

Algoritmi on periaatteellisella tasolla seuraava:

```
Dijkstra( $V, E, \ell, v_0$ ):  
   $S := \{v_0\}$   
   $D[v_0] := 0$   
  for  $v \in V - S$  do  
     $D[v] := \ell(v_0, v)$   
  end for  
  while  $S \neq V$  do  
    valitse  $v \in V - S$  jolle  $D[v]$  on minimaalinen  
     $S := S \cup \{v\}$   
    for  $w \in V - S$  do  
       $D[w] := \min \{ D[w], D[v] + \ell(v, w) \}$   
    end for  
  end while  
  return  $D$ 
```

Tehokkuuden kannalta on merkittävää, millaisella tietorakenteella operaatio "valitse v " toteutetaan. Palataan tähän myöhemmin. Osoitetaan ensin algoritmin oikeellisuus.

Lause Dijkstran algoritmin palautamalle taulukolla D pätee $D[v] = \delta(v)$ kaikilla $v \in V$.

Todistus Ilmeisesti algoritmin suoritus aina päättyy, ja tällöin $S = V$. Lause seuraa osoittamalla induktiolla joukon S koon suhteen seuraava väite:

1. jos $w \in S$ niin $D[w] = \delta(w)$
2. muuten $D[w]$ on lyhimmän sellaisen polun pituus, joka alkaa solmusta v_0 , päättyy solmuun w eikä käy muissa joukon $V - S$ solmuissa.

Perustapaus: Kun $|S| = 1$, niin $S = \{v_0\}$. Koska $D[v_0] = 0 = \delta(v_0)$ ja kaikilla $v \neq v_0$ pätee $D[v] = \ell(v_0, v)$, väite on tosi.

Induktioaskel: Oletetaan, että väite pätee tietyllä joukolla S . Osoitetaan, että se pätee myös joukolla $S \cup \{v\}$, kun v on valittu ja D päivitetty kuten algoritmissa.

Kohta 1: Pitää osoittaa, että päivityksen jälkeen $D[v] = \delta(v)$.

Induktio-oletuksen mukaan $D[v]$ on ennen päivitystä erään polun pituus solmusta v_0 solmuun v . Päivitys ei ainakaan kasvata arvoa $D[v]$, joten sen jälkeenkin erityisesti pätee $\delta(v) \leq D[v]$.

Pitää vielä osoittaa $D[v] \leq \delta(v)$. Tehdään vastaoletus $\delta(v) < D[v]$. Olkoon π (jokin) lyhin polku solmusta v_0 solmuun v .

Induktio-oletuksen mukaan $D[v]$ on ennen päivitystä lyhin polun pituus solmuun v kun käytetään vain joukon $S \cup \{v\}$ solmuja, eikä $D[v]$ muutu päivityksessä. Siis polulla π on ainakin yksi solmu joukon $S \cup \{v\}$ ulkopuolelta.

