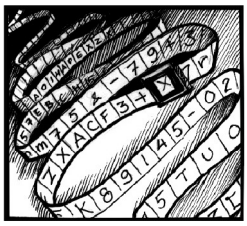

Mitä ei voida laskea?

Tuomas Korppi

Johdanto

Matemaatikkoja kuulee usein syytettävän siitä, että he olettavat, että mitä tahansa voidaan laskea. Perusteluna sille, että kaikkea ei voida laskea, tarjotaan usein rakkauden määrää tai jotain vastaavaa, joka on niin epämääräistä, että matemaattiset metodit eivät pure siihen.

Todellisuudessa matemaatikot eivät oleta, että mitä tahansa voidaan laskea. He nimittäin tietävät, että *matematiikan sisältä* löytyy asioita, jotka ovat eksaktisti määriteltyjä, mutta niin monimutkaisia, että mikään laskentamenetelmä ei tepsi niihin. Tässä kirjoitelmassa tutustumme pariin tällaiseen kysymykseen.

Korostan vielä, että esitämme tässä kirjoitelmassa kysymyksiä, joista voidaan todistaa, että niiden vastauksia ei edes periaatteessa voida laskea. Tässä siis laskemattomuus ei johdu siitä, että laskentamenetelmiä ei ole vielä keksitty.

Mitä tarkoitamme laskemisella?

Tarkoitamme laskennalla prosessia, jonka lähtökohta on *syöte*, jokin (jossain ennaltamäärätyssä äärellisessä aakkostossa annettu) äärellinen merkkijono, ja joka päättyy *tulokseen*, joka on samoin äärellinen merkkijono. Laskenta voi koostua useista välivaiheista, mutta oletamme, että laskennalla on jotkin säännöt, jotka määräävät yksikäsitteisesti kussakin kohdassa, kuinka laskentaa jatketaan. Tämä siis tarkoittaa, että säännöt eivät missään kohdassa anna laskijalle valinnanvaraa jatkon suhteen.

Laskettaessa esimerkiksi kynällä ja paperilla käytettävissä oleva aika ja paperin määrä määräävät, kuinka pitkä laskenta voi olla. Teoreettisessa laskennan käsitteessämme emme tee tällaista rajoitetta, vaan laskenta saa olla vaikka kuinka pitkä, kunhan se on äärellinen. Samoin syöte ja tulos saavat olla kuinka pitkiä tahansa, kunhan ne ovat äärellisiä.

Edellä puhuimme kynällä ja paperilla laskemisesta, mutta yleisemmin laskemme tietokoneella. Tämän johdosta kutsummekin niitä sääntöjä, joilla laskenta etenee, *tietokoneohjelmaksi*. Tässä siis hyväksymme tietokoneohjelmaksi minkä tahansa tavallisen tietokoneohjelman, joka saa aluksi yhden syötteen[1], prosessoi sitä ja palauttaa lopuksi laskennan tuloksen. Ainoa ero tavallisiin tietokoneohjelmiin on se, että oletamme tietokoneessa olevan muistia rajattomasti, eli niin paljon kuin on tarpeen. Laskenta voi myös kestää niin pitkään kuin on tarpeen, vaikka tarvittava aikamäärä olisikin epärealistisen pitkä.

Lukijalle kenties heräsi kysymys, että millä ohjelmointikielellä oletamme ohjelmamme olevan kirjoitettu. Täs-

[1] Tässä siis syöte on yksi merkkijono. Yhteen merkkijonoon voi koodata vaikka kuinka paljon tietoa, esimerkiksi useita lukuja vaikkapa puolipisteellä erotettuna

sä vastaus on: Sillä ei ole merkitystä. Käytännössä kaikki käytössä olevat ohjelmointikielet ovat yhtä vahvoja, eli niillä voidaan toteuttaa samat laskennat. Kun siis puhumme tietokoneohjelmasta, lukija voi vapaasti ajatella sen olevan kirjoitettu hänelle tutuimmalla kielellään, esimerkiksi C:llä, C++:lla tai Javalla. Myös kynällä ja paperilla (kunhan sitä on rajattomasti) voidaan teoriassa toteuttaa täsmälleen samat laskennat kuin ohjelmointikielillä – joskin se on käytännössä huomattavasti työläämpää.

