Palautuva laskenta

Risto Saarelma

Helsinki 15.10.2004

Vaihtoehtoiset laskentaparadigmat -seminaari

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

1 Johdanto

Tietokoneiden laskentateho on tähän asti kasvanut eksponentiaalisesti vuosi vuodelta. Laskentatehon hinta on kuitenkin tietokoneen elektroniikan kuumeneminen. Kotitietokoneiden prosessorit alkoivat vaatia tuulettimia 1990-luvulla, ja nopeimmat nykykoneet tarvitsevat jo nestejäähdytystä. Vaikka laskentaelementtien pieneneminen vähentääkin tuotetun lämmön määrää, fysiikan lakien mukaan perinteisten laskenta-arkkitehtuurien lämmöntuotanto ei voi kutistua mielivaltaisen pieneksi. Olettaen että Mooren laki pysyy voimassa siihen asti, laskentaelementtien lämmöntuotannon fyysiseen alarajaan törmätään vuoteen 2020 mennessä [BTV01].

Kuten myöhemmin havaitaan, laskennan energiahukka ja siitä seuraava lämmöntuotanto ovat väistämättömiä vain siinä tapauksessa, että laskennan aikana hävitetään käytössä olevaa informaatiota. Näin tapahtuu, jos laskentalaitteen tilasta tietyllä hetkellä ei voida varmasti päätellä, mistä aikaisemmasta tilasta nykyiseen tilaan päädyttiin. Palautuvan laskennan perusidea on laskennan suorittaminen niin, että sen aikaisempi tila on aina pääteltävissä nykyisestä.

Teoriassa palautuvaa laskentaa hyödyntävät laitteet voisivat suorittaa mielivaltaisen monimutkaisia laskutoimituksia vakiosuuruisella energiankulutuksella. Tällaisista laitteista voisi rakentaa hyvin pieniä ja nopeita ilman että ne laskentaa suorittaessaan kuumenisivat käyttökelvottomiksi.

2 Laskenta ja termodynamiikka

Termodynamiikan lait kehitettiin alun perin 1800-luvulla selvittämään, miten tehokkaasti höyrykoneet voivat teoriassa toimia. Lait ovat kuitenkin osoittautuneet erittäin yleispäteviksi.

Termodynamiikan kaksi ensimmäistä pääsääntöä ovat seuraavat:

1. Eristetyn järjestelmän energian määrä on vakio.
2. Eristetyn järjestelmän entropia ei voi vähentyä ajan myötä.

Entropia on järjestelmän epäjärjestystä kuvaava ominaisuus, jolla on merkitystä vain tilastollisessa mielessä. Yksittäisiä deterministisesti käyttäytyviä hiukkasia tarkastelevalla tasolla ajan myötä lisääntyvän epäjärjestyksen käsite ei ole mielekäs, koska minkä tahansa fysikaalisen järjestelmän tila on hiukkastasolla peruuntuva. Mahdollinen poikkeus

ovat mustat aukot, joihin joutuneiden hiukkasten ratoja ei välttämättä pystytä myöhemmin edes teoriassa palauttamaan.

Mustat aukot ovat kuitenkin kohtalaisen harvinaisia, joten mitä entropialla sitten oikeastaan tarkoitetaan? Käytännössä fyysisissä järjestelmissä ei pystytä tarkastelemaan joka ikistä hiukkasta, vaan ollaan kiinnostuneempia jostakin järjestelmän makroskooppisesta ominaisuudesta. Tietokoneen tapauksessa näitä ominaisuuksia ovat tietokoneen muodostavista hiukkasista koostuvat muistipaikkojen ja laskentaprosessien esitykset. Jotta hiukkaset olisivat ymmärrettävissä tietokoneeksi, niiden täytyy kuitenkin olla kohtalaisen määrätyssä muodostelmassa. Kasvava epäjärjestys syntyy tietokoneen muodostavan fyysisen järjestelmän vapausasteista, jotka eivät enää osallistu hallittuun laskentaprosessin esittämiseen.

Tilastollisessa mekaniikassa entropia määritellään verrannolliseksi järjestelmän tilojen määrään. Tietokoneen n bittiä sisältävä muisti voi olla 2^n tilassa, joten tietokone pystyy havainnoimaan ainakin tämän verran fyysisiä tilojaan. Jos tietokone kuitenkin hävittää yhden bitin, putoaa sen muistin tilojen määrä puoleen, 2^{n-1}:een. Jos tämä heijastuisi suoraan koko fyysisen tietokoneen hiukkasten tilojen määrään, tietokoneen entropia olisi vähentynyt, mikä olisi termodynamiikan toisen pääsäännön vastaista. Bitin hävittämisen seurauksena täytyy siis olla tilojen lisääntyminen jossain muualla fyysisessä tietokoneessa, mutta koska loogisella tasolla informaatio on jo hävitetty, näitä tiloja ei voida enää hyödyntää. Satunnaistilat ilmenevät käytännössä fyysisen tietokoneen kuumenemisessa.