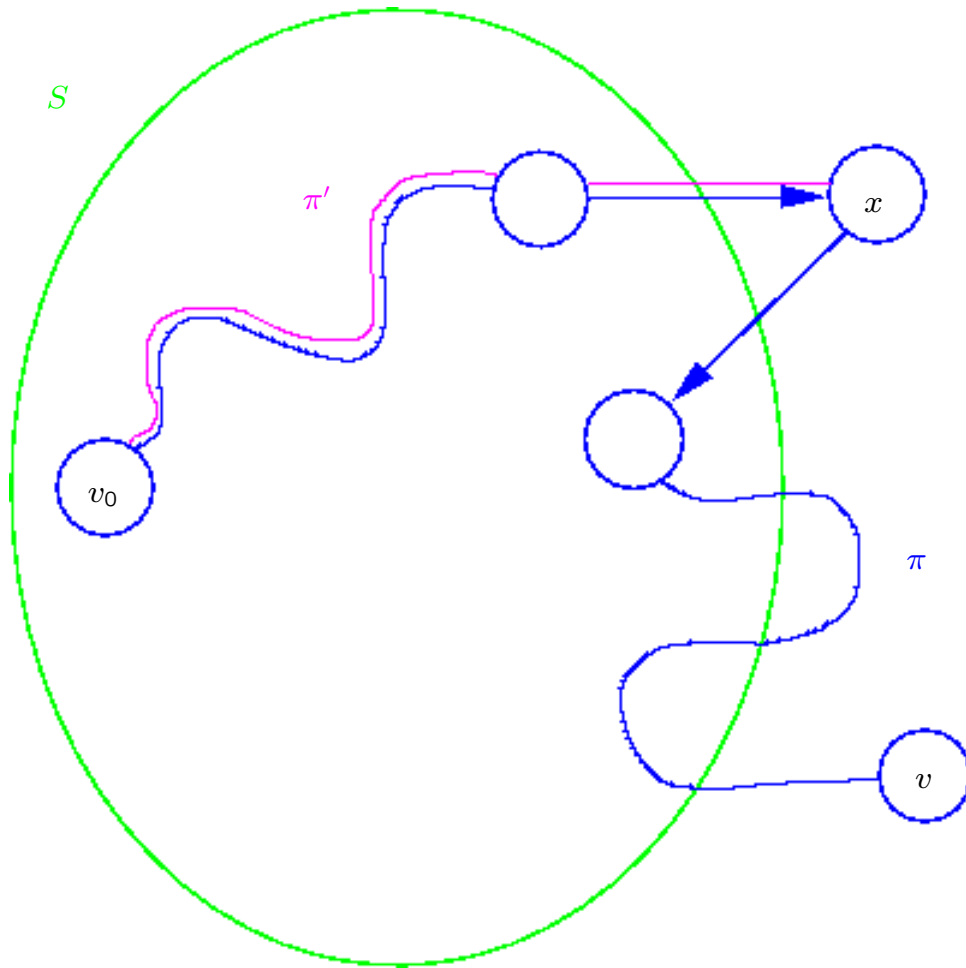
Olkoon x ensimmäinen joukkoon $S \cup \{v\}$ kuulumaton solmu polulla π , ja olkoon π' polun π alkuosa joka päättyy solmuun x .

Selvästi π' on lyhin polku solmuun x ja sisältää vain joukon $S \cup \{x\}$ solmuja, joten induktio-oletuksen nojalla $D[x]$ on polun π' pituus.

Koska π' on polun π alkuosa, $D[x]$ on korkeintaan polun π pituus.

Oletuksen mukaan polun π pituus on $\delta(v) < D[v]$, joten $D[x] < D[v]$.

Tämä on kuitenkin ristiriidassa alkion v valinnan kanssa. Kohta 1 on todistettu.



Kohta 1.

Kohta 2: Olkoon $w \notin S \cup \{v\}$, ja π lyhin polku solmuun w käyttäen vain joukon $S \cup \{v, w\}$ solmuja. Pitää osoittaa, että polun π pituus on

$$\min \{ D[w], D[v] + \ell(v, w) \}$$

missä D viittaa taulukon arvoon ennen päivitystä. On kolme mahdollisuutta:

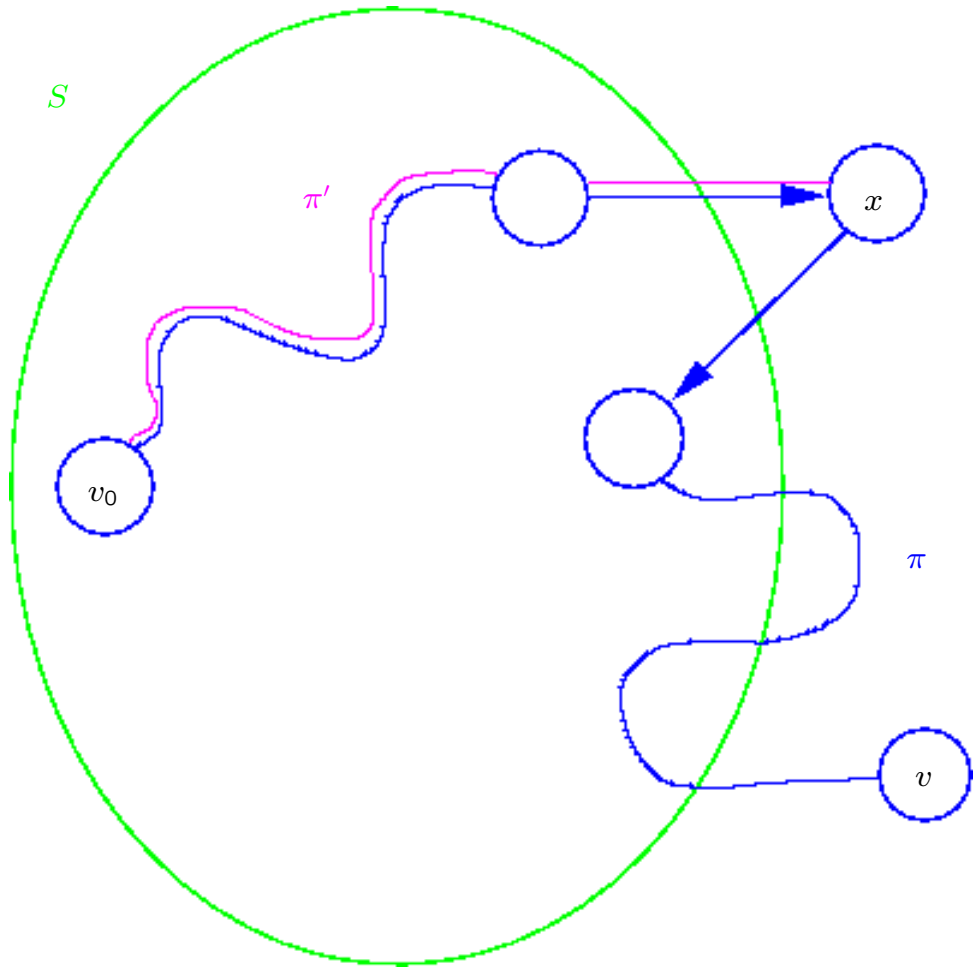
1. Polku π ei kulje solmun v kautta. Nyt induktio-oletuksen nojalla polun π pituus on $D[w]$, ja $D[w] \leq D[v] + \ell(v, w)$.
2. Polku π kulkee joukossa S , kunnes tulee solmuun v ja siitä suoraan solmuun w . Taas induktio-oletuksesta seuraa, että polun π pituus on $D[v] + \ell(v, w)$ ja $D[w] \geq D[v] + \ell(v, w)$.
3. Polku π kulkee solmun v kautta, mutta palaa joukkoon S ennen päätymistä solmuun w .