Teemme vielä yhden lievennyksen edelläesitettyyn laskennan käsitteeseen. Sallimme tietokoneohjelmiksi myös ne, jotka eivät kaikilla syötteillä palauta tulosta, vaan jotka voivat joillain syötteillä ”jäädä jumiin”, eli joillain syötteillä laskenta jatkuu ikuisesti eikä tulosta anneta[2,3]. Jos ohjelma antaa tuloksen jollain syötteellä x, sanomme, että ohjelma *pysähtyy* syötteellä x.

Pysähtymisongelma

Valitaan aluksi ohjelmointikieli. Oletamme, että kaikki tässä luvussa mainitut tietokoneohjelmat on kirjoitettu tällä kielellä. Tutkitaan seuraavaa kysymystä:

On annettu tietokoneohjelma T ja syöte x. Pysähtyykö ohjelma T syötteellä x?

Meitä kiinnostaa se, voidaanko muodostaa tietokoneohjelma T_0, joka vastaa tähän kysymykseen kaikkien parien T, x osalta. Tällainen ohjelma siis saisi syötteenään parin T, x, ja palauttaisi merkkijonon ”Kyllä”, jos T pysähtyy syötteellä x ja merkkijonon ”Ei”, jos T ei pysähdy syötteellä x tai T ei ole kelvollinen (valitulla ohjelmointikielellä kirjoitettu) tietokoneohjelma.

Osoittautuu, että tällaista tietokoneohjelmaa T_0 ei voida muodostaa. Seuraavaksi todistamme kyseiseen väitteen. Käytämme pohjana Wikipediasta [2] löytyvää todistusta. (Voit skipata todistuksen, jos sen lukeminen tuntuu liian raskaalta.)

Tehdään vastaoletus: Tällainen tietokoneohjelma T_0 on olemassa. Muodostetaan T_0:aa muokkaamalla uusi tietokoneohjelma T_1, joka saa syötteenään merkkijonon s ja toimii seuraavasti:

- Jos T_0 palauttaa syöteparilla s, s vastauksen ”Ei”, T_1 palauttaa syötteellä s tuloksena merkkijonon ”ok”.
- Jos T_0 palauttaa syöteparilla s, s vastauksen ”Kyllä”, T_1 jää syötteellä s laskemaan ikuisesti.

Nyt kysymys kuuluu: Kuinka T_1 toimii, jos sille annetaan itsensä (eli T_1) syötteeksi?

- Jos T_1 palauttaa syötteellä T_1 merkkijonon ”ok”, T_0 palauttaa parilla T_1, T_1 vastauksen ”Ei”, eli T_1 jää jumiin syötteellä T_1. Kuitenkin oletimme, että T_1 pysähtyy syötteellä T_1. Ristiriita.
- Jos taas T_1 jää jumiin syötteellä T_1, ohjelma T_0 palauttaa syöteparilla T_1, T_1 vastauksen ”Kyllä”, eli T_1 pysähtyy syötteellä T_1. Kuitenkin oletimme, että T_1 jää jumiin syötteellä T_1. Ristiriita.

Siis kaikki mahdolliset vaihtoehdot johtavat ristiriitaan, eli vastaoletuksemme on väärä, ja tietokoneohjelmaa T_0 ei voida muodostaa.

Olemme siis nyt löytäneet ensimmäisen eksaktisti määritellyn kysymyksen, jonka vastausta ei voida (kaikissa tilanteissa) laskea: Pysähtyykö annettu tietokoneohjelma annetulla syötteellä?

Voidaan tosin muodostaa sellainen tietokoneohjelma T_0, joka saa syötteenään tietokoneohjelman T ja merkkijonon s, ja joka simuloi T:n laskemista syötteellä s. Kuitenkin, jos laskenta kestää kauan, ei missään vaiheessa laskentaa välttämättä ole mahdollista sanoa, että olemme laskeneet niin kauan, että ohjelma ei varmasti tule pysähtymään.

