### 8.4.1.1 Kaarten luokittelu

Syvyyssuuntainen läpikäynti luokittelee jokaisen kaaren

$$u \to v \in E(G) \qquad (22)$$

**kaareksi...**

**puussa** (Tree Edge): lähtösolmu $u$ on maalisolmun $v$ *isä* $\pi$-puussa.

Kaarta (22) käsiteltäessä $v.\text{colour} = \text{WHITE}$.

**taakse** (Back Edge): maalisolmu $v$ on lähtösolmun $u$ *(esi-)isä* $\pi$-puussa.

Erikoistapauksena $u = v$ eli yhden kaaren mittainen silmukka solmun $u$ ympäri.

Kaarta (22) käsiteltäessä $v.\text{colour} = \text{GRAY}$.





**eteen** (Forward Edge): lähtösolmu $u$ on maalisolmun $v$ *esi-isä* $\pi$-puussa.

- Kaarta (22) käsiteltäessä $v.\text{colour} = \text{BLACK}$.
- $u.\text{started} < v.\text{started}$.

**sivulle** (Cross Edge) muuten.

- Kaarta (22) käsiteltäessä $v.\text{colour} = \text{BLACK}$.
- $u.\text{started} > v.\text{started}$.

Silloin $u$ ja $v$ ovat

- saman $\pi$-puun *eri ali*puissa
- kokonaan *eri $\pi$-puissa*.

**Suuntaamattoman** verkon $G$ särmät kulkevat joko **puussa** tai **taakse**.





- Jos halutaan luokitella **kaari sivulle** vielä tarkemmin, niin eri $\pi$-puut voidaan numeroida:
  - Liitetään joka solmuun $u \in V(G)$ kenttä $u.\text{treeOf}$.
  - Liitetään pääohjelmaan DFS globaali laskuri treeIn.
  - Laskuria treeIn kasvatetaan joka kerran kun DFS tekee aliohjelmakutsun DFSVisit($u$) — silloinhan alkaa uuden $\pi$-puun muodostaminen.
  - Asetetaan aliohjelmassa DFSVisit($u$) uusi kenttä $u.\text{treeOf}$ :=treeIn kun väritetään $u$ := GRAY.
  - Kaari sivulle $u \to v \in E(G)$ on samassa $\pi$-puussa kun $u.\text{treeOf} = v.\text{treeOf}$.
  - Kaari sivulle $u \to v \in E(G)$ $\pi$-puiden välillä kulkee suuremmasta pienempään: $u.\text{treeOf} > v.\text{treeOf}$





### 8.4.1.2 Syvyyssuuntaisen läpikäynnin ominaisuuksia

- Syvyyssuntaisesta läpikäynnistä voidaan osoittaa (esimerkiksi)

  **sulkumerkkilause** 8.4.1

  **valkopolkulause** 8.4.2.

- Tällaiset tulokset helpottavat syvyyssuntaisen läpikäynnin käyttämistä toisten algoritmien osana, koska niillä voidaan perustella miksi läpikäynnin tuloksesta
  - voidaan lukea toiselta algoritmilta vaadittu lopputulos
  - saadaan toiseen algoritmiin vaadittu välitulos.
- Kalvojen 8.4.1.1 kaariluokittelua voidaan käyttää samoihin tarkoituksiin.

**Lause 8.4.1.** *Ensin läpikäydään verkko $G$ syvyyssuunnassa.*

*Sitten mielivaltaisille solmuille $u, v \in V(G)$ pätee seuraavista ehdoista tasan yksi:*

1. *Aikavälit*

$$\mathcal{U} = u.\text{started} \ldots u.\text{stopped}$$
$$\mathcal{V} = v.\text{started} \ldots v.\text{stopped}$$

   *ovat kokonaan erilliset, ja kumpikaan solmuista ei ole toisen jälkeläinen $\pi$-metsän missään puussa.*

2. *Aikaväli $\mathcal{V}$ on kokonaan aikavälin $\mathcal{U}$ sisällä, ja solmu $v$ on solmun $u$ jälkeläinen $\pi$-metsän jossakin puussa.*

3. *Ehto 2 symmetrisesti päinvastoin.*

- Tulos ja todistus pätevät sekä suunnatuille että suuntaamattmille verkoille.





