

Vaihto-ominaisuudella on seuraava intuition kannalta keskeinen seuraus: Olkoot  $A \in I$  ja  $B \in I$  samankokoisia riippumattomia joukkoja:  $|A| = |B| = m$  jollain  $m > 0$ . Olkoon vielä  $n = m - |A \cap B|$ , jolloin  $|A - B| = |B - A| = n$ .

Joukko  $A$  voidaan nyt muuntaa joukoksi  $B$  (tai toisinpäin) suorittamalla  $n$  vaihtoa, joissa aina yksi joukon  $A$  alkio vaihdetaan joukon  $B$  alkioksi. Vaihto-ominaisuuden nojalla vaihdot on mahdollista valita niin, että myös kaikki välivaiheet ovat riippumattomia:

Valitaan alkio  $x_1 \in A - B$  ja merkitään  $A'_1 = A - \{x_1\}$ . Perinnöllisyyden nojalla  $A'_1 \in I$ .

Koska  $|A'_1| = |A| - 1 < |B|$ , vaihto-ominaisuuden perusteella jollain  $y_1 \in B - A'_1$  pätee  $A'_1 \cup \{y_1\} \in I$ . Olkoon  $A_1 = A'_1 \cup \{y_1\}$ .

Yleisemmin vaiheessa  $i$  muodostetaan joukko  $A_i$  asettamalla  $A_i = A_{i-1} - \{x_i\} \cup \{y_i\}$  missä  $x_i \in A - B$  ja  $y_i \in B - A$ .

Koska kaikki poistettavat alkiot  $x_i$  ovat joukosta  $A - B$  ja lisättävät alkiot  $y_i$  joukosta  $B - A$ , kerran lisättyä alkioita ei enää poisteta ja kääntäen.

Siis  $A_i$  sisältää  $i$  kappaletta joukon  $B - A$  alkioita,  $n - i$  kappaletta joukon  $A - B$  alkioita, ja kaikki  $m - n$  joukon  $A \cap B$  alkioita. Eryteisesti  $A_n = B$ .

Jos  $A \in I$  ja  $A \cup \{x\} \in I$ , alkio  $x$  on joukon  $A$  jatke.  
Joukko  $A$  on maksimaalinen jos  $A \in I$  mutta joukolla  $A$  ei ole jatkeita.

Vaihto-ominaisuudesta seuraa suoraan

**Propositio** Kaikissa maksimaalisissa joukoissa on sama määrä alkioita.

**Todistus** Jos  $A, B \in I$  ja esim.  $|A| < |B|$ , niin vaihto-ominaisuus antaa joukolle  $A$  jatkeen, joten  $A$  ei ole maksimaalinen.  $\square$

**Esimerkki** Olkoon  $G = (V, E)$  yhtenäinen ja  $M_G = (E, I)$ . Matroidin  $M_G$  maksimaalisia alkioita ovat ne  $T$ , joilla  $(V, T)$  on virittävä puu. Tällöin siis  $|T| = |V| - 1$ .

Olkoot  $(V, T)$  ja  $(V, U)$  kaksi virittävää puuta. Joukko  $T$  voidaan siis muuntaa joukoksi  $U$  suorittamalla jono seuraavanlaisia vaihtoja:

1. Poista jokin joukon  $T - U$  kaari; syntyy kaksi erillistä puuta.
2. Lisää jokin joukon  $U - T$ , joka taas kytkee nämä kaksi puuta yhdeksi.

Matroidi  $M = (S, I)$  on **painotettu** jos siihen liittyy positiivisarvoinen painofunktio  $w: S \rightarrow R^+$ . Painofunktio ulotetaan ilmeisellä tavalla osajoukoille  $A \subseteq S$ :

$$w(A) = \sum_{x \in A} w(x).$$

**Optimaalisia** ovat ne riippumattomat joukot, joiden paino on suurin mahdollinen. Painofunktion positiivisuuden takia optimaaliset joukot ovat aina maksimaalisia.

**Esimerkki** Olkoon  $G = (V, E, \ell)$  yhtenäinen painotettu suuntaamaton verkko. Merkitään  $\ell_0 = \max_e \ell(e) + 1$ , ja asetetaan

$$w(e) = \ell_0 - \ell(e).$$

Siis erityisesti  $w(e) \geq 1$  kaikilla  $e$ .

Otetaan taas  $M_G = (E, I)$ . Mille tahansa  $A \in I$  saadaan

$$\begin{aligned} w(A) &= \sum_{e \in A} (\ell_0 - \ell(e)) \\ &= |A|\ell_0 - \sum_{e \in A} \ell(e). \end{aligned}$$

Maksimaalisia joukkoja ovat siis ne  $A \in I$  joilla  $|A| = |V| - 1$  eli joilla  $(V, A)$  on virittävä puu. Näistä painoltaan suurimpia ovat ne joilla kustannus  $\sum_{e \in A} \ell(e)$  on pienin, eli pienimmät virittävät puut.

Pienimmän virittävän puun etsimiseen on edellä esitetty ahne Kruskalin algoritmi. Tästä saadaan suorana yleistyksenä ahne algoritmi matroidin optimaalisen alkion etsimiseen:

*Greedy*( $S, I, w$ ):

$A := \emptyset$

järjestä  $S$  painon mukaan alenevasti:

$S = \{x_1, \dots, x_{|S|}\}$  missä  $w(x_i) \geq w(x_{i+1})$

**for**  $i := 1$  **to**  $|S|$  **do**

**if**  $A \cup \{x_i\} \in I$  **then**

$A := A \cup \{x_i\}$

**end if**

**end for**

**return**  $A$

Algoritmin aikavaativuus on ilmeisesti  $O(|S| \log |S| + |S|f(|S|))$ , missä  $f(|S|)$  on ehdon  $A \cup \{x_i\} \in I$  testaamiseen kuluva aika.

Todistamme, että algoritmin palauttama  $A$  todella on matroidin  $(S, I)$  optimaalinen joukko.

Olkoon  $A_i$  muuttujan  $A$  arvo kun **for**-silmukkaa on iteroitu  $i$  kertaa. Siis  $A_0 = \emptyset$  ja  $A_{|S|}$  on algoritmin palauttama joukko. Suoraan nähdään, että  $A_i \in I$  kaikilla  $i$ , ja  $A_i \subseteq A_j$  kun  $i < j$ .

**Lemma 1** Jos  $A_i \cup \{z\} \notin I$ , niin  $A_j \cup \{z\} \notin I$  kaikilla  $j > i$ .

**Todistus** Selvästi  $A_i \cup \{z\} \subseteq A_j \cup \{z\}$ . Siis jos olisi  $A_j \cup \{z\} \notin I$ , perinnöllisyydestä seuraisi  $A_i \cup \{z\} \notin I$ .  $\square$

Lemman 1 perusteella algoritmin ei enää tarvitse palata uudestaan tarkastamaan sellaisia  $x$ , jotka on kertaalleen sivuutettu.

Seuraavan lemmän avulla osoitetaan se perusinvariantti, että joukko  $A_i$  on aina laajennettavissa optimaaliseksi. Lemmaa on siis tarkoitus soveltaa tapauksessa  $P = A_i$  ja  $z = x_i$ .

**Lemma 2** Olkoon  $P \in I$  sellainen, että  $P \subseteq B$  jollain optimaalisella  $B$ . Olkoon  $z \in S - P$  painoltaan suurin, jolla  $P \cup \{z\} \in I$ . Nyt on olemassa optimaalinen  $B'$  jolla  $P \cup \{z\} \subseteq B'$ .

**Todistus** Jos  $z \in B$ , valitaan  $B' = B$ . Olkoon  $z \notin B$ .

Muodostetaan nyt jono joukkoja  $P_1, \dots, P_k$  missä  $P_1 = P \cup \{z\}$ ,  $|P_i| = |P| + i$  ja  $P_i \in I$  kaikilla  $i$ . Jonon pituus on  $k = |B| - |P|$ .

Kun on annettu  $P_i \in I$  jolla  $|P_i| = |P| + i < |B|$ , vaihto-ominaisuuden nojalla on olemassa  $y_i \in B - P_i$  jolla  $P_i \cup \{y_i\} \in I$ . Valitaan  $P_{i+1} = P \cup \{y_i\}$ .

Lopputulos on siis

$$P_k = P \cup \{z\} \cup \{y_2, \dots, y_k\}$$

missä  $y_i \in B - P$ . Siis jollain  $y_1 \in B - P$  pätee

$$B = P \cup \{y_1, y_2, \dots, y_k\}$$

eli

$$P_k = B - \{y_1\} \cup \{z\}.$$

Erityisesti  $P \cup \{y_1\} \subseteq B \in I$ , joten perinnöllisyyden nojalla  $P \cup \{y_1\} \in I$ . Alkion  $z$  valinnasta seuraa  $w(y_1) \leq w(z)$ , joten

$$w(P_k) = w(B) - w(y_1) + w(z) \geq w(B).$$

Koska  $B$  on optimaalinen, myös  $P_k$  on. Valitaan siis  $B' = P_k$ .  $\square$

**Lause** Kutsu *Greedy* palauttaa jonkin optimaalisen osajoukon  $A$ .

**Todistus** Tarkastellaan tilannetta, jossa  $A_{i+1} = A_i \cup \{x_i\}$ . Kaikki alkio  $x_j$  joilla  $w(x_j) > w(x_i)$  on jo käyty läpi ja

1. joko hylätty:  $A_j \cup \{x_j\} \notin I$
2. tai otettu mukaan:  $x_j \in A_{j+1}$ .

Jos  $x_j$  hylättiin, Lemman 1 nojalla edelleen pätee  $A_i \cup \{x_j\} \notin I$ .

Jos  $x_j$  otettiin mukaan, edelleen  $x_j \in A_i$ .

Siis  $x_i$  on painoltaan suurin  $z \in S - A_i$ , jolla  $A_i \cup \{z\} \in I$ .

Lemmasta 2 seuraa nyt induktiolla, että kaikilla  $i$  joukko  $A_i$  on jonkin optimaalisen joukon  $B_i$  osajoukko.

Lemman 1 nojalla kun suoritus päättyy,  $A_{|S|}$  on maksimaalinen, joten  $|A_k| = |B_k|$ . Siis palautettava arvo on  $A_k = B_k$  joka on optimaalinen.  $\square$

Tarkastellaan sovelluksena yksinkertaista aikataulutusongelmaa.

Ajatellaan, että yhdessä työpisteessä suoritetaan tietynlaisia työtehtäviä. Yhden tehtävän suorittaminen vie aina yhden aikayksikön.

Kaikkiaan suoritettavana on  $n$  tehtävää. Tehtävään  $i$  liittyy **määräika**  $d_i$  ja **myöhästymissakko**  $c_i \in \mathbf{R}^+$ .

**Aikataulu**  $\pi$  liittää jokaiseen tehtävään  $i$  suoritushetken  $\pi(i)$ . Jos tehtävä  $i$  myöhästyy eli  $\pi(i) > d_i$ , joudutaan maksamaan sakko  $c_i > 0$ . Aikataulun  $\pi$  kustannus on siis

$$C(\pi) = \sum_{\pi(i) > d_i} c_i.$$

Tehtävänä on määrittää kustannukseltaan pienin aikataulu.

Koska vain yksi tehtävä kerrallaan voidaan suorittaa, ja toisaalta koskaan ei ole syytä olla joutilaana, voidaan olettaa että joka ajanhetkellä valmistuu tasan yksi tehtävä, kunnes kaikki on suoritettu. Toisin sanoen rajoitutaan aikatauluihin jotka ovat bijektioita  $\{1, \dots, n\} \rightarrow \{1, \dots, n\}$ ; tässä myös tehtävät on nimetty  $\{1, \dots, n\}$ .

Samoin voidaan olettaa, että kaikki määräajat  $d_i$  ovat joukossa  $\{1, \dots, n\}$ .

Aikataulu  $\pi$  on **normaalimuodossa** jos se täyttää seuraavat kaksi ehtoa:

1. Ajoissa olevat tehtävät ovat ennen myöhästyviä; ts. kaikilla  $i, j$  pätee

$$\text{jos } \pi(i) \leq d_i \text{ ja } \pi(j) > d_j, \text{ niin } \pi(i) < \pi(j)$$

2. Ajoissa olevat tehtävät suoritetaan määräajan mukaan kasvavassa järjestyksessä: kaikilla  $i, j$  pätee

$$\text{jos } \pi(i) \leq d_i, \pi(j) \leq d_j \text{ ja } d_i < d_j, \text{ niin } \pi(i) < \pi(j)$$

Olkoon  $(i, j)$  pari, joka **rikkoo** jompaa kumpaa normalisointiehtoa. Olkoon  $\pi'$  saatu aikataulusta  $\pi$  vaihtamalla tehtävien  $i$  ja  $j$  ajat keskenään. Helposti nähdään, että  $C(\pi') = C(\pi)$ :

1. Jos  $i$  oli ajoissa ja  $j$  myöhässä, vaihdon tuloksena  $i$  tulee vielä aikaisemmaksi ja  $j$  vielä myöhäisemmäksi, joten maksuissa ei tule muutoksia.
2. Jos  $i$  oli kiireellisempi mutta myöhemmäksi sijoitettu, se siirtyy aiemmaksi ja on siis edelleen ajoissa. Vaikka  $j$  siirtyy myöhemmäksi, se ei kuitenkaan myöhästy, koska se saa  $i$ :n vanhan paikan, joka riitti  $i$ :lle ja riittää siis vähemmän kiireiselle  $j$ :llekin.

Tekemällä tarvittava määrä tällaisia vaihtoja saadaan

**Propositio 1** Mille tahansa aikataululle  $\pi$  on olemassa aikataulu  $\pi'$  joka on normaalimuodossa ja jolle  $C(\pi') = C(\pi)$ .  $\square$

Haluamme palauttaa ongelman matroideja koskevaksi, joten haluamme bijektioiden  $\pi$  sijaan puhua joistain sopivista joukoista. Määritellään siis aikataululle  $\pi$  sen ajoissa valmistuvien tehtävien joukko

$$A(\pi) = \{i \in \{1, \dots, n\} \mid \pi(i) \leq d_i\}.$$

Jos aikataulua  $\pi$  ei muuten tunneta, mutta tiedetään kuitenkin  $A(\pi)$ , voidaan helposti muodostaa sellainen normaalimuodossa oleva  $\pi'$ , että  $A(\pi') = A(\pi)$  ja erityisesti siis  $C(\pi') = C(\pi)$ .

Olkoon  $S = \{1, \dots, n\}$ , ja sanotaan että  $B \subseteq S$  on riippumaton jos  $B \subseteq A(\pi)$  jollain  $\pi$ . Siis tehtäväjoukko on riippumaton, jos jollain aikataululla ainakin kaikki tämän joukon tehtävät valmistuvat ajoissa.

Osoitamme kohta, että pari  $(S, I)$ , missä  $I$  on riippumattomien tehtäväjoukkojen joukko, todella on matroidi.

Matroidin  $(S, I)$  painofunktioksi määritellään  $w(i) = c_i$  kaikilla  $i$ . (Huom. oletus  $c_i > 0$ .) Siis

$$w(A(\pi)) = \sum_{i \in A(\pi)} c_i = w_0 - C(\pi)$$

missä  $w_0 = \sum_i c_i$ .

Halvin aikataulu saadaan nyt seuraavasti:

1. Etsi matroidin  $(S, I)$  optimaalinen alkio  $A$  *Greedy*-algoritmilla.
2. Nyt halvimman aikataulun  $\pi$  kustannus on  $C(\pi) = w_0 - W(A)$ .
3. Muodostetaan normaalimuotoinen  $\pi'$ , jolla  $A(\pi') = A$  ja siis  $C(\pi') = C(\pi)$ .

**Propositio 2** Olkoon  $A \subseteq S$ . Seuraavat ehdot ovat yhtäpitävät:

1. Joukko  $A$  on riippumaton.
2. Kaikilla  $1 \leq t \leq n$  pätee  $N_t(A) \leq t$  missä

$$N_t(A) = |\{i \in A \mid d_i \leq t\}|.$$

3. Jos joukon  $A$  tehtävät sijoitetaan ajanhetkiin  $1, \dots, |A|$  määräajan mukaan nousevassa järjestyksessä, ne ovat kaikki ajoissa.

**Todistus** Helposti nähdään  $(1) \Rightarrow (2) \Rightarrow (3) \Rightarrow (1)$ .  $\square$

Kohtaa (2) käyttäen voidaan ajassa  $O(|A|)$  testata, onko  $A$  riippumaton. *Greedy*-algoritmin suoritusajaksi tulee siis  $O(n^2)$ .

Pitää vielä osoittaa, että  $(S, I)$  on matroidi, jolloin tiedämme että *Greedy* tuottaa oikean tuloksen.

**Lause** Edellä määritelty  $M = (S, I)$  on matroidi.

**Todistus** Perinnöllisyys on ilmeinen.

Olkoon  $A, B \in I$  ja  $|A| < |B|$ .

Olkoon  $k$  suurin jolla  $N_k(B) \leq N_k(A)$ . Koska  $N_n(B) = |B| > |A| = N_n(A)$ , on  $k \leq n - 1$  ja siten  $N_t(B) > N_t(A)$  epätyhjällä välillä  $k + 1 \leq t \leq n$ .

Erityisesti  $B$  sisältää ainakin yhden tehtävän  $i$ , jolle  $d_i = k + 1$  ja  $i \notin A$ . Väitetään, että  $A \cup \{i\}$  on riippumaton.

Kun  $t \leq k$ , saadaan proposition 2(2) nojalla

$$N_t(A \cup \{i\}) = N_t(A) \leq t.$$

Kun  $t > k$ , saadaan

$$N_t(A \cup \{i\}) \leq N_t(B) \leq t.$$

Siis  $N_t(A \cup \{i\}) \leq t$  kaikille  $t$ , joten  $A \cup \{i\}$  on riippumaton.  $\square$

Lopputuloksena olemme siis saaneet kaikilla syötteillä oikein ja ajassa  $O(n^2)$  toimivan ahneen algoritmin aikatauluongelmalle.

## Ahne algoritmi heuristiikkana

Ahne algoritmi ei kaikissa ongelmissa tuota parasta tulosta, mutta usein kuitenkin jotain kohtuullista. Koska ahne algoritmi ovat tyypillisesti nopea, sellaista voidaan käyttää heuristiikkana ongelmassa, jolle ei tunneta tehokasta tarkkaa algoritmia.

**Esimerkki** Kauppamatkustajan ongelma (Travelling Salesman Problem, TSP): määrättävä verkossa lyhin polku, joka käy tasan kerran joka solmussa ja palaa lähtöpisteeseensä

Ongelma on NP-kova, eikä tehokasta tarkkaa ratkaisua siis ole näköpiirissä.

**Ahne heuristiikka:** Käsitellään kaaret painon mukaan kasvavassa järjestyksessä.

Käsiteltävä kaari lisätään reittiin, ellei se riko kumpaakaan seuraavista ehdoista:

- Mihinkään solmuun ei saa tulla yli kahta kaarta.
- Reitille ei saa tulla syklejä (paitsi yksi koko verkon käsittävä).

