

# **Molekyylilaskenta**

Janne Kalmari

Helsinki 21.10.2004

Vaihtoehtoiset laskentaparadigmat

HELSINGIN YLIOPISTO

Tietojenkäsittelytieteen laitos

Matemaattis-luonnontieteellinen

Tietojenkäsittelytieteen laitos

Tekijä — Författare — Author

Janne Kalmari

Työn nimi — Arbetets titel — Title

Molekyylilaskenta

Oppiaine — Läroämne — Subject

Tietojenkäsittelytiede

Työn laji — Arbetets art — Level

Seminaariesitelmä

Aika — Datum — Month and year

21.10.2004

Sivumäärä — Sidoantal — Number of pages

9 sivua.

Tiivistelmä — Referat — Abstract

Avainsanat — Nyckelord — Keywords

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

# Sisältö

<b>1 Johdanto</b>	<b>1</b>
<b>2 Leikkaa ja liimaa -paradigma</b>	<b>1</b>
<b>3 Itsestään kokoontumis -paradigma</b>	<b>4</b>
<b>4 Kemiallinen reaktio -paradigma</b>	<b>5</b>
<b>5 Vesilaskentaparadigma</b>	<b>7</b>
<b>6 Yhteenveto</b>	<b>8</b>
<b>Lähteet</b>	<b>9</b>

# 1 Johdanto

Molekyylilaskenta (*molecular computing*) on uusi laskennan malli, jossa laskenta suoritetaan biomolekyyleilla. Aiemmin piihin perustuvien prosessorien uskottiin korvautuvan biologisilla tietokoneilla, mutta nykyään molekyylilaskennan katsotaan tulevaisuudessa lähinnä täydentävän muita tekniikoita.

Tässä esitelmässä esitellään eräitä molekyylilaskennan paradigmoja ja teoreettisia laskennan malleja. Esitelmän runkona toimii Takashi Yokomorin artikkeli [Yok02], jossa molekyylilaskennan mallien tuloksia käsitellään formaalien kielten teoriaan painottuen.

Määritellään aluksi esitelmässä käytettäviä formaalien kieliluokkien lyhenteet ja Chomskyn kielihierarkia:

Äärelliset kielet =  $FIN$

Säännölliset kielet =  $REG$

Kontekstittomat kielet =  $CF$

Kontekstiset kielet =  $CS$

Rajoittamattomat eli rekursiivisesti lueteltavat kielet =  $RE$

$FIN \subset REG \subset CF \subset CS \subset RE$ .

Turingin koneilla tunnistettavat kielet vastaavat rekursiivisesti lueteltavien kielten perhettä.

## 2 Leikkaa ja liimaa -paradigma

Leikkaavat järjestelmät (*splicing systems*) tarjoavat yhden matemaattisen mallin molekyylilaskentaan. Malli on alunperin kehitetty DNA:n rekombinaation biokemiallisten ilmiöiden analysointiin.

On olemassa kahta erityistä entsyymiä jotka osallistuvat biokemiallisiin reaktioihin

elollisissa olioissa. Rajoitusentsyymit (*restriction enzyme*), tunnistavat kaksisäikeisen molekyylin tietyn kohdan ja katkaisevat molekyylit siitä kohdasta. Ligaasi yhdistää kaksisäikeiset molekyylit toisiinsa, jos niiden päät sopivat yhteen. Leikkausoperaatio *splicing operation* määritellään merkkijonojen uudelleen kirjoittamissäänöksi kaksisäikeisessä DNA sekvensseissä rajoitusentsyymien ja ligaasin avulla.

Esimerkiksi rajoitusentsyymi EcoRI voi jakaa kaksisäikeiset molekyylit X ja Y:

$$\begin{array}{l} X = \text{GGGG AATT CGGG} \rightarrow \text{GGGG} \quad + \quad \text{AATT CGGG} \\ \text{CCCC TTAA GCCC} \quad \text{CCCC TTAA} \quad \text{GCCC} \end{array}$$

$$\begin{array}{l} Y = \text{AAAG AATT CTTT} \rightarrow \text{AAAG} \quad + \quad \text{AATT CTTT} \\ \text{TTTC TTAA GAAA} \quad \text{TTTC TTAA} \quad \text{GAAA} \end{array}$$

Ligaasi yhdistää nämä molekyylit uusiksi molekyyliksi Z ja W:

$$\begin{array}{l} Z = \text{GGGG AATT CTTT} \quad \text{ja} \quad W = \text{AAAG AATT CGGG} \\ \text{CCCC TTAA GAAA} \quad \text{TTTC TTAA GCCC} \end{array}$$

Watson-Clarkin komplemetaarisuuden vuoksi operaatioiden toimintaa voidaan tarkastella vain yhden säikeen sekvensseissä, eli merkkijono-operaatioina aakkostossa  $\Sigma = \{A, C, G, T\}$ .

