# 3317332 Optimointi

Timo Erkama
LaTeX-käännös
Marko Lamminsalo
Itä-Suomen yliopisto
6. toukokuuta 2011

# Sisältö

# 1 Peruskäsitteitä

Optimointia tarvitaan mm.

- voiton, hyötysuhteen maksimoinnissa
- kustannusten, $CO_2$-päästöjen minimoinnissa

Kussakin tapauksessa on kysymys ääriarvotehtävästä: optimoinnissa etsitään tilanteita, joissa annettu ns. *kohdefunktio* saavuttaa maksimin tai minimin.

Yleensä tarkasteltava funktio $f$ riippuu useasta muuttujasta $x_1, \ldots, x_n$, joiden arvoja voidaan säädellä asetetuissa rajoissa. Optimointimenetelmissä pyritään hakemaan sellaiset arvot, joilla saavutetaan optimaalinen ratkaisu, so. $f$:n maksimi tai minimi. Probleeman luonteesta johtuen monesti muuttujien $x_1, \ldots, x_n$ valinta ei ole täysin vapaata vaan niiden arvoja saattaa olla sitomassa erilaisia *rajoitteita*: eräät muuttujat ovat aina esim. ei-negatiivisia, kuten tuotteen hinta, työaika, ajomatka jne. Rajoitteet voidaan tavallisesti esittää epäyhtälöinä tai yhtälöinä.

### Rajoitteeton optimointi

Tarkastellaan $n$:n muuttujan funktiota $f(x)$, missä $x = (x_1, \ldots, x_n)$. Määritelmän mukaan $f$:llä on *minimi* pisteessä $X$, jos

$$f(x) \geq f(X)$$

kaikille sen joukon $R$ pisteille $x$, joissa $f$ on määritelty. Vastaavasti $f$:llä on *maksimi* pisteessä $X$, jos

$$f(x) \leq f(X)$$

kaikille $x \in R$.

Sanomme, että $f$:llä on *lokaalinen minimi* (vast. *lokaalinen maksimi*) pisteessä $X$, jos on olemassa $X$-keskinen pallo (kuula), johon rajoitettuna $f$:llä on minimi (vast. maksimi). Jos $f$ on differentioituva ja saa ääriarvon jossakin $R$:n sisäpisteessä $X$, niin $f$:n osittaisderivaatat häviävät pisteessä $X$, so.

$$\nabla f(X) = 0, \tag{1}$$

missä

$$\nabla f = \left( \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n} \right)$$

on funktion $f$ *gradientti*.

Jos (1) pätee pisteessä $X$, sanomme että $X$ on $f$:n *stationaarinen* piste. Stationaarisessa pisteessä ei välttämättä saavuteta edes lokaalista ääriarvoa. Esim. tapauksessa $n = 1$, $f(x) = x^3$, saadaan

$$\nabla f(X) = f'(X) = 3X^2 = 0,$$

kun $X = 0$, mutta $f$:llä ei ole ääriarvoa origossa. Vastaavan esimerkin tapauksessa $n = 2$ antaa funktio $f(x_1, x_2) = x_1 x_2$ jolle jälleen $\nabla f(X) = 0$, kun $X = 0$.

Lokaalisten ääriarvojen löytämiseksi on olemassa analyysistä tuttuja riittäviä ehtoja, jotka takaavat lokaalisen maksimin tai minimin olemassaolon. Esimerkiksi ehdoista $y'(X) = 0$, $y''(X) > 0$ seuraa, että $f$:llä on $X$:ssä lokaali minimi. Käytännössä kuitenkin yhtälön (1) ratkaiseminen voi tuottaa hankaluuksia. Sen vuoksi ratkaisua etsitään yleensä iteratiivisesti lähestymällä ääriarvokohtaa askel askeleelta siten, että funktion arvo kullakin askeleella joko pienenee tai kasvaa sen mukaan, onko haettavana oleva ääriarvo minimi tai maksimi.

*Gradienttimenetelmässä* ratkaistaan useita peräkkäisiä yhden muuttujan funktion ääriarvotehtäviä. Oletetaan esim. että lähtöpisteenä on $x$ ja funktiolla $f$ on minimi pisteessä $X$. Ensimmäisessä iteraatiovaiheessa etsitään $f$:lle minimiä sillä $x$:n kautta kulkevalla suoralla, joka on vektorin $-\nabla f(x)$ suuntainen; tässä suunnassahan $f$:n pieneneminen on nopeinta. Toisin sanoen etsitään minimiä yhdenmuuttujan $t$ funktiolle

$$g(t) = f\big(x - t\nabla f(x)\big) \tag{2}$$

ja valitaan seuraavaksi iteraatiopisteeksi minimikohtaa $t$ vastaava piste

$$z(t) = x - t\nabla f(x). \tag{3}$$

Jos minimikohtia on useampia, valitaan niiden joukosta pienin *positiivinen* $t$:n arvo.

**Esimerkki 1.** Määrätään funktion

$$f(x) = x_1^2 + 3x_2^2 \tag{4}$$

minimikohta gradienttimenetelmällä käyttämällä lähtöpistettä $(6, 3)$.

*Ratkaisu.* Välittömästi nähdään, että minimi saavutetaan origossa, joten gradienttimenetelmä ei tässä tapauksessa ole realistinen ratkaisukeino. Jos sitä kuitenkin käytettäisiin, saataisiin aluksi

$$\nabla f(x) = (2x_1, 6x_2)$$

ja edelleen

$$\begin{aligned} g(t) &= f\big(x - t\nabla f(x)\big) \\ &= f\big((1-2t)x_1, (1-6t)x_2\big) \\ &= (1-2t)^2 x_1^2 + 3(1-6t)^2 x_2^2. \end{aligned}$$

Funktion $g(t)$ minimikohta on yhtälön $g'(t) = 0$ ratkaisu

$$t = \frac{x_1^2 + 9x_2^2}{2x_1^2 + 54x_2^2}.$$

Lähtöpistettä $(6, 3)$ vastaava $t$:n arvo on $0.210$, ja seuraava iteraatiopiste on kaavan (3) mukaisesti $(3.484, -0.774)$. Toistamalla menettelyä saadaan seuraavan taulukon mukainen

pistejono.

| $n$ | $x$ | $t$ |
|---|---|---|
| 0 | $(6, 3)$ | 0.210 |
| 1 | $(3.484, -0.774)$ | 0.310 |
| 2 | $(1.327, 0.664)$ | 0.210 |
| 3 | $(0.771, -0.171)$ | 0.310 |
| 4 | $(0.294, 0.147)$ | 0.210 |
| 5 | $(0.170, -0.038)$ | 0.310 |
| 6 | $(0.065, 0.032)$ | |

# 2 Lineaarinen optimointi

*Lineaarisessa optimoinnissa* etsitään ääriarvokohtaa muuttujien $x_1, \ldots, x_n$ *lineaariselle* funktiolle joukossa, jonka määräävät lineaaristen epäyhtälöiden antamat rajoitteet.

**Esimerkki 1.** Tehtaassa on kaksi konetta $K_1$ ja $K_2$, joilla valmistetaan tuotteita $T_1$ ja $T_2$ siten, että tuotteen $T_1$ valmistamiseen tarvitaan 2 minuuttia konetta $K_1$ ja 4 minuuttia konetta $K_2$, kun taas tuotteen $T_2$ valmistamiseen tarvitaan 8 minuuttia konetta $K_1$ ja 4 minuuttia konetta $K_2$. Tehdas saa $T_1$:n valmistamisesta voittoa 29 € ja $T_2$:n valmistamisesta 45 €. Laadi tuotantosuunnitelma, joka maksimoi voiton.

*Ratkaisu.* Jos tunnissa valmistetaan $x_1$ kpl tuotetta $T_1$ ja $x_2$ kpl tuotetta $T_2$, voitto tunnin aikana on

$$f(x_1, x_2) = 29x_1 + 45x_2.$$

Tälle funktiolle etsitään siis maksimia joukossa, jonka määräävät seuraavat rajoitteet

$$\begin{aligned} 2x_1 + 8x_2 &\leq 60 \\ 4x_1 + 4x_2 &\leq 60 \\ x_1 &\geq 0 \\ x_2 &\geq 0. \end{aligned} \tag{1}$$

Kysymyksessä on nelikulmio, jota rajoittavat suorat $x_1 = 0$, $x_2 = 0$, $2x_1 + 8x_2 = 60$ ja $4x_1 + 4x_2 = 60$. Funktio $f$ saa vakioarvoja suoran $29x_1 + 45x_2 = 0$ kanssa yhdensuuntaisilla suorilla ja suurin arvo nelikulmiossa $OABC$ saavutetaan pisteessä $B = (10, 5)$. Näin ollen voitto maksimoidaan tuottamalla tuotteita $T_1$ ja $T_2$ suhteessa $2 : 1$, jolloin voitto on 515 € tunnissa.

Tämäntyyppisiä ääriarvotehtäviä ei voida ratkaista etsimällä joidenkin osittaisderivaattojen nollakohtia, koska ääriarvo saavutetaan aina alueen reunalla.

### Lineaarisen optimointitehtävän normaalimuoto

Optimointitehtävän epäyhtälörajoitteet voidaan muuttaa yhtälörajoitteiksi käyttämällä apumuuttujia, joita kutsutaan *pelivaramuuttujiksi*. Rajoitteiden (1) ensimmäinen epäyhtälö

$$2x_1 + 8x_2 \leq 60 \tag{2}$$

voidaan kirjoittaa käyttämällä pelivaramuuttujaa $x_3 = 60 - 2x_1 - 8x_2$. Tällöin (2) voidaan kirjoittaa muodossa

$$\begin{aligned} 2x_1 + 8x_2 + x_3 &= 60 \\ x_3 &\geq 0. \end{aligned}$$

**Esimerkki 2.** Esimerkin 1 probleema voidaan esittää yhtälörajoitteisena määrittelemällä kaksi pelivaramuuttujaa $x_3$ ja $x_4$. Tällöin probleema voidaan kirjoittaa seuraavassa muodossa: Etsi funktion

$$f(x_1, x_2) = 29x_1 + 45x_2$$

maksimi rajoitteilla

$$\begin{aligned} 2x_1 + 8x_2 + x_3 &= 60 \\ 4x_1 + 4x_2 + x_4 &= 60 \\ x_i &\geq 0 \quad (i = 1, 2, 3, 4) \end{aligned} \tag{3}$$

Tehtävässä on siis $n = 4$ muuttujaa, ja $m = 2$ (riippumatonta) yhtälöä, jolloin mitkä hyvänsä kaksi annettua muuttujaa määräävät jäljellä olevien muuttujien arvot. Kuvan nelikulmiossa jokainen sivu toteuttaa muotoa $x_i = 0$ olevan yhtälön:

$$OA : x_2 = 0, \quad AB : x_4 = 0, \quad BC : x_3 = 0, \quad CO : x_1 = 0.$$

Kärki on kahden sivun leikkauspiste, esim. $A$:ssa $x_2 = x_4 = 0$. Jokaisessa kärjessä on $n - m = 4 - 2 = 2$ muuttujaa, jotka saavat arvon 0, ja loput muuttujat ovat ei-negatiivisia.

Yleisesti jokainen lineaarinen optimointitehtävä voidaan palauttaa seuraavaan *normaalimuotoon*: Etsi funktion

$$f(x_1, \ldots, x_n) = c_1x_1 + c_2x_2 + \cdots + c_nx_n \tag{4}$$

maksimi rajoitteilla

$$\begin{aligned} a_{11}x_1 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + \cdots + a_{mn}x_n &= b_m \\ x_i &\geq 0 \quad (i = 1, \ldots, n) \end{aligned} \tag{5}$$

Yhtälössä (4) pelivaramuuttujien kertoimet ovat nollia. Oletetaan, että yhtälöt (5) ovat lineaarisesti riippumattomia, jolloin (5):n ratkaisu riippuu $n - m$:stä parametristä. Mikäli etsitään minimiä maksimin sijaan, voidaan $f$ korvata $-f$:llä.

Yhtälöryhmän (5) toteuttavia pisteitä $(x_1, \ldots, x_n)$ sanotaan *rajoiteratkaisuiksi*. Rajoiteratkaisu on *optimaalinen*, jos sillä saavutetaan $f$:n maksimiarvo. Rajoiteratkaisua $(x_1, \ldots, x_n)$ sanotaan *(käyväksi) kantaratkaisuksi*, jos ainakin $n - m$ kappaletta muuttujista $x_1, \ldots, x_n$ on nollia.

Esimerkissä 2 $n = 4$ ja $m = 2$ ja kantaratkaisut ovat nelikulmion $OABC$ kärkipisteet $O$, $A$, $B$ ja $C$. Näistä vain $B$ on optimaalinen.

**Lause 1.** *Jos lineaarisella optimointitehtävällä (4)-(5) on optimaalinen rajoiteratkaisu, tehtävällä on myös optimaalinen kantaratkaisu.*

**Huomautus.** Optimaalinen rajoiteratkaisu ei välttämättä ole kantaratkaisu. Lauseen 1 nojalla kuitenkin optimaalisten rajoiteratkaisuiden joukosta voidaan aina löytää kantaratkaisu. Siten optimaalinen ratkaisu voidaan löytää äärellisellä määrällä askelia asettamalla mitkä tahansa $n - m$ muuttujaa nolliksi, ratkaisemalla yhtälörajoitteet jäljellä olevien muuttujien suhteen ja laskemalla $f$:n arvo jokaisessa ratkaisupisteessä. Työmäärä voi olla kuitenkin hyvin suuri, sillä ratkaistavien yhtälöryhmien määrä on $\binom{n}{n-m}$. Siksi onkin parempi käyttää nopeampia algoritmeja kuten *simpleksimenetelmää*.

