Tietojenkäsittelytiede **23** Kesäkuu 2005 sivut 17–25
Toimittaja: Antti Valmari

Oppimisen teoriasta ja teorian oppimisesta

Tapio Elomaa
Tampereen teknillinen yliopisto
Ohjelmistotekniikan laitos
elomaa@cs.tut.fi

Virkaanastujaisesitelmä 23. 3. 2005, Tapio Elomaa on nimitetty Tampereen teknillisen yliopiston ohjelmistotekniikan professorin virkaan 1. 8. 2003 lukien.

1 Gödel ja oppimisen tehostaminen

Gödel-palkinto myönnetään vuosittain parhaalle viime vuosien aikana julkaistulle teoreettisen tietojenkäsittelytieteen artikkelille. Itävaltalaisen matemaattisen logiikan tutkijan Kurt Gödelin (1906–1978) tunnetuimmat tulokset ovat ensimmäisen kertaluvun logiikan *täydellisyyslause* vuodelta 1930 sekä *epätäydellisyyslause* vuodelta 1931. Gödelin väitöskirjassaan [9] esittämän tuloksen mukaan ensimmäisen kertaluvun logiikan seurauslauseilla on aina olemassa äärellinen todistus, mutta vasta J. Alan Robinsonin resoluutioalgoritmi vuodelta 1965 antoi käytännöllisen menetelmän todistuksen löytämiseksi [23].

Gödelin kuuluisin tulos on epätäydellisyyslause [10]: on olemassa tosia lukuteorian (tai minkä tahansa aksioomasysteemin) väittämiä, joita ei kuitenkaan voi todistaa aksioomista lähtien. Gödelin tuloksen seurauksena emme voi koskaan todistaa kaikkia matematiikan lauseita mistään aksioomista lähtien. Epätäydellisyyslauseen on myös katsottu kiistävän tekoälyn mahdollisuuden, koska tietokoneet sinänsä ovat formaaleja systeemejä ja siten niillä on olemassa Gödel-lauseensa, jota ei voida todistaa, kun taas inhimillinen ajattelu ei kärsi vastaavista rajoitteista [19, 22]. Tämä päättely ei kuitenkaan ole aukotonta, eikä osoita tekoälyä mahdottomaksi.

Vuonna 2003 Gödel-palkinto myönnettiin yhdysvaltalaisille tutkijoille Yoav Freund ja Robert E. Schapire heidän *oppimisen tehostamista* (boosting) koskevasta artikkelistaan [8]. Tämä on ensimmäinen kerta kun koneoppimisen alaan kuuluva julkaisu on saanut kyseisen tunnustuksen. Ohjatussa oppimisessa oppimisalgoritmi saa luokiteltuja *esimerkkejä* jonkin tuntemattoman, mutta kiinteän, todennäköisyysjakauman perusteella ja sen tehtävänä on oppia luokittelemaan saman jakauman mukaan vedettyjä uusia tapauksia. Oppimisen tehostamisessa kyse on ns. *heikkojen oppijoiden*, joista jonkin luokittelutarkkuus minkä tahansa esimerkkien todennäköisyysjakauman mukaan on vain hieman satunnaista arvausta parempi, yhdistämisestä siten, että muodostuva yhteisluokittelija saavuttaa mielivaltaisen

suuren luokittelutarkkuuden.

Palkittu artikkeli esitteli AdaBoost-algoritmin, elegantin ja yksinkertaisen menetelmän oppimisen tehostamiseen. Palkintoperusteissa arveltiin algoritmin olevan pysyvä kontribuutio tieteelle jopa ohi tietojenkäsittelyn. AdaBoost avasi uusia mahdollisuuksia datan analysoinnissa ja käynnisti paljon uutta tutkimusta tilastotieteen, tekoälyn, empiirisen koneoppimisen ja tietämyksen muodostamisen piirissä. AdaBoostin käytännön menestys on hyvä osoitus siitä, kuinka teoreettinen tietojenkäsittelytiede voi tuoda täysin uusia vahvoja ideoita, joilla on merkittäviä suoria vaikutuksia jo runsaasti tutkituilla aloilla kuten data-analyysi. Vaikkakin oppimisen tehostamisen perusoletus heikkojen oppijoiden luokan olemassaolosta on puhtaan teoreettinen ajatus — se ei nouse esiin empiirisistä havainnoista — on tämä ajatusmalli tarjonnut uuden viitekehikon olemassa olevien menetelmien analyysiin ja suunnittelumallin uusille oppimisalgoritmeille.