Luettelevat tietokoneohjelmat

Aiemmin tutkimme tietokoneohjelmia, jotka yleensä pysähtyivät. Seuraavaksi määrittelemme käsitteen *luetteleva tietokoneohjelma*, joka ei saa syötettä, vaan alkaa laskennan tyhjästä, eikä välttämättä pysähdy. Luetteleva tietokoneohjelma antaa kuitenkin laskennan edetessä tulosteita. Koska laskenta voi jatkua äärettömästi, se voi laskennan kuluessa antaa yhteensä äärettömän määrän tulosteita.

Laskennan edetessä luetteleva tietokoneohjelma voi käyttää yhä enemmän ja enemmän muistia, ja oletamme, että luettelevalla tietokoneohjelmalla on käytössään rajattomasti muistia niin, että ohjelman suoritus ei tyssää muistin loppumiseen[4].

Ne lukijat, jotka eivät tunne oloaan kotoisaksi tietokoneiden parissa, voivat yhä ajatella kynällä ja paperilla suoritettavaa laskentaa, joka jatkuu ja jatkuu, ja laskennan edetessä määrättyjä välituloksia kutsutaan tulosteiksi.

[2] Jokainen ohjelmointia harrastanut lukija lienee tehnyt vähintään kerran elämässään sellaisen ohjelmointivirheen, jonka johdosta ohjelma on jäänyt ikuiseen silmukkaan.

[3] Tällainen ikuisesti jatkuva laskenta voi vaatia laskennan edetessä yhä enemmän ja enemmän muistia. Oletamme, että tällaisella laskennalla on käytössään rajaton määrä muistia, niin, ettei se lopu.

[4] Lukija voi puolileikillisesti ajatella äärellisellä muistilla varustetun luettelevaa tietokoneohjelmaa suorittavan tietokoneen, joka osaa ilmoittaa muistin loppumisesta, ja koneen vieressä istuvan juoksupojan, joka käy aina tarpeen vaatiessa ostamassa lisää muistia ja asentaa sen koneeseen laskennan jatkamiseksi.

Totuus lukuteoriassa

Olkoon $(n, n+2)$ pari luonnollisia lukuja. Sanomme, että n ja $n+2$ ovat alkulukukaksoset, jos sekä n että $n+2$ ovat alkulukuja. On avoin ongelma, onko alkulukukaksosia äärellinen vai ääretön määrä. Jos kävisimme läpi kaikki luonnolliset luvut n ja testaisimme jokaisen kohdalla, ovatko n ja $n+2$ alkulukukaksosia, joutuisimme käymään läpi äärettömän monta lukua n, joten tällainen läpikäynti ei ole laskenta tarkoittamassamme mielessä[5]. Tällaisia äärettömästä määrästä luonnollisia lukuja puhuvia lauseita on muitakin, ja herää kysymys, olisiko mahdollista muodostaa jokin ääretöntä läpikäyntiä ovelampi laskentamenetelmä, jolla ratkaista kaikkien tällaisten lauseiden totuus. Esittelemme tässä luvussa tuloksen, joka sanoo, että tämä ei ole mahdollista.

Tulos, jonka aiomme esitellä, puhuu luonnollisia lukuja koskevista lauseista. Koska tässä meillä lauseet ovat *matematiikan tutkimuksen kohde*, eivät *matematiikan tutkimuksen väline*, tarvitsemme eksaktin määritelmän niille lauseille, josta tuloksemme puhuu.

Jatkon kannalta olennaista on ymmärtää, että tarkoitamme lukuteorian lauseilla lauseita, jotka puhuvat luonnollisista luvuista, ja joilla on tietty, tarkasti määrätty muoto. Muoto on sellainen, että voidaan laskea, onko jokin merkkijono tätä muotoa oleva lause. Lisäksi kyseistä muotoa olevia lauseita on ääretön määrä. Tässä on huomattava, että muoto on sellainen, että kyseistä muotoa oleva lause voi olla joko tosi tai epätosi; muoto määrää vain sen, että kyseessä on mielekäs lause, jolla on totuusarvo.

Annamme hiukan tarkemman luonnehdinnan (joskaan emme tarkkaa määritelmää). Sen voi halutessa sivuuttaa. Tarkkakin määritelmä on mahdollista antaa, mutta emme halua rasittaa lukijaa sen yksityiskohtien läpikäynnillä.