Olkoon x solmua v seuraava solmu polulla π .

Nyt $x \in S$, joten oletuksen mukaan lyhin polku solmuun x kulkee joukon S sisällä.

Ottamalla tämä lyhin polku korvaamaan polun π alkuosan solmuun x saakka saadaan polku π' , joka ei ole pitempi kuin π eikä kulje solmun v kautta. Tämä tapaus palautuu siis tapaukseen 1.

Kohta 2 ja siis lause on todistettu. \square



Kohta 2, tapaus 3.

Aikavaativuus Suoraviivainen vierusmatriisiin perustuva toteutus antaa helposti aikavaativuuden $O(|V|^2)$.

Vieruslistaesityksellä ja sopivilla aputietorakenteilla saadaan aikavaativuus $O(|V| \log |V| + |E|)$, joka on parempi harvoille verkoille ($|E| \ll |V|^2$).

Palautetaan mieliin (käännetty) **prioriteettijono**. On kiinnitetty järjestetty joukko K ("avaimet") ja mielivaltainen joukko D ("data"). Prioriteettijono ylläpitää joukkoja $Q \subseteq K \times D$ seuraavilla operaatioilla:

Empty(Q): $Q := \emptyset$

Insert(Q, x, d): $Q := Q \cup \{ (x, d) \}$

Minimum(Q): palauttaa sellaisen parin $(x, d) \in Q$ että x on suurin mahdollinen

Extract-Min(S): palauttaa sellaisen parin $(x, d) \in Q$, että x on suurin mahdollinen, ja poistaa sen joukosta

Decrease-Key(Q, x, y, d): korvaa alkio $(x, d) \in Q$ alkiolla (y, d) ; operaatio on sallittu vain jos oletetaan $y \leq x$

("Alkuperäisessä" prioriteettijonossa on vastaavasti operaatiot Maximum, Extract-Max ja Increase-Key.)

Esitetään Dijkstran algoritmi käyttäen prioriteettijonoa, jonka avaimia ovat polunpituudet ja dataa solmujen nimet. Oletetaan, että $A[v]$ on solmun v vieruslista verkossa (V, E) .

Dijkstra (V, E, ℓ, v_0) :

$S := \{v_0\}$

$D[v_0] := 0$

Empty(Q)

for $v \in V - S$ **do**

$D[v] := \ell(v_0, v)$

Insert($Q, D[v], v$)

end for

while $|S| < n$ **do**

1. $(x, v) :=$ Extract-Min(S)

2. $S := S \cup \{v\}$

3. **for** $w \in A[v]$ **do**

4. $x := D[v] + \ell(v, w)$

5. **if** $x < D[w]$ **then**

6. Decrease-Key($Q, w, D[w], x$)

end if

end for

end while

return D

Aikavaatimus selvästi määräytyy **while**-silmukan sisällä olevista prioriteettijono-operaatioista.