*Todistus.* Oletetaan aluksi että $u.\text{started} < v.\text{started}$. On 2 tapausta:

1. $v.\text{started} < u.\text{stopped}$:

   - Solmu $v$ löydettiin (eli muutettiin valkeasta harmaaksi) silloin kun solmu $u$ oli yhä harmaa (eli ei vielä musta).
   - Silloin solmu $v$ on solmun $u$ jälkeläinen:
     - Solmu $v$ löydettiin kulkemalla rekursiossa verkon $G$ kaaria eteenpäin solmusta $v$ alkaen.
     - Rekursiossa kuljetusta kaarista tulee saman $\pi$-puun kaaria.
   - Kaikki solmusta $v$ lähtevät kaaret käsitellään ennen kuin solmu $u$ mustuu.

     Siis $v.\text{stopped} < u.\text{stopped}$

   Siis ehto 2 pätee.





2. $v.\text{started} > u.\text{stopped}$:

   - Jokaiselle solmulle $p \in V(G)$ pätee selvästi

$$p.\text{started} < p.\text{stopped}. \qquad (23)$$

   - Siis välit $\mathcal{U}$ ja $\mathcal{V}$ ovat erilliset.
   - Siis kun solmu $v$ löydettiin, ei solmu $u$ ollut harmaa.
   - Siis solmusta $v$ ei voinut tulla solmun $u$ jälkeläistä.
   - Samalla perustelulla solmusta $u$ ei voinut tulla solmun $v$ jälkeläistä.

   Siis ehto 1 pätee.

Toisesta oletuksesta $u.\text{started} > v.\text{started}$ voidaan päätellä symmetrisesti. □





**Lause 8.4.2.** *Läpikäydään verkko $G$ syvyyssuunnassa.*

*Solmu $v \in V(G)$ on solmun $u \in V(G)$ jälkeläinen jossakin syntyneessä $\pi$-puussa*

*jos ja vain jos*

*ajanhetkellä $u.\text{started}$ oli polku*

$$u \to \cdots \to v$$

*jonka kaikki solmut olivat vielä valkeita.*

- Tulos ja todistus pätevät sekä suunnatuille että suuntaamattmille verkoille.

*Todistus.*

**Suunta $\Rightarrow$:**

- Oletetaan, että solmu $v$ on solmun $u$ jälkeläinen $\pi$-puussa.
- Olkoon $w \in V(G)$ mielivaltainen solmu sillä $\pi$-puun polulla, joka kulkee solmusta $u$ solmuun $v$.
- Silloin myös solmu $w$ on solmun $u$ jälkeläinen samassa $\pi$-puussa.
- Lauseesta 8.4.1 nähdään välittömästi, että solmu $y$ on solmun $x$ jälkeläinen syvyyssuuntaisen läpikäynnin $\pi$-puussa täsmälleen silloin, kun

$$x.\text{started} < y.\text{started} < y.\text{stopped} < x.\text{stopped}. \quad (24)$$

- Kun valitaan yhtälössä (24) $x = u$ ja $y = w$ niin saadaan

$$u.\text{started} < w.\text{started}$$

eli $w$ oli valkea vielä ajanhetkellä $u.\text{started}$.





**Suunta $\Leftarrow$:**

- Oletetaan, että ajanhetkellä $u.\text{started}$ solmusta $u$ solmuun $v$ on sellainen polku, joka kaikki solmut ovat vielä valkeita.
- Tehdään *vastaväite:*

  Silti solmusta $v$ ei tule solmun $u$ jälkeläistä $\pi$-puussa.
- Voidaan olettaa lisäksi, että solmu $v$ on valkean polun *ensimmäinen* solmu, josta ei tule jälkeläistä.

  Muuten siirretään mielenkiinto sitä edelliseen sellaiseen solmuun $v'$.
- Olkoon solmu $w$ solmun $v$ välitön edeltäjä valkealla polulla.

