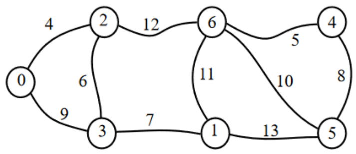
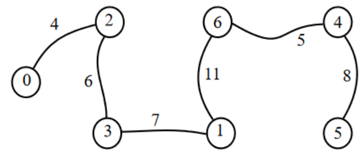
MAT.APP.270 Kevät 2024
Harjoitus 4:

1. Ohessa on esimerkkinä painotettu suuntaamaton graafi G_1.

G_1


T_1


Graafin G_1 kevein virittävä puu on esitetty vieressä, puuna T_1. Seuraavaa lemmaa voidaan käyttää rakentamaan T_1 graafista G_1:

> **Lemma** Olkoon graafissa G silmukka σ. Tällöin graafin G virittävä puu ei voi sisältää silmukan σ raskainta kaarta.

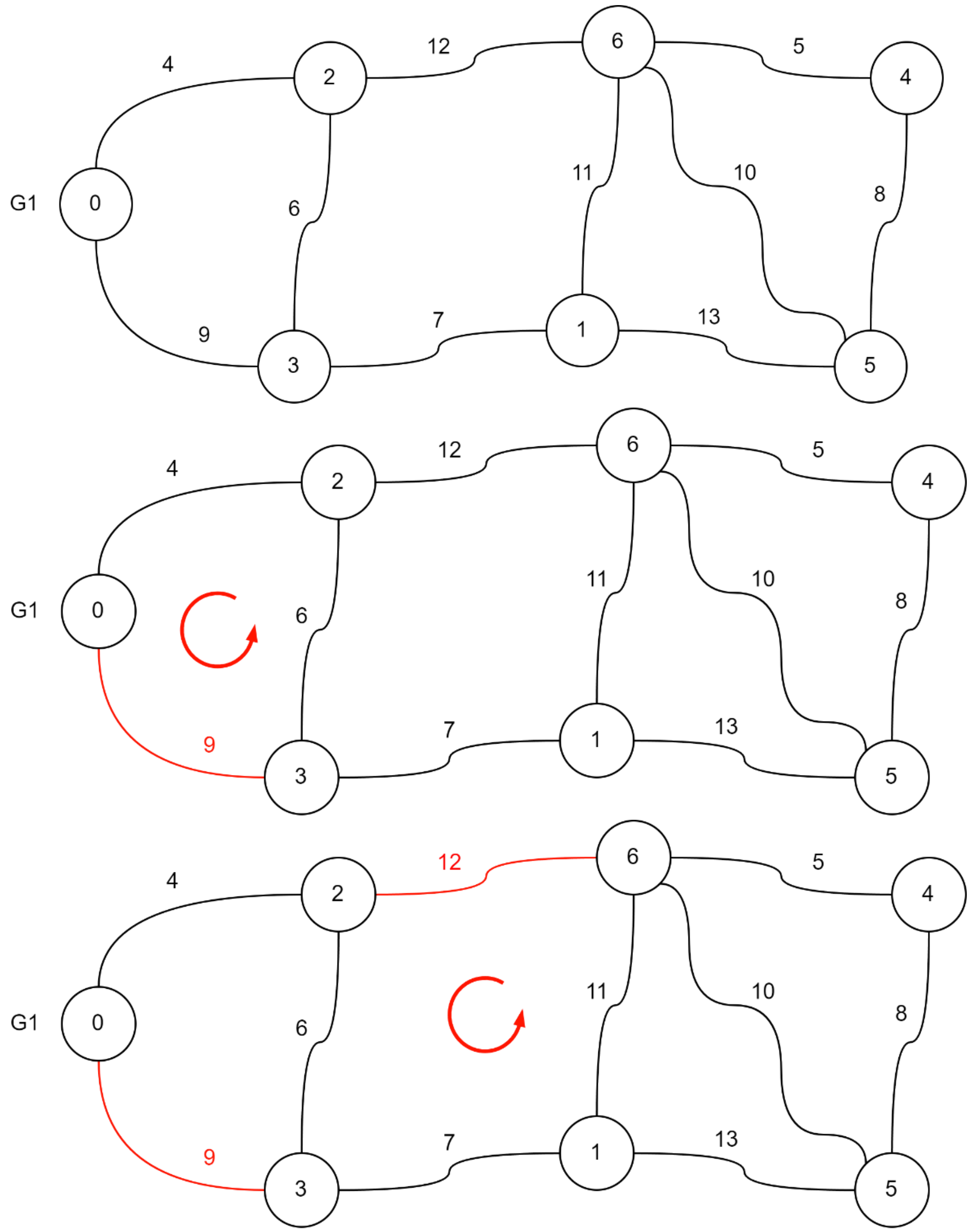
(a) Todista väittämä oikeaksi

(b) Piirrä jono graafeja yksi kerrallaan siten, että T_1 syntyy graafista G_1 käyttäen edellä olevaa lemmaa Kun muodostat graafista G_i graafin G_{i+1}:

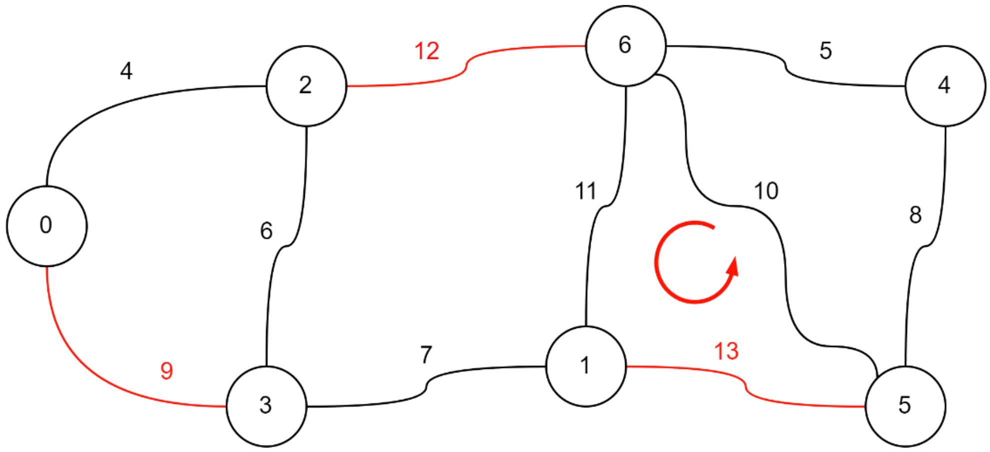
- Tunnista silmukka σ_i graafista G_i
- Poista silmukan raskain kaari, jolloin jäljelle jää graafi G_{i+1}

a) Todistetaan väite todeksi vastaväitteellä. Jos meillä on graafi *G*, jonka kevein virittävä puu on *T* ja *T* pitää sisällään graafin *G* raskaimman kaaren. Kuvitellaan, että meillä on graafi G' joka saadaan, kun graafin *G* silmukasta *σ* poistetaan sen raskain kaari *e*, eli graafi *G'* on graafin *G* aligraafi. Graafin *G'* kevein virittävä puu *T'* on siis oltava myös graafin *G* virittävä puu ja se ei pidä sisällään silmukan *σ* raskainta kaarta *e*. Meillä on siis graafille G kaksi virittävää puuta *T* ja *T'*. Vertailemalla virittävien puiden kaarten painoja huomataan, että puun *T'* on oltava kevyempi kuin puun *T* sillä *T* pitää sisällään silmukan *σ* raskaimman kaaren *e*, kun taas *T'* ei. Siis T ei voi olla graafin *G* kevein virittävä puu. Eli oletus että T voi pitää sisällään silmukan *σ* raskaimman kaaren *e* on väärin, eli väittämä pitää paikkansa.

