



Algoritmitutkimuksen menetelmistä ***Tutkimusmenetelmät-kurssi, s-2004***

Pekka Kilpeläinen

Kuopion yliopisto

Tietojenkäsittelytieteen laitos

Tänään

- ⑥ Tietojenkäsittelytiede tutkimusalana
- ⑥ Algoritmitutkimuksen käsitteet ja tulokset
- ⑥ Analyttiset menetelmät

Ensi viikolla

- ⑥ Kokeelliset menetelmät
- ⑥ Algoritmisten tulosten raportointi

TKT tutkimusalana

Ks. esim. E. Ukkonen: Tietojenkäsittelytiede. Luku 1.2 teoksessa Hyvönen, Karanta, Syrjänen (toim.) *Tekoälyn ensyklopedia*, Gaudeamus, 1993.

TKT (Computer Science) on menetelmätiede, kohteena tietojenkäsittelymenetelmät

- ⑥ tietojenkäsittelyongelmat, algoritmit
- ⑥ abstraktiot
 - △ kuvaustavat: tiedon esitystavat, ohjelmointi- ja kuvauskielet, ...
 - △ rajapinnat
 - △ tietorakenteet, tietomallit
- ⑥ suunnittelumenetelmät
- ⑥ e.m. toteuttaminen ja soveltaminen

TKT:n tutkimustulokset?

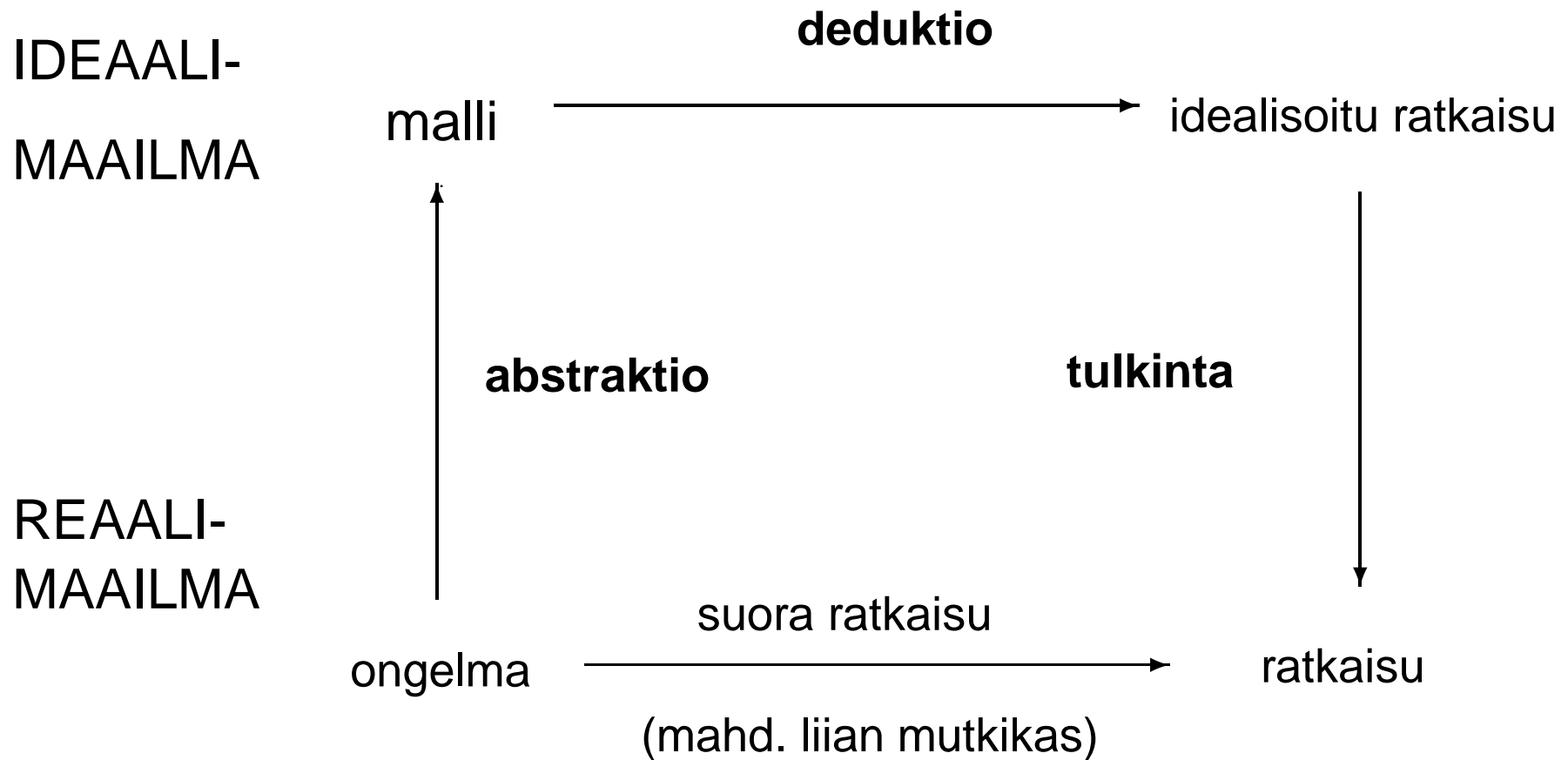
- ⑥ Uusi menetelmä, toteutustapa tai sovellus
- ⑥ Menetelmään, toteutustapaan tai soveltamiseen liittyvä uusi tieto
- ⑥ Tieteellisyys: tiedon paikkansapitävyys **perusteltu**
 - ← olemassaoleva tieto
 - ← formaali todistus
 - ← empiirinen koe