Olkoon  $x_1 = \alpha_1\beta_1$  ja  $x_2 = \alpha_2\beta_2$  kaksi merkkijonoa aakkostossa  $\Sigma$ . Leikkaussääntö (*splicing rule*)  $r$  on muotoa  $\alpha_1\#\beta_1\$ \alpha_2\#\beta_2$ , missä  $\#$  ja  $\$$  ovat aakkostoon kuulumattomia uusia symboleja.

$x$ :n ja  $y$ :n leikkaus (*splicing*)  $r$ :llä on:

$$(x, y) \stackrel{r}{\models} (z, w) \Leftrightarrow \begin{cases} x = x_1\alpha_1\beta_1x'_1 & z = x_1\alpha_1\beta_2y'_2, \\ & \text{ja} \\ y = y_2\alpha_2\beta_2y'_2 & w = y_2\alpha_2\beta_1x'_1, \end{cases}$$

Leikkausoperaatiota käyttämällä voidaan muodostaa kielen tuottava malli, ns. H-järjestelmä. H-kaava on pari  $\sigma = (V, R)$  missä  $V$  on äärellinen aakkosto ja  $R \subseteq V^* \# V^* \$ V^* \# V^*$  on äärellinen joukko leikkaussääntöjä. Olkoon  $\sigma = (V, R)$  ja  $L \subseteq V^*$ .

Määritellään  $\sigma(L) = \{z \in V^* \mid \exists x, y \in L, r \in R \text{ se. } (x, y) \vdash_r z\}$

$\sigma(L)$  on siis merkkijonojoukko, joka saadaan  $L$ :stä yhdellä leikkausoperaatiolla. Kun kuvitellaan todellista laskentatilannetta, reagoivat entsyymit molekyylien kanssa toistuvasti. Siksi määritellään

$$\sigma^0(L) = L, \quad \sigma^{i+1}(L) = \sigma^i(L) \cup \sigma(\sigma^i(L)), i > 0, \quad \sigma^*(L) = \bigcup_{i \geq 0} \sigma^i(L).$$

Kolmikkoa  $\gamma = (V, R, L)$  kutsutaan H-järjestelmäksi eli leikkaavaksi järjestelmäksi. Kieltä  $L'$  kutsutaan leikkaavaksi kieleksi (*splicing language*), jos  $L' = \sigma^*(L)$  jollekin H-järjestelmälle  $\gamma = (V, R, L)$ . Kahdelle kieliperheelle  $FL_1, FL_2$  määritellään

$$H(FL_1, FL_2) = \{\sigma^*(L) \mid L \in FL_1, \sigma = (V, R), R \in FL_2\}$$

Ylläolevia määritelmiä käyttäen voidaan esittää leikkausoperaatioita käyttävän DNA-ohjelmoinnin eräitä perustuloksia:

- (1)  $H(FIN, FIN) \subset REG$
- (2)  $H(REG, FIN) = REG$

Yhden leikkausoperaation H-järjestelmien, jotka käyttävät säännöllistä joukkoa aksioomia ja äärellistä joukkoa sääntöjä, laskentakyvyllä on siis yläraja. Siksi järjestelmiin on kehitetty laajennoksia, joilla laskentakykyä voidaan parantaa. Eräs sellainen on pyöreät H-järjestelmät, jotka perustuvat pyöreisiin DNA-molekyyleihin. Pyöreitä DNA-molekyylejä esiintyy luonnossa, esimerkiksi bakteereissa. Pyöreiden molekyylien mukaanotto leikkausoperaatioihin nostaa H-järjestelmien laskentakyvyn rekursiivisesti lueteltavien kielten tasolle [YKF97].

### 3 Itsestään kokoontumis -paradigma

Itsestään kokoontuvassa laskennassa (*self-assembly computation*) käytettävät molekyylit yhdistyvät suuremmiksi kun ne sekoitetaan keskenään. Laskennan tulos voidaan lukea seoksesta. Yhdessä astiassa tapahtuva laskenta voidaan esittää seuraavasti:

1. Suunnittele äärellinen joukko itsestään kokoontuvan laskennan perusyksiköjä (tätä joukkoa voidaan kutsua ohjelmaksi).
2. Laita kaikki perusyksiköt riittävän korkeassa konsentraatiossa samaan astiaan, jolloin astiaan muodostuu satunnaisallas (*random pool*). (=Sopivan ajanjakson kuluttua astiaan on muodostunut kaikki mahdolliset perusyksiköiden kokoelmat.)
3. Käytä seulontamekanismia eristääksesi tarpeelliset (tai tarpeettomat) perusyksiköiden kokoelmat.
4. Päättele onko satunnaisaltaaseen muodostunut halutut ominaisuudet sisältävä perusyksiköiden kokoelma ja vastaa kyllä, jos on, ja ei muuten.