# 3 Simpleksimenetelmä

Tarkastellaan lineaarista optimointiongelmaa normaalimuodossa kuten edellä: Etsi funktion

$$f(x_1, \ldots, x_n) = c_1x_1 + c_2x_2 + \cdots + c_nx_n \tag{1}$$

maksimi rajoitteilla

$$\begin{aligned} a_{11}x_1 + \cdots + a_{1n}x_n &= b_1 \\ &\vdots \\ a_{m1}x_1 + \cdots + a_{mn}x_n &= b_m \\ x_i &\geq 0 \quad (i = 1, \ldots, n) \end{aligned} \tag{2}$$

Lauseen 1 nojalla riittää tarkastella kantaratkaisuja optimaalisen ratkaisun löytämiseksi. Simpleksimenetelmässä kuljetaan kantaratkaisusta toiseen siten, että kohdefunktion arvo ei vähene. Aloituspiste on jokin kantaratkaisu, joka valitaan algoritmin alussa. Menetelmässä voidaan erottaa kolme vaihetta:

$V_1$: optimaalisuuden testaus

$V_2$: paremman kantaratkaisun etsiminen

$V_3$: muuttujan vaihto siirryttäessä parempaan kantaratkaisuun

**Lähtöpisteen valinta**

Valitaan muuttujien $x_1, \ldots, x_n$ joukosta (umpimähkään) $m$ kpl. Näitä kutsutaan *kantamuuttujiksi*; jäljelle jäävät $n - m$ muuttujaa ovat *oikeanpuoleisia muuttujia*. Ratkaistaan systeemi (2) kantamuuttujien suhteen. Luvun 2 esimerkissä systeemi (2)

$$\begin{aligned} 2x_1 + 8x_2 + x_3 &= 60 \\ 4x_1 + 4x_2 + x_4 &= 60 \\ x_i &\geq 0 \quad (i = 1, 2, 3, 4) \end{aligned} \tag{3}$$

Valitsemalla kantamuuttujiksi esim. $x_3, x_4$ saadaan ratkaisu

$$\begin{aligned} x_3 &= 60 - 2x_1 - 8x_2 \\ x_4 &= 60 - 4x_1 - 4x_2 \end{aligned} \tag{4}$$

Hyvällä tuurilla kantaratkaisu löytyy asettamalla oikeanpuoleiset muuttujat nolliksi. Tässä esimerkissä kantaratkaisuksi tulee näin piste $(x_1, x_2, x_3, x_4) = (0, 0, 60, 60)$ (kuvassa origo 0).

On kuitenkin mahdollista, että systeemillä (3) ei ole sellaista ratkaisua, jossa oikeanpuoleiset muuttujat häviävät. Tällöin on kantamuuttujat valittava jollakin muulla tavalla.

Kun lähtöpiste on löydetty, on vuorossa

$V_1$: optimaalisuuden testaus

Lausutaan maksimoitava funktio $f$ oikeanpuoleisten muuttujien avulla. Jos tällöin saadussa lausekkeessa kaikkien oikeanpuoleisten muuttujien kertoimet ovat ei-positiivisia, valittu kantaratkaisu on optimaalinen. Tämä johtuu siitä, että muuttujien sallitut arvot ovat ei-negatiivisia, jolloin $f$:n suurin mahdollinen arvo saavutetaan oikeanpuoleisten muuttujien ollessa nollia.

Esimerkissämme oikeanpuoleiset muuttujat ovat $x_1$ ja $x_2$, ja $f(x) = 29x_1 + 45x_2$. Koska kertoimet 29 ja 45 ovat positiivisia, lähtöpisteeksi valittu kantaratkaisu ei ole optimaalinen.

$V_2$: paremman kantaratkaisun etsiminen

Valitaan sellainen oikeanpuoleinen muuttuja $x_R$, jonka kerroin $f$:n lausekkeessa on positiivinen. Etsitään sellainen rajoiteratkaisu, jossa $x_R$ on mahdollisimman suuri ja muut oikeanpuoleiset muuttujat ovat nollia. Merkitään $\Delta x_R$:llä kyseistä (mahdollisimman suurta) $x_R$:n arvoa, ja $\Delta f$:llä $f$:n arvon muutosta siirryttäessä lähtöpisteestä uuteen rajoiteratkaisuun.

Esimerkiksi yhtälöissä (4) valitaan aluksi $x_R = x_1$, jolloin

$$\begin{aligned} x_3 &= 60 - 2x_1 - 8x_2 \\ x_4 &= 60 - 4x_1 - 4x_2 \end{aligned}$$

Nyt etsitään siis sellaista rajoiteratkaisua, jossa $x_2 = 0$ ja $x_1$ on mahdollisimman suuri. Tällöin

$$\begin{aligned} x_3 = 60 - 2x_1 \geq 0 \;&\Rightarrow\; x_1 \leq \frac{60}{2} = 30 \\ x_4 = 60 - 4x_1 \geq 0 \;&\Rightarrow\; x_1 \leq \frac{60}{4} = 15 \end{aligned} \tag{5}$$

Alemman epäyhtälön perusteella suurin mahdollinen $x_1$:n arvo on $\Delta x_1 = 15$, ja vastaava $f$:n arvon muutos

$$\Delta f = 29\Delta x_1 = 435.$$

Toistetaan äskeinen menettely kaikkien niiden oikeanpuoleisten muuttujien suhteen, joiden kerroin $f$:n lausekkeessa on positiivinen. Esimerkissämme on siis vielä valittava $x_R = x_2$ ja haettava sellaista rajoiteratkaisua, jossa $x_1 = 0$ ja $x_2$ on mahdollisimman suuri. Tällöin

$$\begin{aligned} x_3 = 60 - 8x_2 \geq 0 \;&\Rightarrow\; x_1 \leq \frac{60}{8} = 7.5 \\ x_4 = 60 - 4x_2 \geq 0 \;&\Rightarrow\; x_1 \leq \frac{60}{4} = 15 \end{aligned}$$

Suurin mahdollinen $x_2$:n arvo on siis $\Delta x_2 = 7.5$, ja vastaava $f$:n arvon muutos

$$\Delta f = 45\Delta x_2 = 337.5.$$

$V_3$: muuttujien vaihto siirryttäessä parempaan kantaratkaisuun

Vaiheessa $V_2$ haetuista kantaratkaisuista valitaan se, johon liittyi suurin $\Delta f$. Esimerkissämme suurin $\Delta f$:n arvo $\Delta f = 435$ saavutettiin valittaessa $x_R = x_1 = 15$, jolloin toinen kantamuuttuja $x_4 = 60 - 4x_1$ joutui saamaan arvon nolla. Vaiheessa $V_3$ valitaan uudeksi kantamuuttujaksi $x_R = x_1$, ja $x_4$ siirtyy sen tilalle oikeanpuoleiseksi muuttujaksi.

Uudeksi kantamuuttujaksi valitaan siis sellainen $x_R$, jolla vaiheessa $V_2$ saatiin suurin $\Delta f$:n arvo, ja tämän tilalle oikeanpuoleiseksi muuttujaksi siirtyy se kantamuuttujista, joka $x_R$:n kasvaessa joutui saamaan arvon nolla. Tämän jälkeen ratkaistaan (2) uusien kantamuuttujien suhteen.

Esimerkissämme tulos on

$$\begin{aligned} x_1 &= 15 - x_2 - \frac{1}{4}x_4 \\ x_3 &= 30 - 6x_2 + \frac{1}{3}x_4 \end{aligned} \tag{6}$$

Uudesta kantaratkaisusta lähtien suoritetaan tämän jälkeen uudestaan vaiheet $V_1, V_2$ ja $V_3$. Aluksi on siis $f$ lausuttava uusien oikeanpuoleisten muuttujien $x_2$ ja $x_4$ avulla. Käyttämällä ensimmäistä yhtälöistä (6) saadaan

$$\begin{aligned} f &= 29x_1 + 45x_2 \\ &= 29(15 - x_2 - \frac{1}{4}x_4) + 45x_2 \\ &= 435 + 16x_2 - 7.25x_4 \end{aligned}$$

Uusi kantaratkaisu ei tämän perusteella ole optimaalinen, koska $x_2$:n kerroin on positiivinen.

Vaiheessa $V_2$ valitaan $x_R = x_2$. Asettamalla $x_4 = 0$ yhtälöistä (6) seuraa

$$\begin{aligned} x_1 = 15 - x_2 \geq 0 \;&\Rightarrow\; x_2 \leq 15 \\ x_3 = 30 - 6x_2 \geq 0 \;&\Rightarrow\; x_2 \leq \frac{30}{6} = 5 \end{aligned}$$

Alemman epäyhtälön perusteella suurin mahdollinen $x_2$:n arvo on $\Delta x_2 = 5$, ja vastaava $f$:n arvon muutos

$$\Delta f = 16\Delta x_2 = 80.$$

Koska $x_4$:n kerroin $f$:n lausekkeessa on negatiivinen, tapaus $x_R = x_4$ ei tässä tule kysymykseen. Voidaan siis siirtyä vaiheeseen $V_3$.

Uudeksi kantamuuttujaksi valitaan siis $x_R = x_2$, ja sen tilalle oikeanpuoleiseksi muuttujaksi siirtyy $x_3$, joka arvolla $x_R = 5$ tuli nollaksi. Ratkaistaan (4) uusien kantamuuttujien suhteen

$$\begin{aligned} x_1 &= 10 + \frac{1}{6}x_3 - \frac{1}{3}x_4 \\ x_2 &= 5 - \frac{1}{6}x_3 + \frac{1}{12}x_4 \end{aligned} \tag{7}$$

Algoritmi jatkuu toistamalla taas vaiheet $V_1, V_2$ ja $V_3$ lähtien yhtälöistä (7). Vaiheessa $V_1$ saadaan käyttämällä jälkimmäistä yhtälöistä (7)

$$f = 515 - 2.667x_3 - 5.917x_4$$

Koska $x_3$:n ja $x_4$:n kertoimet ovat negatiivisia, on löydetty optimaalinen kantaratkaisu.

$f$:n maksimiarvo $f_{opt} = 515$ saavutetaan siis, kun $x_3 = x_4 = 0$, jolloin (7):n perusteella $x_1 = 10$, $x_2 = 5$ kuten luvun 2 esimerkissä 1.

# 4 Simpleksimenetelmän ongelmia

Sanomme, että lineaarisen optimointitehtävän kantaratkaisu $(x_1, \ldots, x_n)$ on *degeneroitunut*, jos vähintään $n - m + 1$ luvuista $x_1, \ldots, x_n$ on nolla. Tässä $m$:llä on tietenkin sama merkitys kuin edellä. Degeneroituneen kantaratkaisun tapauksessa algoritmi ei aina toimi normaalisti: parempaan kantaratkaisuun ei välttämättä päästä yhdellä ainoalla muuttujan vaihdolla, vaan vaihtoja voi joutua suorittamaan useampia.

Algoritmi saadaan yleensä toimimaan siten, että arvon nolla saavat kantamuuttujat vaihdetaan sopivien oikeanpuoleisten muuttujien kanssa. Teoreettisesti on kuitenkin mahdollista että näin menetellen palataan yhä uudestaan samaan kantaratkaisuun eikä optimaalista ratkaisua löydetä.

**Esimerkki 1.** Tehtaassa tuotetaan seosmetalleja $L_1$, $L_2$ raaka-aineista $R_1$, $R_2$, $R_3$ allaolevan taulukon mukaisesti. Maksimoi päivittäinen voitto.

| | Raaka-aineen tarve (tn) | | Päivittäinen käytettävissä |
|---|---|---|---|
| Raaka-aine | $L_1$ | $L_2$ | oleva raaka-aine (tn) |
| $R_1$ | 2 | 1 | 16 |
| $R_2$ | 1 | 1 | 8 |
| $R_3$ | 0 | 1 | 3.5 |
| Voitto/tn | 150 | 300 | |

*Ratkaisu.* Olkoon $x_1$ ja $x_2$ vastaavasti tuotteiden $L_1$ ja $L_2$ päivittäiset tuotantomäärät (tonneissa). Maksimoitava voitto on

$$f = 150x_1 + 300x_2 \tag{1}$$

rajoitteiden ollessa $x_1 \geq 0$, $x_2 \geq 0$ ja

$$\begin{aligned} 2x_1 + x_2 &\leq 16 \\ x_1 + x_2 &\leq 8 \\ x_2 &\leq 3.5 \end{aligned}$$

Ottamalla käyttöön pelivaramuuttujat $x_3$, $x_4$ ja $x_5$ päästään normaalimuotoon

$$\begin{aligned} 2x_1 + x_2 + x_3 &= 16 \\ x_1 + x_2 + x_4 &= 8 \\ x_2 + x_5 &= 3.5 \\ x_i &\geq 0 \quad (1 \leq i \leq 5) \end{aligned} \tag{2}$$

Valitaan ensin kantamuuttujaksi nämä pelivaramuuttujat ja ratkaistaan (2) näiden suhteen:

$$\begin{aligned} x_3 &= 16 - 2x_1 - x_2 \\ x_4 &= 8 - x_1 - x_2 \\ x_5 &= 3.5 - x_2 \end{aligned} \tag{3}$$

Asettamalla $x_1 = 0$, $x_2 = 0$ löydetään kantaratkaisu $x_3 = 16$, $x_4 = 8$, $x_5 = 3.5$. Tämä valitaan lähtöpisteeksi.

*Vaihe* $V_1$: optimaalisuuden testaus. Kaavassa (1) on $f$ lausuttuna oikeanpuoleisten muuttujien avulla. Koska kertoimet ovat positiivisia, valittu kantaratkaisu ei ole optimaalinen.