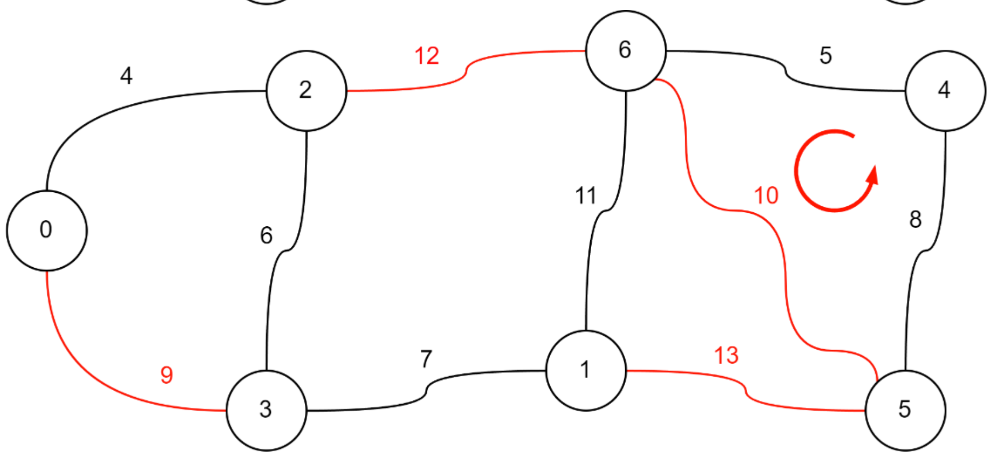
b)


G1
0
2
6
4
3
1
5
4
12
5
11
10
6
8
9
7
13
G1
0
2
6
4
3
1
5
4
12
5
11
10
6
8
9
7
13
G1
0
2
6
4
3
1
5
4
12
5
11
10
6
8
9
7
13

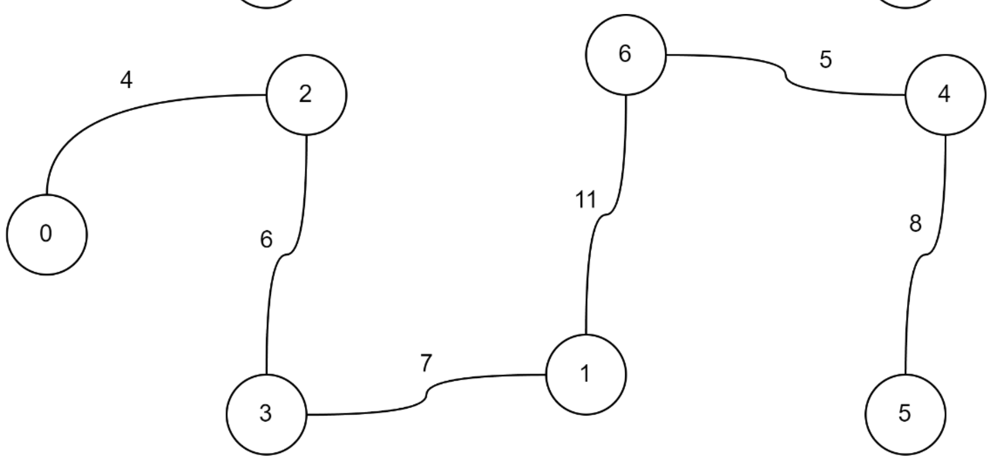
G1



G1



T1



2. Etsi kirjallisuudesta jokin sellainen vahvasti kytkettyjen komponettien algoritmi, jota ei ole esitelty prujussa. (Esim Kosaraju) Selitä sen toimintaperiaate (voit käyttää pseudokoodia).

Kosarajun algoritmi on O(n) algoritmi, jolla voidaan löytää suunnatun graafin vahvasti kytketyt komponentit.