Käytännön esimerkki itsestään kokoontuvasta laskennasta on Adlemanin kokeellinen työ [Adl94]. Siinä molekyylilaskennalla ratkaistiin sunnattun Hamiltonin polun ongelma *Directed Hamiltonian Path Problem (DHPP)* seitsemän solmun kokoisessa verkossa. Ongelmassa etsitään suunnattun verkon kahden solmun välistä polkua, joka sisältää kaikki solmut täsmälleen kerran. Algorimi ongelman ratkaisemiseksi on seuraava:

**Input:** Suunnattu verkko  $G = (V, E)$ , alkusolmu 0 ja maalisolmu 6.

**Output:** *Kyllä*, jos alku ja maalisolmujen välillä on hamiltonin polku *ei* muuten.

**Askel 1:** Koodaa V ja E molekyyleihin siten että jos solmuja x ja y kuvaavat sekvenssit  $x=ATTGAC$  ja  $y=AATGGC$ , yhteys  $x \rightarrow y$  on CTGTTA.

**Askel 2:** Luo kaikki mahdolliset polut satunnaisaltaaseen käyttämällä suurta määrää koodattuja molekyylejä.

**Askel 3:** Eristä altaasta yhdistyneet molekyylit, jotka alkavat alkusolmusta ja päättyvät loppusolmuun ja siirrä tulos uuteen astiaan T.

**Askel 4:** Eristä T:stä molekyylit, joissa on täsmälleen  $|V|$  solmua ja siirrä tulos uuteen astiaan T.

**Askel 5:** Eristä T:stä molekyylit, joissa on jokainen V:n solmu ja siirrä lopulliseen astiaan T.

**Askel 6:** Vastaa *kyllä*, jos T:ssä on molekyylejä, *ei* muuten.

Itsestään kokoontuvan laskennan laskentakykyä voidaan selvittää laskennassa syntyneiden lopullisten DNA-molekyyliä monimutkaisuutta tarkastelemalla. Esimerkiksi Adlemanin kokeessa käytettyjen yksinkertaisten molekyyliä laskentakyky on sama kuin säännöllisten kielten. Monimutkaisempia molekyylejä käyttämällä päästään rekursiivisesti lueteltaviin kieliin.

## 4 Kemiallinen reaktio -paradigma

Elävän solun sisäisiä prosesseja voidaan pitää kemiallisen reaktion aikaansaamana laskentana. Tähän ajatukseen perustuu esimerkiksi P-järjestelmien teoria, jossa peruselementtinä on kalvorakenne (*membrane structure*). Rakenne sisältää kalvoja, jotka rajoittavat alueita. Kalvot voivat olla sisäkkäisiä. Ulointa kalvoa kutsutaan ihoksi. Alueet sisältävät olioita, jotka kehittyvät kehittymssääntöjen (*evolving rules*) mukaisesti. Sääntöjä sovelletaan epädeterminisesti ja maksimaalisen rinnakkaisesti.

Oliot voivat kommunikoida alueiden välillä, jolloin järjestelmän tila muuttuu. Tilojen muutokset muodostavat laskennan, jonka lopputulos on niiden olioiden joukko, jotka ovat tietyn tuloskalvon rajoittamalla alueella.

Eräs yksinkertainen P-järjestelmä on *P-järjestelmä (astetta  $m \geq 1$ ) symport/antiport-säännöillä* [PaP02]. Siinä järjestelmän oliot eivät muutu, päinvastoin kuin useimmissa muissa P-järjestelmissä, vaan vaihtavat ainoastaan sijaintiaan. Järjestelmässä on kaksi mekanismia olioiden väliseen kommunikointiin:

*symport* aiheuttaa olioiden  $a$  ja  $b$  kulkemisen kalvon läpi vain yhdessä samaan suuntaan:  $(ab, in)$  tai  $(ab, out)$ .

*antiport* sallii olioiden  $a$  ja  $b$  kulkemisen kalvon läpi samanaikaisesti vain vastakkaisiin suuntiin:  $(a, in; b, out)$

Järjestelmä määritellään seuraavalla konstruktiolla:

$\Pi = (V, \mu, M_1, \dots, M_m, M_e, R_1, \dots, R_m, i_0)$ , missä

1.  $V$  on olioiden aakkosto
2.  $\mu$  on  $m$  kalvon rakenne, joka kuvaa kalvojen sisäkkäisyydet
3.  $M_1, \dots, M_m$  ovat alueissa sijaitsevien olioiden joukkojen joukot (*multisets*),  
 $M_e$  on kalvojen ulkopuolella olevien olioiden joukkojen joukko,
4.  $\forall a \in V, M_e(a) = \emptyset$  tai  $M_a = \infty$
5.  $R_1, \dots, R_m$  ovat äärellisiä sääntöjoukkoja muodoltaan:  
 $(a, in), (a, out) a \in V$   
 $(ab, in), (ab, out) a, b \in V$   
 $(a, out; b, in), a, b \in V$
6.  $i_0 \in \{1, \dots, m\}$  on sen kalvon numero, johon laskennan tulos valmistuu