*Vaihe* $V_2$: valitaan aluksi $x_R = x_1$, jolloin $x_2$ asetetaan nollaksi, ja yhtälöstä (3) saadaan

$$\begin{aligned} x_3 &= 16 - 2x_1 \geq 0 \;\Rightarrow\; x_1 \leq \frac{16}{2} = 8 \\ x_4 &= 8 - x_1 \geq 0 \;\Rightarrow\; x_1 \leq 8 \end{aligned}$$

Suurin mahdollinen $x_1$:n arvo on $\Delta x_1 = 8$, ja vastaava $f$:n arvon muutos

$$\Delta f = 150\Delta x_1 = 1200.$$

Jos $x_R = x_2$ ja $x_1 = 0$, saadaan vastaavasti

$$\begin{aligned} x_3 &= 16 - x_2 \geq 0 \;\Rightarrow\; x_2 \leq 16 \\ x_4 &= 8 - x_2 \geq 0 \;\Rightarrow\; x_2 \leq 8 \\ x_5 &= 3.5 - x_2 \geq 0 \;\Rightarrow\; x_2 \leq 3.5 \end{aligned}$$

Suurin mahdollinen $x_2$:n arvo on $\Delta x_2 = 3.5$, ja vastaava $f$:n arvon muutos

$$\Delta f = 300\Delta x_2 = 1050.$$

*Vaihe* $V_3$: Suurin $\Delta f$:n arvo $\Delta f = 1200$ saavutetaan valittaessa $x_R = x_1 = 8$, jolloin kumpikin kantamuuttujista $x_3$ ja $x_4$ tuli nollaksi. Uudeksi kantamuuttujaksi on siis valittava $x_1$, ja sen tilalle oikeanpuoleiseksi muuttujaksi nimetään joko $x_3$ tai $x_4$.

Vaihdetaan oikeanpuoleiseksi muuttujaksi $x_3$ ja ratkaistaan (3) uusien kantamuuttujien suhteen:

$$\begin{aligned} x_1 &= 8 - 0.5x_2 - 0.5x_3 \\ x_4 &= -0.5x_2 + 0.5x_3 \\ x_5 &= 3.5 - x_2 \end{aligned} \tag{4}$$

Merkitsemällä oikeanpuoleiset muuttujat nolliksi saadaan kantaratkaisu

$$x_1 = 8,\; x_2 = 0,\; x_3 = 0,\; x_4 = 0,\; x_5 = 3.5$$

Tämä kantaratkaisu on degeneroitunut, koska $n-m+1 = 5-3+1 = 3$ muuttujista on nollia.

Toinen iteraatiokierros sisältää myös vaiheet $V_1$, $V_2$ ja $V_3$.

*Vaihe* $V_1$: Yhdistämällä (1) ja (4) saadaan

$$f = 1200 + 225x_2 - 75x_3.$$

Koska $x_2$:n kerroin on positiivinen, tarkasteltava kantaratkaisu ei ole optimaalinen.

*Vaihe* $V_2$: Koska $x_3$:n kerroin on negatiivinen, riittää tarkastella tapausta $x_R = x_2$. Kun $x_3 = 0$, kaavoista (4) seuraa

$$\begin{aligned} x_1 &= 8 - 0.5x_2 \geq 0 \;\Rightarrow\; x_2 \leq 16 \\ x_4 &= -0.5x_2 \geq 0 \;\Rightarrow\; x_2 \leq 0 \\ x_5 &= 3.5 - x_2 \geq 0 \;\Rightarrow\; x_2 \leq 3.5 \end{aligned}$$

Suurin mahdollinen $x_2$:n arvo on $\Delta x_2 = 0$, jolloin $x_4$ saa myös arvon nolla.

*Vaihe* $V_3$: Tässä vaiheessa $x_4$ siirtyy $x_2$:n tilalle oikeanpuoleiseksi muuttujaksi. Ratkaistaan (4) uusien kantamuuttujien suhteen:

$$\begin{aligned} x_1 &= 8 - x_3 + x_4 \\ x_2 &= x_3 - 2x_4 \\ x_5 &= 3.5 - x_3 + 2x_4 \end{aligned} \tag{5}$$

Tässä muuttujien vaihdossa ei löydetty parempaa kantaratkaisua, sillä merkitsemällä oikeanpuoleiset muuttujat nolliksi saadaan tulokseksi aikaisempi kantaratkaisu.

Kolmannella iteraatiokierroksella suoritetaan jälleen vaiheet $V_1$, $V_2$ ja $V_3$.

*Vaihe* $V_1$: Kaavojen (1) ja (5) avulla saadaan $f$:lle esitys

$$f = 1200 + 150x_3 - 450x_4.$$

Optimaalisuusehto ei ole täytetty; tämä oli tietenkin selvää jo ennakolta, koska edellisessä iteraatiovaiheessa tarkasteltiin jo samaa kantaratkaisua.

*Vaihe* $V_2$: Koska $x_4$:n kerroin on negatiivinen, riittää tarkastella tapausta $x_R = x_3$. Kun $x_4 = 0$, kaavoista (5) seuraa

$$\begin{aligned} x_1 &= 8 - x_3 \geq 0 \;\Rightarrow\; x_3 \leq 8 \\ x_2 &= x_3 \geq 0 \;\Rightarrow\; x_3 \geq 0 \\ x_5 &= 3.5 - x_3 \geq 0 \;\Rightarrow\; x_3 \leq 3.5 \end{aligned}$$

Suurin mahdollinen $x_3$:n arvo on $\Delta x_3 = 3.5$, jota vastaava $f$:n arvon muutos

$$\Delta f = 150\Delta x_3 = 525.$$

*Vaihe* $V_3$: Nyt $x_5$ siirtyy $x_3$:n tilalle oikeanpuoleiseksi muuttujaksi. Ratkaistaan (5) uusien kantamuuttujien $x_1$, $x_2$, $x_3$ suhteen:

$$\begin{aligned} x_1 &= 4.5 - x_4 + x_5 \\ x_2 &= 3.5 - x_5 \\ x_3 &= 3.5 + 2x_4 - x_5 \end{aligned} \tag{6}$$

Neljäs iteraatiokierros:

*Vaihe* $V_1$: Kaavojen (1) ja (6) avulla saadaan $f$:lle esitys

$$f = 1725 - 150x_4 - 150x_5. \tag{7}$$

Optimaalisuusehdon perusteella edellisessä vaiheessa löydetty kantaratkaisu on optimaalinen. Kyseinen kantaratkaisu saadaan asettamalla (6):ssa oikeanpuoleiset muuttujat nolliksi:

$$x_1 = 4.5, \quad x_2 = 3.5, \quad x_3 = 3.5, \quad x_4 = x_5 = 0$$

Optimaalinen $f$:n arvo on (7):n mukaan $f_{opt} = 1725$. Voitto on siis maksimaalinen, jos päivittäin tuotetaan $x_1 = 4.5$ tonnia seosta $L_1$ ja $x_2 = 3.5$ tonnia seosta $L_2$. Raaka-ainetta $R_1$ jää tällöin $x_3 = 3.5$ tonnia ylitarpeen.

Lähtöpisteen löytäminen voi joskus tuottaa vaikeuksia simpleksimenetelmässä. Tällöin voi olla paikallaan ottaa käyttöön pelivaramuuttujien lisäksi keinotekoisia apumuuttujia.

**Esimerkki 2.** Maksimoidaan

$$f = 2x_1 + x_2 \tag{8}$$

ehdoilla $x_1 \geq 0$, $x_2 \geq 0$ ja

$$\begin{aligned} x_1 - \frac{1}{2}x_2 &\geq 1 \\ x_1 - x_2 &\leq 2 \\ x_1 + x_2 &\leq 4 \end{aligned}$$

*Ratkaisu.* Ottamalla käyttöön pelivaramuuttujat päästään normaalimuotoon

$$\begin{aligned} -x_1 + \frac{1}{2}x_2 + x_3 &= -1 \\ x_1 - x_2 + x_4 &= 2 \\ x_1 + x_2 + x_5 &= 4 \\ x_i &\geq 0 \quad (i = 1, \ldots, 5) \end{aligned} \tag{9}$$

Valitaan kantamuuttujiksi $x_3$, $x_4$, $x_5$:

$$\begin{aligned} x_3 &= -1 + x_1 - \frac{1}{2}x_2 \\ x_4 &= 2 - x_1 + x_2 \\ x_5 &= 4 - x_1 - x_2 \end{aligned}$$

Jos tässä $x_1 = x_2 = 0$, niin $x_3 = -1$ on negatiivinen eikä tulokseksi saada lähtöpisteessä hyväksyttävää kantaratkaisua. Nyt voitaisiin tietenkin valita kantamuuttujat toisella tavoin, mutta vaihtoehtoinen keino on ottaa käyttöön uusi *apumuuttuja* $x_6 \geq 0$ siten, että

$$x_3 = -1 + x_1 - \frac{1}{2}x_2 + x_6. \tag{10}$$

Kun kantamuuttujiksi valitaan $x_4$, $x_5$, $x_6$, saadaan

$$\begin{aligned} x_4 &= 2 - x_1 + x_2 \\ x_5 &= 4 - x_1 - x_2 \\ x_6 &= 1 - x_1 + \frac{1}{2}x_2 + x_3 \end{aligned} \tag{11}$$

Näillä kantamuuttujilla on positiiviset arvot silloin kun oikeanpuoleiset muuttujat asetetaan nolliksi.

Tällöin saadaan siis kantaratkaisu laajennetulle optimointitehtävälle, jossa on 3 yhtälöä ja 6 muuttujaa. Tämän uuden tehtävän kantaratkaisusta saadaan kantaratkaisu alkuperäiselle tehtävälle, mikäli $x_6 = 0$.

Yhtälöiden (11) perusteella saadussa kantaratkaisussa

$$x_1 = x_2 = x_3 = 0, \quad x_4 = 2, \quad x_5 = 4, \quad x_6 = 1 \tag{12}$$

$x_6$ on positiivinen. Jotta $x_6$ saataisiin nollaksi, muutetaan maksimoitavaa funktiota lisäämällä siihen termi $-Mx_6$, missä $M$ on hyvin suuri. Uusi maksimoitava funktio on siis

$$\tilde{f} = f - Mx_6 = (M+2)x_1 - \big(\frac{1}{2}M - 1\big)x_2 - Mx_3 - M. \tag{13}$$

Jos tätä funktiota pyritään maksimoimaan simpleksimentelmällä käyttämällä lähtöpisteenä kantaratkaisua (12), päädytään ennen pitkää sellaiseen kantaratkaisuun, jossa $x_6 = 0$. Funktion $\tilde{f}$ määritelmästä johtuen sen arvo ei näet voi olla maksimaalinen mikäli $x_6 > 0$.

Algoritmin ensimmäisessä vaiheessa asetetaan $x_R = x_1$, sillä $x_1$:n kerroin on $\tilde{f}$:n lausekkeessa positiivinen jos $M$ on suuri. Kun muut oikeanpuoleiset muuttujat merkitään nollaksi, kaavoista (11) sauraa

$$\begin{aligned} x_4 = 2 - x_1 \geq 0 \;&\Rightarrow\; x_1 \leq 2 \\ x_5 = 4 - x_1 \geq 0 \;&\Rightarrow\; x_1 \leq 4 \\ x_6 = 1 - x_1 \geq 0 \;&\Rightarrow\; x_1 \leq 1 \end{aligned}$$

Suurin mahdollinen $x_1$:n arvo on $\Delta x_1 = 1$, ja vastaava $\tilde{f}$:n arvon muutos

$$\Delta\tilde{f} = (M+2)\Delta x_1 = M + 2.$$

Muuttujien $x_1$ ja $x_6$ vaihto johtaa systeemiin

$$\begin{aligned} x_1 &= 1 + \frac{1}{2}x_2 + x_3 - x_6 \\ x_4 &= 1 + \frac{1}{2}x_2 - x_3 + x_6 \\ x_5 &= 3 - \frac{3}{2}x_2 - x_3 + x_6 \end{aligned}$$

jolloin (13):n perusteella

$$\tilde{f} = 2 + 2x_2 + 2x_3 - (M+2)x_6.$$

Kun nyt oikeanpuoleiset muuttujat asetetaan nolliksi, saadaan kantaratkaisu

$$x_1 = 1, \quad x_2 = 0, \quad x_3 = 0, \quad x_4 = 1, \quad x_5 = 3, \quad x_6 = 0.$$

Koska $x_6 = 0$, tämä antaa kantaratkaisun myös alkuperäiselle tehtävälle, jonka ratkaisua voidaan tämän jälkeen jatkaa normaalisti.

# 5 Verkot

*Verkko* $G$ koostuu kahdesta äärellisestä joukosta $V$ ja $E$ siten, että jokaiseen $E$:n alkioon on liitetty kaksi $V$:n alkiota. Joukon $V$ alkiot ovat *solmuja* ja $E$:n alkiot ovat *kaaria*. Jokaiseen kaareen liittyy siis kaksi solmua, joita kutsutaan kaaren *päätepisteiksi*. Merkitsemme $G = (V, E)$. Yksinkertaisuuden vuoksi rajoitumme tarkastelemaan verkkoja, joilla on seuraavat ominaisuudet:

- jokainen solmu on jonkin kaaren päätepiste
- kaari ei ole koskaan ns. silmukka, so. sen päätepisteet ovat aina eri pisteitä
- jokaista solmuparia yhdistää korkeintaan yksi kaari

Merkitsemme solmuja kirjaimin $u$, $v$, ... tai $v_1$, $v_2$, ..., tai numeroin 1, 2, .... Kaaria merkitään $e_1$, $e_2$, ... tai päätepisteitä käyttäen esim. $e_1 = (1, 4)$, $e_2 = (1, 2)$. Sanomme että solmu $v$ *kuuluu* kaareen $e$, jos $v$ on $e$:n päätepiste. Solmun $v$ *aste* on niiden kaarien lukumäärä joihin $v$ kuuluu.

*Suunnattu verkko* $G = (V, E)$ on verkko, jossa jokaisella kaarella $(i, j)$ on annettu suunta *alkupisteestä* $i$ *päätepisteeseen* $j$. Yllämainituista rajoituksista poiketen sallitaan suunnatuille verkoille myös tilanne, jossa solmuparia yhdistää kaksi vastakkaissuuntaista kaarta.

Verkko $G_1 = (V_1, E_1)$ on verkon $G = (V, E)$ *aliverkko*, jos $V_1 \subset V$ ja $E_1 \subset E$. Jos $G$ on suunnattu verkko, myös $G_1$ on suunnattu verkko, jossa jokaisen $E_1$:n alkion suunta on sama kuin sen suunta suunnatussa verkossa $G$.