Algoritmi: Kosaraju

```
Kosaraju(G)
    V, E = G                          // graafin solmut ja kaaret
    L := {}                           // alustetaan tyhjä pino
    visited = []                      // Lista siitä missä solmuissa on vierailtu

    for u ∈ V do
        visited[u] = 0                // alustetaan kaikki graafin solmut vierailemattomiksi
    for u ∈ V do                      // Kaikille graafin solmuille
        if u not visited do           // Jos solmussa ei ole jo vierailtu
            DFS-Visit(u)              // Toteutetaan sille DFS
    for u ∈ V do
        visited[u] = 0                // alustetaan kaikki graafin solmut vierailemattomiksi
    while L is not empty              // Kun kaikissa solmuissa on vierailtu, käydään pino läpi
        u = L.pop()                   // Otetaan pinon päällimmäinen solmu
        if u not visited do           // Jos solmussa ei ole jo vierailtu
            SCC = Find-SCC(u)         // Etsitään solmujen vahvasti kytketyt komponentit
            // SCC eli vahvasti kytketty komponentti voidaan nyt lisätä johonkin listaan, tai
            prosessoida muuten halutulla tavalla


DFS-Visit(u)
    visited[u] = 1                    // Merkataan solmu vierailluksi
    for (v, u) ∈ E                    // Eli kaikille solmun u naapureille v
        if v not visited do           // Jos solmussa ei ole jo vierailtu
            DFS-Visit(v)              // Toteutetaan sille DFS
    L.push(u)                         // Lisätään u pinoon


Find-SCC(u)
    visited[u] = 1                    // Merkataan solmu vierailluksi
    SCC = [u]                         // Alustetaan uusi vahvasti kytketty komponentti
    for (v, u) ∈ E do                 // Eli kaikille solmun u naapureille v
        if v not visited do           // Jos solmussa ei ole jo vierailtu
            SCC += Find-SCC(v)        // Lisätään löydetyt vahvasti kytketyt solmut tähän
    Return SCC                        // Palautetaan vahvasti löydetty komponentti
```

3. Olkoon graafi $G = (V, E)$ kytketty. Muodostetaan niin sanottu *komponenttigraafi* $G_c = (V_c, E_c)$ siten että

$$V_c = \{C_u \mid u \in V\}$$

$$E_c = \{(C_u, C_v) \mid C_u \neq C_v \land \exists x \in C_u, y \in C_v : (x, y) \in E\}$$

, eli komponenttigraafin solmut koostuvat alkuperäisen graafin komponenteista ja komponentista C_u on kaari komponenttiin C_v jos ja vain komponentit ovat eri ja jostain komponentin C_u solmusta on kaari johonkin komponentin C_v solmuun alkuperäisessä graafissa. Totea ensimmäisenä itsellesi ettei graafissa G_c ole suunnattuja silmukoita.

(a) Seuraako edellä mainitusta että G_c puu? Jos seuraa, niin todista, jos ei, niin anna kaksi graafia siten että ensimmäisen komponettigraafi ei ole puu ja toisen komponenttigraafi on puu.

(b) Sanomme että komponentti on *lopullinen* jos mistään sen solmusta ei ole kaarta solmuun joka kuuluisi eri komponenttiin. Komponenttigraafissa tällaista komponenttia vastaavasta solmusta ei siten lähde yhtään kaarta. Osoita että jokaisessa graafissa on ainakin yksi lopullinen komponentti.

Graafissa G_c ei voi olla suunnattuja silmukoita. Tehdään vastaoletus että komponenttigraafissa G_c olisi suunnattu silmukka $C_1, C_2, \ldots, Ck$, missä C_i on joku graafin G komponenteista. On siis olemassa kaari kaikkien solmujen C_i ja C_{i+1} välillä, sillä C_{k+1} on C_1.

a) Tästä seuraa, että G_c on puu. Koska se on kytketyn graafin G komponentti, sen on myös oltava kytketty ja koska se ei voi pitää sisällään silmukkaa on se siis puu.

b) Koska G_c on puu, on sillä oltava vähintään yksi lehti. Lehti on sellainen solmu, josta ei lähde yhtäkään kaarta. Ja jos graafin G komponenttigraafi on aina puu, on aina löydyttävä lehti. Siis väittämä pitää paikkansa.