Kurt Gödel on yksi tunnetuista viime vuosisadan aikalaismatemaatikoista, jotka loivat pohjan tietojenkäsittelyn teorialle jo ennen ensimmäisten tietokoneiden rakentamista. Kuuluisimmat näistä lienevät tietokoneiden rakentamiseen suoraan vaikuttaneet englantilainen Alan Turing (1912–1954) ja unkarilaissyntyinen John von Neumann (1903–1957). Turing määritteli abstraktin laskentalaitteen, *Turingin koneen*, joka formalisoi algoritmin käsitteen matemaattisesti [29]. Von Neumann puolestaan on, monien muiden tieteellisten saavutustensa lisäksi, nykyaikaisen tietokoneen arkkitehtuurin isä. Turing ja von Neumann tapasivat ollessaan molemmat 1936–1938 Princetonin *Institute for Advanced Studyssa* (IAS) ja epäillään Turingin teoreettisten ajatusten vaikuttaneen von Neumannin tietojenkäsittelyä koskevaan ajatteluun. Myös Gödel työskenteli elämänsä loppuvuodet IAS:ssa solmien mm. ystävyyssuhteen von Neumannin ja Albert Einsteinin (1879–1955) kanssa. Teoreettisen tietojenkäsittelyn kannalta merkittäviä aikalaismatemaatikkoja ovat lisäksi mm. informaatioteorian luonut Claude Shannon (1916–2001) [7, 24, 25] ja todennäköisyyslaskennan aksiomaattisen järjestelmän kehittänyt venäläinen Andrei Kolmogorov (1903–1987) [15, 16, 18].

2 Laskennallisten ongelmien vaikeus

Vasta suhteellisen äskettäin on paljastunut Kurt Gödelin mielenkiinto teoreettisen tietojenkäsittelytieteen tunnetuimpaan avoimeen ongelmaan. Tämän ns. ***P* vs. *NP* -*ongelman*** hän esitti ensimmäisen kerran jo vuonna 1956, mutta vasta 1970-luvun alkuvuosina Stephen A. Cook [5] ja Richard Karp [14] muotoilivat ongelman sen nykyiseen asuunsa. Luokka **P** on polynomisessa ajassa deterministisesti toimivalla Turingin koneella ratkaistavien laskennallisten ongelmien joukko. Se vastaa likimain (nykyaikaisella) tietokoneella käytännössä ratkaistavissa olevien ongelmien joukkoa. Luokka **NP** puolestaan tarkoittaa epädeterministisesti toimivin Turingin konein polynomisessa ajassa ratkaistavien ongelmien joukkoa. Epädeterministisessä laskennassa voidaan ajatella laskentalaitteen kykenevän arvaamaan oikean ratkaisun ja tehokkaasti tarkastamaan arvauksen oikeellisuuden. Tällaiseen eivät tietokoneet tietenkään pysty, vaan ne on pääsääntöisesti ohjelmoitava deterministisesti oikean tuloksen saamiseksi.

Luokka **P** sisältyy luokkaan **NP**, mutta ei ole osattu todistaa, että **NP** olisi aidosti **P**:tä laajempi luokka. Jos osoittau-

tuu että **P** = **NP**, niin tietokonein ratkaistavien ongelmien joukko on paljon laajempi kuin mitä yleisesti oletetaan. Tämä ei kuitenkaan olisi pelkästään positiivinen tulos, koska esimerkiksi tietoliikenteessä käytetyt salausmenetelmät perustuvat joidenkin laskennallisten ongelmien oletettuun vaativuuteen,[1] jonka takia niiden murtaminen on käytännössä mahdotonta nykyaikaisella tietokoneella. Nevanlinna- ja Gödel-palkinnon työstään sittemmin saanut Peter Shor tosin esitti noin kymmenen vuotta sitten miten epädeterministisesti toimivalla *kvanttitietokoneella* voitaisiin jakaa tekijöihin suuriakin kokonaislukuja tehokkaasti, joka murtaisi yleisesti käytetyn salausmenetelmän [28]. Ainakin toistaiseksi yleiskäyttöiset kvanttitietokoneet ovat kuitenkin pelkkää tieteisfantasiaa.