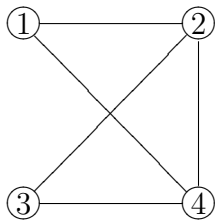
### Verkkojen esittäminen tietokoneessa

Verkon $G$ *yhteysmatriisi* $A = [a_{ij}]$ määritellään siten, että

$$a_{ij} = \begin{cases} 1, \text{ jos } G \text{ sisältää kaaren } (i, j) \\ 0 \text{ muulloin} \end{cases}$$

$A$ on siis symmetrinen neliömatriisi, jonka sarakkeiden lukumäärä on sama kuin solmujen määrä verkossa $G$. Sopimuksemme mukaisesti kaikki $A$:n diagonaaliset alkiot ovat nollia.

**Esimerkki 1.** Tarkastellaan seuraavaa suuntaamatonta verkkoa.



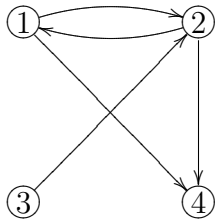
Kuvan esittämän verkon yhteysmatriisi on

$$\begin{array}{r c} \text{Solmun n:o} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{r} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

*Suunnatun verkon yhteysmatriisi* $A = [a_{ij}]$ määritellään siten, että

$$a_{ij} = \begin{cases} 1, \text{ jos } G \text{ sisältää kaaren } (i,j), \text{ jonka päätepiste on } j \\ 0 \text{ muulloin} \end{cases}$$

**Esimerkki 2.** Terkastellaan seuraavanlaista suunnattua verkkoa.




Kuvan esittämän suunnatun verkon yhteysmatriisi on

$$\begin{array}{r c} \text{Solmun n:o} & \begin{array}{cccc} 1 & 2 & 3 & 4 \end{array} \\ \begin{array}{r} 1 \\ 2 \\ 3 \\ 4 \end{array} & \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

### Vastaavuustaulukot

Verkon solmujen *vastaavuustaulukko* osoittaa kullekin solmulle ne kaaret, joilla on päätepisteinä kyseinen solmu. *Kaarien vastaavuustaulukko* osoittaa jokaisen kaaren päätepisteet.

Suunnatun verkon solmujen vastaavuustaulukossa varustetaan miinusmerkillä ne kaaret, joilla on *alkupisteenä* tarkasteltava solmu, ja kaarien vastaavuustaulukossa jokaista kaarta vastaa *järjestetty* solmupari.

**Esimerkki 3.** Esimerkin 1 verkon solmujen ja kaarien vastaavuustaulukot.

| Solmu | Vastaavat kaaret | Kaari | Päätepisteet |
|---|---|---|---|
| $v_1$ | $e_1, e_5$ | $e_1$ | $v_1, v_2$ |
| $v_2$ | $e_1, e_2, e_3$ | $e_2$ | $v_2, v_3$ |
| $v_3$ | $e_2, e_4$ | $e_3$ | $v_2, v_4$ |
| $v_4$ | $e_3, e_4, e_5$ | $e_4$ | $v_3, v_4$ |
| | | $e_5$ | $v_1, v_4$ |

Verkossa, jossa on $n$ solmua, kaarien lukumäärä on korkeintaan $\binom{n}{2} = \frac{n(n-1)}{2}$. Verkko on *harva*, jos kaarien lukumäärä on tätä oleellisesti pienempi. Harva verkko on usein parempi esittää käyttämällä vastaavuustaulukkoa, sillä yhteysmatriisi vaatii tällöin enemmän muistitilaa tietokoneessa.

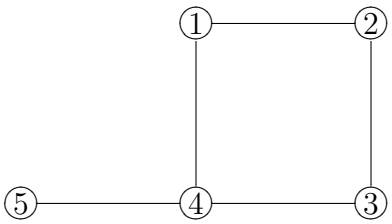
# 6 Minimietäisyydet ja -reitit

*Polku* verkon $G = (V, E)$ solmusta $v_1$ solmuun $v_k$ on muotoa

$$(v_1, v_2),\ (v_2, v_3),\ \ldots,\ (v_{k-1}, v_k). \tag{1}$$

Polku koostuu siis peräkkäisistä kaarista. Sama kaari voi esiintyä polussa useampia kertoja. Polku on *ketju*, jos kukin kaari esiintyy siinä vain kerran. Ketju on *yksinkertainen*, mikäli se ei sisällä risteyksiä, so. jokaisessa solmussa käydään vain kerran (poikkeus: alku- ja loppupiste voivat olla samat, jolloin (1):ssä $v_k = v_1$). Yksinkertainen umpinainen ketju on *sykli*.

**Esimerkki.** Tarkastellaan seuraavanlaista suuntaamatonta verkkoa.




Verkosta löytyy mm. seuraavat reitit solmujen avulla ilmaistuna:

| | |
|---|---|
| 1-2-3-2 | polku (ei ketju) |
| 4-1-2-3-4-5 | ketju (ei yksinkertainen) |
| 1-2-3-4-5 | yksinkertainen ketju (ei sykli) |
| 1-2-3-4-1 | sykli |

Koska jokaista solmuparia yhdistää korkeintaan yksi kaari (Luvun 5 sopimus), niin jokaisessa syklissä on vähintään kolme kaarta.

Oletetaan, että verkon $G = (V, E)$ jokaisella kaarella $(v_i, v_j)$ on annettu "pituus" $l_{ij} > 0$. Polun (1) kokonaispituus on silloin $l_{12} + l_{23} + l_{34} + \cdots + l_{k-1,k}$.

**Probleema.** Löydettävä muotoa (1) oleva yksinkertainen ketju solmusta $v_1$ solmuun $v_k$ siten, että ketjun kokonaispituus on mahdollisimman pieni (vast. mahdollisimman suuri).

Käytännön sovelluksissa "pituus" $l_{ij}$ voi tarkoittaa tilanteesta riippuen esim. matkaa, aikaa tai kustannuksia.

**Esimerkki 1.** "Kauppamatkustajan ongelma": Etsittävä annetussa verkossa lyhyin ns. *Hamiltonin sykli*, so. sellainen umpinainen ketju joka sisältää verkon kaikki solmut.

**Esimerkki 2.** Kauppamatkustajaa voi kiinnostaa myös "tuottoisin matkareitti". Tällöin pyritään maksimoimaan $\sum l_{ij}$, missä $l_{ij}$ on arvioitu provisio vähennettynä matkakustannuksilla kaupungista $i$ kaupunkiin $j$.

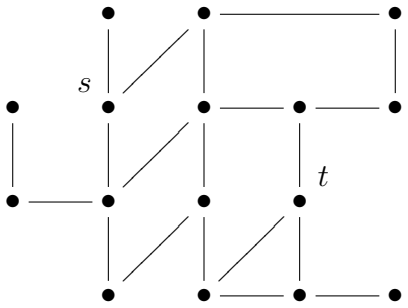
Tarkastellaan aluksi erikoistapausta, jossa jokaisen kaaren pituus on 1. Silloin lyhin ketju solmusta $v_1$ solmuun $v_k$ on tietenkin mikä hyvänsä ketju $v_1 \to v_k$, jossa kaarien lukumäärä on mahdollisimman pieni. Tällainen ketju löydetään esim. ns. *BFS-algoritmilla* ("Breadth First Search") seuraavasti.

Jotta tehtävällä olisi aina ratkaisu, oletetaan että verkko $G$ on *yhtenäinen* so. mitkä hyvänsä kaksi $G$:n solmua voidaan yhdistää polulla. Olkoon $a$ etsittävän ketjun alkupiste ja $p$ sen päätepiste. BFS-algoritmissa liitetään verkon solmuun kokonaislukuarvo $k$, jos kyseinen solmu on sellaisen kaaren päätepiste, jonka pituus on $k$ ja alkupiste $a$.

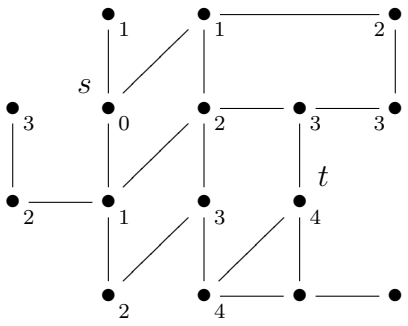
Algoritmi etenee seuraavasti:

1. Liitetään solmuun $a$ kokonaislukuarvo 0
2. Asetetaan $i = 0$
3. Etsitään kaikki kaaret, joiden toiseen päätepisteeseen on liitetty kokonaislukuarvo $i$
4. Liitetään kokonaislukuarvo $i + 1$ kaikkiin niihin solmuihin, joihin ei vielä ole liitetty kokonaislukuarvoa ja jotka ovat päätepisteenä jossakin kohdassa 3 löydetyssä kaaressa
5. Jos solmuun $p$ on liitetty kokonaislukuarvo $k$, etsitty ketju löydetään tarkastelemalla algoritmia takaperin ja etsimällä sellainen jono solmuja, joihin liittyvät kokonaislukuarvot ovat $k$, $k-1$, $k-2$, $\ldots$, 0.
6. Jos solmuun $p$ ei ole liitetty kokonaislukuarvoa, asetetaan $i = i + 1$ ja palataan kohtaan 3

**Esimerkki 3.** Etsitään kuvan osoittamassa verkossa lyhin polku $s \to t$.




Soveltamalla BFS-algoritmia saadaan seuraavanlainen verkko.



Ratkaisu ei ole yksikäsitteinen. Jos verkon solmut on numeroitu $1, \ldots, n$ saadaan yksikäsitteisyys aikaan esim. valitsemalla vaiheessa 5 sellainen solmujono, johon kuuluvien solmujen järjestysnumerot ovat mahdollisimman pienet. On huomioitava, että tässä järjestysnumero ei ole yleensä sama kuin BFS-algoritmissa solmuun liitetty kokonaislukuarvo.

### Algoritmin kompleksisuus

Äskeisessä BFS-algoritmissa joudutaan kutakin kaarta tarkastelemaan yleensä kahdesti. Esim. arvolla $i = 0$ tarkastellaan aluksi kaikkia kaaria, joilla on päätepisteenä $a$. Koska näiden kaarien toiseen päätepisteeseen liitetään kokonaislukuarvo 1, kyseisiä kaaria tullaan tarkastelemaan uudelleen algoritmin seuraavalla kierroksella, jolloin $i = 1$. Jos verkon kaarien lukumäärä on $m$, algoritmin maksimaalinen suoritusaika on siis verrannollinen lukuun $2m$.

Yleisesti algoritmin $\mathcal{A}$ *aikakompleksisuus* $c_{\mathcal{A}}$ tarkoittaa algoritmin maksimaalista tai keskimääräistä suoritusaikaa. Tällöin $c_{\mathcal{A}}$ on funktio, joka riippuu probleeman koosta. Verkkoteorian probleemoissa koko on usein joko kaarien lukumäärä $m$ tai solmujen lukumäärä $n$. Äskeisessä BFS-algoritmissa $c_{\mathcal{A}}(m)$ on verrannollinen lukuun $2m$.

Algoritmin käyttökelpoisuuden kannalta on olennaista, että $c_{\mathcal{A}}(m)$ ei kasva liian nopeasti $m$:n kasvaessa. Seuraavassa $O(m^k)$ tarkoittaa mitä hyvänsä sellaista $m$:n funktiota, jonka suhde potenssiin $m^k$ pysyy rajoitettuna kun $m \to \infty$. Jos algoritmin $\mathcal{A}$ maksimaalinen suoritusaika $c_{\mathcal{A}}(m) = O(m^k)$ jollekin kokonaisluvulle $k$, sanomme että $\mathcal{A}$ on *polynomisesti rajoitettu*. Esim. äskeinen BFS-algoritmi on polynomisesti rajoitettu, sillä siinä $c_{\mathcal{A}}(m) = O(m^1)$.

### Bellmanin periaate

Tarkastellaan jälleen tilannetta, jossa verkon $G = (V, E)$ jokaiselle kaarelle $(i, j)$ on annettu pituus $l_{ij} > 0$. Seuraava ns. Bellmanin periaate antaa välttämättömän ehdon sille, että jokin ketju $P : 1 \to j$ solmusta 1 solmuun $j$ on mahdollisimman lyhyt.

**Bellmanin periaate.** *Olkoon $P : 1 \to j$ mahdollisimman lyhyt ketju solmusta 1 solmuun $j$, ja olkoon $(i, j)$ ketjun $P$ viimeinen kaari. Silloin lyhyin ketju solmusta 1 solmuun $i$ saadaan poistamalla $P$:stä viimeinen kaari $(i, j)$.*

*Todistus.* Olkoon $P_i : 1 \to i$ se ketju, joka saadaan $P$:stä poistamalla siitä viimeinen kaari $(i, j)$, ja olkoon $P_i^* : 1 \to i$ mikä hyvänsä ketju, jonka pituus $l^*$ on mahdollisimman pieni. Silloin $l^*$ on korkeintaan yhtäsuuri kuin $P_i$:n pituus $l_i$ ($l^* \leq l_i$). Toisaalta liittämällä ketjuun $P_i^*$ kaari $(i, j)$ saadaan ketju $1 \to j$, jonka pituus $l^* + l_{ij}$ on vähintään $P$:n pituus $l_i + l_{ij}$. Siis

$$l^* \leq l_i \quad \text{ja} \quad l^* + l_{ij} \geq l_i + l_{ij}$$

joista seuraa $l^* = l_i$. Siis $P_i$ on mahdollisimman lyhyt. □

Bellmanin periaatteesta on helppo johtaa ns. *Bellmanin yhtälöt*. Olkoon $P_i$ lyhyin ketju solmusta 1 solmuun $i$, ja olkoon $L_i$ ketjun $P_i$ pituus. Asetetaan $l_{ij} = \infty$ aina kun $E$ ei

sisällä kaarta $(i, j)$. Silloin

$$\begin{aligned} L_1 &= 0 \\ L_j &= \min_{i \neq j}(L_i + l_{ij}) \quad (2 \leq j \leq n) \end{aligned} \tag{$*$}$$

Jälkimmäinen näistä ns. Bellmanin yhtälöistä seuraa välittömästi Bellmanin periaatteesta. Itse asiassa

$$L_j = \min_{i}(L_i + l_{ij}) \quad (2 \leq j \leq n)$$

sillä $l_{ii} = 0$ (sopimus).