  Voi olla myös $u = w$.
- Silloin solmusta $w$ tulee solmun $u$ jälkeläinen $\pi$-puussa.
- Kun valitaan yhtälössä (24) $x = u$ ja $y = w$ niin saadaan

$$w.\text{stopped} \leq u.\text{stopped}.$$





- Koska solmu $v$ on vielä valkea ajanhetkellä $u.\text{started}$, täytyy olla

$$u.\text{started} < v.\text{started}.$$

- Koska solmusta $w$ tulee solmun $u$ jälkeläinen $\pi$-puussa, niin solmusta $u$ alkanut rekursio tavoittaa solmun $w$.
- Koska valkealla polulla on kaari $w \to v \in E(G)$, niin solmu $v$ löydetään (tavalla tai toisella) ennen kuin solmu $w$ mustuu:

$$v.\text{started} < w.\text{stopped}.$$

- Siis

$$u.\text{started} < v.\text{started} < u.\text{stopped}.$$

- Sulkumerkkilauseen 8.4.1 nojalla

$$v.\text{stopped} < u.\text{stopped}.$$

- Kun valitaan yhtälössä (24) $x = u$ ja $y = v$ niin saadaan haluttu *ristiriita:*

  Solmusta $v$ tulee sittenkin solmun $u$ jälkeläinen $\pi$-puussa. □





## 8.4.1.3 Onko verkossa sykli?

- Syötteenä annetaan suunnattu verkko $G$. Vastauksena halutaan tietää onko siinä syklejä vaiko ei.
- *Verkossa $G$ on sykli*

  *jos ja vain jos*

  *sen syvyyssuuntainen läpikäynti luokittelee jonkin kaarista kaareksi* ***taakse****.*
- Solmujen läpikäyntijärjestys
  - ei vaikuta itse vastaukseen
  - vaikuttaa siihen mikä kaarista luokitellaan **taakse**.
- Vastaus voidaan laskea syvyyssuuntaisella läpikäynnillä kalvojen 8.4.1.1 tapaan.
- Menetelmän toimivuus voidaan perustella kalvojen 8.4.1.2 lauseilla.

*Todistus.*

**Suunta $\Rightarrow$:**

- Olkoon verkossa $G$ jokin sykli $c$.
- Olkoon solmu $v \in V(G)$ se syklin $c$ solmuista, joka löydetään ensimmäisenä, kun verkko $G$ läpikäydään syvyyssuunnassa.
- Olkoon
$$u \to v \in E(G) \qquad (25)$$
se syklin $c$ kaarista, joka tulee solmuun $v$.
- Ajanhetkellä $v.$started kaikki syklin $c$ solmut ovat vielä valkeita.
- Erityisesti solmu $u$ on silloin vielä valkea.
- Siis valkopolkulauseen 8.4.2 nojalla solmusta $u$ tulee solmun $v$ jälkeläinen $\pi$-puussa.
- Silloin kaari (25) luokitellaan **taakse**.





**Suunta $\Leftarrow$:**

- Olkoon
$$u \to v \in E(G) \qquad (26)$$
kaari **taakse**.
- Silloin solmu $v$ on solmun $u$ esi-isä $\pi$-puussa.
- Verkossa $G$ on silloin polku
$$u \leftarrow u.\pi \leftarrow u.\pi.\pi \leftarrow u.\pi.\pi.\pi \leftarrow \cdots \leftarrow v.$$
- Kaari (26) täydentää tämän polun kehäksi. □





### 8.4.1.4 Topologinen järjestäminen

- Suunnatun syklittömän verkon $G$ *topologinen* järjestäminen (topological sort) on sen solmujen $V(G)$ luettelointi sellaisessä järjestyksessä, että kaaren
$$p \to q \in E(G)$$
lähtösolmu $p$ mainitaan aina luettelossa *ennen* maalisolmua $q$.
- Tällöin kaari tarkoittaa usein ehtoa

  "Edellinen työvaihe $p$ pitää olla valmis ennen kuin seuraavaan työvaiheeseen $q$ voidaan ryhtyä."
- Tällöin (jokainen) topologinen järjestys on (yksi) tapa tehdä eri työvaiheet oikeassa järjestyksessä.
- Sykliselle verkolle $G$ ei ole tällaista luettelointia:

  Syklissä oleva solmu pitäisi saada valmiiksi jo ennen kuin se itse voidaan aloittaa.





- Osoitetaan, että seuraava menetelmä toimii:
  1. Läpikäy verkko $G$ syvyyssuuntaisesti.
  2. Luettele solmut $p \in V(G)$ niiden loppuaikojen $p.$stopped suhteen *käänteisessä* järjestyksessä.

     Siis
     (a) ensimmäisenä loppunut luetellaan viimeisenä
     (b) toisena loppunut luetellaan toiseksi viimeisenä
     (c) ja niin edelleen.
- Askel 2 voidaan tehdä askeleen 1 osana:
  - Luettelo on (aluksi tyhjä) lista.
  - Kun solmun $u \in V(G)$ loppuaika asetetaan rivillä 12, niin $u$ lisätään luettelon alkuun.