Tiettävästi John von Neumann esitti vuonna 1949 luennollaan [Ben82], että alkeislaskentaoperaatioon tarvittava energia on vähintään $kT \ln 2$, missä k on Bolzmannin vakio ($1.380 \times 10^{-23} J \cdot K^{-1}$) ja T järjestelmän absoluuttinen lämpötila. Alkeislaskentaoperaatiossa oletetaan joko siirrettävän tai tuotettavan jollakin valintaoperaatiolla yhden bitin verran informaatiota. Von Neumann ei kuitenkaan julkaissut tulostaan, eikä ole selvää, kuinka hän siihen tarkalleen päätyi. Rolf Landauer lähti johtamaan todistusta, mutta ei päätynyt samaan tulokseen kuin von Neumann. Landauer päätyi sensijaan siihen tulokseen, että vähintään $kT \ln 2$ verran energiaa on kulutettava vain siinä tapauksessa että laskentajärjestelmä *hävittää* yhden bitin verran informaatiota [Lan61].

3 Palautuva laskenta

Laskennassa informaatiota hävittävä operaatio on sellainen, jonka tuloksesta ei voida varmuudella päätellä sen lähtöarvoja. Loogiset operaatiot JA sekä TAI ovat tällaisia, looginen operaatio EI sensijaan ei ole. Tallentamalla ylimääräistä tietoa, joka mahdollistaa infor-

maatiota hävittävien loogisten operaatioiden lähtöarvojen palauttamisen, voidaan palautumaton laskenta muuttaa palautuvaksi. Palautuvuusominaisuuden voi esittää formaalisti myös toteamalla, että palautuvan laskentajärjestelmän tilasiirtymäkuvaus on bijektiivinen.

Vaikka palautuvalla laskennalla voitaisiinkin ehkäistä energian kulutus, ei Landauer pitänyt sitä toteuttamiskelpoisena ratkaisuna. Palautuvasta laskennasta on nimittäin laskentatuloksen lisäksi seurauksena muistiin talletettua ”roskaa”, joka tarvitaan jokaisen suoritetun normaalisti palautumattoman loogisen operaation perumiseen. Tämän roskan siivoaminen vaatisi sen saman energian joka olisi muuten mennyt normaalien palautumattomien operaatioiden suorittamiseen, joten Landauer päätyi siihen tulokseen, että informaation hävittämisen energiakustannus on väistämätöntä laskentaa suoritettaessa.

Palautuvan laskennan tuottama ylimääräinen muistintäyte ei kuitenkaan ole satunnaista informaatiota, vaan deterministisen laskentaprosessin tuote. Jos loppuun suoriutunutta laskentaa aletaan suorittaa takaperin sen alkua kohden, palautuvat laskuoperaatiot pyyhkivät muistin tyhjäksi. Valitettavasti tämä peruutus pyyhkii muistista myös laskennan tuloksen, joten prosessi ei ole sellaisenaan erityisen hyödyllinen. Charles Bennett esitti kuitenkin vuonna 1973 menetelmän, jonka avulla voidaan hyödyntää palautuvuutta niin, että laskentatulos kuitenkin säilyy. Bennettin mallissa yksinkertaisesti laskennan päätyttyä tehdään kopio lopputuloksesta, ja palautettaessa perutaan kaikki laskenta-askeleet *paitsi* tämä kopiointi. Lopputuloksena muistiin jää alkuperäinen syöte ja kopio laskennan tuloksesta.

Jatkossa esitettävissä palautuvan laskennan malleissa oletetaan aina käytettävän tätä menetelmää sekä mahdollisten osatulosten että lopullisten laskentatulosten tallentamiseen.

4 Palautuvia laskentamekanismeja

Artikkelissaan *The thermodynamics of computation — a review* [Ben82] Bennett esittää useita vaihtelevassa määrin idealisoituja fysikaalisia toteutusmalleja palautuvaa laskentaa suorittavalle koneelle.