Bellmanin periaatteeseen nojaa myöskin seuraava *Dijkstran algoritmi*, joka etsii annetussa äärellisessä yhtenäisessä verkossa lyhyimmän ketjun solmusta 1 verkon muihin solmuihin.

Jos verkossa on $n$ solmua, Dijkstran algoritmi etsii bijektion

$$f : \{1, \ldots, n\} \to \{1, \ldots, n\}$$

siten, että $f(1) = 1$ ja Bellmanin yhtälöissä käytetyin merkinnöin pätee

$$L_{f(1)} \leq L_{f(2)} \leq \cdots \leq L_{f(n)}.$$

Algoritmi etenee seuraavasti:

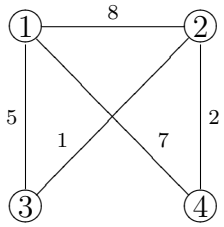
1. Määritellään $f(1) = 1$, jolloin $L_{f(1)} = 0$
2. Valitaan mahdollisimman pieni $j$ siten, että $l_{1j} = \min\{l_{1k} : k \neq 1\}$ ja määritellään $f(2) = j$; Tällöin $L_{f(2)} = l_{1j}$ on lyhyin matka solmusta 1 solmuun $f(2)$
3. Kun $f(1)$, $f(2)$, …, $f(k)$ on määritelty, valitaan mahdollisimman pieni $j$ siten, että $j$ ei kuulu joukkoon $\{f(1), f(2), \ldots, f(k)\}$ ja

   $$\min_{1 \leq i \leq k}\{L_{f(i)} + l_{f(i),j}\} \tag{2}$$

   on mahdollisimman pieni. Määritellään $f(k+1) = j$, jolloin $L_{f(k+1)}$ on lausekkeen (2) arvo kun $j = f(k+1)$.

Dijkstran algoritmin avulla löydetään lyhyin ketju solmusta 1 verkon jokaiseen solmuun seuraavasti. Jos vaiheessa 2 $f(2) = j$, niin lyhyin ketju solmusta 1 solmuun $f(2)$ käsittää kaaren $(1, j)$. Jos lyhyin ketju solmusta 1 solmuihin $f(1)$, $f(2)$, …, $f(k)$ on löydetty, saadaan lyhyin ketju solmuun $f(k+1)$ tarkastelemalla vaiheessa 3 sitä $i$:n arvoa, jolla lauseke (2) saa minimin ja lisäämällä kaari $(f(i), f(k+1))$ lyhyimpään solmuun $f(i)$ päättyvään ketjuun.

**Esimerkki.** Etsitään Dijkstran algoritmin avulla lyhyimmät ketjut oheisen verkon solmusta 1 solmuihin 2, 3 ja 4.



1. $l_{12} = 8$, $l_{13} = 5$ ja $l_{14} = 7$; siis

$$l_{13} = \min\{l_{1k} : k \neq 1\} = 5$$

ja $f(2) = 3$, $L_{f(2)} = L_3 = 5$.

2. j=2:

$$L_{f(1)} + l_{f(1),2} = 0 + l_{12} = 8$$
$$L_{f(2)} + l_{f(2),2} = 5 + l_{32} = 5 + 1 = 6$$

j=4:

$$L_{f(1)} + l_{f(1),4} = 0 + l_{14} = 7$$
$$L_{f(2)} + l_{f(2),4} = 5 + l_{34} = 5 + \infty = \infty$$

Arvolla $k = 2$ lausekkeen (2) pienin mahdollinen arvo on siis 6, ja se saavutetaan kun $j = 2$. Näin ollen $f(3) = 2$ ja $L_{f(3)} = L_2 = 6$.

3. Seuraavaksi on etsittävä

$$\min_{1 \leq i \leq 3} \{L_{f(i)} + l_{f(i),4}\}.$$

$L_{f(1)} + l_{f(1),4} = L_1 + l_{14} = 0 + 7 = 7$
$L_{f(2)} + l_{f(2),4} = L_3 + \infty = \infty$ ja
$L_{f(3)} + l_{f(2),4} = 6 + 2 = 8$

Lyhyimpien ketjujen pituudet ovat siis $L_2 = 6$, $L_3 = 5$ ja $L_4 = 7$, ja vastaavat ketjut ovat 1-3-2, 1-3 ja 1-4.

**Väite.** Dijkstran algoritmin kompleksisuus on $O(n^3)$.

*Todistus.* Muotoa (2) oleva lauseke joudutaan algoritmin $k$:nnellä kierroksella laskemaan $(n - k)$:lla eri $j$:n arvolla. Jokaista $j$:n arvoa kohden tarvitaan tällöin $k$ kappaletta yhteenlaskutoimituksia, joiden lopputuloksia on lisäksi vertailtava keskenään minimin löytämiseksi. Yhteensä $k$:nnella kierroksella tarvitaan siis $k(n - k)$ yhteenlaskua ja $k(n - k)$ vertailuoperaatiota. Koska algoritmissa tarvitaan

$$1 \cdot (n-1) + 2 \cdot (n-2) + \cdots + (n-1) \cdot 1 = \frac{1}{6}(n^3 - n)$$

yhteenlaskua ja yhtä monta vertailuoperaatiota. □

# 7 Virittävät puut

Verkko on *yhtenäinen*, jos mitkä hyvänsä kaksi solmua voidaan yhdistää ketjulla. *Puu* on yhtenäinen verkko, joka ei sisällä syklejä. Annetun yhtenäisen verkon $G = (V, E)$ aliverkko $T$ on $G$:n *virittävä puu*, jos $T$ on sellainen puu joka sisältää kaikki $G$:n solmut. Jos $G$:ssä on $n$ solmua, niin jokaisessa $G$:n virittävässä puussa on $n - 1$ kaarta.

Jos verkon $G$ jokaiseen kaareen $(i, j)$ liittyy pituus $l_{ij} > 0$, voidaan etsiä *lyhintä $G$:n virittävää puuta*, so. sellaista virittävää puuta jonka kaarien pituuksien summa on mahdollisimman pieni.

**Esimerkki 1.** Rataverkon suunnittelu: $l_{ij}$ on kaupunkeja $i$ ja $j$ yhdistävän rautatien rakennushinta. Pyritään suunnittelemaan sellainen kaikki annetut kaupungit yhdistävä rataverkko (puu) että $\sum l_{ij}$ on mahdollisimman pieni.

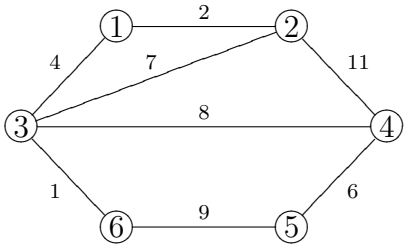
**Esimerkki 2.** Rahtiliikenteen suunnittelu: $l_{ij}$ on laivanvarustamon arvioima nettovoitto kaupunkien $i$ ja $j$ välisessä linjaliikenteessä. Pyritään löytämään sellainen annetut kaupungit yhdistävä puu, että $\sum l_{ij}$ on mahdollisimman suuri.

Seuraava ns. *Kruskalin algoritmi* etsii annetussa äärellisessä yhtenäisessä verkossa $G = (V, E)$ lyhyimmän virittävän puun, kun verkon kaarien $(i, j)$ pituudet $l_{ij} > 0$ on annettu.

1. Järjestetään $G$:n kaaret pituusjärjestykseen
2. Valitaan virittävään puuhun aluksi $G$:n lyhyin kaari, sitten toiseksi lyhyin jne. pituusjärjestyksessä ja jätetään valitsematta vain ne kaaret, jotka yhdessä aikaisemmin valittujen kaarien kanssa muodostaisivat jonkun syklin
3. Kun $n - 1$ kaarta on valittu yo. tavalla, lopputuloksena on etsitty virittävä puu

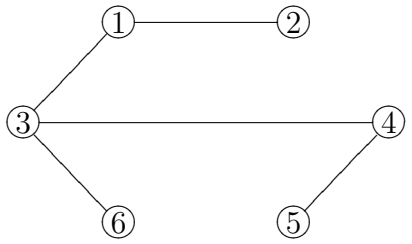
**Huomautus.** Ensimmäiset valituista kaarista eivät yleensä muodosta yhtenäistä verkkoa, mutta lopputulos on tietenkin aina yhtenäinen.

**Esimerkki 3.** Sovelletaan Kruskalin algoritmia seuraavanlaiseen verkkoon.




| Kaari | Pituus | Valinta |
|---|---|---|
| $(3, 6)$ | 1 | 1 |
| $(1, 2)$ | 2 | 2 |
| $(1, 3)$ | 4 | 3 |
| $(4, 5)$ | 6 | 4 |
| $(2, 3)$ | 7 | ei valita |
| $(3, 4)$ | 8 | 5 |
| $(5, 6)$ | 9 | |
| $(2, 4)$ | 11 | |

Tuloksena saadaan siis seuraavanlainen virittävä puu:




Kruskalin algoritmi soveltuu erityisesti *harvoihin* verkkoihin, so. verkkoihin joissa on solmujen lukumäärään nähden vähän kaaria. Käytännössä algoritmi voidaan toteuttaa seuraavasti.

Olkoon $T_k$ se (ei välttämättä yhtenäinen) $G$:n aliverkko joka koostuu algoritmin $k$:nnessa vaiheessa valituista $k$:sta kaaresta (jotka siis ovat lopullisen puun $k$ lyhyintä kaarta). Liitetään jokaiseen $T_k$:n solmuun $i$ lukupari $(r_i, p_i)$ seuraavasti:

$r_i$ on mahdollisimman pieni kokonaisluku siten, että solmut $i$ ja $r_i$ voidaan yhdistää verkkoon $T_k$ sisältyvällä ketjulla. $p_i = 0$, jos $i = r_i$; muulloin $p_i$ on solmua $i$ seuraava solmu yllämainitussa ketjussa solmusta $i$ solmuun $r_i$.

Kun $T_k$:n solmuihin on liitetty lukuparit $(r_i, p_i)$, algoritmia on helpompi jatkaa. Verkon $G$ jokin kaari $(i, j)$ muodostaa nimittäin $T_k$:n kaarien kanssa syklin täsmälleen silloin kun $r_i = r_j$. Näin ollen algoritmin $k + 1$ kierroksella kaari $(i, j)$ valittaisiin virittävään puuhun vain jos $r_i \neq r_j$.

Painon $(r_i, p_i)$ avulla voidaan ilmoittaa myös $T_k$:n kaaret, jotka ovat kaikki muotoa $(i, p_i)$. On huomattava, että solmuun $i$ voi algoritmin kuluessa liittyä useita eri pareja $(r_i, p_i)$; tämä seuraa siitä, että liitettäessä verkkoon $T_k$ uusia kaaria sen komponentit sulautuvat lopulta yhteen. Algoritmin $k$:nnessa vaiheessa on tiedostossa kuitenkin säilytettävä vain kyseisessä vaiheessa voimassa olevat lukuparit.

Esimerkin 3 tapauksessa saadaan seuraava taulukko.

| | 1. valinta | 2. valinta | 3. valinta | 4. valinta | 5. valinta |
|---|---|---|---|---|---|
| Solmu | (3,6) | (1,2) | (1,3) | (4,5) | (3,4) |
| 1 | | $(1, 0)$ | | | |
| 2 | | $(1, 1)$ | | | |
| 3 | $(3, 0)$ | | $(1, 1)$ | | |
| 4 | | | | $(4, 0)$ | $(1, 3)$ |
| 5 | | | | $(4, 4)$ | $(1, 3)$ |
| 6 | $(3, 3)$ | | $(1, 3)$ | | |

Huomataan, että algoritmin 5. kierroksella ei valita kaarta $(2, 3)$, koska solmuihin 2 ja 3 liittyy sama $r_i$. Merkittävä osa Kruskalin algoritmin vaatimasta työmäärästä kuluu asetettaessa $G$:n kaaret pituusjärjestykseen. Aikakompleksisuus on $O(m \log n)$.

Lyhyin virittävä puu voidaan konstruoida myös käyttämällä ns. *Primin algoritmia*, jossa mistä hyvänsä solmusta lähtien etsitään virittävä puu kaari kaarelta. Algoritmin $k$:nnella

kierroksella syntyy tällöin jonkin $G$:n aliverkon virittävä puu $T_k$, joka toisin kuin Kruskalin algoritmissa on aina yhtenäinen. Lisäksi jokaiseen $T_k$:hon kuulumattomaan solmuun $i$ liitetään reaaliluku

$$\lambda_i = \min\{l_{ij} : j \neq i \text{ on } T_k\text{:n solmu}\} \tag{1}$$

missä aikaisemman sopimuksen mukaisesti $l_{ij} = \infty$ jos kaari $(i, j)$ ei sisälly verkkoon $G$.

Lähdettäessä liikkeelle solmusta 1 määritellään aluksi

$$\lambda_i = l_{i1} \qquad (2 \leq i \leq n)$$

ja valitaan pienin indeksi $j$ siten että

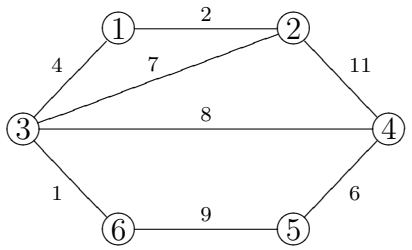
$$\lambda_j = \min\{\lambda_i : 2 \leq i \leq n\}.$$

Määritellään $T_1$ siten, että $T_1$ koostuu kaaresta $(1, j)$ ja sen päätepisteistä. Kun $T_1$, $T_2$, ..., $T_k$ on konstruoitu, määritellään aluksi luvut $\lambda_i$ kaavan (1) mukaisesti ja valitaan pienin indeksi $j$ siten, että $j$ ei ole $T_k$:n solmu ja

$$\lambda_j = \min\{\lambda_i : i \text{ ei ole } T_k\text{:n solmu}\}.$$

Edelleen (1):n perusteella valitaan pienin indeksi $i$ siten, että $\lambda_j = l_{ji}$ ja solmu $i$ kuuluu puuhun $T_k$. Tämän jälkeen $T_{k+1}$ saadaan $T_k$:sta liittämällä siihen kaari $(i, j)$. Arvolla $k = n - 1$ saadaan etsitty lyhyin virittävä puu $T_{n-1}$.