Lukuteorian lauseella tarkoitamme ”mielekästä”, äärellistä lausetta, joka saadaan muodostettua käyttäen merkkejä $(,), 0, 1, +, \times, =, \wedge$ (ja), $\neg$ (ei), $\forall$ (kaikilla) sekä rajatonta määrää muuttujasymboleja $x_0, x_1, \ldots$. Muuttujien ajatellaan saavan arvoikseen luonnollisia lukuja.

Esimerkkejä lukuteorian lauseista ovat

$$1 + 1 + 1 = 1 + 1,$$

joka on hyvinmuodostettu (joskin epätosi) lause, joka väittää, että kaksi on yhtäsuuri kuin kolme,

$$\forall x_0(x_0 = x_0 + 0),$$

joka väittää, että jos mihin tahansa luonnolliseen lukuun lisätään nolla, saadaan alkuperäinen luku,

$$\forall x_0 \neg(x_0 = x_0 + 1),$$

joka väittää, että mikään luonnollinen luku ei ole sellainen, että kun siihen lisätään yksi, saadaan alkuperäinen luku,

$$(0 = 0) \wedge (1 = 1),$$

joka väittää, että sekä nolla että yksi ovat yhtäsuuria itsensä kanssa,

$$\forall x_0 \neg(x_0 \times 0 = 0),$$

joka on epätosi lause, joka väittää, että mikään luonnollinen luku kerrottuna nollalla ei ole nolla, sekä

$$\forall x_0 \neg \forall x_1 \neg(x_1 = x_0 + 1),$$

joka väittää, että jokaiselle luonnolliselle luvulle x_0 on olemassa toinen luonnollinen luku x_1 siten, että $x_1 = x_0 + 1$. (Tässä kannattaa huomata, että ”Ei ole niin, että millään x ei päde…” tarkoittaa samaa kuin ”On olemassa x, jolle pätee…”.)

Myös väite, että alkulukukaksosia on ääretön määrä, on mahdollista kirjoittaa lukuteorian lauseena, joskin lauseesta tulisi melko pitkä.

Lukuteorian lauseen käsitteeseemme sisältyy myös se, että jokaiseen x_i:n esiintymään vaikuttaa kvanttori $\forall x_i$, eli

$$x_2 = x_2 + 0$$

ei ole lukuteorian lause tarkoittamassamme mielessä.

Nyt voidaan todistaa seuraavaa teoreemat:

Teoreema 1. *Ei voida muodostaa luettelevaa tietokoneohjelmaa, joka luettelee kaikki todet lukuteorian lauseet ja vain ne.*

Teoreema 2. *Ei voida muodostaa tietokoneohjelmaa, joka saadessaan toden lukuteorian lauseen syötteeksi antaa tuloksen ”kyllä” ja saadessaan epätoden lukuteorian lauseen syötteeksi antaa tuloksen ”ei”.*

Todistukset ovat vaikeita, ja ne löytyvät teoksesta Väänänen [1]. (Tulosten saamiseksi tarvitsemme Määritelmän 13.1, Lauseen 12.8 ja Lauseen 12.3.)

Olemme nyt löytäneet toisen hyvinmuotoillun ongelman, jonka vastausta ei voida (kaikissa tapauksissa) laskea, nimittäin matemaattisten väitteiden totuuden. Sanon edellä ”matemaattisten väitteiden”, mutta itse asiassa olemme todenneet, että laskemattomuus koskee jo hyvin rajallista muotoa olevia matemaattisia väitteitä.

[5] Useissa tapauksissa tällaisten kaikista luonnollisista luvuista puhuvien lauseiden totuus voidaan ratkaista äärellisellä todistuksella, ja useimmat uskonevatkin, että alkulukukaksoskysymys saadaan ennemmin tai myöhemmin ratkaistua tällä tavoin.

Seuraus matematiikan harjoittamiselle

Oletetaan, että olemme saaneet matemaattisen todistuksen käsitteen niin hyvin määritellyksi, että voidaan laskea, onko annettu merkkijono todistus. Ts. voidaan muodostaa tietokoneohjelma T_0, joka saa syötteenään parin P, R, ja palauttaa merkkijonon "Ok", jos merkkijono P on lauseen R todistus ja merkkijonon "Ei-Ok", jos näin ei ole. Tämä oletus on realistinen, koska näin voidaan tehdä[6]. Käytännössä vain todistusten saaminen tällaiseen eksaktiin muotoon on äärimmäisen työlästä, joten matemaatikot eivät koskaan esitä todistuksia tässä muodossa.

Nyt voidaan muodostaa luetteleva tietokoneohjelma T_1, joka toimii seuraavasti: Se käy läpi kaikki parit P, R, ja aina kohdatessaan parin P, R, missä P on R:n hyväksyttävä todistus, se tulostaa R:n[7]. Ohjelma voidaan tehdä mm. niin, että ensin käydään läpi kaikki kahden merkin mittaiset parit P, R, sitten kaikki kolmen merkin mittaiset parit P, R, sitten kaikki neljän merkin mittaiset ja niin edelleen.

Edelleen T_1:n avulla voidaan muodostaa luetteleva tietokoneohjelma T_2, joka poimii T_1:n tulosteista kaikki lukuteorian lauseet. T_2 siis luettelee kaikki lukuteorian lauseet, joille on olemassa hyväksyttävä todistus.

Edellisessä luvussa totesimme, että ei voida muodostaa luettelevaa tietokoneohjelmaa, joka luettelee kaikki todet lukuteorian lauseet. Koska T_2 luettelee kaikki lukuteorian lauseet, joille on hyväksyttävä todistus, teemme seuraavan johtopäätöksen: On olemassa tosia lukuteorian lauseita, joita ei voida todistaa nykyisen todistuskäsityksen mukaisesti. Sama pätee mille tahansa todistuskäsitykselle, jossa todistuksen pätevyys voidaan laskea.

Pähkinöitä

1. Osoita, että ei voida muodostaa tietokoneohjelmaa T_0 siten, että T_0 saa syötteenään tietokoneohjelman T, ja palauttaa merkkijonon "Tosi", jos T pysähtyy kaikilla syötteillä, ja merkkijonon "Epätosi", jos on vähintään yksi syöte, jolla T ei pysähdy. (Tässä tehtävässä valitaan ohjelmointikieli ja oletetaan, että kaikki tehtävässä mainitut ohjelmat on kirjoitettu tällä kielellä.)

2. Edellisen kappaleen lopussa annoimme argumentin sille, että on olemassa vähintään yksi tosi lukuteorian lause, jota ei voida todistaa nykyisen todistuskäsityksen mukaisesti. Osoita käyttäen tässä kirjoitelmassa mainittuja tuloksia, että tällaisia lukuteorian lauseita on ääretön määrä.

3. Olkoon T luetteleva tietokoneohjelma. Osoita, että voidaan muodostaa tietokoneohjelma, joka antaa vastauksen "Kyllä" täsmälleen niillä syötteillä, jotka T luettelee (ja muilla syötteillä jää jumiin).

4. Olkoon T tietokoneohjelma, joka joillain (ennaltamäärätyssä äärellisessä aakkostossa annetuilla) syötteillä antaa tuloksen "Kyllä", joillain (samassa aakkostossa annetuilla) syötteillä antaa tuloksen "Ei" ja jää jumiin lopuilla (samassa aakkostossa annetuilla) syötteillä. Osoita, että voidaan muodostaa luetteleva tietokoneohjelma, joka luettelee täsmälleen ne syötteet, joilla T antaa tuloksen "Kyllä".

Viitteet

[1] Väänänen, Jouko, *Matemaattinen logiikka*, luentomoniste, `https://wiki.helsinki.fi/download/attachments/86367413/luentoteksti2010.pdf?version=1&modificationDate=1345108822439`

[2] Wikipedia, *Halting problem*, `http://en.wikipedia.org/wiki/Halting_problem`, haettu 25.12.2012.

[6] Siihen, kuinka tämä tehdään, emme tässä mene, mutta taikasanat ovat 1. kertaluvun predikaattilogiikka + ZFC.

[7] Vaikka tällainen tietokoneohjelma voidaan teoriassa tehdä, käytännössä se on niin hidas, että sen käyttäminen todistusten etsimiseen on aivan toivotonta.