Vuonna 1938 Gödel [11] todisti, ettei äärettömien joukkojen ns. Cantorin *kontinuumihypoteesia* — onko olemassa kardinaliteettia, joka on luonnollisten lukujen ja reaalilukujen kardinaliteetin välissä? — voi osoittaa paikkaansa pitämättömäksi joukko-opin Zermelo-Fraenkel -aksioomista lähtien (myöhemmin Paul Cohen on osoittanut, ettei sitä myöskään voi todistaa samoista aksioomista lähtien [4]). Koska **P** vs. **NP** -ongelma on runsaasta tutkimuksesta huolimatta säilynyt avoimena ongelmana jo pitkälti yli kolmekymmentä vuotta, on viimeaikoina yhä useammin epäilty sen olevan kontinuumihypoteesin kaltainen aksioomista riippumaton ominaisuus. Matematiikan historia on kuitenkin kerta toisensa jälkeen osoittanut, että vaikeiden matemaattisten kysymysten ratkaiseminen usein vaatii kymmenien jopa satojen vuosien tietämyksen kertymisen ennen kuin ratkaisu löytyy. Esimerkiksi kymmenen vuotta sitten (Princetonissa toimiva) brittimatemaatikko Andrew Wiles todisti yli kolmesataa vuotta matemaatikkoja kiusanneen *Fermat'n suuren lauseen* ($a^n + b^n \neq c^n$, kaikilla kokonaisluvuilla a, b, c ja $n > 2$) käyttäen hyväkseen tuloksia (mm. topologian ja lukuteorian yhteyksiä), joita Pierre de Fermat'n (1601–1665) aikoina ei tunnettu [32].

Laskennan vaativuusteoriassakin saavutettiin ratkaisu vuosisatoja avoimena pysyneeseen ongelmaan, kun intialaiset tutkijat Agrawal, Kayal ja Saxena vuonna 2002 löysivät polynomiaikaisen deterministisen algoritmin alkulukutestaukseen[2] [1]. Alkulukuhan ei ole jaollinen millään muilla luvuilla kuin yhdellä ja itsellään. Myös tämä tulos perustui vuosikymmenten kertyneeseen tietämykseen alkulukutestauksesta. Näiden esimerkkien valossa lienee siis ennenaikaista julistaa **P** vs. **NP** -ongelmaa aksioomista riippumattomaksi vielä sen suhteellisen lyhyen historian perusteella. Ongelman merkittävyydestä kuitenkin kertoo se, että se on sisällytetty niiden *Clay Mathematics Institute*n seitsemän matematiikan avoimen *Millenium*-ongelman joukkoon, joista kustakin on luvassa miljoonan dollarin palkkio ratkaisijalleen.

3 Tekoäly ja tietokonepelit

Alan Turing pohti koneälyn mahdollisuuksia kun ensimmäisiä tietokoneita vasta rakennettiin ja hahmotteli samalla pit-

[1] Tarkkaan ottaen julkisen avaimen salakirjoituksen taustalla olevan ongelman, kokonaislukujen tekijöihin jakamisen, ei tiedetä olevan luokkaan **NP** kuuluva ongelma, vaan sen oletetaan olevan luokan **P** ongelma, joskaan tätä ei vielä toistaiseksi ole osattu todistaa [12].