**Esimerkki.** Sovelletaan Primin algoritmia alla olevaan verkkoon.




1. Ensimmäisessä vaiheessa $\min\{\lambda_i; 2 \leq i \leq 6\} = \lambda_2 = 2$. Siis $T_1$ koostuu kaaresta $(1, 2)$ ja sen päätepisteistä.

2. Arvolla $k = 1$ saadaan $\lambda_3 = \min\{\lambda_i;\ i \text{ ei ole } T_1\text{:n solmu}\}$; $T_2$ saadaan $T_1$:stä liittämällä siihen kaari $(1, 3)$.

3. Arvolla $k = 2$ saadaan $\lambda_6 = \min\{\lambda_i;\ i \text{ ei ole } T_2\text{:n solmu}\} = l_{63} = 1$; $T_3$ saadaan $T_2$:sta liittämällä siihen kaari $(3, 6)$.

Luvut $\lambda_i$ algoritmin eri vaiheissa

| Solmu | $k = 0$ | $k = 1$ | $k = 2$ | $k = 3$ | $k = 4$ |
|---|---|---|---|---|---|
| 2 | $l_{21} = 2$ | | | | |
| 3 | $l_{31} = 4$ | $l_{31} = 4$ | | | |
| 4 | $\infty$ | $l_{42} = 11$ | $l_{43} = 8$ | $l_{43} = 8$ | |
| 5 | $\infty$ | $\infty$ | $\infty$ | $l_{56} = 9$ | $l_{54} = 6$ |
| 6 | $\infty$ | $\infty$ | $l_{63} = 1$ | | |

# 8 Virtaukset verkoissa

Olkoon $G = (V, E)$ yhtenäinen suunnattu verkko, jonka jokaisella kaarella $(i, j)$ on annettu "kapasiteetti" $c_{ij} > 0$, so. $c_{ij}$ on suurin mahdollinen virtaus kaarella $(i, j)$. *Virtaus* verkossa $G$ on joukossa $E$ määritelty reaaliarvoinen funktio $(i, j) \mapsto f_{ij}$ siten, että

$$0 \leq f_{ij} \leq c_{ij} \qquad \text{kaikille } (i, j) \in E \tag{1}$$

**Huomautus.** Jos $e = (i, j)$, merkitään $f_e = f_{ij}$.

Solmuun $i$ *saapuvien kaarien joukko* $E_\sigma(i)$ käsittää ne $E$:n kaaret, joiden päätepiste on $i$, ja solmusta $i$ *lähtevien kaarien joukko* $E_\tau(i)$ käsittää ne $E$:n alkiot joiden alkupiste on $i$. Solmuun $i$ *saapuva kokonaisvirtaus* on

$$\sigma(i) = \sum_{e \in E_\sigma(i)} f_e$$

ja solmusta $i$ *lähtevä kokonaisvirtaus* on

$$\tau(i) = \sum_{e \in E_\tau(i)} f_e.$$

Sanomme, että solmu $i \in V$ on *lähde*, jos $\tau(i) > \sigma(i)$, ja *kohde* eli *nielu*, jos $\tau(i) < \sigma(i)$.

Seuraavassa rajoitumme virtauksiin, joilla on vain yksi lähde $s$ ja yksi kohde $t$. Myöhemmin nähdään, että tällöin aina $\tau(s) = \sigma(t)$; tämä luku on *kokonaisvirtaus* verkossa $G$. Merkitään sitä $f$:llä, siis

$$f = \tau(s) = \sigma(t),$$

edellyttäen, että $\sigma(s) = \tau(t) = 0$. Lisäksi $\tau(i) = \sigma(i)$ jokaiselle solmulle $i \in V \setminus \{s, t\}$ (Kirchhoffin laki).

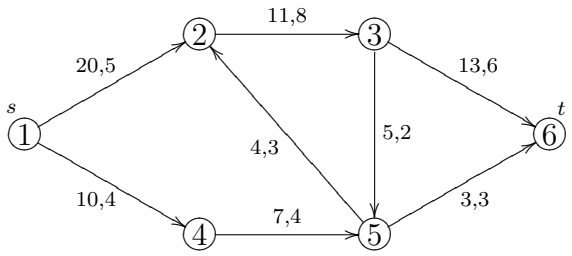
Suunnattuun verkkoon $G = (V, E)$ liittyy vastaava suuntaamaton verkko, jonka kaarina ovat kaikki $G$:n kaaret ilman suuntaa. *Ketju* suunnatun verkon $G$ solmusta $v_1$ solmuun $v_k$ tarkoittaa ketjua $v_1 \to v_k$ vastaavassa suuntaamattomassa verkossa. Tällainen ketju koostuu siis peräkkäisistä kaarista

$$(v_1, v_2),\ (v_2, v_3),\ \ldots,\ (v_{k-1}, v_k)$$

missä jokaisella $i \in \{1, \ldots, k-1\}$ joko $(v_i, v_{i+1})$ tai $(v_{i+1}, v_i)$ on kaari suunnatussa verkossa $G = (V, E)$, so. joko $(v_i, v_{i+1})$ tai $(v_{i+1}, v_i)$ kuuluu joukkoon $E$. Sanomme, että $(v_i, v_{i+1})$ on *myötävirtainen*, jos $(v_i, v_{i+1}) \in E$, ja *vastavirtainen*, jos $(v_{i+1}, v_i) \in E$.

Sanomme, että verkon $G = (V, E)$ ketju $v_1 \to v_k$ on *vajaavirtainen*, jos ketjun jokaiselle myötävirtaiselle kaarelle $(i, j)$ pätee $f_{ij} < c_{ij}$ ja jokaiselle vastavirtaiselle kaarelle $(i, j)$ pätee $f_{ji} > 0$. Osoittautuu, että vajaavirtaisilla ketjuilla on tärkeä merkitys silloin kun pyritään löytämään maksimaalisia virtauksia verkossa $G$.

**Esimerkki 1.** Tarkastellaan seuraavaa verkkoa.




Kuvan osoittamassa suunnatussa verkossa jokaiseen kaareen liittyy kaksi lukua, jotka ovat ko. kaaren kapasiteetti ja annettu virtaus $f_{ij}$. Jokaista solmuparia $(i, j)$ kohden asetetaan

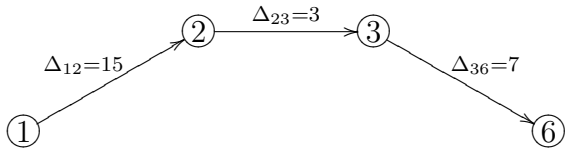
$$\Delta_{ij} = \begin{cases} c_{ij} - f_{ij}, \text{ jos } (i,j) \text{ on myötävirtainen} \\ f_{ji}, \qquad \text{jos } (i,j) \text{ on vastavirtainen} \end{cases}$$

Silloin mikä hyvänsä ketju lähteestä 1 kohteeseen 6 on vajaavirtainen, jos ja vain jos jokaiselle ko. ketjun kaarelle $(i, j)$ pätee $\Delta_{ij} > 0$. Ketjun *vajaavirtaisuus* $\Delta$ on pienin ko. ketjuun liittyvistä luvuista $\Delta_{ij}$.

Esimerkiksi ketju $P : 1 \to 2 \to 3 \to 6$ (ks. alla) on vajaavirtainen, sillä

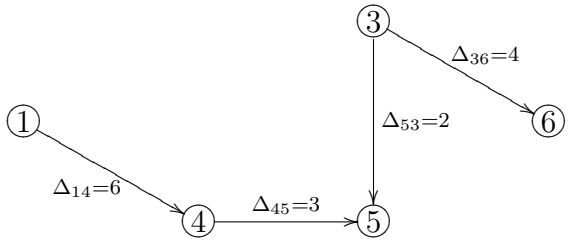
$$\Delta_{12} = 20 - 5 = 15, \ \Delta_{23} = 11 - 8 = 3 \ \text{ ja } \Delta_{36} = 13 - 6 = 7.$$

Kaikki ketjun kaaret ovat myötävirtaisia, ja ketjun vajaavirtaisuus $\Delta = 3$.




Vajaavirtaisuuden ansiosta verkon kokonaisvirtausta $f = 9$ voidaan kasvattaa $\Delta$:n verran asettamalla $f_{12} = 8$, $f_{23} = 11$ ja $f_{36} = 9$ muiden virtausten pysyessä ennallaan. Silloin virtaus kaarella $(2, 3)$ tulee maksimaaliseksi, ja kokonaisvirtaus saa arvon $f = 12$.

Kokonaisvirtausta voidaan kasvattaa vieläkin suuremmaksi etsimällä uusi vajaavirtainen ketju. on ketju $P_2 : 1 \to 4 \to 5 \leftarrow 3 \to 6$, jonka vajaavirtaisuus $\Delta = 2$, sillä kaari $(5, 3)$ on vastavirtainen ja $\Delta_{53} = f_{35} = 2$. Kokonaisvirtausta voidaan jälleen kasvattaa $\Delta$:n verran asettamalla $f_{14} = 6$, $f_{45} = 6$, $f_{35} = 0$, $f_{36} = 11$.



**Leikkaukset**

Sanomme, että pari $(S,T)$ joukon $V$ epätyhjiä osajoukkoja on $G$:n *leikkaus*, jos

$$S \cap T = \emptyset \quad \text{ja} \quad S \cup T = V.$$

Jos $G$:ssä on määritelty virtaus, vaaditaan lisäksi että lähde $s$ ja kohde $t$ kuuluvat vastaavasti joukkoihin $S$ ja $T$.

Jos $(S,T)$ on $G$:n leikkaus, merkitään

$$E_{ST} = \{(i,j) \in E : i \in S \text{ ja } j \in T\}$$

ja

$$E_{TS} = \{(i,j) \in E : i \in T \text{ ja } j \in S\}.$$

*Leikkauksen* $(S,T)$ *kapasiteetti* on

$$\text{cap}(S,T) = \sum_{(i,j)\in E_{ST}} c_{ij}$$

ja *leikkauksen* $(S,T)$ *nettovirtaus* on

$$\sum_{e\in E_{ST}} f_e - \sum_{e\in E_{TS}} f_e.$$

**Esimerkki 2.** Tarkastellaan Esimerkin 1 verkossa leikkausta, jossa $S = \{1,2,4\}$ ja $T = \{3,5,6\}$. Tällöin $E_{ST}$ koostuu kaarista $(2,3)$ ja $(4,5)$, joten

$$\text{cap}(S,T) = c_{23} + c_{45} = 11 + 7 = 18$$

ja nettovirtaus on

$$f_{23} + f_{45} - f_{52}.$$

Virtauksen alkuperäisillä arvoilla leikkauksen $(S,T)$ nettovirtaus on siis $8 + 4 - 3 = 9$, ja korotetuilla arvoilla nettovirtaus on $11+6-3 = 14$. Kummassakin tapauksessa nettovirtaus on kokonaisvirtauksen suuruinen. Näin on laita yleisestikin:

**Lause 2.** *Verkon $G = (V,E)$ jokaisen leikkauksen $(S,T)$ nettovirtaus on yhtäsuuri kuin $G$:n kokonaisvirtaus.*

*Todistus.* Merkitään $E_S$:llä niiden $E$:n alkioiden joukkoa, joiden molemmat päätepisteet kuuluvat joukkoon $S$. Vähentämällä puolittain yhtälöt

$$\sum_{i\in S} \tau(i) = \sum_{e\in E_S} f_e + \sum_{e\in E_{ST}} f_e$$

ja

$$\sum_{i\in S} \sigma(i) = \sum_{e\in E_S} f_e + \sum_{e\in E_{TS}} f_e$$

saadaan

$$\sum_{i \in S} [\tau(i) - \sigma(i)] = \sum_{e \in E_{ST}} f_e - \sum_{e \in E_{TS}} f_e.$$

Tässä oikea puoli on yhtäsuuri kuin leikkauksen $(S, T)$ nettovirtaus, ja vasen puoli on Kirchhoffin lain perusteella

$$\tau(s) - \sigma(s) = \tau(s) = f.$$

□

**Seuraus.** $\tau(s) = \sigma(t)$

*Todistus.* Tarkastellaan leikkausta $\{V \setminus \{t\}, \{t\}\} = \{S, T\}$ ja käytetään tietoa $\tau(s) - \sigma(s) = \sigma(t) - \tau(t) = f$. □

Lauseen 2 avulla saadaan verkon kokonaisvirtaukselle yläraja:

**Lause 3.** *Olkoon $(S, T)$ verkon $G$ leikkaus. Silloin $G$:n kokonaisvirtaus on enintään $\mathrm{cap}(S, T)$.*

*Todistus.* Lauseen 2 nojalla

$$f = \sum_{e \in E_{ST}} f_e - \sum_{e \in E_{TS}} f_e \leq \sum_{e \in E_{ST}} f_e.$$

Tässä $f_e \leq c_{ij}$ aina kun $e = (i, j) \in E_{ST}$. Siis

$$f \leq \sum_{(i,j) \in E_{ST}} c_{ij} = \mathrm{cap}(S, T).$$

□

Vajaavirtaisten ketjujen merkitys maksimaalista kokonaisvirtausta etsittäessä perustuu seuraavaan lauseeseen.

**Lause 4.** *Virtaus verkon $G = (V, E)$ lähteestä $s$ kohteeseen $t$ on maksimaalinen, jos ja vain jos mikään ketju $s \to t$ ei ole vajaavirtainen.*

*Todistus.* (a) Jos jokin ketju $P : s \to t$ on vajaavirtainen, verkon kokonaisvirtausta voidaan kasvattaa kuten Esimerkissä 1. Lauseen ehto on siis välttämätön.

(b) Oletetaan, että mikään ketju $s \to t$ ei ole vajaavirtainen. Olkoon $S$ niiden verkon solmujen $i$ joukko, joille pätee

$(*)$ joko $i = s$ tai on olemassa vajaavirtainen ketju $s \to i$.

Silloin $S \neq \emptyset$ ja myös $T = V \setminus S \neq \emptyset$, koska $t \in T$. Edelleen $S \cap T = \emptyset$ ja $S \cup T = V$. Siis $(S, T)$ on $G$:n leikkaus.

