Kolmogorov-kompleksiteetti

Laskennan teorian opintopiiri

Sebastian Björkqvist

3. helmikuuta 2014

Tiivistelmä

Työssä esitetään määritelmät merkkijonon lyhimmälle kuvaukselle sekä Kolmogorov-kompleksiteetille ja todistetaan muutamia niihin liittyviä perustuloksia. Lisäksi työssä käsitellään Kolmogorov-kompleksiteetin laskemisen ratkeamattomuutta sekä tiivistymättömien jonojen yhteyttä satunnaisuuteen.

Johdanto

Tietojenkäsittelytieteessä ovat keskeisessä roolissa käsitteet algoritmi ja informaatio. Algoritmien kohdalla käytämme määritelmänä Churchin-Turingin teesiä, joka sanoo että kaikki laskettavissa oleva voidaan laskea Turingin koneella. Teesiä perustellaan sillä, että moni muu laskennan malli on osoitettu laskentateholtaan ekvivalentiksi Turingin koneen kanssa.

Informaation ollessa kyseessä pitää toki ensiksi miettiä, mitä tarkoitamme käsitteellä informaatio? Yksi tapa nähdä asia on yrittää löytää jokin mahdollisimman lyhyt tapa kuvata tietty merkkijono. Jotta tätä lyhyempää kuvausta, eli erityisesti sen pituutta, voitaisiin verrata merkkijonoon itseensä, haluamme määritellä myös mahdolliset lyhyemmät kuvaukset merkkijonoina.

Tämän lähestymistavan formalisaatiota Turingin koneiden avulla kutsutaan **Kolmogorov-kompleksiteetiksi**. Formalismin avulla voidaan nähdä yhteyksiä satunnaisuuden ja informaation välillä; voidaan esimerkiksi osoittaa, että n.s. tiivistymättömät jonot ovat satunnaisjonojen kaltaisia.

Työssä seurataan pääosin Sipserin kirjan *Introduction to the Theory of Computation* ([1]) lukua 6.4 (s. 237-245). Kolmogorov-kompleksiteettia ja sen sovelluksia käsitellään laajemmin Lin ja Vitányin kirjassa *An Introduction to Kolmogorov Complexity and Its Applications* ([2]).

Informaatiomäärä

Huomautus. Tässä työssä käsitellään ainoastaan binäärimerkkijonoja. Kaikki merkkijonot voidaan esittää binäärijonoina, joten tämä ei aseta mitään rajoituksia esitettäville määritelmille.

Millä tavalla voidaan kuvailla merkkijonon sisältämää informaatiomäärää? Lienee selvää että merkkijono kuvaa aina itseään, eli toisin sanoen jos välitämme jonkin merkkijonon kokonaisuudessaan saadaan varmasti selville sen sisältämä informaatio. Toisaalta intuitiivisesti tuntuu siltä, että samanpituiset merkkijonot voivat sisältää eri määrän informaatiota. Valotetaan asiaa kahdella merkkijonolla:

$$A = 111111111111111111111111111111$$
$$B = 101000010101101101110000101011$$

Molemmat yllä olevat merkkijonot ovat pituudeltaan 30 merkkiä, mutta merkkijono A vaikuttaa sisältävän vähemmän informaatiota kuin merkkijono B, sillä A vain toistaa merkkiä 1 30 kertaa, kun taas B näyttää melko satunnaiselta. Täten, jos haluamme kuvata merkkijonon sisältämää informaatiomäärää, pitää tarkastella jotain muuta kun pelkästään jonon pituutta.

Merkkijonon lyhin kuvaus

Mahdollistaaksemme eri merkkijonojen sisältämän informaatiomäärän vertailun haluamme löytää jonkin tavan kuvata jokainen merkkijono mahdollisimman lyhyesti. Jotta tätä lyhintä kuvausta pystytään vertaamaan merkkijonoon itseensä sekä muihin lyhimpiin kuvauksiin, haluamme määritellä myös lyhimmät kuvaukset merkkijonoina.

Luonnollinen lähestymistapa on ottaa Turingin koneet avuksi, sillä kuten jo edellisessä kappaleessa jonon A kohdalla huomasimme, voi hyvinkin olla mahdollista kuvata jotakin merkkijonoa lyhyesti esimerkiksi sanomalla että se toistaa jotain tunnettua kaavaa. Täten päädymme kuvaamaan merkkijonoja Turingin koneen ja sille annettavan syötteen avulla:

Määritelmä. Merkkijonon x **lyhin kuvaus** $d(x)$ on lyhin jono $\langle M, w\rangle$, jolle pätee, että Turingin kone M palauttaa jonon x syötteellä w.