[2] Yhdistettyjen kokonaislukujen tunnistaminen on alkulukujen tunnistamiselle käänteinen ongelma ja siis myös luokassa **P**, mutta luvun tunnistaminen yhdistetyksi ei vielä anna menetelmää sen jakamiseksi tekijöihinsä.

kälti koneoppimisen myöhempiä linjoja [30]. Tunnetuin hänen tekoälyyn liittyvistä kontribuutioistaan lienee ns. *Turingin testi*, jossa tutkitaan pystyykö inhimillinen tarkkailija erottamaan tietokoneen ja ihmisen toisistaan pelkästään niiden kanssa käymänsä kirjallisen vuorovaikutuksen perusteella. Turingin testistä järjestetään vuosittain kilpailu, jossa tietokoneohjelmat pyrkivät vakuuttamaan arvioijat omasta inhimillisyydestään. Mainittakoon, että varmimmin ei-inhimillisiksi keskustelukumppaneiksi näissä ajaltaan rajoitetuissa keskusteluissa yleensä tulevat tulkituiksi ihmisasiantuntijat kuten lääkärit, joilla on näennäisesti epäinhimillinen määrä yksityiskohtaista tietoa omasta erityisalueestaan. Turingin testiä ja monia muita tekoälyyn liittyviä asioita on popularisoitu hauskasti *Pulitzer*-palkitussa kirjassa [13].

Claude Shannonin ja Alan Turingin mielenkiinto yhtyi tekoälyn ja koneoppimisen alueella. Shannon mm. hahmotteli tarkasti modernien peliohjelmien perusteet jo vuonna 1950 [26] ja Turing puolestaan kirjoitti ensimmäisen täydellisen shakkiohjelman vuonna 1951 [3]. Käsisimulaatiossa Turingin ohjelma kuitenkin hävisi kehnohkolle shakin pelaajalle. Shakkiohjelmista tuli Turingin vastoinkäymisistä huolimatta yleistä viihdykettä viimeistään mikrotietokoneiden yleistyttyä 1980-luvulla. Jonkinlainen kulminaatiopiste tietokoneshakissa saavutettiin vuonna 1997 kun IBM:n *Deep Blue* -laitteisto voitti hallitsevan maailmanmestarin Garri Kasparovin haasteturnauksessa. Tietokoneiden osaaminen shakissa lisääntyy tasaisesti, mutta toisissa lautapeleissä (esimerkiksi Go) on tietokoneiden kyky vielä toistaiseksi auttamattomasti jäljessä ihmisten kyvyistä.

Tietokoneiden yleistyessä kotona on niiden viihdekäyttö kasvanut räjähdysmäisesti. Samalla muiden yleisten elektronisten laitteiden (matkapuhelinten) laskentakapasiteetin lisääntyminen on johtanut siihen, että tietokonepelit ovat viime vuosina olleet nopeasti kasvava ohjelmistotuotteiden ala. Modernit tietokonepelit ovat edenneet kauas lautapelien maailmasta, ne luovat monitahoisia virtuaalimaailmoja ja perustuvat usein reaaliaikaiseen toimintaseikkailuun. Viihteellisyydestään huolimatta tietokonepelit yhdistävät monia teoreettisen tietojenkäsittelyn alueita: Reaaliajassa suoritettavat tietokonegrafiikan algoritmit edellyttävät edistyneiden tietorakenteiden ylläpitoa ja innovatiivisia laskennallisia ratkaisuja. Tietokonepelien tekoäly on ratkaisevassa asemassa niiden pelattavuuden kannalta. Myös koneoppimisen algoritmeilla on nouseva osuutensa pelien kehittämisessä ja toteuttamisessa.

Suomalaiset ohjelmistoalan yritykset ovat edelläkävijöitä matkapuhelimiin tarkoitettujen pelien suunnittelijoina ja saavuttaneet maailmanlaajuista menestystä myös tietokoneilla pelattavien pelien kehittäjinä. TTY:n ohjelmistotekniikan laitos on jo aloittanut panostuksen tietokonepeleihin liittyvässä opetuksessa ja tutkimuksessa ja tutkimusalan voi nähdä tulevaisuudessa edelleen kasvavan, kunhan vain opiskelijoiden teoreettiset perustaidot kyetään turvaamaan.