4.1 Ballistinen tietokone

Havainnollinen, mutta kaukana todellisuudessa toteutettavasta on alunperin Edward Fredkinin ja Tommaso Toffolin esittämä ballistinen tietokone [FT82]. Tässä laitteessa laskentaa kuvaa joukko tasolla vieriviä ja tasolle asetelluista pystypinnoista täysin elastisesti

kimpoilevia palloja. Laskennan syöte esitetään rivinä palloja, jotka sysätään täysin rinnakkain ja täsmälleen samalla nopeudella ballistisen tietokoneen sisään.

Sopivasti asetelluilla esteillä voidaan pallojen radoilla suorittaa minkä tahansa perinteisen loogisen operaation sisältävä palautuva operaatio. Operandit määräytyvät biteiksi 1 tai 0 riippuen siitä tuleeko operaatioelementtiin tätä operandia vastaavaa rataa pitkin pallo vai ei.

Pallojen vieriminen tasolla ja täysin elastiset törmäykset eivät kuluta energiaa, ja pallojen liikkeelle sysäämisen vaatima energia voidaan kerätä takaisin kun ballistisesta tietokoneesta ulos laskennan lopputulosta ilmaisevassa rivissä vierivät pallot pysäytetään. Tällaista konetta ei kuitenkaan voida todellisuudessa rakentaa toimivaksi. Ballistinen logiikka on äärimmäisen herkkä alkuarvoille, ja vaikka alkuarvot olisivatkin täysin kohdallaan, pieninkin lämpöliike tai painovoiman vaihtelu aiheuttaa järjestelmään kumuloituvan epätarkkuuden joka alkaa varsin pienen törmäysmäärän jälkeen aiheuttaa virheitä [Ben82]. Ballistinen tietokone on kuitenkin fysikaaliseti hyvin yksinkertainen malli, joten se havainnollistaa hyvin energiaa kuluttamattoman laskennan teoreettista mahdollisuutta.

4.2 Brownin tietokone

Brownin liike on fysikaalinen ilmiö, jossa nesteessä tai kaasussa sijaitsevat pienet kappaleet liikahtelevat satunnaisesti edestakaisin ilman mitään näkyvää syytä. Ilmiön aiheuttajana on nesteen tai kaasun omien hiukkasten lämpöliike. Liikkuvat hiukkaset törmäilevät kevyeen kappaleeseen ja antavat sille hetkellisen liike-impulssin johonkin suuntaan [Ein05]. Seurauksena kappale alkaa liikkua satunnaiskävelyrataa.

Bennett esittää, että palautuvaa laskentaa suorittava kone voisi edetä tällaisella satunnaiskävelyllä. Kone olisi rakennettu sellaiseksi, että laskentapolulla eteen- tai taaksepäin liikkuminen onnistuisi mitättömän pienellä energialla, mutta laskentapolulta poikkeaminen vaatisi kohtalaisen suurta energiaa. Tällöin ilman ulkoa tulevaa ylimääräistä energiaa järjestelmän voisi olettaa liikkuvan satunnaiskävelyllä eteen- ja taaksepäin haluttua laskentalinjaa. Yhtä todennäköisesti eteen- tai taaksepäin laskeva tietokone saattaa tuntua tehottomalta, mutta koneen tarvitsee laskea vain äärellinen määrä askeleita oikeaan suuntaan saadakseen laskennan valmiiksi. Usein Brownin tietokone on myös sellainen, että syöttämällä jollain tapaa ylimääräistä energiaa järjestelmään voidaan laskentaa haluttuun suuntaan jouduttaa.

Bennett esittää Brownin tietokoneeksi kitkattomalla kellokoneistolla rakennetun Turingin koneen. Laskenta etenee koneen osien pyöriessä ja liukuessa kitkattomasti, kun taas ko-

neiston jäykät osat estävät järjestelmää eksymästä laskentapolun ulkopuolisille radoille. Julkaisussa [Ben82] kuvailtu kohtalaisen monimutkainen rakennelma on jo jonkin verran ballistista tietokonetta realistisempi malli. Käytännössä kitkaton kellokoneisto on tavoittamaton idealisaatio. Kitkan merkitys kuitenkin pienenee kun mekanismin osat kutistuvat, ja koneiston rakennetta muuttamalla voidaan osien välistä kosketusta vähentää. Tämän vuoksi tällaista kirjaimellisesti mekaanista laskentaakin on pidetty todellisten palautuvien tietokoneiden mahdollisena toteuttamistapana nanomittakaavan koneissa [Mer93].

Paljolti Brownin tietokonetta vastaava mekanismi löytyy kuitenkin todellisuudestakin, biologisten solujen proteiinikoodauksesta.