Algoritmitutkimuksen luonnehdintaa

Algoritmitutkimus klassista tietojenkäsittelytiedettä

- ⑥ Eleganttia: eksakteja ja yleisiä tuloksia
- ⑥ Haastavaa: hyviä (= osin vaikeita) ongelmia
- ⑥ Relevanttia: läheinen yhteys sovelluksiin

Korkeantason näkemys



Algoritmit

Algoritmi: täsmällisesti määritellyn tk-tehtävän automatisoitavissa oleva ratkaisu \approx tietokoneohjelma

- ⑥ ei sama; pikemmin ohjelmien komponentteja
- ⑥ mielellään yleisen ongelman ratkaisu

Oikea/hyvä algoritmi?

- ⑥ Toimii *oikein*
- ⑥ Toimii *tehokkaasti*

Seuraavaksi esimerkki formaalista oikeellisuus- ja tehokkuustarkastelusta

Algoritmin oikeellisuus

Annettava oikea vastaus jokaiseen ratkaistavan ongelman tapaukseen

Esimerkki: Pikalajittelu

Lajittelutehtävä: Järjestä jono $S = s_1, \dots, s_n$ jonoksi $S' = s'_1, \dots, s'_n$ s.e. $s'_1 \leq s'_2 \leq \dots \leq s'_n$

Huom: Täsmällisyyden ja määritelmien merkitys;
Tutkija luo tietoon järjestystä!

Pikalajittelu

(“Hajota-ja-hallitse”)

```
procedure QuickSort( $S$ ):  
  if  $|S| \leq 1$  then return  $S$ ;  
  else  
    Valitse jakoalkio  $s \in S$ ;  
    Osita  $S$  jonoiksi  $S_{<s}$ ,  $S_{=s}$  ja  $S_{>s}$ ;  
    return QuickSort( $S_{<s}$ )  $\circ$   $S_{=s}$   $\circ$  QuickSort( $S_{>s}$ );  
  endif;
```

Metodi 1: Mallintaminen:

Algoritmi on toteutusten oleelliset piirteet säilyttävä yksinkertaistus

Pikalajittelun oikeellisuus

Lause Proseduuri QuickSort toimii oikein.

Todistus *Induktio* alkioden lkm $n = |S|$ suhteen. □

Metodi 2: Matemaattinen induktio:

$$1^\circ P(k)$$

$$2^\circ P(k), P(k + 1), \dots, P(n - 1) \Rightarrow P(n)$$

$$\therefore P(n) \text{ kaikilla } n \geq k$$

Algoritmin tehokkuus

Resurssien (aika, muisti) tarve tietyn kokoisilla syötteillä?

— *riippuvuutena* syötteen koosta (n)

→ huomion kohteena algoritmin *skaalautuvuus*

Yksinkertaistuksia:

⑥ keskitytään “riittävän suuriin” syötteisiin ($n \geq n_0$)

⑥ eliminoidaan ohjelmointitaidon sekä suoritusympäristön ja laitteiston vaikutus: analysoidaan “perusoperaatioiden” lukumäärää

↪ Asymptoottiset kertaluokka-arviot

(Mielekkäitä? Yleensä, mutta hyvä olla kriittinen.)

Kertaluokka-arviot

Yläraja: $T(n) = O(f(n))$, jos $T(n) \leq cf(n)$ kun $n \geq n_0$ joillain c, n_0 .

⑥ Aito pienemmyys: $T(n) = o(f(n))$ jos $\lim_{n \rightarrow \infty} T(n)/f(n) = 0$

Vast. alaraja $T(n) = \Omega(f(n))$ ja
tarkka kertaluokka $T(n) = \Theta(f(n))$

Välittämiä sievennystekniikoita:

$$c \times f(n) = \begin{cases} O(f(n)) \\ \Theta(f(n)) \\ \Omega(f(n)) \end{cases} \quad \begin{array}{l} \text{(vakio kertoimien } c > 0 \\ \text{merkityksettömyys)} \end{array}$$

$$f(n) \pm g(n) = \Theta(f(n)), \quad \begin{array}{l} \text{(alempiasteisten termien} \\ \text{merkityksettömyys)} \end{array}$$

jos $g(n) = o(f(n))$

Pahin ja keskimääräinen tapaus

Huom: Samankokoistenkin syötteiden resurssintarve voi vaihdella (Kuva)

→ **Pahimman tapauksen analysointi:**

$$T_{\max}(n) = \max_{|w|=n} T(w)$$

Edut ja haitat:

- + : analyysi yksinkertaistuu
- + : takuu: kompleksisuus ei koskaan huonompi
- : liian pessimistinen, jos pahin tapaus harvinainen

Pikalajittelun pahin tapaus

Lause Pikalajittelun $T_{\max}(n) = \Theta(n^2)$.