4 Teorian opettamisen vaikeus ja tärkeys

Vaikka opiskelijat niin TTY:lla kuin muissakin yliopistoissa ovat tietokonepelien ehdottomia osaajia, ei heillä useinkaan ole motivaatiota teoreettisen tausta-aineksen omaksumiseen. Mistä tämä ristiriita voi johtua peliohjelmistoja valmistavien yritysten ollessa suosikkityönantajia opiske-

lijoiden keskuudessa? Ilmeiseltä vaikuttava selitys on virheellinen käsitys tietotekniikan ammattilaisen tarvitsemasta osaamisesta. Valitettavasti tämä virhekäsitys ei vaivaa yksinomaan teekkareita, vaan on yleinen myös alan yrityksissä toimivien ammattilaisten keskuudessa. Vaikeiden algoritmisten ongelmien ratkaiseminen edellyttää kykyä formalisoida ja ratkaista ongelma abstraktilla tasolla sekä tarvittavan tausta-aineksen tuntemusta. Tämä teorian ja käytännön vastakkainasettelu ei suinkaan ole uusi, vaan on kiusannut tietotekniikan yliopistokoulutusta kautta sen historian.

Edeltäjäni, emeritusprofessori Reino Kurki-Suonio käynnisti suomalaisen tietojenkäsittelyopin koulutuksen naapuriyliopistossamme jo kun nyt 40-vuotista taivaltaan juhlivaa TTY:a vasta perustettiin. Suomessa oltiin uuden oppiaineen kanssa eturintamassa maailman mittakaavassakin, sillä ensimmäinen tietojenkäsittelytieteen laitos perustettiin Yhdysvalloissa *Purduen* yliopistoon vuonna 1962.

Tampereella ensin Kurki-Suonion ja pian sen jälkeen Helsingissä emeritusprofessori Martti Tienarin johdolla käynnistetty tietojenkäsittelyopin opetus nojasi alusta alkaen vahvasti alan emämaasta Yhdysvalloista haettuun oppiin sekä käytännössä hankittuun kokemukseen. Kurki-Suonion ja Tienarin kuten monen muunkin kokemus oli hankittu Kaapelitehtaalla akateemikko Olli Lehdon johdolla.

Virkaanastujaisesitelmässään lähes neljäkymmentä vuotta sitten professori Kurki-Suonio pohti nuoren oppialansa teoreettisen ja käytännöllisen oppimateriaalin suhdetta todeten mm. [17]

“Kun … monet muut oppiaineet kaipaavat tietojenkäsittelyoppia tukiaineekseen, on oppiaineella suuri vaara jäädä pintapuolisen tietojenkäsittelytekniikan opetukseksi tai kokoelmaksi irrallisia, erilaisiin sovelluksiin käyttökelpoisia menetelmiä.”

Tämä huolenaihe ei ole hälvennyt näihin päiviin tultaessakaan. Kaksi vuotta sitten pääjohtaja Jorma Ollila piti Nokia Oyj:n Säätiön juhlatilaisuudessa puheen, jossa totesi mm., että [21]

“… on nykyisen kilpailurytmin kiihkeys tuonut mukanaan kärsimättömyyden, mikä saattaa suosia pintapuolisiin tietoihin perustuvaa riuskan suoraviivaista toimintaa. Tällainen ilmapiiri demotivoi paneutumista usein vaikeastikin omaksuttavien teoreettisten perustietojen ja -taitojen hankintaan ja ylikorostaa sinänsä tarpeellista käytännön läheisyyttä ja työelämään osallistumista opintojen kaikissa vaiheissa.

On todellinen harhaluulo, että puutteellista perusosaamista voitaisiin kompensoida paneutumisella käytännönläheisiin kysymyksiin. Merkittäviin tuloksiin pyrittäessä käytännön läheisyys tuo suuresti lisäosaamistarvetta — ei suinkaan vähennä vaadittavaa teoreettisen osaamisen välttämättömyyttä.

… Ei voi liikaa korostaa sitä, että hyvien teoreettisten perustietojen hankkiminen ja niiden jatkuva huipputasolla pitäminen ovat keskeisen tärkeitä sekä yksilön että yrityksen kannalta — mahdotonta sanoa kumman kannalta tärkeämpää erityisesti nykyisessä murrosvaiheessa.”