Tällä yksinkertaisella P-järjestelmällä voidaan simuloida Turingin konetta vastaavaa kielioppia, jos kalvoja on vähintään viisi. Lukumäärä voidaan pudottaa kahteen jos antiport säännöissä hyväksytään yli kahden olion kulkevan kalvon läpi samanaikaisesti. Jos P-järjestelmällä on kyky jakaa ja luoda kalvoja on niiden osoitettu ratkaisevan NP-täydelliset ongelmat vakioajassa

## 5 Vesilaskentaparadigma

Vesilaskennassa (*aqueous computing*) veteen tehdään liuos DNA-molekyyleistä. Molekyylit toimivat muistina ja astia rinnakkaislaskentakoneena. Satunnaisallasta ei muodosteta vaan liuoksen molekyyleihin tehdään muutoksia ja liuosta jaetaan ja yhdistetään halutun laskennan suorittamiseksi. Esimerkkinä laskennasta esitetään Headin ratkaisu suuntaamattoman verkon suurimman riippumattoman osajoukon etsimiseen [Hea99]. Menetelmässä käytetään pieniä rengasmaisia DNA koodeja, keinotekoisia plasmideja. Menetelmällä voidaan ratkaista NP-täydellisiä ongelmia lineaarisessa ajassa.

Plasmidi on koottu seuraavalla tavalla:

$\dots C_1 S_1 C_1 C_2 S_2 C_2 \dots C_i S_i C_i \dots C_n S_n C_n \dots$  missä:

- (1)  $C_1, \dots, C_n$  ovat kohtia, joista entsyymit  $RE_1, \dots, RE_n$  voivat leikata plasmidin.
- (2)  $S_1, \dots, S_n$  ovat plasmidin tiloja.

Algoritmissa käytetään seuraavia prosesseja:

**INITIALIZE** , muodostaa käytettävän plasmidinesteen,

**DELETE**( $S_i$ ) , poistaa tilan  $S_i$  leikkaamalla plasmidin entsyymillä  $RE_i$  ja korjaamalla sen ligaasilla,

**DIVIDE(j)** , jakaa nesteen  $j$  eri astiaan joita voidaan käsitellä erikseen,

**UNITE** , yhdistää eri astioiden nesteet,

**DROP(molekyylien pituus)** , poistaa halutun pituiset molekyylit.

Olkoon  $G = (V, A)$  suuntamaton verkko. Plasmidiksi koodataan verkko, jossa tilat vastaavat solmuja  $p$ . Algoritmi suurimman riippumattoman osajoukon etsimiseksi on seuraava:

```
INITIALIZE;
FOR kaikille yhteyksille (p,q) joukossa A DO
  DIVIDE(2)
    1:DELETE(p)
    2:DELETE(q)
  UNITE
END_FOR;
DROP(tilojen lukumäärä<suurin tilojen lukumäärä)
```

Laskenta on siis rinnakkaista ja alussa tarvittavien plasmidien lukumäärä riippuu verkon koosta.

## 6 Yhteenveto

Tässä tekstissä on esitelty vain muutamia molekyyllilaskennan malleja. Molekyyllilaskenta on varsin tuore laskennan osa-alue, joka on laajentunut nopeasti. Teoreettisia laskennan malleja on kehitetty pitkälle, mutta käytännön sovellukset ovat vasta alkutekijöissään. Tällaisia kuitenkin on luultavasti tulossa. Todellinen läpimurto voi tulla bioteknologiassa, nanoteknologiassa tai rinnakkaislaskennassa.

## Lähteet

- Adl94      Adleman, L., Molecular computation of solutions of combinatorial problems. *Science*, 266, sivut 1021–1024.
- Hea99      Head, T., Circular suggestions for dna computing. Teoksessa *Pattern Formation in Biology, Vision and Dynamics*, Carbone, A., Gromov, M. ja Pruzinkiewicz, P., toimittajat, World Scientific, 1999, sivut 325–335.
- PaP02      Paun, A. ja Paun, G., The power of communication: P-systems with symport/antiport. *New Generation Computing*, 20,3(2002), sivut 295–305.
- YKF97      Yokomori, T., Kobayashi, S. ja Ferretti, C., On the power of circular splicing systems and dna computability. *Proceedings of IEEE International Conference on Evolutionary Computation*, Indianapolis, 1997, sivut 219–224.
- Yok02      Yokomori, T., Molecular computing paradigm - toward freedom from turing's charm. *Natural Computing*, 1,4(2002), sivut 333–390.