Tod. Merk. $T(n) = T_{\max}(n)$. Ositus vaatii (jollain b) ajan $\geq bn$

\rightsquigarrow Kun $n \geq 2$,

$$T(n) \geq bn + T(n-1) \quad (\text{jakoalkio min tai max})$$

$$\geq bn + b(n-1) + T(n-2)$$

\vdots

$$\geq b(n + (n-1) + \dots + 2)$$

$$= b \sum_{i=2}^n i = b \left(\frac{n(n+1)}{2} - 1 \right) = \Omega(n^2)$$

Metodi 3: Kaavamanipulointi

Pikalajittelun pahin tapaus (2)

Toisaalta kukin alkio $s \in S$ jakoalkiona enintään kerran \rightsquigarrow rekursiivisia kutsuja $O(n)$ kpl, ja kussakin ositustyö $O(n)$; Siten $T(n) = O(n^2)$, joten $T(n) = \Theta(n^2)$. \square

Metodi 4: Kombinatorinen havainnointi

Käytännössä pikalajittelu huomattavasti pahinta tapaustaan parempi (Ks. seur.)

Keskimääräinen kompleksisuus

$$T_{avg}(n) = \sum_{|w|=n} p(w)T(w)$$

Kompleksisuuden odotusarvo: keskikompleksisuus painotettuna tapausten w esiintymistodennäköisyydellä $p(w)$

Edut ja haitat:

- + : kuva keskimääräisestä käyttäytymisestä
- : tapausten todennäköisyysjakauma?
- : analyysin vaikeutuminen

Pikalajittelu keskimäärin

(Aho, Hopcroft & Ullman, 1974, s. 93–95)

Lause Pikalajittelun $T_{avg}(n) = O(n \log n)$.

Tod. Merk. $T(n) = T_{avg}(n)$. Jakaumaoletus: syötteet erillisten alkioden permutaatioita, kukin yhtä usein. $T(0) = T(1) = a$ (vakio). Onnistukoon proseduurin paikallinen työ ajassa bn . Jakaumaoletus \Rightarrow kullakin $i = 1, \dots, n$ on $|S_{<s}| = i - 1$ ja $|S_{>s}| = n - i$ todennäköisyydellä $1/n \rightsquigarrow$ Kun $n \geq 2$,

$$\begin{aligned} T(n) &\leq bn + 1/n \sum_{i=1}^n (T(i-1) + T(n-i)) \\ &= bn + 2/n \sum_{i=0}^{n-1} T(i) \end{aligned} \tag{1}$$

Pikalajittelu keskimäärin (2)

Osoitetaan $T(n) \leq cn \ln n$, kun $n \geq 2$ ja $c = 2(a + b)$:

Perustapaus: $T(2) \leq 2b + 2a \leq 2(a + b)2 \ln 2$; OK

Induktioaskel: Kirjoitetaan yläraja (1) muotoon

$$bn + 2/n[T(0) + T(1) + \sum_{i=2}^{n-1} T(i)] \leq bn + 4a/n + 2c/n \sum_{i=2}^{n-1} i \ln i$$

Arvioidaan: $\sum_{i=2}^{n-1} i \ln i \leq \int_2^n x \ln x dx \leq (n^2 \ln n)/2 - n^2/4$

Siten $T(n) \leq bn + 4a/n + cn \ln n - cn/2$

Nyt $bn + 4a/n \leq cn/2$, joten $T(n) \leq cn \ln n$ □

Metodi 5: Reaalianalyysi, tn-laskenta, lukuteoria ym.

Pikalajittelu käytännössä

Logaritmit kasvavat hitaasti, joten $O(n \log n)$ -algoritmi skaalautuu lähes yhtä hyvin kuin lineaarinen

Pikalajittelu käytännössä parhaita lajittelumenetelmiä (esim. virtuaalimuistin käytön suhteen)

⇒ Kokeellinen algoritmitutkimus

Yhteenveto

Teoreettisen algoritmitutkimuksen tulokset

- ⑥ algoritmit,
- ⑥ niiden oikeellisuus ja
- ⑥ kompleksisuus

Metodit analyyttis-deduktiivisia, “matemaattisia”

Muita näkökohtia: toteutettavuus, käyttökelpoisuus, teoreettisen tarkastelun validius, todellinen tehokkuus, ...

~>

Kokeellinen algoritmitutkimus