Teoreettisen taustatietämyksen keskeinen asema tietotekniikan ammattilaisen työkalupakissa on siis nähty niin kauan kuin Suomessa on yliopistotasoista alan koulutusta tarjottu. Onko nykypäivänä tarvittava teoreettinen aines samaa kuin neljäkymmentä vuotta sitten? Tampereen yliopistossa tietojenkäsittelyopin cum laude approbatur -opetukseen sisällytettiin Boolen algebraa ja muuta diskreettiä matematiikkaa pakolliseen ainekseen Kurki-Suonion *Carnegie Institute of Technolo-*

gyssa (sittemmin *Carnegie Mellon University*) saamien kokemusten perusteella [17]. Sama logiikkaan, diskreettiin matematiikkaan ja kombinatoriikkaan keskittyvä aines onkin säilynyt tietotekniikan taustamateriaalina ja vaadittuna matematiikan aineksena useissa maamme yliopistoissa aina näihin päiviin asti. Tänä päivänä uudet suuntaukset tietojenkäsittelyssä kuitenkin haastavat tämän pohjan yksinoikeuden.

5 Tietotekniikan matematiikan taso ja aines

Matematiikka on edelleenkin tietotekniikan keskeinen tausta-aine, jonka osaamiseen joudutaan nojautumaan teoreettisen aineksen opetuksessa. Valitettavasti on vain niin, että käytännön opetustyössä yliopistomaailmassa koulumatematiikan tason yleinen romahtaminen on jo pitkään ollut ilmiselvää. Kuitenkin suomalaisen koulutusjärjestelmän menestys erityisesti matematiikan alalla on OECD:n Pisa-tutkimuksissa useaan kertaan todettu maailman huippuluokkaan kuuluvaksi. Helsingin yliopiston matematiikan professori Kari Astala (ja 205 muuta yliopistojen ja ammattikorkeakoulujen matematiikan opettajaa) nosti tämän ristiriidan esiin äskettäisessä kirjoituksessaan todeten, että Pisa-tutkimus mittaa matematiikan lukutaitoa, eikä niinkään yliopisto-opintojen kannalta keskeistä osaamista [2]. Koska esimerkiksi tietotekniikan opetettava aines ei yliopistoissa suinkaan yksinkertaistu ja sen vaatima matematiikan hallinta vain kasvaa, johtaa koulumatematiikan tason heikkeneminen valitettavasti siihen, että yliopistoille jää entistä suurempi vastuu ja tehtävä opiskelijoiden perustietojen paikkaamisessa. Kuten Astala kirjoituksessaan huomauttaa [2], tämä ongelma ei kohdistu yksinomaan teknillisten alojen opiskelijoihin, vaan koskee myös mm. biologian opiskelijoita.

Tekoäly ja koneoppiminen ovat esimerkkejä nousevasta tietojenkäsittelytrendistä, jossa logiikka ja kombinatoriikka eivät enää ole yhtä keskeisessä asemassa kuin perinteisessä laskennallisten ongelmien irrallaan ulkomaailmasta tapahtuvassa ratkomisessa. Toki tekoälytutkimuksen alkuaikojen menestykset saavutettiin nimenomaan logiikkaan perustuvan tietämyksen esityksen ja päättelyn turvin, mutta pian havaittiin, ettei tämä lähestyminen riitä siirryttäessä esimerkkiongelmista reaalimaailman tehtäviin. Eksakti ja täydellinen looginen päättely ei ole mahdollista suuressa mittakaavassa, vaan todellisissa ympäristöissä toimivat agentit joutuvat sietämään epävarmuutta. Parhaaksi kuvausformalismiksi näihin tilanteisiin on nähty todennäköisyyslaskentaan perustuva tietämyksen esittäminen ja päättely. Erityisen hyödylliseksi on osoittautunut Bayesilainen päättely. Yhä lisääntyvä luonnollista dataa käyttävä tietojenkäsittely on vaatinut mm. informaatioteoreettisten ja tilastollisten menetelmien laajenevaa käyttöä.

Myös koneoppimisen teoriassa tärkein matematiikan alue on todennäköisyyslaskenta, mutta muitakin matematiikan osa-alueita sovelletaan. Oppimisen tehostamisen lisäksi tuore koneoppimisen piirissä kehittynyt menetelmä on *tukivektorikoneet* (support vector machines) [6, 27]. Tukivektorikone kuvaa annetun aineiston epälineaarisesti moniulotteiseen *piirreavaruuteen*, jossa aineiston luokittelu on mahdollista yksinkertaisella lineaarisella koneella. Moniulotteiseen piirreavaruuteen siirtyminen kuitenkin johtaisi laskennalliseen vaativuuteen, jonka välttämiseksi kuvaus tapahtuu käyttäen *ydinfunktiota* (kernel function). Se mahdollis-

taa tarvittavan *sisätulon* tehokkaan laskemisen alkuperäisessä tapausavaruudessa. Tukivektorikoneiden teoria nojaa vahvasti lineaarialgebran ja matemaattisen optimoinnin tutkimustraditioon yhdistyneenä viime vuosisadalla kehittyneeseen lineaarisen luokittelun teoriaan.