Jotta määritelmämme olisi järkevä, täytyy meidän lyödä lukkoon jokin tapa kuvata pari $\langle M, w\rangle$ merkkijonona. Koodaamme ensiksi Turingin koneen M merkkijonoksi $\langle M\rangle$ jollain tavalla, ja konkatenoimme perään syötteen w. Lopputuloksena on siis merkkijono $\langle M\rangle w$.

Huomautus. Voimme yllä koodata Turingin koneen merkkijonoksi haluamallamme tavalla kunhan jotenkin pidämme huolen siitä, että tiedämme mikä osa konkatenoidusta jonosta $\langle M \rangle w$ on osa Turingin koneen kuvausta ja mikä osa on sille annettavaa syötettä. Yksi mahdollisuus on tuplata jonon $\langle M \rangle$ bitit, liittää perään jono 01 ja lisätä sen jälkeen jono w. Tätä esitystapaa käytetään myöhemmin eräässä todistuksessa.

Määritelmä. Merkkijonon x **Kolmogorov-kompleksiteetti** on merkkijonon lyhimmän kuvauksen pituus, toisin sanoen

$$K(x) = |d(x)|.$$

Perustuloksia

Lause. *On olemassa vakio $c \geq 0$ jolle pätee kaikilla merkkijonoilla x että*

$$K(x) \leq |x| + c.$$

Merkkijonon Kolmogorov-kompleksiteetti on siis aina korkeintaan vakion verran suurempi kuin merkkijonon pituus, ja vakio ei riipu jonosta itsestään.

Todistus. [1, Thm 6.24] Haluamme siis löytää ylärajan jonon Kolmogorov-kompleksiteetille. Koska Kolmogorov-kompleksiteetti on jonon lyhimmän kuvauksen pituus, tiedämme, että mikäli löydämme minkä tahansa kuvauksen jonolle x, voi jonon lyhin kuvaus olla korkeintaan tämän kuvauksen pituinen.

Voimme muodostaa ylärajan seuraavasti: Olkoon M Turingin kone joka toimii kuten identtinen kuvaus, eli M palauttaa saamansa syötteen. Tällöin eräs kuvaus jonolle x on jono $\langle M \rangle x$. Väitteen vakio c on jonon $\langle M \rangle$ pituus. □

Seuraava lause sanoo, että merkkijono xx ei sisällä olennaisesti enempää informaatiota kuin jono x.

Lause. *On olemassa vakio $c \geq 0$ jolle pätee kaikilla merkkijonoilla x että*

$$K(xx) \leq K(x) + c.$$

Vakio c ei siis riipu jonosta x.

Todistus. [1, Thm 6.25] Todistuksen ideana on käyttää merkkijonon x lyhintä kuvausta $d(x)$ merkkijonon xx kuvauksen muodostamiseen. Koska määrittelimme myöskin lyhimmät kuvaukset merkkijonoina, voimme antaa jollekin Turingin koneelle syötteenä jonkin toisen merkkijonon lyhimmän kuvauksen. Tässä tapauksessa määrittelemme Turingin koneen M, joka saa syötteenä jonon $d(x)$ ja laskee sen avulla jonon x seuraavalla tavalla:

$M =$ "Syötteellä $\langle N, w \rangle$, missä N on Turingin kone ja w on merkkijono:

1. Aja konetta N syötteellä w kunnes se palauttaa jonon s.
2. Palauta jono ss."

Nyt eräs kuvaus jonolle xx on jono $\langle M \rangle d(x)$. Tämän kuvauksen pituus on $|\langle M \rangle| + |d(x)|$, eli toisin sanoen $c + K(x)$, missä vakio c on jonon $\langle M \rangle$ pituus. $\square$

Kahden mielivaltaisen jonon konkatenaation Kolmogorov-kompleksiteetille voimme todistaa seuraavanlaisen ylärajan:

Lause. *On olemassa vakio $c \geq 0$ jolle pätee kaikilla jonoilla x ja y että*

$$K(xy) \leq 2 \cdot K(x) + K(y) + c.$$

Todistus. [1, Thm 6.26] Käytämme kuten edellisessä todistuksessa merkkijonojen x ja y lyhimpiä kuvauksia apuna kun muodostamme ylärajan jonon xy lyhimmälle kuvaukselle. Idea on seuraava: Luodaan jono w, joka alkaa jonolla $d(x)$, paitsi että jokainen jonon $d(x)$ bitti kirjoitetaan kahdesti. Kun tämä on tehty, lisätään perään merkit 01, jonka jälkeen lisätään vielä koko jono $d(y)$. Jos meillä esimerkiksi on jonot $d(x) = 0110110$ ja $d(y) = 1110001$, niin saadaan jono

$$w = \underbrace{00111100111100}_{d(x) \text{ tuplattuna}} 01 \underbrace{1110001}_{d(y)}.$$

Seuraavaksi luomme Turingin koneen M, joka saatuaan syötteen w purkaa siitä jonot $d(x)$ ja $d(y)$. Määritelmän mukaan $d(x)$ on pari $\langle N, v \rangle$, jolle Turingin kone N palauttaa jonon x kun sille annetaan syötteeksi v. Täten M saa käsiinsä jonon x käyttämällä kuvauksen $d(x)$ sisältämää Turingin konetta. Vastaavalla tavalla M hankkii jonon y, jonka jälkeen M konkatenoi jonot jonoksi xy ja palauttaa sen.

Kuvauksen $\langle M \rangle w$ pituus on $|\langle M \rangle| + 2|d(x)| + |d(y)| + 2$, eli lauseen väite pätee kun valitaan vakioksi $c = |\langle M \rangle| + 2$. $\square$

Edellisen lauseen ylärajaa voidaan parantaa esimerkiksi muotoon

$$K(xy) \leq 2 \cdot log_2(K(x)) + K(x) + K(y) + c,$$

jos koko jonon $d(x)$ tuplaamisen sijasta tuplataan vain jono joka kertoo jonon $d(x)$ pituuden. Osoittautuu kuitenkin mahdottomaksi saavuttaa yläraja $K(xy) \leq K(x) + K(y) + c$, toisin sanoen jokaiselle positiiviselle vakiolle c

löytyy jonot x ja y jolle pätee että $K(xy) > K(x) + K(y) + c$ ([2, Ex. 2.2.3, s. 118]). Tämä johtuu siitä, että jos vain konkatenoimme jonot $d(x)$ ja $d(y)$ peräkkäin, meillä ei ole mitään tapaa tietää missä kohtaa jono $d(x)$ loppuu ja $d(y)$ alkaa. Voi hyvinkin käydä niin, että jono voidaan jakaa useammasta kohtaa kahteen osaan joista molemmat osat sisältävät syntaktisesti korrektin Turingin koneen sekä sille annettavan syötteen. Täten jonoja $d(x)$ ja $d(y)$ ei välttämättä pystytä purkamaan yksiselitteisesti.

Tiivistymättömät jonot

Määritelmä. Merkkijono x on **c-tiivistyvä**, jos

$$K(x) \leq |x| - c.$$

Jos merkkijono ei ole 1-tiivistyvä, on se **tiivistymätön**.

Seuraavan lauseen nojalla on olemassa kaikenpituisia tiivistymättömiä jonoja.

Lause. *Olkoon n mielivaltainen ei-negatiivinen kokonaisluku. On olemassa ainakin yksi tiivistymätön pituutta n oleva merkkijono.*

Todistus. [1, Thm 6.29] Jotta pituutta n oleva jono x olisi tiivistyvä, täytyy sen lyhimmän kuvauksen $d(x)$ pituus olla pienempi kuin n. Huomataan, että jokainen merkkijono voi olla ainoastaan yhden merkkijonon lyhin kuvaus, sillä muuten alkuperäisiä jonoja ei voitaisi yksikäsitteisesti palauttaa. Pituutta n olevia merkkijonoja on 2^n kappaletta, mutta tätä lyhyempiä merkkijonoja on ainoastaan

$$2^0 + 2^1 + \cdots + 2^{n-1} = 2^n - 1$$

kappaletta. Täten löytyy ainakin yksi pituutta n oleva jono jota ei voida kuvata merkkijonolla joka on lyhyempi kuin n. □

Millaisilta tiivistymättömät jonot sitten näyttävät? Voisi kuvitella, että tiivistymätön jono ei toista mitään selkeää kaavaa koska sitä ei pystytä mitenkään kuvaamaan lyhyesti. Tämä intuitiivinen havainto osoittautuu itseasiassa oikeaksi; pätee nimittäin, että tiivistymättömät jonot ovat hyvin paljon satunnaisjonojen kaltaisia. Voidaan osoittaa, että ominaisuudet jotka pätevät lähes jokaiselle merkkijonolle, ja siten myös satunnaisesti valitulle jonolle, pätevät myös tiivistymättömille merkkijonoille.