Väitämme, että leikkauksen $(S, T)$ nettovirtaus on yhtäsuuri kuin

$$\mathrm{cap}(S, T) = \sum_{(i,j) \in E_{ST}} c_{ij}.$$

Valitaan jokaista $i \in S \setminus \{s\}$ kohden vajaavirtainen ketju $P_i : s \to i$. Jos $(i,j) \in E_{ST}$, niin yhdistämällä $P_i$ kaareen $(i,j)$ saatu ketju $s \to j$ ei ole vajaavirtainen, sillä $j \notin S$. Tämä on mahdollista vain mikäli $f_{ij} = c_{ij}$, koska $P_i$ on vajaavirtainen. Samalla tavoin nähdään, että $f_{ji} = 0$ jos $(j,i) \in E_{TS}$. Leikkauksen $(S,T)$ nettovirtaus on siis

$$\sum_{e \in E_{ST}} f_e - \sum_{e \in E_{TS}} f_e = \sum_{(i,j) \in E_{ST}} c_{ij} = \mathrm{cap}(S,T)$$

Lauseen 2 nojalla tämä on yhtäsuuri kuin $G$:n kokonaisvirtaus, joka Lauseen 3 perusteella siten on maksimaalinen. □

Lauseessa 4 konstruoidun leikkauksen $(S,T)$ kapasiteetti on minimaalinen.

**Lause 5.** *Verkon $G$ leikkausten kapasiteettien suurin alaraja on yhtäsuuri kuin $G$:n maksimaalinen kokonaisvirtaus.*

*Todistus.* Jos $f$ on $G$:n maksimaalinen kokonaisvirtaus, niin Lauseen 4 todistuksen perusteella on olemassa $G$:n leikkaus $(S,T)$ siten, että $f = \mathrm{cap}(S,T)$. Toisaalta Lauseen 3 nojalla $f \leq \mathrm{cap}(S',T')$ jokaiselle $G$:n leikkaukselle $(S',T')$. Siis $f = \mathrm{cap}(S,T)$ on $G$:n leikkausten kapasiteettien suurin alaraja. □

### Algoritmi maksimivirtauksen löytämiseksi

Seuraavassa *Fordin-Fulkerssonin algoritmissa* lähdetään liikkeelle annetusta virtauksesta (esim. $f_{ij} = 0$ kaikille $i,j$), ja etsitään kullakin iteraatioaskeleella sellainen vajaavirtainen ketju $s \to t$ jonka avulla voidaan kasvattaa kokonaisvirtauksen arvoa kuten Esimerkissä 1. Kun vajaavirtaista ketjua ei enää löydy, algoritmi päättyy ja kokonaisvirtaus on maksimaalinen.

Yksinkertaisuuden vuoksi rajoitutaan verkkoihin, joissa ehdosta $(i,j) \in E$ seuraa, että $(j,i) \notin E$. Algoritmi toimii myös tapauksissa, joilloin $\sigma(s) \neq 0$ tai $\tau(t) \neq 0$. Tällöin kokonaisvirtaus on (vrt. Lauseen 2 todistus)

$$f = \tau(s) - \sigma(s) = \sigma(t) - \tau(t).$$

1. Aluksi käydään läpi joukon $E_\tau(s)$ kaaret $(s,j)$ ja liitetään kuhunkin päätepisteeseen $j$ lukupari $(s, c_{sj} - f_{sj})$ *mikäli* $c_{sj} - f_{sj} > 0$. Jos $c_{sj} - f_{sj} = 0$, solmuun $j$ ei liitetä tällaista lukuparia.

2. Seuraavaksi käydään läpi kaikki joukon $E_\sigma(s)$ kaaret $(j,s)$ ja liitetään kuhunkin alkupisteeseen $j$ lukupari $(s, f_{js})$ mikäli $f_{sj} > 0$. Jos $f_{sj} = 0$, solmuun $j$ ei liitetä tällaista lukuparia.

3. Kun kaikki joukon $E_\tau(s) \cup E_\sigma(s)$ kaaret on käyty läpi kohtien 1 ja 2 mukaisesti, siirrytään tarkastelemaan sitä solmua $i$, johon ensimmäisenä liitettiin lukupari $(s, d_i)$

4. Käydään läpi ne joukon $E_\tau(i)$ kaaret $(i,j)$, joita ei tarkasteltu vaiheissa 1 ja 2 ja joiden päätepisteeseen ei ole liitetty lukuparia. Jos $c_{ij} - f_{ij} > 0$, solmuun $j$ liitetään lukupari $(i, d_j)$, missä
$$d_j = \min(d_i, c_{ij} - f_{ij})$$

5. Käydään läpi ne joukon $E_\sigma(i)$ kaaret $(j, i)$, joita ei ole tarkasteltu aikaisemmissa vaiheissa. Jos $f_{ij} > 0$ eikä solmuun $j$ ole vielä liitetty lukuparia, solmuun $j$ liitetään lukupari $(i, d_j)$, missä
$$d_j = \min(d_i, f_{ji})$$

6. Vaiheita 4 ja 5 kutsutaan solmun $i$ *selaamiseksi*. Tämän jälkeen selataan samalla tavoin verkon muut solmut siinä järjestyksessä missä niihin on liitetty lukupareja. Kutakin solmua selattaessa käydään läpi vain ne ko. solmuun liitettyvät kaaret, joiden toiseen päätepisteeseen ei ole vielä liitetty lukuparia ja jotka eivät kuulu joukkoon $E_\tau(s) \cup E_\sigma(s)$. Mielivaltaiseen solmuun liitetty lukupari $(i, d_j)$ osoittaa tällöin, että kaari $(i, j)$ sisältyy sellaiseen ketjuun $s \to j$, jonka vajaavirtaisuus on $d_j$.

7. Jos verkon alkuperäinen virtaus on maksimaalinen, solmuun $t$ ei vaiheessa 6 tulla liittämään lukuparia (Lause 4). Tällöin algoritmi päättyy kun kaikki mahdolliset solmut on selattu.

8. Jos virtaus ei ole maksimaalinen, solmuun $t$ liitetyn lukuparin avulla voidaan löytää vajaavirtainen ketju $s \to t$, jonka vajaavirtaisuus on $d_t$. Tähän ketjuun kuuluvat kaaret löytyvät helposti päätepisteisiin liittyvien lukuparien ensimmäisten komponentin avulla (näin löydetään oikeastaan ketju $t \to s$).

9. Vaiheessa 8 löydetyn vajaavirtaisen ketjun avulla voidaan verkon virtausta kasvattaa kuten Esimerkissä 1 siten, että kokonaisvirtaus tulee $d_t$:n verran alkuperäistä suuremmaksi. Tämän uuden virtauksen pohjalta käydään uudelleen läpi algoritmin vaiheet 1-8, jolloin algoritmi joko päättyy vaiheeseen 7 tai tuottaa vajaavirtaisen ketjun $s \to t$ vaiheessa 8.

**Esimerkki 3.** Etsitään Fordin-Fulkerssonin algoritmilla maksimaalinen kokonaisvirtaus $s \to t$ Esimerkin 1 verkossa. Alussa annettu kokonaisvirtaus $f = 0$.

1. kierros

   1. $c_{12} - f_{12} = 20 - 5 = 15$; solmuun 2 liitetään lukupari $(5, 15)$

   2. $c_{14} - f_{14} = 10 - 4 = 6$; solmuun 4 liitetään lukupari $(5, 6)$

   3. Siirrytään solmuun 2

   4. $c_{23} - f_{23} = 11 - 8 = 3$; solmuun 3 liitetään $(2, 3)$, sillä $3 = \min(d_2, 3) = \min(15, 3)$

   5. $f_{52} = 3 > 0$; solmuun 5 liitetään lukupari $(2, 3)$, sillä $3 = \min(d_2, 3)$

   6. Solmua 4 selattaessa ei synny uusia lukupareja, sillä solmuun 5 on jo liitetty lukupari ja kaarta $(1, 4) \in E_\tau(s)$ ei tarkastella.

      Solmua 3 selattaessa saadaan
      $c_{36} - f_{36} = 13 - 6 > 0$; solmuun 6 liitetään lukupari $(3, \min(d_3, 7)) = (3, 3)$

   8. Ketjun $P : 1 - 2 - 3 - 6$ vajaavirtaisuus on $d_6 = 3$

9. Virtauksiin $f_{12}$, $f_{23}$ ja $f_{36}$ lisätään $d_6 = 3$, jonka jälkeen $f_{12} = 8$, $f_{23} = 11$ ja $f_{36} = 9$ ja uusi kokonaisvirtaus $f = 12$.

2. kierros

1. $c_{12} - f_{12} = 20 - 8 = 12$; solmuun 2 liitetään lukupari $(5, 12)$
2. $c_{14} - f_{14} = 10 - 4 = 6$; solmuun 4 liitetään lukupari $(5, 6)$
3. Siirrytään solmuun 2
4. $c_{23} - f_{23} = 11 - 11 = 0$
5. $f_{52} = 3 > 0$; solmuun 5 liitetään lukupari $(2, 3)$, sillä $3 = \min(d_2, 3)$
6. Kuten edellä Solmua 4 selattaessa ei synny uusia lukupareja.

   Solmua 5 selattaessa saadaan
   $c_{56} - f_{56} = 3 - 3 = 0$
   $f_{35} = 2 > 0$; solmuun 3 liitetään lukupari $(5, 2)$, sillä $2 = \min(d_5, 2)$

   Solmua 3 selatessa saadaan
   $c_{36} - f_{36} = 13 - 9 > 0$; solmuun 6 liitetään lukupari $(3, \min(d_3, 4)) = (3, 2)$
8. Ketjun $P : 1 \to 2 \leftarrow 5 \leftarrow 3 \to 6$ vajaavirtaisuus on $d_6 = 2$
9. Virtauksiin $f_{12}$ ja $f_{36}$ lisätään 2 ja virtauksista $f_{52}$, $f_{35}$ vähennetään 2. Tämän jälkeen $f_{12} = 10$, $f_{36} = 11$, $f_{52} = 1$, $f_{35} = 0$ ja uusi kokonaisvirtaus $f = 14$.

3. kierros

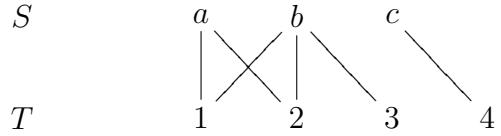
1. $c_{52} - f_{52} = 10 > 0$; solmuun 2 liitetään lukupari $(5, 10)$
2. $c_{54} - f_{54} = 10 - 4 = 6$; solmuun 4 liitetään lukupari $(5, 6)$
3. Siirrytään solmuun 2
4. $c_{23} - f_{23} = 11 - 11 = 0$
5. $f_{52} = 1 > 0$; solmuun 5 liitetään lukupari $(2, 1)$, sillä $1 = \min(d_2, 1)$
6. Kuten edellä Solmua 4 selattaessa ei synny uusia lukupareja.

   Solmua 5 selattaessa saadaan
   $c_{56} - f_{56} = 0$ ja $f_{35} = 0$; ei synny uusia lukupareja.
7. Koska kaikki mahdolliset solmut on selattu ja solmuun $t$ ei ole liitetty lukuparia, algoritmi päättyy ja kokonaisvirtaus on maksimaalinen.

# 9 Pareittain yhdistely

**Määritelmä.** Verkko $G = (V, E)$ on *kaksijakoinen*, jos on olemassa $G$:n leikkaus $(S, T)$ siten, että $E_S = E_T = \emptyset$, so. jokaisen $G$:n kaaren toinen päätepiste kuuluu joukkoon $S$ ja toinen joukkoon $T$. Minkään kaaren molemmat päätepisteet eivät siis samanaikaisesti kuulu $S$:ään eivätkä myöskään $T$:hen. Tällaista verkkoa merkitään myös $G = (S, T; E)$.

**Esimerkki.** Työnvälitys: $S$ on työttömien työnhakijoiden ja $T$ avointen työpaikkojen joukko. Tehtävänä on jakaa työpaikat siten, että kaikille hakijoille riittää töitä.




**Määritelmä.** *Parite* verkossa $G = (S, T; E)$ on joukko $M \subset E$ siten, että millään kahdella $M$:n alkiolla ei ole yhteistä päätepistettä. Parite $M$ on *maksimaalinen*, jos siinä on mahdollisimman monta alkiota.

Äskeisessä työnvälitysesimerkissä paritteita ovat esim. $M_1 = \{(a, 2), (b, 1)\}$ ja $M_2 = \{(a, 1), (b, 3), (c, 4)\}$. Näistä on jälkimmäinen on selvästi maksimaalinen.

**Määritelmä.** Solmu $v$ on *erillään* paritteesta $M$, jos $v$ ei ole minkään $M$:n kaaren päätepiste. Parite $M$ on täydellinen, jos mikään $G$:n solmu ei ole siitä erillään.

Selvästi täydellinen parite määrittelee bijektion $S \to T$. Näin ollen verkossa $G = (S, T; E)$ voi olla täydellinen parite vain jos joukoissa $S$ ja $T$ on yhtä monta alkiota.

Verkon $G$ ketju $P$ on *vuorotteleva paritteen $M$ suhteen*, jos $P$:n joka toinen kaari kuuluu joukkoon $M$ ja joka toinen kaari joukkoon $E \setminus M$. Paritteen $M$ suhteen vuorotteleva ketju $P$ on *parittava*, jos sen alku- ja loppupisteet ovat erillään paritteesta $M$.

**Esimerkki.** Tarkastellaan seuraavia vuorottelevia ketjuja, joissa kahdella viivalla merkityt kaaret kuuluvat paritteeseen $M$. Ketjuista vasemmanpuoleinen on parittava.

Seuraavassa osoitamme, että parittavaa ketjua käyttämällä voidaan annetusta paritteesta $M$ lähtien lisätä $M$:n alkioiden lukumäärää siten, että lopputuloksena syntyy maksimaalinen parite.