Yleisemmin algoritmien suunnittelussa ja analyysissä arkipäiväiseksi tekniikaksi ovat nousseet *satunnaisalgoritmit*, jotka "kolikkoa heittäen" toimivat epädeterministisesti mutta kuitenkin suurella todennäköisyydellä halutulla tavalla [20]. Toki niillä on juurensa algoritmien kehittämisen alkuvuosissa saakka. *Las Vegas -algoritmi* tuottaa aina oikean vastauksen, mutta sen vaatima laskenta-aika voi vaihdella, kun taas *Monte Carlo -algoritmi* aina toimii tehokkaasti, mutta voi joskus antaa väärän vastauksen. Esimerkiksi edellä mainittu alkulukutestaus, johon äskettäin löydettiin polynomiaikainen deterministinen algoritmi, hoitunee vielä tänäkin päivänä tehokkaimmin satunnaisalgoritmilla. Itse asiassa Agrawal ja kumppanit lainasivat oman algoritminsa kehittämiseen todistustekniikoita, joita ensin oltiin sovellettu polynomiaikaisten satunnaisalgoritmien laatimiseen. Yksi tällainen tulos on yksinkertainen versio *Fermat'n pienestä lauseesta* ($n^p \equiv n \pmod{p}$, kun n on kokonaisluku ja p on alkuluku).

Kaikkiaan satunnaisalgoritmien teoria perustuu tietotekniikan perinteistä matematiikkataustaa laajempaan ainekseen. Keskeisessä asemassa tässäkin on todennäköisyyslaskenta, mutta myös mm. algebralliset tekniikat ovat käytössä. Toinen algoritmiteorian uudehko tutkimuskenttä on *approksimaatioalgoritmit* [31]. Kun laskennallisesti vaativassa tehtävässä ei voida tehokkaasti ratkaista oikeaa tulosta, voi edelleenkin olla mahdollista löytää ratkaisu, joka varmasti on vain hieman oikeasta tuloksesta poikkeava. Likimääräisten ratkaisujen etsimisessä keskeinen matemaattinen aines on taas todennäköisyyslaskenta ja mm. optimointiteoria.

Tietojenkäsittelyn teoreettinen aines on siis mitä tärkeintä osaamista ja pääomaa tietotekniikan ammattilaiselle. Sen modernit menetelmät edellyttävät laajaa matemaattista osaamista. Diskreetin matematiikan ja logiikan merkitystä ei voi kiistää ohjelmien laatimisen ja toimintalogiikan varmistamisen välineinä. Kun Freundin ja Schapiren koneoppimisen teoriaan lukeutuva — joskin vahvasti käytäntöön suuntautunut — artikkeli on voittanut Gödel-palkinnon, lienee aika tunnustaa myös esimerkiksi todennäköisyyslaskennan entisestään kasvanut merkitys ohjelmien ulkoisen toiminnan tarkasteluvälineenä.

Kiitokset

Kiitän artikkelin toimittajaa Antti Valmaria tekstin huolellisesta läpikäymisestä ja monista hyödyllisistä korjausehdotuksista.

Viitteet

[1] Manindra Agrawal, Neeraj Kayal, Nitin Saxena (2002). PRIMES is in P. Indian Institute of Technology, Kanpur, India.

[2] Kari Astala (2005). Matematiikan taidot ovat heikentyneet. *Helsingin Sanomat*, mielipidekirjoitus, 17. 2. 2005.

[3] M. Audrey Bates, Bertram V. Bowden, Christopher Strachey, Alan Turing (1953). Digital computers applied to games. Ss. 286–310 teoksessa Bowden, B. V. (toim.), *Faster than Thought*. Pitman.