Rivin 1. Extract-Min suoritetaan $|V| - 1$ kertaa.

Rivin 6. Decrease-Key-operaation suorituskertojen lukumäärä on korkeintaan vieruslistojen yhteispituus eli $|E|$.

Kurssilla Tietorakenteet on opittu, miten n alkion prioriteettijono voidaan toteuttaa *kekoa* käyttäen ajassa $O(\log n)$ per operaatio. Aikavaatimukseksi tulee

$$(|V| + |E|) \log |V|.$$

Jatkossa esiteltävillä **Fibonacci-keoilla** aikavaatimusta saadaan parannetuksi sikäli, että operaation Decrease-Key **tasoitetuksi** aikavaatimukseksi tuleekin vain $O(1)$. Dijkstran algoritmin aikavaatimukseksi tulee tällöin

$$|V| \log |V| + |E|.$$

(Fibonacci-keot ovat suhteellisen monimutkainen tietorakenne eikä välttämättä kovin käytännöllinen.)

Pienin virittävä puu (minimum spanning tree)

Olkoon $G = (V, E)$ suuntaamaton verkko ja $G' = (V', E')$ sen osaverkko ($V' \subseteq V$ ja $E' \subseteq E \cap (V' \times V')$).

Jos $V' = V$, sanomme että G' virittää verkon G .

Jos lisäksi G' on syklitön, se on virittävä metsä.

Jos lisäksi G' on yhtenäinen, se on virittävä puu. (Huom. virittävä metsä on virittävä puu joss $|E'| = |V| - 1$.)

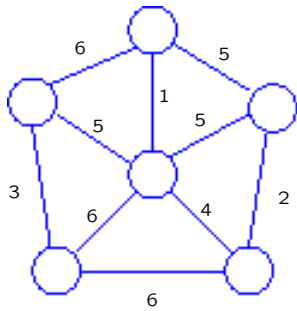
Oletetaan vielä, että kuhunkin verkon kaareen $e \in E$ liittyy paino $\ell(e) \geq 0$. Aliverkon G' kustannus on

$$\ell(G') = \sum_{e \in E'} \ell(e).$$

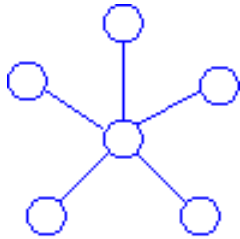
Yhtenäisen verkon pienin virittävä puu on kustannukseltaan minimaalinen virittävä puu. Tällainen on olemassa joss verkko on yhtenäinen; se ei välttämättä ole yksikäsitteinen.

Sovelluksia: tietoliikenneverkot, piirisuunnittelu, osana muita verkkoalgoritmeja.

Virittäviä puita esim. täydellisellä verkolla on eksponentiaalinen määrä, joten kaikkien mahdollisuuksien kokeileminen on tehotonta.

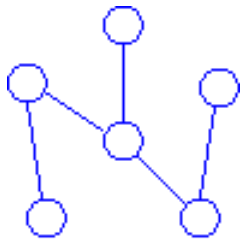


Verkko G



Eräs virittävä puu G'

$$\ell(G') = 1 + 5 + 4 + 6 + 5 = 21$$



Pienin virittävä puu G''

$$\ell(G'') = 3 + 5 + 1 + 4 + 2 = 15$$

Esitetään ahne **Kruskalin algoritmi** pienimmän virittävän puun löytämiseksi yhtenäiselle verkolle.

Algoritmi pitää yllä kaarijoukkoa $T \subseteq E$, jolla (V, T) on metsä. Joukkoa T kasvatetaan yksi kaari kerrallaan, kunnes saadaan yhtenäinen osaverkko.

```
Kruskal( $V, E, \ell$ ):  
   $T := \emptyset$   
   $L := E$   
  while  $|T| < |V| - 1$  do  
    valitse  $e = (v, w) \in Q$  jolla  $\ell(e)$  on minimaalinen  
     $L := L - \{e\}$   
    if  $v$  ja  $w$  kuuluvat eri puihin  
      metsässä  $(V, T)$   
    then  $T := T \cup \{e\}$   
  end while  
  return  $(V, T)$ 
```

Aikavaativuusanalyysissä keskeinen kysymys on, miten testataan solmujen kuuluminen eri puihin. Tämä on esimerkki **erillisten joukkojen yhdisteistä** (Union-Find), johon sopiva tehokas tietorakenne esitetään myöhemmin. Lopputulos on, että algoritmi toimii ajassa

$$O(|E| \log |E|) = O(|E| \log |V|)$$

eli joukon E järjestäminen painojen suhteen dominoi aikavaativuutta. (Huom. $|V| - 1 \leq |E| \leq |V|^2$).

