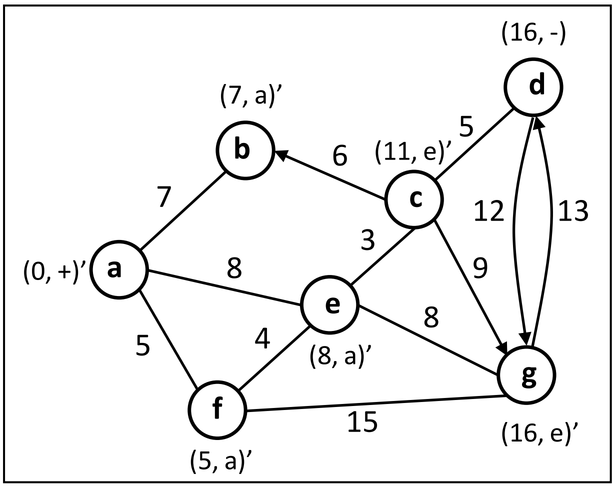
$d_{ag} = \min(16, d_{ac} + d(c,g)) = \min(16, 11+9) = 16$

$d_{\min} = \min(d_{ag}, d_{ad}) = \min(16, 16) = 16 = d_{ag} \Rightarrow \mathbf{c} = \mathbf{g}$

$d_{ag} - d(c,g) = 16 - 9 = 7 \neq d_{ac}$

$d_{ag} - d(e,g) = 16 - 8 = 8 = d_{ae} \Rightarrow \mathbf{q_g} = \mathbf{e}$

$d_{ag} - d(f,g) = 16 - 15 = 1 \neq d_{af}$

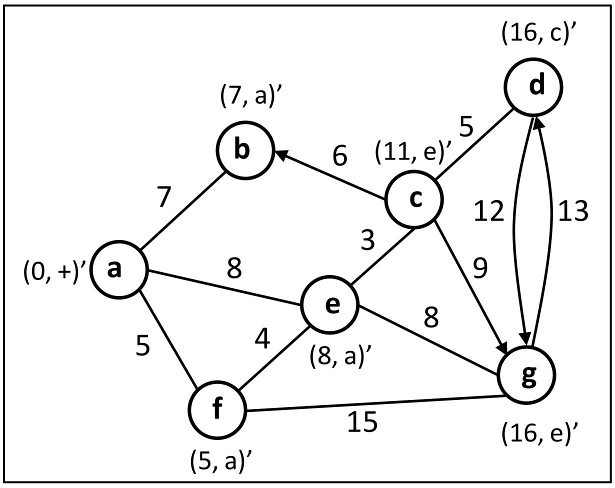


Čvorovi u **zatvorenom** stanju	Čvorovi u **otvorenom** stanju
a, f, b, e, c, g	~~g~~, d

7° $d_{ad} = \min\left(16, d_{ag} + d(g,d)\right) = \min(16, 16+13) = 16 \Rightarrow \mathbf{c} = \mathbf{d}$

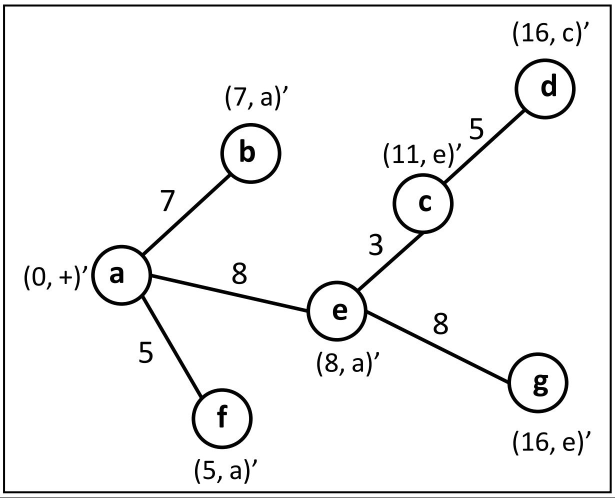
$d_{ad} - d(c,d) = 16 - 5 = 11 = d_{ac} \Rightarrow \mathbf{q_d} = \mathbf{c}$

$d_{ad} - d(g,d) = 16 - 13 = 3 \neq d_{ag}$



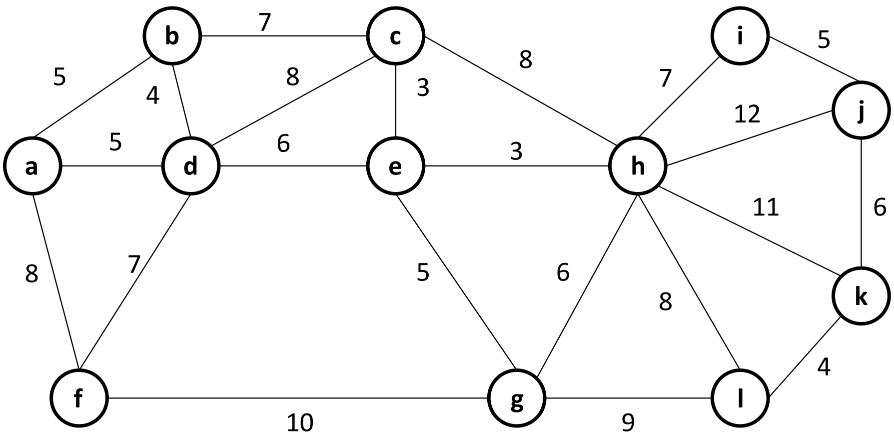
Čvorovi u **zatvorenom** stanju	Čvorovi u **otvorenom** stanju
a, f, b, e, c, g, d	~~d~~

Приказ најкраћих путева од чвора *a* до осталих чворова:



Пример 3-2

Користећи алгоритам Dijkstra-е одредити најкраће путеве од чвора a до свих осталих чворова транспортне мреже, приказане на доњој слици.



3.2 Floyd-ов алгоритам за изналажење најкраћих путева између свих парова чворова

Посматрајмо транспортну мрежу $G = (N, A)$, која има n чворова. Уведимо следеће појмове:

D_0 - почетна матрица дужина најкраћих путева;

Q_0 - почетна матрица чворова-претходника;

d_{ij}^{k} – дужина најкраћег пута од чвора i до чвора j, који је откривен у k–том пролазу кроз алгоритам;

q_{ij}^{k} – чвор-претходник чвору j на најкраћем путу од чвора i, који је откривен у k–том пролазу кроз алгоритам;

Елементи матрице D_0 су дефинисани на следећи начин:

$$d_{ij}^0 = \begin{cases} d(i,j)\,, \quad \text{ако између } i \text{ и } j \text{ постоји грана} \\ min\{d_1(i,j), d_2(i,j), \dots, d_m(i,j)\}, \text{ако између } i \text{ и } j \text{ постоји више грана} \\ \infty, \text{ако између } i \text{ и } j \text{ не постоји грана} \\ 0, \text{ако } i = j \end{cases}$$

Елементи матрице Q_0 су дефинисани на следећи начин:

$$q_{ij}^0 = \begin{cases} 0, i \neq j \\ -, i = j \end{cases}$$

Floyd-ов алгоритам се састоји из следећих корака:

<u>Корак 1:</u> Нека је $k = 1$.

<u>Корак 2:</u> Елементе d_{ij}^k матрице дужина најкраћих путева, откривених завршно са k-тим пролазом кроз алгоритам, D_k, израчунавамо помоћу релације:

$$d_{ij}^k = min\{d_{ij}^{k-1},\ d_{ik}^{k-1} + d_{kj}^{k-1}\}\,.$$

<u>Корак 3:</u> Елементи q_{ij}^k матрице чворова-претходника, откривених завршно са k-тим пролазом кроз алгоритам, Q_k, израчунавамо помоћу релације:

$$q_{ij}^k = \begin{cases} q_{kj}^{k-1}, d_{ij}^k \neq d_{ij}^{k-1} \\ q_{ij}^{k-1}, \text{у осталим случајевима} \end{cases}$$

<u>Корак 4:</u> Ако је $k = n$, завршити са алгоритмом. Ако је $k < n$, повећати k за 1, тј. ставити да је $k = k + 1$ и вратити се на корак 2.

4 ПРОБЛЕМ ТРГОВАЧКОГ ПУТНИКА

4.1 Кристофидесов хеуристички алгоритам за конструкцију руте трговачког путника

Кристофидесов хеуристички алгоритам припада класи алгоритама за конструкцију руте трговачког путника. Састоји се од следећих корака:

Корак 1: Пронаћи најкраће припадајуће дрво које спаја n чворова. Означити то дрво са A.

Корак 2: Нека је k од n чворова дрвета А непарног степена (k је увек паран број). Извршити спаривање по 2 од ових k чворова тако да укупна дужина грана које спајају ове чворове буде најмања. Ових k чворова са одговарајућим гранама, добијеним спаривањем, чине мрежу B. Нацртати мрежу C која представља унију мреже B и мреже A.

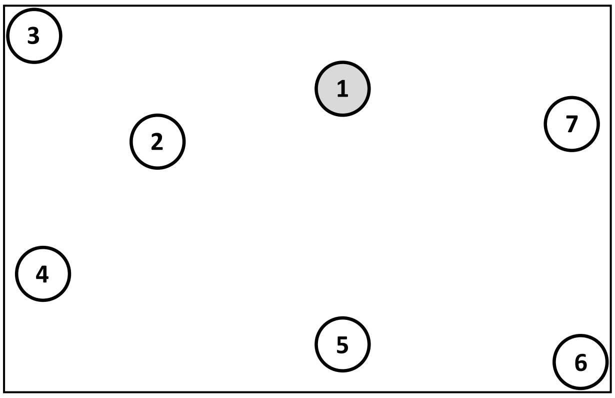
Корак 3: Мрежа C не садржи чворове непарног степена. Нацртати Ојлерову туру у мрежи C. Ова Ојлерова тура је приближно решење за проблем трговачког путника.

Корак 4: Проверити који су све чворови посећени више од једног пута и побољшати руту трговачког путника добијену после трећег корака узимањем у обзир неједнакости:

$$d(a,b) \leq d(a,c) + d(c,b)$$

Пример 4-1

Применом Кристофидесовог хеуристичког алгоритма конструисати руту трговачког путника у случају мреже приказане на слици. Растојања између свих парова чворова дата су у табели. Претпоставимо да рута мора да започне и да се заврши у чвору 1.

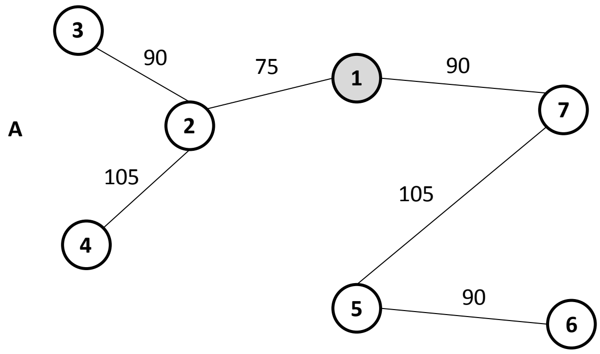


	1	2	3	4	5	6	7
1	0	75	135	165	135	180	90
2	75	0	90	105	135	210	150
3	135	90	0	150	210	300	210

4	165	105	150	0	135	210	210
5	135	135	210	135	0	90	105
6	180	210	300	210	90	0	120
7	90	150	210	210	105	120	0

Решење:

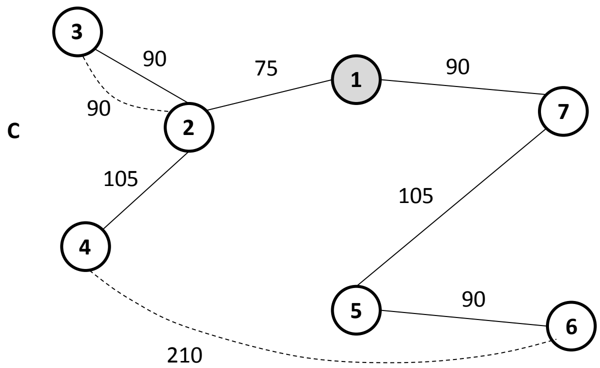
1° За конструкцију разапињућег дрвета најмање дужине може се користити Примов или Крускалов алгоритам. С обзиром да је удаљеност између чворова дата табеларно, коришћен је Примов алгоритам. Тако је добијена мрежа *А*.



2° У мрежи А има четири чвора непарног степена (чворови 2, 3, 4 и 6). Могућа су следећа спаривања ова четири чвора: (2-3) и (4-6), (2-4) и (3-6) и (2-6) и (3-4). Најмања дужина грана добија се за спаривање (2-3) и (4-6). Ове две нове гране чине мрежу В.

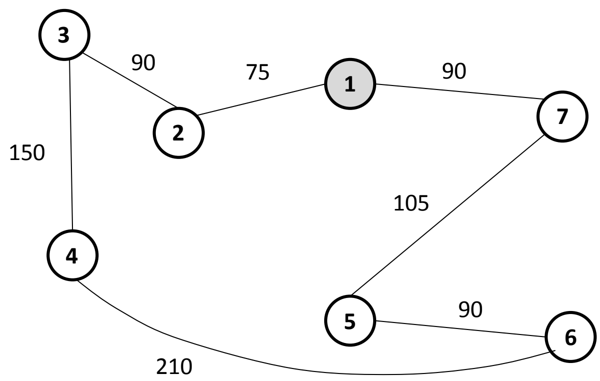
Начин спаривања	Укупна дужина грана
(2-3) и (4-6)	90+210 = 300
(2-4) и (3-6)	105+300 = 405
(2-6) и (3-4)	210+150 = 360

Мрежа С представља унију мрежа А и В.



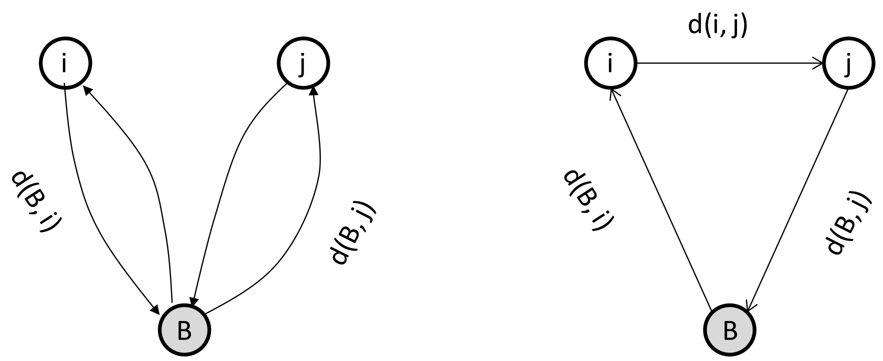
3° Ојлерова тура у мрежи С: (1, 2, 3, 2, 4, 6, 5, 7, 1).

4° У Ојлеровој тури је чвор 2 посећен више од једног пута - (1, **2**, 3, **2**, 4, 6, 5, 7, 1). Део Ојлерове туре, почев од чвора испред прве појаве чвора који је два пута посећен, па до чвора иза друге појаве чвора који је два пута посећен, тј. 1, 2, 3, 2, 4 може се заменити са 1, 2, 3, 4 или 1, 3, 2, 4 (замене се добијају када се са једног места уклони чвор који је два пута посећен). Дужина дела 1, 2, 3, 4 је 75+90+150 = 315, а дела 1, 3, 2, 4 је 135+90+105 = 330. Пошто је дужина дела 1, 2, 3, 4 краћа, онда део Ојлерове туре 1, 2, 3, 2, 4 мењамо са 1, 2, 3, 4. Коначна рута је (1, 2, 3, 4, 6, 5, 7, 1). Дужина ове руте је 810.



4.2 Кларк-Рајтов алгоритам „уштеда" за конструкцију руте трговачког путника и пројектовање рута превозних средстава

Кларк-Рајтов алгоритам „уштеда" је један од најпознатијих алгоритама за пројектовање рута превозних средстава, а може се користити и за конструкцију руте трговачког путника. Заснива се на „уштедама" у кретању трговачког путника које се остварују обилажењем више чворова у једној рути.



Нека је B чвор из кога трговачки путник полази и у који се враћа. Претпоставимо да трговачки путник треба да обиђе чворове *i* и *j*. Ако се креће тако да из базе B прво оде у чвор *i*, затим се врати у базу B, па оде у чвор *j* и врати се у базу B, онда је пређени пут једнак 2d(B, i) + 2d(B, j). Међутим, ако из базе B оде прво у чвор *i*, а затим у чвор *j* и врати се у базу B, онда је трговачки пут прешао пут једнак d(B, i) + d(i, j) + d(B, j). Ако се креће на други начин, онда је „уштеда" у пређеном путу једнака разлици дужина ова два пута: 2d(B, i) + 2d(B, j) – (d(B, i) + d(i, j) + d(B, j)), тј.

$$s(i,j) = d(B,i) + d(B,j) - d(i,j)$$

где је $s(i,j)$ уштеда коју трговачки путник остварује када чворове *i* и *j* споји у једну руту.

Кларк-Рајтов алгоритам „уштеда“ за конструкцију руте трговачког путника (пројектовање рута саобраћајних средстава) састоји се од следећих корака:

Корак 1: Израчунати уштеде $s(i,j)$ за сваки пар чворова (i,j) које треба посетити/опслужити.

Корак 2: Извршити рангирање свих уштеда и поређати их по величини. Направити листу уштеда која започиње са највећом уштедом.

Корак 3: При разматрању уштеде $s(i,j)$, одговарајућу грану (i,j) укључити у делимичну руту ако:

(а) ни чвор i ни чвор j нису били укључени ни у једну делимичну руту;

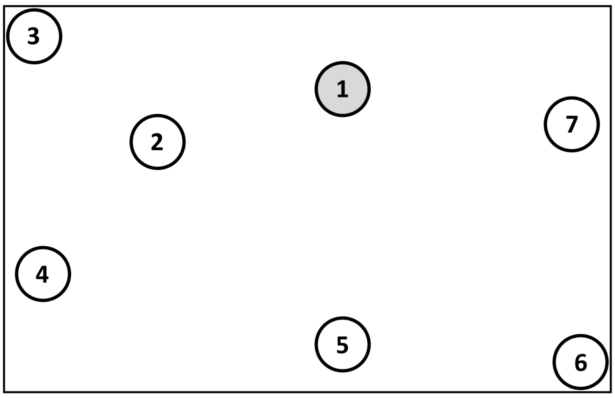
(б) је један од чворова i или j већ укључен у неку постојећу делимичну руту и уколико тај чвор није унутрашњи чвор у рути (чвор је унутрашњи у рути уколико није суседан почетном чвору Б),

(в) су оба чвора i и j већ укључена у две различите делимичне руте и ниједан од тих чворова није унутрашњи у тим рутама (оба су спољна), у ком случају је могуће спојити делимичне руте у једну.

Корак 4: Када је листа „уштеда“ потрошена до краја, завршити са алгоритмом.

Пример 4-2:

Применом Кларк-Рајтовог алгоритма конструисати руту трговачког путника у случају мреже приказане на слици. Растојања између свих парова чворова дата су у табели. Претпоставимо да рута мора да започне и да се заврши у чвору 1.



	1	2	3	4	5	6	7
1	0	75	135	165	135	180	90
2	75	0	90	105	135	210	150
3	135	90	0	150	210	300	210
4	165	105	150	0	135	210	210
5	135	135	210	135	0	90	105
6	180	210	300	210	90	0	120
7	90	150	210	210	105	120	0

Решење:

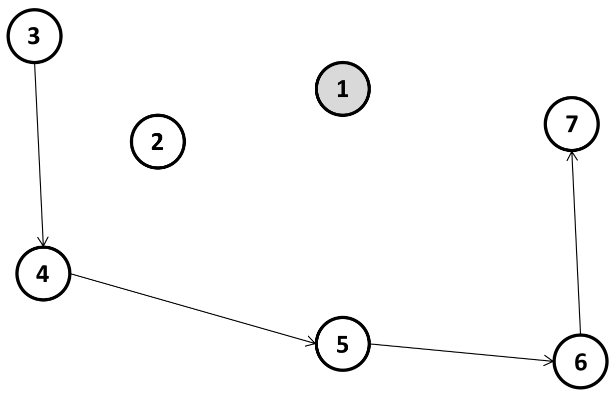
1° Прво се рачунају уштеде између свих парова чворова (не рачунајући 1, који је почетни чвор (база)):