- Menetelmän toiminnan varmistamiseksi riittää osoittaa:

  *Kun suunnattu syklitön verkko $G$ on läpikäyty syvyyssuuntaisesti, niin sen mielivaltaiselle kaarelle*

  $$u \to v \in E(G) \tag{27}$$

  *pätee ehto*

  $$u.\,\text{stopped} > v.\,\text{stopped}\,. \tag{28}$$

*Todistus.* Mikä on kaaren (27) maalisolmun $v$ väri sillä hetkellä kun kun kaari (27) käsitellään?

WHITE: Silloin maalisolmusta $v$ tulee lähtösolmun $u$ lapsi $\pi$-puussa.

Ehto (28) pätee valitsemalla $x = u$ ja $y = v$ yhtälössä (24).





GRAY: Silloin kaari (27) luokiteltaisiin kalvoilla 8.4.1.1 **taakse**.

Silloin verkko $G$ ei olisikaan syklitön kalvojen 8.4.1.3 mukaan.

BLACK: Silloin solmu $v$ on käsitelty loppuun jo ennen kuin kaarta (27) ryhdytään käsittelemään.

- Siis kenttä $v.\,\text{stopped}$ on jo asetettu.
- Toisaalta kaaren (27) lähtösolmun $u \in V(G)$ käsittely on vielä kesken.
- Siis kenttä $u.\,\text{stopped}$ on vielä asettamatta.
- Siis sen arvoksi tullaan asettamaan jotakin aidosti suurempaa kuin kentälle $v.\,\text{stopped}$ asetettu arvo.

Siis ehto (28) pätee. □





### 8.4.1.5 Kriittiset työvaiheet

- (Ohjelmisto)projektissa on tiettyjä *määrähetkiä:*
  - dokumenttien ja komponenttien määräpäiviä
  - (laadunvarmistus)katselmuksia
  - asiakasdemoja
  - ...
- Määrähetkillä on *osittainen järjestys:*
  - Käyttöliittymä(n ensimmäinen versio) on saatava valmiiksi ennen asiakasdemoa.
  - Käyttöliittymää ei voida aloittaa ennen kuin tietokantaliittymä on valmis.
  - ...





- Eri vaiheilla on *kestoja:*
  - Käyttöliittymän aloittamisesta sen valmistumiseen kuluu (arviolta) 4 viikkoa.
  - ...
- Mallinnetaan verkko-ongelmana:

  **Solmuina** määrähetket

  **Kaarina** määrähetkien järjestysrajoitteet

  **Kaaripainoina** kahden määrähetken väliin tarvittava aika.

  **Halutaan** tietää se aika, joka projektiin *vähintään* kuluu.
  - Tarvitaan vastaavan verkon *pisin polku*.
  - Sen pituus on vähimmäiskesto.
  - Sen vaiheet ovat ne jotka *eivät saa myöhästyä* aikataulustaan, tai muuten koko projekti myöhästyy!

    *Kriittinen* (critical) polku.

- Tässä mallinnuksessa unohdetaan käytännön rajoitteista esimerkiksi

  **resurssit** kuten

  "$X$ on ainoa työtekijämme, joka hallitsee sekä käyttöliittymä- että verkkoyhteysohjelmoinnin."

  **disjunktiiviset** kuten

  "Siispä käyttöliittymä on ohjelmoitava *joko* ennen *tai* jälkeen verkkoyhteyden, mutta ei yhtä aikaa."

- Olisi luontevampaa mallintaa toisinpäin:

  **solmuina** olisivatkin työvaiheet

  **kaarina** olisivatkin eri työvaiheiden välttämätön järjestys

  **painoina** olisivat työvaiheiden kestot *solmuissa* ja kaaret painottomia.

  Vastaava muutos esiteltävään menetelmään olisi helppo ja sivuutetaan.





- Esimerkki tilanteesta jossa *tulos solmussa $p$ voidaan laskea kun tunnetaan tulokset sen seuraajissa $q$* (eli $p \to q \in E(G)$):
  - Liitetään jokaiseen solmuun $u \in V(G)$ uusi kenttä $u.\mathrm{dist} =$ pisimmän solmusta $u$ alkavan polun pituus.
  - Silloin

    $$u.\mathrm{dist} = \max(0, \eta) \tag{29}$$

    kun

    $$\eta = \max\left\{w + v.\mathrm{dist} : u \xrightarrow{w} v \in E(G)\right\}$$

    missä

    $$\max \emptyset = -\infty \tag{30}$$

    koska
    * vaihtoehto 0 edustaa kaaretonta polkua $u$
    * vaihtoehto $\eta$ valitsee pisimmän sellaisen polun, jolla on ainakin yksi kaari.





- Sellainen yhtälö kuin (29) *ei ole kehä*määritelmä silloin kun verkko $G$ on (suunnattu ja) kehätön.

  Projektiesimerkissämme kehällisyys olisi mahdoton projekti: jokin välitavoite on välttämätön edellytys itsensä saavuttamiselle!

- Yhtälö (30) johtuu seuraavasta yleisperiaatteesta:
  - Jos halutaan laskea jokin suure

    $$x_1 \oplus x_2 \oplus x_3 \oplus \cdots$$

  - kun parametrejä $x_i$ *ei ole* yhtään kappaletta
  - niin algebrallisista syistä kannattaa määritellä tulokseksi
  - laskutoimituksen $\oplus$ *neutraalialkio*
  - eli se $y$ jolla tulos ei muutu:

    $$y \oplus x = x \oplus y = x.$$





- Sellaisen yhtälön kuin (29) oikea *soveltamisjärjestys* on kalvojen 8.4.1.4 topologinen järjestys käänteisenä:

  Kenttä $u.\mathrm{dist}$ lasketaan vasta sen jälkeen kun kaikki siihen tarvittavat seuraajakentät $v.\mathrm{dist}$ on laskettu.

- Siis oikea soveltamishetki on verkon $G$ syvyyssuuntaisen läpikäynnin rivi 12, kun juuri loppuun käsitelty solmu $u$ laitettaisiin topologisen järjestyksen alkuun.

- Yhtälön (29) tapauksessa riittävät lisäykset

  $3\frac{1}{2}$: $u.\mathrm{dist} := 0$;
  $8\frac{1}{2}$: $u.\mathrm{dist} := \max(u.\mathrm{dist}, w + v.\mathrm{dist})$
  missä $u \xrightarrow{w} v \in E(G)$;

- Kaari $p \xrightarrow{c} q \in E(G)$ kuuluu johonkin kriittiseen polkuun jos ja vain jos
  1. sen pidentyminen (painosta $c$) kasvattaa kenttää $p.\mathrm{dist}$
  2. sen lähtösolmu $p$ kuuluu johonkin kriittiseen polkuun, jolloin kentän kasvatus vaikuttaa koko verkkoon.

- Verkon $G$ syvyyssuuntaisen läpikäynnin

  **aikana** muistetaan jokaisessa solmussa $u \in V(G)$ myös ne kaaret $u \xrightarrow{w} v \in E(G)$ joilla yhtälö (29) antoi lopullisen arvon $u.\mathrm{dist}$

  (koska juuri niiden pidentyminen kasvattaa lähtösolmun kenttää $u.\mathrm{dist}$)

  **jälkeen** tehdään toinen syvyyssuuntainen läpikäynti

  **aloittaen vain solmuista** $u \in V(G)$ joissa $u.\mathrm{dist}$ saavutti sen kaikkein suurimman löytyneen arvon

  (koska juuri niistä alkavat kaikki kriittiset polut)

  **seuraten vain kaaria** jotka muistettiin edellisen läpikäynnin aikana.

  Seuratut kaaret ovat siis kriittiset kaaret.





## 8.4.2 Leveyssuuntainen läpikäynti

Verkon $G$ *leveyssuuntainen läpikäynti* (Breadth-First Search) saadaan kalvojen 8.4.1 syvyyssuuntaisesta siten, että harmaat solmut viedäänkin (rekursio)pinon sijasta kalvojen 4.3 jonoon.

- Tehdään vain yhdestä *alkusolmusta* $s \in V(G)$ alkaen.

  Osa verkosta $G$ voi jäädä käsittelemättä.

- Alku- ja loppuajoilla ei ole selkeää tulkintaa.

  Ei siis talleteta niitä.

- Vaan talletetaankin solmun $v \in V(G)$ *syvyys* $v.\mathrm{depth}$ = pienin $k$ jolla verkossa $G$ on polku

  $$s = p_0 \to p_1 \to p_2 \to p_3 \to \cdots \to p_k = v.$$

  Jos solmua $v$ ei voi saavuttaa alkusolmusta $s$, niin $v.\mathrm{depth} = +\infty$.





- Verkon $S$ leveyssuuntainen läpikäynti laskee
  - jokaiselle solmulle $v$ joka on saavutettavissa alkusolmusta $s$
  - jonkin *vähäkaarisimman* saavuttavan polun
  - muodossa

  $$v \leftarrow v.\pi \leftarrow v.\pi.\pi \leftarrow \cdots \leftarrow s. \qquad (31)$$

- Periaate kuten kalvojen 6.3.4.2 vaiheessa 1:

  **Valkeaan** solmuun $v$ ei vielä ole edetty mitään polkua (31) pitkin.

  **Harmaaseen** solmuun $v$
  - on jo edetty jotakin polkua (31) pitkin
  - mutta solmusta $v$ ei vielä ole edetty eteenpäin tätä polkua (31) pitkin
  - vaan tämä polku (31) odottaa pidentymistä sen loppusolmu $v$ työjonossa $Q$.

  **Musta** solmu on jo pidennetty.





**procedure** BFS($G$: **verkko**, $s$: **solmu**)

```
 1: for all u ∈ V(G) \ {s} do
 2:     u.colour := WHITE;
 3:     u.depth := +∞;
 4:     u.π := NULL
 5: end for;
 6: s.colour := GRAY;
 7: s.depth := 0;
 8: s.π := NULL;
 9: Q := makeEmptyQueue( );
10: enqueue(Q, s);
11: while not isEmpty(Q) do
12:     u := dequeue(Q);
13:     for all u → v ∈ E(G) do
14:         if v.colour = WHITE then
15:             v.colour := GRAY;
16:             v.depth := u.depth +1;
17:             v.π := u;
18:             enqueue(Q, v);
19:         end if
20:     end for;
21:     u.colour := BLACK
22: end while.
```

**Huomioita leveyssuuntaisesta läpikäynnistä:**

(Sekä suunnatuille että suuntaamattomille verkoille.)

1. Kalvojen 8.3 vieruslistaesitys on luonteva syötteenä saadulle verkolle $G$ samasta syystä kuin kalvojen 8.4.1 syvyyssuuntaisessa läpikäynnissä:

   Rivien 13–20 silmukka käy läpi solmun $u$ vieruslistan.

   Silloin aikavaatimuskin on sama

   $$\mathcal{O}(|V(G)| + |E(G)|).$$

2. Rivit 15–18 suoritetaan solmulle $v \in V(G)$ *korkeintaan* kerran.

   *Todistus.* Solmu $v$

   - harmaantuu rivillä 15
   - eikä enää valkene koska rivi 2 on jo ohitettu

   joten rivin 14 ehto ei enää toiste toteudu solmulle $v$. □





3. Kaarelle $u \to v \in E(G)$ pätee

   $$\delta(s, v) \leq \delta(s, u) + 1$$

   missä

   $$\begin{aligned}\delta(s, x) &= \text{solmun } x \text{ syvyys} \\ &\quad\ \text{(solmusta } s \text{ alkaen)} \\ &= \text{kentän } x.\mathsf{depth} \text{ haluttu oikea arvo.}\end{aligned}$$

   *Todistus.* Jos solmu $u$ *on* saavutettavissa solmusta $s$ pitkin jotakin polkua

   $$s \to \cdots \to u$$

   niin myös solmu $v$ on saavutettavissa solmusta $s$ *ainakin* yhden kaaren verran pidempää polkua

   $$s \to \cdots \to u \to v$$

   pitkin (ja mahdollisesti jotakin suorempaakin polkua pitkin). Siis ehto on tosi.

   Jos taas solmu $u$ *ei* ole lainkaan saavutettavissa solmusta $s$, niin $\delta(s, u) = +\infty$ ja ehto automaattisesti tosi. □





4. Läpikäydään verkko $G$ leveyssuuntaisesti alkusolmusta $s \in V(G)$. Lopuksi jokaisen solmun $v \in V(G)$ kenttä $v.\mathsf{depth} \geq \delta(s, v)$.

   *Todistus.* Osoitetaan induktiolla yli tehtyjen enqueue-operaatioiden lukumäärän, että ehto pätee *koko ajan* eikä vain lopuksi.

   - Ensimmäinen enqueue-operaatio on rivin 10 alustus.

     Alkusolmulle $s.\mathsf{depth} = 0 = \delta(s, s)$ riviltä 7.

     Muille solmuille $u \in V(G) \setminus \{s\}$ on $u.\mathsf{depth} = +\infty \geq \delta(s, u)$ riviltä 3.

   - Muut enqueue-operaatiot tapahtuvat sisäsilmukassa rivillä 18.

     Tällä silmukkakierroksella harmaatuvalle solmulle $v$ pätee

     $$\begin{aligned}v.\mathsf{depth} &= u.\mathsf{depth} + 1 \quad \text{(rivi 16)} \\ &\geq \delta(s, u) + 1 \quad \text{(induktio)} \\ &\geq \delta(s, v) \quad \text{(huomio 3).}\end{aligned}$$

     Huomion 2 mukaan kenttä $v.\mathsf{depth}$ ei enää tämän jälkeen muutu. Muut kentät eivät muutu edes nyt. □





5. Verkon $G$ leveyssuuntaisen läpikäynnin aikana työjonon $Q$ sisältö

   $$\overrightarrow{v_1, v_2, v_3, \ldots, v_r}$$

   toteuttaa väitteet

   - $v_r.\mathsf{depth} \leq v_1.\mathsf{depth} + 1$
   - $v_i.\mathsf{depth} \leq v_{i+1}.\mathsf{depth} + 1$ kaikilla $1 \leq i < r$.

   *Todistus.* Induktiolla yli tehtyjen jono-operaatioiden lukumäärän.

   - Kun alustusrivi 10 on suoritettu, on $r = 1$ ja väitteet toteutuvat automaattisesti.
   - Kun jonosta poistetaan $u = v_1$ rivillä 12, niin

     **joko** jono $Q$ tyhjenee, jolloin väitteet toteutuvat automaattisesti.

     **tai** jonon $Q$ uudeksi ensimmäiseksi alkioksi tulee solmu $v_2$. Silloin

     $$\begin{aligned}v_r.\mathsf{depth} &\leq v_1.\mathsf{depth} + 1 \quad \text{(induktiolla)} \\ &\leq v_2.\mathsf{depth} + 1 \quad \text{(induktiolla)}\end{aligned}$$

     ja muut epäyhtälöt eivät muutu.

- Kun jonoon $Q$ lisätään $v = v_{r+1}$ rivillä 18:
  - Käsiteltävä solmu $u$ on jo poistettu rivillä 12:

    **joko** jono $Q$ tyhjeni, jolloin $v$ päästään lisäämään tyhjään jonoon, ja väitteet toteutuvat kuten alustuksessa

    **tai** entinen $v_2$ on nykyinen $v_1$, ja tarkastellaan sitä.
  - Ylempi väite:

$$\begin{aligned} v.\text{depth} &= u.\text{depth} + 1 \quad \text{(rivi 16)} \\ &\leq v_1.\text{depth} + 1 \quad \text{(induktio)} \end{aligned}$$

  - Alempi väite:

$$\begin{aligned} v_r.\text{depth} &\leq u.\text{depth} + 1 \quad \text{(induktio)} \\ &= v.\text{depth} \quad \text{(rivi 16)} \end{aligned}$$

  ja muut epäyhtälöt eivät muutu. □





6. Jos verkon $G$ leveyssuuntainen läpikäynti vie työjonoon $Q$
   (a) ensin solmun $x \in V(G)$
   (b) joskus myöhemmin solmun $y \in V(G)$

   niin $x.\text{depth} \leq y.\text{depth}$.

   *Todistus.* Tarkastellaan solmujen luetteloa

$$\mathcal{Z} = z_1, z_2, z_3, \ldots, z_m$$

   missä $z_j =$ se solmu joka lisättiin työjonoon $Q$ algoritmin $j$. lisäysoperaatiossa.
   - Huomion 2 mukaan mikään solmu ei esiinny luettelossa $\mathcal{Z}$ kahdesti.
   - Huomion 5 mukaan luettelossa $\mathcal{Z}$ pätee $z_j \leq z_{j+1}$ kaikilla $1 \leq j < m$.
   - Alkio $x$ esiintyy ennen alkiota $y$ luettelossa $\mathcal{Z}$. □