Kruskalin algoritmin oikeellisuustodistus perustuu invarianttiin, että (V, T) on aina jonkin pienimmän virittävän puun osaverkko:

Lemma Oletetaan että (V, E) on yhtenäinen ja $T \subseteq E$. Oletetaan että

- on olemassa pienin virittävä puu (V, S) jolla $T \subseteq S$ ja
- $e \in E$ on painoltaan minimaalinen kaari, jonka päätepisteet kuuluvat metsän (V, T) eri puihin.

Nyt

- on olemassa pienin virittävä puu (V, S') jolla $T \cup \{e\} \subseteq S'$.

Lemmasta seuraa suoraan induktiolla joukon T koon suhteen, että Kruskalin algoritmi palauttaa jonkin pienimmän virittävän puun.

Lemman todistus Jos $e \in S$, voidaan valita $S' = S$.
Olkoon $e \notin S$.

Koska (V, S) on yhtenäinen ja syklitön, verkossa $(V, S \cup \{e\})$ on tasan yksi sykli.

Olkoon e' jokin tämän syklin kaari, joka ei kuulu joukkoon $T \cup \{e\}$. Tällainen kaari on olemassa, koska $(V, T \cup \{e\})$ on syklitön.

Valitaan $S' = S - \{e'\} \cup \{e\}$. Koska (V, S') on syklitön ja $|S'| = |S| = |V| - 1$, niin (V, S') on virittävä puu.

Koska $e' \notin T$ ja $(V, T \cup e')$ on syklitön, kaaren e' päätepisteet ovat metsän (V, T) eri puissa. Kaaren e valinnasta seuraa, että $\ell(e) \leq \ell(e')$.

Siis

$$\begin{aligned} \ell((V, S')) &= \ell((V, S)) - \ell(e') + \ell(e) \\ &\leq \ell((V, S)) \end{aligned}$$

joten myös (V, S') on pienin virittävä puu.

□

Ahneiden algoritmien teoriaa

Kruskalin algoritmi on esimerkki laajasta joukosta ahneita optimointialgoritmeja, joiden toimivuus voidaan perustella **matroidien** teorian avulla.

Olkoon $S \neq \emptyset$ äärellinen, ja olkoon I kokoelma joukon S osajoukkoja. Pari (S, I) on **matroidi**, jos seuraavat ehdot ovat voimassa:

Perinnöllisyys: jos $B \in I$ ja $A \subseteq B$ niin $A \in I$

Vaihto-ominaisuus: jos $A \in I$, $B \in I$ ja $|A| < |B|$, niin on olemassa $x \in B - A$ jolla $A \cup \{x\} \in I$

Kokoelman I joukkoja sanotaan **riippumattomiksi**.

Esimerkki Olkoon S äärellinen ja $I = \{A \subseteq S \mid |A| \leq k\}$ jollain k .

Selvästi I on perinnöllinen.

Olkoon $A \in I$, $B \in I$ ja $|A| < |B|$. Siis $|A| \leq k - 1$, joten $A \cup \{x\} \in I$ millä tahansa x . Koska $|A| < |B|$, on olemassa $x \in B - A$. Siis vaihto-ominaisuus pätee, ja (S, I) on matroidi.

Esimerkki Olkoon $G = (V, E)$ ja

$$I = \{T \subseteq E \mid (V, T) \text{ on sykliiton}\}.$$

Merkitään $M_G = (E, I)$.

Propositio M_G on matroidi.

Todistus Perinnöllisyys on ilmeistä.

Olkoon $A \in I$, $B \in I$ ja $|A| < |B|$. Siis (V, A) ja (V, B) ovat metsiä. Koska metsässä (V, A) on vähemmän kaaria, siinä on enemmän yhtenäisiä komponentteja (eli puita).

Siis ainakin yksi metsän (V, B) puu sisältää solmuja ainakin kahdesta metsän (V, A) puusta. Voidaan siis valita kaari $e \in B$, jonka päätepisteet ovat metsän (V, A) eri puissa. Tällöin $A \cup \{e\} \in I$. \square