4. Sanomme että graafi on *puolikytketty* (semi-connected) jos kaikille sen solmuille u, v pätee että $d(u,v) < \infty$ tai $d(v,u) < \infty$, eli polku on aina olemassa jompaan kumpaan suuntaan, mutta ei välttämättä molempiin. Anna algoritmi joka selvittää onko graafi puolikytketty. *Vihje:* Vahvasti kytketty komponentti liittyy asiaan erittäin olennaisesti. Mieti edellistä tehtävää. Käytä tarvittaessa internet-lähteitä.

Tehtävässä 2 esitetyllä Kosarajun algoritmilla voidaan etsiä kaikki graafin vahvasti kytketyt komponentit. Jos vahvasti kytkettyjä komponentteja on useampi kuin 1 ja yksi vahvasti kytketyistä komponenteista pitää sisällään kaikki graafin solmut niin graafi on puolikytketty. Siis käytetään esim. Kosarajun algoritmia etsimään kaikki graafin vahvasti kytketyt komponentit, minkä jälkeen käydään ne läpi ja tarkistetaan jos jokin niistä pitää sisällään kaikki graafin solmut.

5. Osoita että virtaukselle (s, t, f) pätee:

$$|f| = \sum_{u \in V} f(u,t) = \sum_{u \in V} f(s,u)$$

Kaikille solmuille $u \in V$ poissulkien lähdesolmu s ja nielusolmu t pätee, että kaiken solmuun tulevan virtauksen on myös lähdettävä siitä:

$$\sum_{v \in V} f(v,u) - \sum_{v \in V} f(u,v) = 0$$

Nielusolmuun tulevien virtausten summa on:

$$\sum_{u \in V} f(u,t)$$

Lähdesolmusta lähtevien virtausten summa on:

$$\sum_{u \in V} f(s,u)$$

Kun kaikkien solmujen väliset virtaukset summataan yhteen saadaan:

$$\sum_{u \in V} (\sum_{v \in V} f(v,u) - \sum_{v \in V} f(u,v)) = 0$$

$$\sum_{u \in V} \sum_{v \in V} f(v,u) - \sum_{u \in V} \sum_{v \in V} f(u,v) = 0$$

Summat yhtälön vasemmalla puolella kuvaavat graafin solmuihin tulevaa virtausta ja niistä lähtevää virtausta. Kun tiedetään että kaikilla muilla paitsi lähde- ja nielusolmuilla summien erotus on 0 voidaan esittää

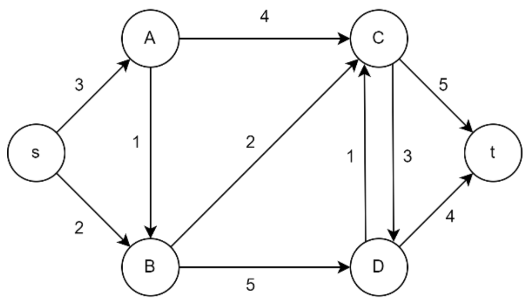
$$\sum_{u \in V} f(u,t) - \sum_{u \in V} f(s,u) = 0$$

Eli

$$|f| = \sum_{u \in V} f(u,t) = \sum_{u \in V} f(s,u)$$

6. Piirrä virtausverkko jossa on ainakin 6 solmua (mukaan lukien s ja t) ja 10 kaarta. Valitse kapasiteetit kokonaislukuina ja esitä verkolle kolme erilaista virtausta.

Virtausverkko 6 solmulla ja 10 kaarella:



Eri virtaukset virtausverkossa: