

Esimerkkejä vaativuusluokista

Seuraaville kalvoille on poimittu joitain esimerkkejä havainnollistamaan algoritmien aikavaativuusluokkia.

Esimerkit on valittu melko mielivaltaisesti laitoksella tehtävään tutkimukseen liittyen. Ne eivät pyri antamaan mitenkään tasapainoista kuvaa siitä, millaisia aikavaativuudet ”yleensä” ovat (sikäli kuin tällainen tavoite olisi edes mielekäs).

(Tämä materiaali täydentää luentokalvon 46 listaa.)

Assosiaatiosääntöjen louhiminen

Tarkastellaan taulukkoa $A[1..m, 1..n]$, missä

- sarakkeet $1 \leq j \leq n$ esittävät **attribuutteja**; esim. kaupassa myytävät tuotteet
- rivit $1 \leq j \leq m$ esittävät **transaktioita**; esim. kunkin asiakkaan "ostoskori"
- $A[i, j] = 1$ tarkoittaa, että transaktiolla i on attribuutti j ; esim. asiakkaan i ostoskorissa oli tuote j

Halutaan löytää **assosiaatiosääntöjä**, jotka kertovat, mitkä attribuuttiyhdistelmät tyypillisesti esiintyvät samoissa transaktioissa; esim.

asiakkaat, jotka ostivat olutta ja sinappia, yleensä ostivat myös makkaraa.

Kyseessä on **tiedonlouhinnan** perusongelma, joka voidaan ratkaista ajassa $O(mn2^n)$ soveltamalla tunnettua Apriori-algoritmia.

Assosiosääntöjen louhimisen aikavaativuudesta $O(mn2^n)$ huomattavaa:

- Riippuvuus transaktioiden lukumäärästä m on **lineaarinen**, joten ratkaisu skaalautuu hyvin suurellekin joukolle transaktioita.
- Riippuvuus attribuuttien lukumäärästä n on **eksponentiaalinen**, joten attribuuttien lisääminen johtaa nopeasti tehokkuusongelmiin.
- **Mutta:** kerroin 2^n on hyvin pessimistinen yläraja, todellinen suoritus aika vaihtelee paljon sen mukaan, millaista data todella on.
- Käytännössä esim. m voi olla joitakin miljoonia ja $n \approx 1000$, jolloin suoritus aika voi olla minuutteja tai tunteja.

Sivuutamme tässä O -merkinnän täsmällisen määrittelyn kun muuttujia on useita, esim. tässä m ja n .

Likimääräinen hahmonsovitus

On annettu kaksi merkkijonoa:

- pitkä **teksti**; esim. ihmisen koko DNA n. 3 gigatavua
- lyhyempi **hahmo**; esim. n. 100 tavun mittainen DNA-pätkä

Halutaan tietää, esiintyykö hahmo osamerkkijonona tekstissä, kun sallitaan vähäinen määrä virheitä.

Esim. tekstissä KAALILAATIKKO hahmolla AALI on

- yksi virheetön esiintymä: KAALILAATIKKO
- yksi esiintymä, jossa yksi virhe: KAALILAATIKKO

Likimääräinen hahmonsovitus ratkeaa **dynaaminen ohjelmointi** -nimisellä perustekniikalla ajassa $O(mn)$, missä m on hahmon ja n tekstin pituus.

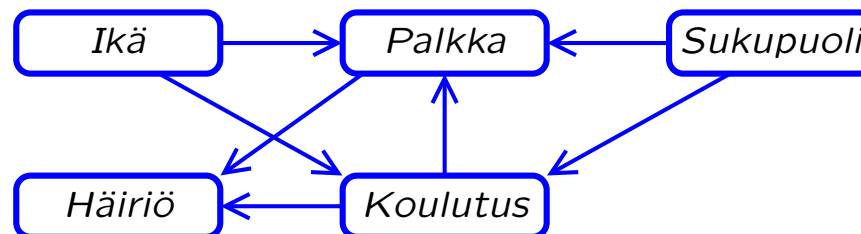
- Aikavaativuus $O(mn)$ on periaatteessa neliöllinen syötteen koon $m + n$ suhteen, mutta tämä on hieman harhaanjohtava ilmaus, koska yleensä $n \ll m$.
- Esimerkissä $m = 3 \cdot 10^9$ ja $n = 100$ ajoaika voisi olla suunnilleen 5 tuntia.
- Pelkkä ihmisen täyden DNA:n lukeminen levytä vie pari minuuttia.
- Jos halutaan vain virheettömät esiintymät, tehokkaammilla algoritmeilla päästään aikavaativuuteen $O(m)$, jolloin aikaa ei käytännössä kulu juuri enempää kuin pelkkään syötteen lukemiseen levytä.

Parhaan todennäköisyysmallin etsiminen

Leikkikaluesimerkki: Henkilön luottotiedot kuvataan muuttujilla *Ikä*, *Sukupuoli*, *Koulutus*, *Palkka* ja *Maksuhäiriö*.

Kun valitaan satunnainen henkilö populaatiosta ja annetaan häntä koskevat muuttujien arvot, saadaan [esimerkki](#).

Muuttujien tilastollisia riippuvuuksia voidaan kuvata [graafisilla malleilla](#) eli [Bayes-verkoilla](#). Esim. graafinen malli



esittää tilannetta, jossa *Maksuhäiriö* riippuu suoraan muuttujista *Palkka* ja *Koulutus*, mutta vain välillisesti muuttujista *Ikä* ja *Sukupuoli*. Tavoitteena on löytää jossain mielessä parhaiten esimerkkijoukon jakaumaa kuvaava graafinen malli; ts. mitkä nuolet kaaviossa pitäisi olla.

Jos muuttujia on n , niin erilaisten mallien lukumäärälle saadaan yläraja $O(2^{n^2})$, mikä kasvaa **todella** nopeasti muuttujan n funktiona.

Tekemällä pieniä lisäoletuksia ja suunnittelemalla algoritmi hyvin saadaan aikavaativuus kuitenkin luokkaan $O(n2^n)$. Tämä on edelleen iso, mutta ei aivan mahdoton.

(Lisäksi aikavaativuus skaalautuu lineaarisesti esimerkkien lukumäärän suhteen, samaan tapaan kuin assosiaatiosääntöjen louhinnassa.)

Tyypillisillä esimerkkijoukoilla voidaan optimoida esim. 30 muuttujan malli 10 tunnissa käyttäen 8 prosessoria. Algoritmin laskennallisesti vaativin osa rinnakkaistuu helposti.

Käytännössä muuttujia on usein paljon enemmän kuin 30 ja käytetään **heuristisia** menetelmiä, jotka eivät takaa parhaan ratkaisun löytymistä.

Kun Mooren lain mukaan tietokoneiden nopeus kaksinkertaistuu 18 kuukaudessa, voidaan tehokkaampia koneita ostamalla kasvattaa optimoitavien mallien kokoa vajaan yhden muuttujan verran vuodessa.