[4] Paul J. Cohen (1963). The independence of the Continuum Hypothesis.

Proceedings of the National Academy of Sciences of U.S.A. **50**, 1143–1148.

[5] Stephen A. Cook (1971). The complexity of theorem-proving procedures. *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing* (ss. 151–158). ACM Press.

[6] Corinna Cortes, Vladimir Vapnik (1995). Support-vector networks. *Machine Learning* **20**, 273–297.

[7] Thomas M. Cover, Joy A. Thomas (1991). *Elements of Information Theory*. John Wiley & Sons.

[8] Yoav Freund, Robert E. Schapire (1997). A decision theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* **55**, 119–139.

[9] Kurt Gödel (1930). Über die Vollständigkeit des Logikkalküls. Dissertation. Universität Wien.

[10] Kurt Gödel (1931). Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Monatshefte für Mathematik und Physik* **38**, 173–198.

[11] Kurt Gödel (1940). *The Consistency of the Axiom of Choice and the Generalized Continuum Hypothesis with the Axioms of Set Theory*. Princeton University Press.

[12] Mika Hirvensalo (2004). Matemaattisen ajattelun luonne, **P** vs **NP**. *Dimensio* 5/2004, 16–20.

[13] Douglas Hofstadter (1979). *Gödel, Escher, Bach: an Eternal Golden Braid*. Basic Books.

[14] Richard M. Karp (1972). Reducibility among combinatorial problems. Ss. 85–103 teoksessa Miller, R. E., Thatcher, J. W. (toim.), *Complexity of Computer Communications*. Plenum Press.

[15] Andrei N. Kolmogorov (1933). *Grundbegriffe der Wahrscheinlichkeitsrechnung*. Springer.

[16] Andrei N. Kolmogorov (1965). Three approaches to the quantitative definition of information. *Problems in Information Transmission* **1**, 1–7.

[17] Reino Kurki-Suonio (1993). Tietojenkäsittelyopin korkeakouluopetuksen käynnistyminen. Ss. 24–47 teoksessa Tienari, M. (toim.) *Tietotekniikan alkuvuodet Suomessa*. Gummerus.

[18] Ming Li, Paul Vitányi (1997). *An Introduction to Kolmogorov Complexity and Its Applications*. Second Ed. Springer Verlag.

[19] John R. Lucas (1961). Minds, machines, and Gödel. *Philosophy* **36**, 112–127.

[20] Rajeev Motwani, Prabhakar Raghavan (1995). *Randomized Algorithms*. Cambridge University Press.

[21] Jorma Ollila (2003). Tiedolla ja oppimisella jatkuvaan uusiutumiseen. *Tietojenkäsittelytiede* 19 (kesäkuu 2003), 10–14. Nokia Oyj:n Säätiön tilaisuudessa 9.12.2002 pidetty juhlapuhe.

[22] Roger Penrose (1994). *Shadows of the Mind*. Oxford University Press.

[23] J. Alan Robinson (1965). A machine-oriented logic based on the resolution principle. *Journal of the ACM* **12**, 23–41.

[24] Claude E. Shannon (1948). A mathematical theory of communication. *Bell System Technical Journal* **27**, 397–423 ja 623–656.

[25] Claude E. Shannon, W. Weaver (1949). *The Mathematical Theory of Communication*. University of Illinois Press.

[26] Claude E. Shannon (1950). Programming a computer for playing chess. *Philosophical Magazine, Series 7* **41**, 256–275.

[27] John Shawe-Taylor, Nello Cristianini (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

[28] Peter Shor (1994). Algorithms for quantum computation: discrete logarithms and factoring. *Proceedings of the 35th Annual IEEE Symposium on Foundations of Computer Science* (ss. 124–134). IEEE Computer Society Press.

[29] Alan Turing (1936). On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society, 2nd Series* **42**, 230–265.

[30] Alan Turing (1950). Computing machinery and intelligence. *Mind* **59**, 433–460.

[31] Vijay V. Vazirani (2001). *Approximation Algorithms*. Springer.

[32] Andrew Wiles (1995). Modular elliptic-curves and Fermat's Last Theorem. *Annals of Mathematics* **141**, 443–551.