Huomautus. Tässä **ominaisuus** on kuvaus, joka liittää merkkijonoon arvon tosi tai epätosi. Sanotaan, että ominaisuus **pätee lähes jokaiselle jonolle** jos se osa pituutta n olevista jonoista jolla ominaisuus on epätosi lähestyy nollaa kun n lähestyy ääretöntä.

Lause. *Olkoon f ominaisuus joka pätee lähes jokaiselle merkkijonolle. Tällöin jokaisella $c > 0$ pätee, että ominaisuus f on epätosi ainoastaan äärellisen monelle jonolle joka ei ole c-tiivistyvä.*

Todistushahmotelma. Saamme aikaiseksi lyhyen kuvauksen jonoista joille ominaisuus f ei päde muodostamalla lista kaikista jonoista joille f on epätosi ja käymällä läpi tätä listaa. Täten, jos jono x on tarpeeksi pitkä, voimme löytää jonolle x kuvauksen joka on lyhyempi kuin $|x| - c$ viittaamalla siihen ylläolevan listan indeksiin, josta x löytyy. Todistus kokonaisuudessaan löytyy kirjasta [1, Thm 6.31]. □

Emme voi yleisesti antaa esimerkkejä pitkistä tiivistymättömistä jonoista (tätä perustellaan tarkemmin seuraavassa osiossa), mutta voimme kuitenkin osoittaa, että merkkijonon lyhin kuvaus on lähes tiivistymätön. Intuitiivisesti tämä tuntuu selvältä, sillä jos lyhintä kuvausta voitaisiin merkittävästi tiivistää, se tuskin olisi lyhin tapa kuvata jotain jonoa.

Lause. *On olemassa vakio $c > 0$, jolle pätee jokaisella merkkijonolla x että sen lyhin kuvaus $d(x)$ ei ole c-tiivistyvä.*

Todistus. Kts [1, Thm 6.32]. □

Kompleksiteetin määrittäminen

Millä tavalla voitaisiin määrittää merkkijonon x Kolmogorov-kompleksiteetti? Määritelmän mukaan jonon x kompleksiteetti on jonon lyhimmän kuvauksen $d(x)$ pituus. Koska lyhimmät kuvaukset ovat merkkijonoja, voimme käydä läpi kaikki merkkijonot pituusjärjestyksessä, jakaa jokainen jono kahteen osaan kaikilla mahdollisilla tavoilla ja katsoa muodostuuko ensimmäisestä osasta Turingin kone joka palauttaa jonon x kun sille annetaan syötteeksi merkkijonon toinen osa. Ensimmäinen vastaantuleva jono jolle edellämainittu pätee olisi tällöin jonon x lyhin kuvaus.

Tämän lähestymistavan ongelma on se, että emme voi mitenkään olla varmoja siitä, että merkkijonosta löytyvä Turingin kone pysähtyy sille annetulla syötteellä. Saattaa siis käydä niin, että algoritmimme ei koskaan pysähdy, koska jonkin merkkijonon kohdalla vastaantuleva Turingin kone ei pysähdy. Koska pysähtymisongelma on ratkeamaton [1, Thm 5.1], emme myöskään voi mitenkään etukäteen poimia pois niitä merkkijonoja jotka jäävät ikuiseen silmukkaan. Itse asiassa mikään algoritmi ei pysty määrittämään kompleksiteettia kaikissa tapauksissa, minkä osoitamme seuraavaksi:

Lause. *Mikään algoritmi ei pysty laskemaan Kolmogorov-kompleksiteettia kaikille merkkijonoille.*

Todistus. [3, Thm 3] Tehdään vastaoletus, eli oletetaan että Kolmogorov-kompleksiteetti $K(x)$ voidaan laskea kaikille jonoille x. Tarkastellaan seuraavaa Turingin konetta M:

$M =$ "Syötteellä n, missä n on kokonaisluku:

1. Käy järjestyksessä läpi merkkijonoja lyhyimmistä alkaen.
2. Kun löytyy merkkijono s, jolle $K(s) \geq n$, palauta s."

Osoitimme aikaisemmin että kaikenpituisia tiivistymättömiä jonoja on olemassa, joten M pysähtyy kaikilla syötteillä. Olkoon s jono jonka M palauttaa jollain kokonaisluvulla n. Tällöin $\langle M \rangle n$ on eräs jonon s kuvaus, ja tämän kuvauksen pituus on $|\langle M \rangle| + \log_2 n$, sillä kokonaisluvun n kuvaus binäärijonona on pituudeltaan $\log_2 n$. Koska n kasvaa nopeammin kuin $\log_2 n$ ja $|\langle M \rangle|$ on vakio, löytyy jokin kokonaisluku n jolle $n > |\langle M \rangle| + \log_2 n$. Koneen M toiminnan nojalla tiedämme, että $K(s) \geq n$, mikä ei ole mahdollista, sillä kuvauksen $\langle M \rangle n$ pituus on pienempi kuin n. Täten kompleksiteettia ei voida laskea. □

Emme siis pysty yleisesti laskemaan jonon Kolmogorov-kompleksiteettia. Jos pysähtymisongelma olisi ratkeava, pystyisimme edellä esitetyn naiivin algoritmin avulla selvittämään kaikkien jonojen kompleksiteetin. Osoittautuu, että sama pätee myös toisinpäin: Jos Kolmogorov-kompleksiteetti pystyttäisiin laskemaan kaikille merkkijonoille, olisi pysähtymisongelma ratkeava ([4])! Nämä ongelmat ovat siis Turing-yhtäpitäviä.

Ei myöskään ole yleisesti mahdollista selvittää onko jokin merkkijono tiivistymätön. Tämän todistus etenee samalla lailla kuin edellisen lauseen todistus, sillä M löytää jonon jonka kompleksiteetti on vähintään n tarkistamalla onko jono tiivistymätön. Tiivistymättömille jonoille voidaan todistaa myös vahvempi tulos:

Lause. *Olkoon A kaikkien tiivistymättömien jonojen joukko. Kaikki joukon A äärettömät osajoukot ovat Turing-tunnistamattomia.*

Todistus. Tehdään vastaoletus: Olkoon N Turingin kone, joka tunnistaa jonkin joukon $B \subset A$, missä B on ääretön. Tällöin on olemassa kone E (ns.

luetteloija, *enumerator*), joka luetteloi kaikki joukon B alkiot ([1, Thm 3.21]). Koska joukko B on ääretön, tulee kone E jossain vaiheessa tulostamaan mielivaltaisen pitkiä jonoja. Voimme siis kuten edellisen lauseen todistuksessa muodostaa luetteloijan E avulla koneen M, joka syötteellä n palauttaa jonon jonka Kolmogorov-kompleksiteetti on vähintään n, ja saamme aikaiseksi ristiriidan. $\square$

Määritelmän optimaalisuus

Merkkijonon lyhimmän kuvauksen määritelmässä käytimme laskentamallina Turingin konetta. Osoittautuu kuitenkin että kompleksiteetti, eli lyhimmän kuvauksen pituus, ei oleellisesti muutu vaikka käyttäisimme määritelmässä jotain toista algoritmia. Tämä johtuu siitä, että Churchin-Turingin teesin mukaan voimme Turingin koneen avulla simuloida mitä tahansa muuta algoritmia. Kompleksiteetti siis muuttuu vain vakion verran laskentamallia vaihdettaessa, ja vakion suuruus on sen Turingin koneen kuvauksen suuruus, joka simuloi valittua laskentamallia. Tämän havainnon formaali esitys löytyy kirjasta [1, Thm 6.27].

Viitteet

[1] Sipser, M. *Introduction to the Theory of Computation*, second edition. Thomson, 2006.

[2] Li, M., Vitányi, P. *An Introduction to Kolmogorov Complexity and Its Applications*, third edition. Springer, 2008.

[3] Miltersen, P. B. *Course notes for Data Compression - 2 Kolmogorov complexity.* Fall 2005.
`http://www.daimi.au.dk/~bromille/DC05/Kolmogorov.pdf`

[4] Chaitin, G.J., Arslanov, A., Calude, C. *Program-Size Complexity Computes the Halting Problem.* CDMTCS Research Reports CDMTCS-008, 1995
`https://researchspace.auckland.ac.nz/handle/2292/3517`