5 Palautuvan laskennan teoriaa

Turingin koneella suoritettava laskenta on triviaalia muuntaa palautettavaksi. On ainoastaan otettava käyttöön ylimääräinen nauha, jolle kirjoitetaan jokaisen Turingin koneen suorittaman normaalisti palautumattoman operaation palauttamiseen tarvittava tieto. Tästä nähdään heti, että palautuva laskenta on yhtä ilmaisuvoimaista kuin normaalilla Turingin koneella suoritettava laskenta. Ylimääräisen kirjanpidon vuoksi palautuva laskenta ei kuitenkaan ole välttämättä yhtä tehokasta kuin palautumaton. Koska palautuvien ohjelmien kirjoittaminen suoraan on vaikeaa, palautuvan laskennan mallien tarkastelu keskittyy toistaiseksi tapauksiin, joissa palautuvalla laskennalla simuloidaan perinteistä palautumatonta laskentaa.

Palautuvassa laskennassa aikavaativuus T vaikuttaa helposti tilavaativuuteen S, koska laskennan on perinteistä laskentaa simuloidessaan jatkuvasti talletettava muistiin lisää välituloksia. Ensimmäinen ongelma palautuvan laskennan mallintamisessa onkin se, voidaanko yksinkertaisessa toteutuksessa suoraan aikavaativuuteen verrannollista tilankäyttöä jotenkin pienentää.

5.1 Bennettin helmipeli

Palautuva helmipeli (*engl. pebbling game*) on Charles Bennettin esittämä suoritusmalli palautuvalle laskennalle, joka toimii ajassa $O(ST^{\log 3})$ ja vaatii tilan $O(S \log T) = O(S^2)$ [Ben89] [BTV01].

Helmipelin perusta on solmujono $G = 1, 2, \ldots, 2^k$. Bennettin mallissa jono kuvaa T askeleen laskentaprosessia, ja jokainen solmu vastaa $T2^{-k}$ askeleen pituista jaksoa tästä.

Helmipelissä käytettävänä on n helmeä, ja jokaisessa G:n solmussa voi sijaita enintään yksi helmi. Alussa yksikään helmi ei ole missään G:n solmussa.

Helmipeliä pelataan seuraavalla säännöllä: Jos solmussa G_i sijaitsee helmi, voidaan solmuun G_{i+1} laittaa helmi jos siellä ei sellaista ole, tai sieltä voidaan ottaa pois siellä oleva helmi. Jotta pelin aloittaminen olisi mahdollista, sovitaan että solmuun G_1 voi aina asettaa tai siitä poistaa helmen.

Palautuvaa laskentaa esittävässä helmipelissä solmujono G vastaa etenevän laskennan kokonaisuudessaan vaatimaa muistia, ja käytössä olevat n helmeä vastaavat käytössä todellisuudessa olevaa muistia. Vaikka G:tä vastaava muistialue on laskettava läpi kokonaisuudessaan jotta laskennan tulos voitaisiin saavuttaa ja laskennan välitulokset sen jälkeen palautuvasti hävittää, ei koko G:n tarvitse olla kerralla fyysisessä muistissa.

Helmen b lisääminen helmen a perään kuvaa sitä, miten helmeä a vastaavalla muistialueelle tallennetun laskennan tilatiedon perusteella lasketaan seuraavan jakson tilatieto. Tämä ei luonnollisestikaan ole mahdollista jos aikaisempaa tilatietoa ei ole käytössä, helmi a:n on siis oltava pelissä. Helmen b poistaminen puolestaan kuvaa b:n esittämän laskentajakson palauttamista ja poistamista muistista. Koska laskentaprosessi on palautettava alkupisteeseensä asti, täytyy tässäkin tapauksessa helmen a olla pelissä.

Helmipeliä sovellettaessa ratkotaan tilanteita, joissa n helmellä pitää edetä 2^k laskentajaksoa ja laskentajakson alkupisteen vasemmalla puolella oletetaan olevan helmi. Mikäli $k = 0$, suoritus onnistuu triviaalisti asettamalla helmi jakson ainoaan solmuun, tällöin $n = 1$.

Jos jakso on pidempi, edetään ensin $n - 1$ helmellä pisteeseen $2^{k-1} - 1$ ja sitten asetetaan viimeinen helmi pisteeseen 2^{k-1}. Seuraavaksi suoritetaan käänteisesti kaikki viimeisen helmen asettamista edeltäneet siirrot, jolloin $n-1$ helmeä palautuvat käyttöön. Laskettava väli on siis puolittunut ja käytössä olevien helmien määrä on vähentynyt yhdellä. Jotta voitaisiin laskea 2^k jaksoa tarvitaan siis k helmeä, josta seuraa että $n = k$ ja $n \propto \log_2 T$. Tästä voidaan nähdä helmipelin tilavaativuusluokan olevan $S \log T$.