**Lause 6.** *Parite $M$ on maksimaalinen, jos ja vain jos $G$ ei sisällä yhtään $M$:n suhteen parittavaa ketjua.*

*Todistus.* 1°) Olkoon $P : a \to b$ jokin $M$:n suhteen parittava ketju, jossa on $q$ kappaletta $M$:n alkioita. Silloin $P$ sisältää $q+1$ joukon $E \setminus M$ alkiota. Olkoon $M'$ niiden $M$:n alkioiden joukko, joiden kumpikaan päätepiste ei kuulu ketjuun $P$. Liittämällä paritteeseen $M'$ kaikki

$P$:hen kuuluvat joukon $E \setminus M$ alkiot saadaan silloin uusi parite, jossa on yksi kaari enemmän kuin paritteessa $M$. Siis $M$ ei ole maksimaalinen.

2°) Oletetaan kääntäen, että $M$ ei ole maksimaalinen. Olkoon $M^*$ mielivaltainen maksimaalinen parite, ja olkoon

$$H = (M^* \cup M) \setminus (M^* \cap M).$$

Silloin joukkoon $H$ kuuluvat kaaret päätepisteineen muodostavat verkon $G_H$, jonka jokainen ketju on vuorotteleva paritteen $M$ suhteen. Tämä seuraa siitä, että mikään parite ei voi sisältää ketjua jossa on vähintään 2 kaarta; paritteen määräämässä aliverkossa jokaisen solmun aste on 1. Edelleen jokaisen $G_H$:n solmun aste $n \leq 2$, koska $M$ ja $M^*$ ovat paritteita. Koska myös jokaisen $G_H$:n sykli on vuorotteleva paritteen $M$ suhteen, näissä sykleissä on yhtä paljon $M$:n ja $M^*$:n alkioita.

Poistamalla $G_H$:sta nämä syklit saadaan uusi verkko $G'_H$, joka voidaan esittää äärellisenä yhdisteenä paritteen $M$ suhteen vuorottelevista ketjuista, joiden päätepisteiden asteet verkossa $G'_H$ ovat $= 1$. Koska $M^*$ on maksimaalinen, ainakin yhdessä näistä ketjuista on enemmän $M^*$:n kuin $M$:n alkioita. Tällainen ketju on parittava $M$:n suhteen. □

Seuraavan ns. *vappuheila-algoritmin* avulla löydetään annetusta paritteesta $M$ lähtien maksimaalinen parite konstruoimalla jono parittavia ketjuja. Algoritmin jokaisessa vaiheessa varustetaan verkon solmuja leimoilla seuraavasti.

1. Jos joukossa $S$ ei ole yhtään paritteesta $M$ erillään olevaa solmua, $M$ on selvästi maksimaalinen.

2. Leimataan kaikki joukon $S$ paritteesta $M$ erillään olevat solmut leimalla $\emptyset$.

3. Jos solmu $i \in S$ on leimattu ja $(i, j) \in E$, varustetaan solmu $j \in T$ leimalla $i$ ellei sitä aiemmin ole varustettu jollakin muulla leimalla.

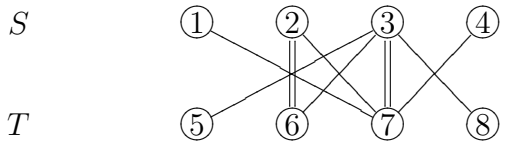
4. Käydään läpi vaiheessa 3 leimatut solmut $j \in T$. Jokaisen solmun kohdalla on kaksi mahdollisuutta:

   4.1 On olemassa solmu $i \in S$ siten, että $(i, j) \in M$. Tällöin $i$ varustetaan luvulla $j$.

   4.2 Solmu $j$ on erillään paritteesta $M$. Tällöin on olemassa parittava ketju $\emptyset \to j$, johon kuuluvat kaaret löydetään solmuihin liitettyjen leimojen avulla.

5. Jos kohdassa 4 ei löydy parittavaa ketjua, algoritmi jatkuu kohdasta 3.

**Esimerkki.**



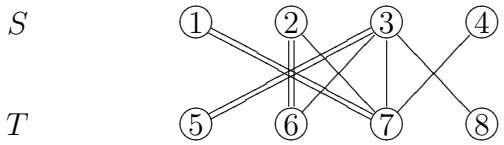

2. Leimataan solmut 1 ja 4 leimalla $\emptyset$

3. Varustetaan 7 leimalla 1.

4. Varustetaan 3 leimalla 7.

3. Varustetaan 5 ja 8 leimalla 3.

4. $P_1 : 1-7-3-5$ ja $P_2 : 1-7-3-8$ ovat parittavia. Lisätään $P_1$:n avulla paritteen alkioiden lukumäärää; saadaan uusi parite $M_2$, joka koostuu kaarista $(1,7)$, $(3,5)$, $(2,6)$.




2. Leimataan solmu 4 leimalla $\emptyset$.

3. Varustetaan solmu 7 leimalla 4.

4. Varustetaan 1 leimalla 7.

3. Ei uusia leimoja

Koska algoritmi päättyi eikä parittavaa ketjua löytynyt, $M_2$ on maksimaalinen.

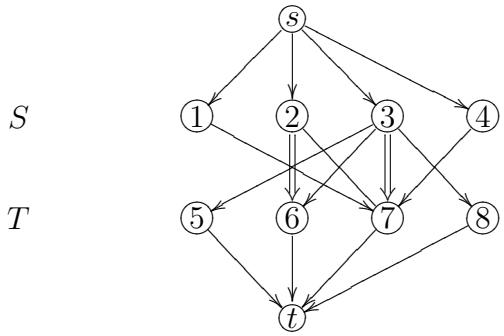
Maksimaalista paritetta voitaisiin etsiä myös Fordin-Fulkerssonin algoritmilla. Liitetään verkkoon $G = (S, T; E)$ suunnattu verkko $\tilde{G} = (\tilde{V}, \tilde{E})$ siten, että

$$\tilde{V} = V \cup \{s\} \cup \{t\}$$

ja $\tilde{E}$ koostuu kaarista $(\sigma, \tau) \in E$, missä $\sigma \in S$ ja $\tau \in T$ sekä lisäksi kaikista kaarista $(s, \sigma)$ ja $(\tau, t)$, missä $\sigma \in S$ ja $\tau \in T$.

Oletetaan, että $\tilde{G}$:n jokaisen kaaren kaaren kapasiteetti on 1. Silloin jokainen $G$:n parite $M$ määrittelee virtauksen $\tilde{G}$ siten, että virtaus muotoa $(\sigma, \tau)$ olevalla kaarella on 1, jos ja vain jos $(\sigma, \tau) \in M$; muulloin virtaus tällaisella kaarella on 0. Kokonaisvirtaus $s \to t$ on sama kuin $M$:n alkioiden lukumäärä.

Jokainen $\tilde{G}$:n vajaavirtainen ketju $s \to t$ määrittelee parittavan ketjun $G$:ssä (kun siitä poistetaan ensimmäinen ja viimeinen kaari), ja kääntäen kaikki $G$:n parittavat ketjut saadaan tällä tavalla.



# 10 Duaalisuus

Olkoon $A$ tyyppiä $m \times n$ oleva matriisi, $b$ $m$-vektori ja $c$ $n$-vektori. Normaalimuotoinen lineaarinen optimointitehtävä on:

$$\text{minimoi } c^T x \text{ ehdoilla } Ax = b \text{ ja } x \geq 0. \tag{1}$$

Tässä $c^T x = c_1 x_1 + \cdots + c_n x_n$ on vektorien sisätulo, ja merkintä $x \geq 0$ tarkoittaa, että $x_i \geq 0$ kaikilla $i \in \{1, \ldots, n\}$.

Tehtävään (1) liitetään vastaava *duaalinen* optimointitehtävä

$$\text{maksimoi } y^T b \text{ ehdolla } y^T A \leq c^T. \tag{2}$$

Duaalisen tehtävän ratkaisu on $m$-vektori, jonka komponenttien ei tarvitse olla ei-negatiivisia.

**Esimerkki.** Olkoot $A = \begin{pmatrix} 1 & 1 \end{pmatrix}$, $b = 1$ ja $c = \begin{pmatrix} 5 & 4 \end{pmatrix}^T$. Silloin "primaari" tehtävä (1) ja duaali tehtävä (2) ovat

Primaari: minimoi $5x_1 + 4x_2$ ehdoilla $x_1 + x_2 = 1$ ja $x \geq 0$.
Duaali: maksimoi $y_1$ ehdoilla $y_1 \leq 5$, $y_1 \leq 4$.

Primaaritehtävässä minimi 4 saavutetaan, kun $x_1 = 0$ ja $x_2 = 1$. Duaalitehtävän maksimi on samoin 4 ja saavutetaan kun $y_1 = 4$. Tässä esimerkissä duaalitehtävä on hieman helpompi, koska siinä on vain yksi tuntematon.

Yleisesti voidaan osoittaa, että tehtävillä (1) ja (2) on samat ratkaisut, mutta todistus ei ole triviaali. Todistuksen helpompi osa on ns. *heikko duaalisuuslause*:

**Heikko duaalisuuslause.** *Jos $x$ ja $y$ ovat tehtävien (1) ja (2) mielivaltaisia rajoiteratkaisuja, niin $y^T b \leq c^T x$. Jos $y^T b = c^T x$, niin rajoiteratkaisut $x$ ja $y$ ovat optimaalisia.*

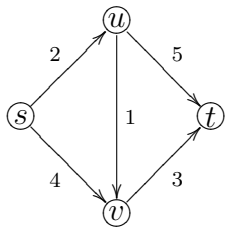
*Todistus.*

$$\left.\begin{array}{rcl} Ax = b & \Rightarrow & y^T Ax = y^T b \\ y^T A \leq c^T & \Rightarrow & y^T Ax \leq c^T x \end{array}\right\} \Rightarrow \; y^T b \leq c^T x$$

Jos $y^T b = c^T x$, rajoiteratkaisu $x$ on optimaalinen, koska mikä hyvänsä toinen rajoiteratkaisu $\tilde{x}$ toteuttaa juuri todistetun yhtälön $y^T b \leq c^T x$, jolloin myös $c^T x \leq c^T \tilde{x}$. Samoin tavoin $y$ on optimaalinen. □

Heikosta duaalisuuslauseesta seuraa myös, että jos tehtävän (1) minimi on $-\infty$, tehtävän (2) rajoiteratkaisujen joukko on tyhjä. Muutenhan lukujen $c^T x$ joukko olisi alhaalta rajoitettu. Vastaavasti tehtävällä (1) ei ole rajoiteratkaisuja, jos tehtävän (2) maksimi on $+\infty$. Jos molemmilla tehtävillä löytyy rajoiteratkaisuja, tehtävän (1) minimi on sama kuin tehtävän (2) maksimi.

**Esimerkki 1.** Etsittävä lyhyin ketju oheisen verkon lähteestä $s$ kohteeseen $t$.




Jos verkon solmut ovat rautatieasemia, kaaren pituus ilmoittaa esim. kustannuksen annetun tavaraerän kuljettamiseksi solmusta toiseen. Kysymys kuuluu tällöin: Miten tavara siirtyy halvimmillaan $s$:stä $t$:hen?

Vastaus voidaan arvata: koko tavaraerä kulkee pitkin ketjua $s \to u \to v \to t$, merkitään

$$x_{su} = x_{uv} = x_{vt} = 1 \quad \text{ja} \quad x_{sv} = s_{ut} = 0.$$

Kustannus on tällöin $2 + 1 + 3 = 6$.

Ongelma voidaan esittää lineaarisena optimointitehtävänä: Minimoi kustannus

$$c^T x = 2x_{su} + 4x_{sv} + x_{uv} + 5x_{ut} + 3x_{vt}$$

ehdoilla

$$\begin{cases} x \geq 0 \\ x_{su} - x_{uv} - x_{ut} = 0 \\ x_{sv} + x_{uv} - x_{vt} = 0 \\ x_{ut} + x_{vt} = 1 \end{cases}$$

Laskemalla rajoiteyhtälöt yhteen saadaan $x_{su} + x_{sv} = 1$ (koko tavara lähtee $s$:stä). Matriisimuodossa yhtälöt voidaan kirjoittaa

$$Ax = \begin{pmatrix} 1 & 0 & -1 & -1 & 0 \\ 0 & 1 & 1 & 0 & -1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_{su} \\ x_{sv} \\ x_{uv} \\ x_{ut} \\ x_{vt} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = b$$

Duaalisessa probleemassa on maksimoitava lauseketta $y^T b = y_t$ ehdolla $y^T A \leq c^T$. Vektorin $y$ komponentit $y_u$, $y_v$ ja $y_t$ vastaavat solmuja $u$, $v$ ja $t$, ja rajoitteet ovat

$$\begin{cases} y_u \leq 2 \\ y_v \leq 4 \\ y_v - y_u \leq 1 \\ y_t - y_u \leq 5 \\ y_t - y_v \leq 3 \end{cases} \qquad (*)$$

Duaalisen probleeman tulkinta: $y_u$, $y_v$ ja $y_t$ ovat hintoja, jolla kilpaileva kuljetusyritys suostuu kuljettamaan tavaran solmusta $s$ vastaavasti solmuihin $u$, $v$ ja $t$. Kuljetushinta solmusta $u$ solmuun $v$ on $y_v - y_u$ ja solmusta $u$ solmuun $t$ on $y_t - y_u$. Rajoitteet $(*)$ aiheutuvat siitä, että kilpailukyvyn säilyttämiseksi hinnat eivät saa ylittää rautatieyhtiön kustannusta, joka määräytyy kaarien pituuksista.

Tehtävänä on maksimoida tuotto $y_t$, joka saadaan tavaran kuljettamisesta solmusta $s$ solmuun $t$ ($y_s = 0$). Jokaiselle duaalisen probleeman rajoiteratkaisulle pätee

$$y_t = (y_t - y_v) + (y_v - y_u) + y_u \leq 3 + 1 + 2 = 6.$$

Näin ollen rajoiteratkaisu $(y_u, y_v, y_t) = (2, 3, 6)$ on optimaalinen ja duaaliprobleeman maksimi $y_t = 6$ on sama kuin alkuperäisen probleeman minimi.