Пар (i, j)	Уштеда
(2,3)	$s(2,3) = d(1,2) + d(1,3) - d(2,3) = 75 + 135 - 90 = 120$
(2,4)	$s(2,4) = d(1,2) + d(1,4) - d(2,4) = 75 + 165 - 105 = 135$
(2,5)	$s(2,5) = d(1,2) + d(1,5) - d(2,5) = 75 + 135 - 135 = 75$
(2,6)	$s(2,6) = d(1,2) + d(1,6) - d(2,6) = 75 + 180 - 210 = 45$
(2,7)	$s(2,7) = d(1,2) + d(1,7) - d(2,7) = 75 + 90 - 150 = 15$
(3,4)	$s(3,4) = d(1,3) + d(1,4) - d(3,4) = 135 + 165 - 150 = 150$
(3,5)	$s(3,5) = d(1,3) + d(1,5) - d(3,5) = 135 + 135 - 210 = 60$
(3,6)	$s(3,6) = d(1,3) + d(1,6) - d(3,6) = 135 + 180 - 300 = 15$
(3,7)	$s(3,7) = d(1,3) + d(1,7) - d(3,7) = 135 + 90 - 210 = 15$
(4,5)	$s(4,5) = d(1,4) + d(1,5) - d(4,5) = 165 + 135 - 135 = 165$
(4,6)	$s(4,6) = d(1,4) + d(1,6) - d(4,6) = 165 + 180 - 210 = 135$
(4,7)	$s(4,7) = d(1,4) + d(1,7) - d(4,7) = 165 + 90 - 210 = 45$
(5,6)	$s(5,6) = d(1,5) + d(1,6) - d(5,6) = 135 + 180 - 90 = 225$
(5,7)	$s(5,7) = d(1,5) + d(1,7) - d(5,7) = 135 + 90 - 105 = 120$
(6,7)	$s(6,7) = d(1,6) + d(1,7) - d(6,7) = 180 + 90 - 120 = 150$

2° Затим се гране (парови чворова) рангирају према уштедама, у опадајућем поретку:

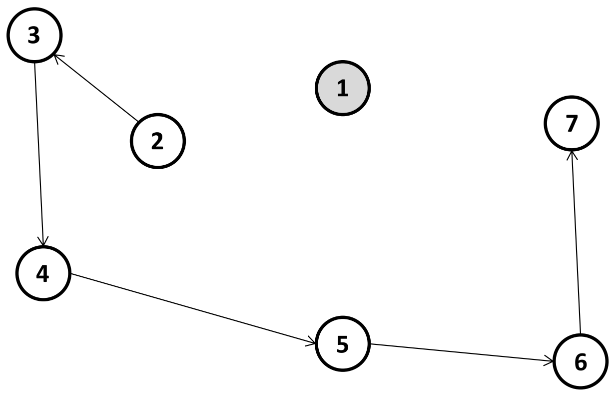
Пар (i, j)	Уштеда
(5,6)	$s(5,6) = 225$
(4,5)	$s(4,5) = 165$
(3,4)	$s(3,4) = 150$
(6,7)	$s(6,7) = 150$
(2,4)	$s(2,4) = 135$
(4,6)	$s(4,6) = 135$
(2,3)	$s(2,3) = 120$
(5,7)	$s(5,7) = 120$
(2,5)	$s(2,5) = 75$
(3,5)	$s(3,5) = 60$
(2,6)	$s(2,6) = 45$
(4,7)	$s(4,7) = 45$
(2,7)	$s(2,7) = 15$
(3,6)	$s(3,6) = 15$
(3,7)	$s(3,7) = 15$

3° Конструкција прве делимичне руте почиње граном са највећом уштедом, тј. са граном (5,6). Затим у прву делимичну рути додајемо грану (4,5) по услову (б) из корака 3 алгоритма. Затим у прву делимичну рути редом додајемо гране (3,4) и (6,7) по услову (б) из корака 3 алгоритма.



Када треба да додамо грану (2,4), видимо да се она не може надовезати на прву делимичну руту, јер чвор 2 не припада ниједној руту, а чвор 4 је унутрашњи у првој делимичној рути. Грана (4,6) такође не може да се дода у прву делимичну руту, јер су и чвор 4 и чвор 6 већ укључени у ту руту.

Грану (2,3) надовезујемо на прву делимичну руту по услову (б) из корака 3 алгоритма.



С обзиром да су сви чворови укључени у ту руту, коначна рута трговачког путника гласи: (1, 2, 3, 4, 5, 6, 7, 1).