5.2 Tilavaativuuden pienentäminen

Bennettin helmipeli pienentää tilavaativuuden aikavaativuuden logaritmista riippuvaiseksi, mutta se vaatii kuitenkin yhä enemmän tilaa laskennan pitkittyessä. Klaus-Jörn Lange, Pierre McKenzie ja Alain Tapp ovat esittäneet menetelmän, jolla determinististä palautumatonta laskentaa voidaan simuloida palautuvasti samassa pysyen palautumattoman laskennan tilavaativuusluokassa S [LMT97]. Keksijöidensä nimikirjaimien mukaan *LMT-*

simulaatioksi nimetyn menetelmän aikavaativuus on kuitenkin eksponentiaalinen.

Menetelmän idea on tarkastella palautumatonta laskentaa äärettömänä suunnattuna verkkona, jonka jokainen solmu on yksi mahdollinen laskentatila. Laskentatilaa a esittävästä solmusta lähtee kaari a:n seuraajatilaa a' esittävään solmuun, ja siihen saapuu kaari jokaista a:n mahdollista edeltäjätilaa esittävästä solmusta. Tätä palautumattoman laskennan siirtymäkaaviota voidaan ajatella puuna, jonka juuri on lopputila ja jonka yksi oksa on alkutila. Menetelmässä kuljetaan puun reunaa pitkin etsien lopputilaa ja pitäytyen enintään k kokoisiksi kasvavissa laskentatiloissa, jotta tilavaativuusehtoa ei rikottaisi.

Mikäli simuloidun laskennan tilavaativuus on pienempi tai yhtä suuri kuin k, haku löytää laskennan lopputilan. Muuten kierros aloitetaan uudelleen suuremmalla k:n arvolla. LMT-simulaatio on palautuva prosessi, koska jokaisen palautumattoman laskentaoperaation kohdalla se käy läpi kaikki mahdolliset laskentatilat, joista tämän operaation lopputulokseen on päädytty. Tavallaan tässä muutetaan palautumatonta laskentaa kuvaava verkko, jossa solmun sisäaste voi olla suurempi kuin yksi, palautuvaa laskentaa kuvaavaksi verkoksi, jossa sekä solmun sisäasteen että ulkoasteen on oltava enintään yksi.

Kehitelmän tarkemmat yksityiskohdat löytyvät viitteestä [LMT97].

6 Lopuksi

Palautuva laskenta vaikuttaisi olevan tärkeä perusta tietotekniikan tulevaisuudelle. Sen lisäksi että perinteiset tietokoneet alkavat seuraavan kahdenkymmenen vuoden aikana vaatia ainakin osittain palautuvaa arkkitehtuuria prosessoreiden pakkaustiheyden kasvaessa, monet uudenlaiset laskentatavat ovat palautuvia Mahdollisista tulevaisuuden laskentatekniikoista sekä molekyylilaskenta että kvanttilaskenta pohjautuvat palautuviin prosesseihin.

Lähteet

Ben82 C. H. Bennett. The thermodynamics of computation — a review. *International Journal of Theoretical Physics*, 21(12):905–940, 1982.

Ben89 C. H. Bennett. Time-space tradeoffs for reversible computation. *SIAM Journal on Computing*, 18(4):766–776, 1989.

BTV01 H. Buhrman, J. Tromp ja P. Vitányi. Time and space bounds for reversible

simulation. *Journal of Physics A: Mathematical and General*, 34(35):6821–6830, 2001.

Ein05 A. Einstein. Über die von der molekularkinetischen theorie der wärme geforderte bewegung von in ruhenden flüssigkeiten suspendierten teilchen. *Annalen der Physik*, 17:549, 1905.

FT82 E. Fredkin ja T. Toffoli. Conservative logic. *MIT Report MIT/LCS/TM-197, International Journal of Theoretical Physics*, 21:219, 1982.

Lan61 R. Landauer. Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 5(4):183–191, 1961.

LMT97 K-J Lange, P. McKenzie ja A. Tapp. Reversible space equals deterministic space. *Proc. 12th Annual IEEE Conf. on Computational Complexity*, sivut 45–50, 1997.

Mer93 R. C. Merkle. Two types of mechanical reverse logic. *Nanotechnology*, 4:114–131, 1993.