Algoritmisen merkitsevyyden menetelmä DNA–sekvenssien hahmontunnistuksessa

Niina Haiminen
niina.haiminen@cs.helsinki.fi

Tiedon louhinta biomolekyyliaineistoista
Helsingin yliopisto, tietojenkäsittelytieteen laitos
Raportti C–2003–52, s. 1–10, marraskuu 2003

Tiivistelmä

Algoritmisen merkitsevyyden menetelmä (algorithmic significance method) on hahmontunnistusmenetelmä, joka perustuu tutkittavassa datassa esiintyvän tiivistettävyyden tarkasteluun. Tämän yleiskäyttöisen menetelmän perustana on Occamin partaveitsiteoriaan, jonka mukaan havainnot selittävistä malleista on parasta valita yksinkertaisin. Tätä periaatetta sovelletaan siten, että sekvenssin oletusarvoista koodauspituutta tarkastellaan suhteessa sekvenssin koodauspituuteen jotain tiettyä koodausmenetelmää A käyttäen.

Sekvenssin oletusarvoinen koodauspituus tarkoittaa sen koodauspituutta silloin kun datan ei oleteta sisältävän tiivistettävissä olevaa rakennetta. Koodausmenetelmä A on sovelluskohtainen ja se määrittelee sekvenssistä löytyvien hahmojen tyypin.

Jos sekvenssin esitystä voidaan tiivistää huomattavasti käyttäen tiettyä menetelmää A niin sekvenssistä on löydettävissä merkitsevää sisäistä rakennetta. Sellaiset sekvenssit, joiden esitystä voidaan tiivistää ennalta asetettua kynnysarvoa enemmän, tunnistetaan hahmoja sisältäviksi.

Tässä raportissa käydään läpi algoritmisen merkitsevyyden menetelmän perusteet sekä esitetään menetelmästä johdettu laajennettu versio. Menetelmän alkuperäistä versiota on käytetty yksinkertaisten DNA–sekvenssien etsimiseen ja laajennettua versiota muun muassa samankaltaisten sekvenssiparien etsimiseen. Raportissa esitellään nämä kaksi sovellusta niihin liittyvine koodausmenetelmineen.

1 Johdanto

Algoritmisen merkitsevyyden menetelmä on yleiskäyttöinen hahmontunnistusmenetelmä, joka perustuu yleiseen olettamukseen siitä, että datan selittävistä malleista paras on kaikkein yksinkertaisin malli [Mil99]. Yksinkertaisinta mahdollista selitystä suosiva periaate tunnetaan yleisesti 1300–luvulta peräisin olevana *Occamin partaveitsiteoriana*: "Ilmiön kanssa yhtenevistä selityksistä tulisi valita yksinkertaisin"[LiV97, s. 317].

Tässä tapauksessa yksinkertaisuudella tarkoitetaan datan vaatiman koodauksen pituutta. Datana tässä raportissa tarkastellaan merkeistä A, C, G ja T koostuvia DNA–jaksoja. Yksinkertainen selitys vastaa nyt sellaista jaksoa, joka on mahdollista esittää tiiviisti koodatussa muodossa eli pienellä bittimäärällä. Tällaiset jaksot määritellään *hahmoiksi*, joita datasta halutaan tunnistaa. Hahmot siis sisältävät jotain tiivistettävissä olevaa rakennetta.

Esimerkiksi jos jonkun jakson sisällä toistuu usein sama yksinkertainen sana niin jakso voidaan kuvata vähemmällä bittimäärällä kuin alkuperäinen kirjain kirjaimelta koodaus vaatisi. Tällöin sana voidaan korvata osoittimella sanan edelliseen esiintymään, jolloin saadaan kirjaimista ja osoittimista koostuva merkkijono. Tämän jonon koodaus on alkuperäistä lyhyempi, kunhan osoitin vie vähemmän tilaa kuin alkuperäinen sana.

Samaa sanaa toistuvasti sisältävät jaksot ovatkin eräs mielenkiintoinen etsittävien hahmojen ryhmä. Tällaisia hahmoja sisältävien jaksojen tunnistaminen voisi auttaa DNA:n toimintaan ja syntyhistoriaan liittyvän uuden tiedon kartuttamisessa.

Algoritmisen merkitsevyyden menetelmän ideana on datassa esiintyvien hahmojen löytäminen koodaamalla tutkittava data jollain tietyllä koodausmenetelmällä. Koodatun datan esityksen pituutta tarkastellaan suhteessa datan oletettavissa olevaan esityspituuteen silloin, kun datan ei oleteta sisältävän tiivistettävissä olevaa rakennetta [MiJ93a].

Tässä raportissa esitellään menetelmän käyttöön liittyvät lauseet, joiden avulla koodauspituuksien suhteesta voidaan päätellä datan sisältämän hahmon merkitsevyys. Samoin esitellään menetelmän kaksi sovellusta: yksinkertaisten DNA–jaksojen etsiminen ja samankaltaisten jaksojen tunnistaminen. Raportin luvussa 2 esitellään menetelmän idea ja peruslauseet sekä kerrotaan miten menetelmää voidaan käytännössä soveltaa. Luvussa 3 tarkastellaan edellä mainittuja ongelmia, joiden ratkaisemiseen menetelmää on ansiokkaasti sovellettu. Luku 4 sisältää yhteenvedon raportissa esitetystä asiasta.

2 Algoritmisen merkitsevyyden menetelmä: taustaa ja teoriaa

Tässä luvussa esitellään tarkemmin algoritmisen merkitsevyyden menetelmä ja siihen liittyvää teoriaa. Ensin annetaan sekvenssien koodaamisen ja hypoteesitestaukseen liittyvät taustatiedot ja sitten esitellään menetelmän keskeinen idea ja siihen liittyvät teoreemat.

2.1 Sekvenssien koodaaminen

Algoritmisen merkitsevyyden menetelmän soveltaminen DNA–sekvensseihin perustuu sekvenssien koodaamiseen jollain tietyllä koodausmenetelmällä, jonka tuottaman esityksen koko kertoo tietyn tyyppisten hahmojen esiintymisestä [Mil99]. Mitä pienempi sekvenssin koodaamiseen tarvittava bittimäärä on niin sitä suurempi osa sekvenssin sisältämästä tiedosta on toistuvaa tai jonkun muun tiedon avulla esitettävissä. Tällöin sekvenssissä on joko toistuvia hahmoja tai yhtäläisyyksiä johonkin toiseen sekvenssiin.

Yhdessä sekvenssissä toistuvia sanoja voidaan etsiä koodaamalla sekvenssi aliluvussa 3.1 esitetyllä tavalla. Toinen koodausmenetelmä ja sen sovellus esitellään aliluvussa 3.2, jossa tarkastellaan samankaltaisten sekvenssien tunnistamista. Erilaiset koodausmenetelmät sopivat siis eri tyyppisten hahmojen etsintään.

Sekvenssi t (*target*) voidaan koodata johonkin toiseen sekvenssiin s (*source*) perustuen. Tällöin $I_A(t|s)$ ilmoittaa tarvittavien bittien määrän, kun sekvenssi t koodataan sekvenssin s avulla käyttäen koodausmenetelmää A[1]. Sekvenssi t voidaan koodata myös ilman lähdettä s,

[1]Milosavljević [Mil99] käyttää määreestä A sekä termiä *koodausmenetelmä* (encoding scheme) että *koodauksen purkava algoritmi* (decoding algorithm) asiayhteydesetä riippuen. Eräässä toisessa artikkelissaan [Mil95] hän määrittelee algoritmin A olevan koodauksen purkava algoritmi, joka on suunniteltu siten että hahmoja sisältävien jaksojen koodatut pituudet ovat lyhyitä. Tässä raportissa on yksinkertaisuuden vuoksi valittu A tarkoittamaan koodausmenetelmää, joka sisältää sekä koodauksen muodostamisen että koodauksen purkamisen.

jolloin tarvittavien bittien määrä on $I_A(t)$. Koodausmenetelmien käytön ideana on tunnistaa ne sekvenssien alueet, joiden esitys voidaan tiivistää kyseistä koodausta käyttämällä. Koodausmenetelmiä voidaankin suunnitella tunnistamaan erityyppisiä hahmoja, kuten yhdessä sekvenssissä toistuvia sanoja tai kahden sekvenssin välisiä yhtäläisyyksiä.

Algoritmisen merkitsevyyden menetelmää käytettäessä täytyy ensin päättää millaisia hahmoja halutaan tarkastella ja valita sitten sopiva koodausmenetelmä, jonka avulla näitä hahmoja voidaan käytännössä tunnistaa. Kuvissa 1 ja 2 (sivuilla 7 ja 8) on esitetty kaksi eri tarkoitukseen soveltuvaa koodausmenetelmää. Menetelmät esitellään tarkemmin niitä käyttävien sovellusten yhteydessä, mutta niiden tarkasteleminen voi olla hyödyksi algoritmisen merkittävyyden periaatteen ymmärtämisessä.

2.2 Hypoteesitestaus

Hypoteesitestaus (hypothesis testing) on tilastotieteen menetelmä, jota käytetään tässä algoritmisen merkitsevyyden menetelmää johdettaessa ja määriteltäessä. Tässä yhteydessä sen avulla mitataan datassa esiintyvän tiivistettävissä olevaa rakennetta [Mil99].

Määritellään ensin dataan liittyvä *nollahypoteesi* P_0, joka halutaan kumota. Nollahypoteesi voisi olla esimerkiksi "sekvenssin kukin kirjain on x, $x \in \{A, G, C, T\}$, muista kirjaimista riippumattomalla todennäköisyydellä p_x". Tällöin sekvenssin t todennäköisyys nollahypoteesilla P_0, merkitään $p_0(t)$, saadaan kertomalla kutakin sekvenssin kirjainta vastaavat todennäköisyydet keskenänsä. Siis $p_0(t) = \Pi_x p_x^{n_x(t)}$, missä $n_x(t)$ on kirjaimen x esiintymien määrä sekvenssissä t.

Vaihtoehtoinen hypoteesi P_A puolestaan kuvaa jonkun muun mallin, josta halutaan tietää selittäisikö se datan paremmin kuin nollahypoteesi. Nyt vaihtoehtoiseksi hypoteesiksi haluttaisiin sellainen hypoteesi, joka ilmaisee jollain tavalla sen, että data ei ole täysin satunnaista vaan että sen kuvaamiseen soveltuisi paremmin joku tietty koodausmenetelmä A. Tällöin sekvenssin koodattu esitys voidaan kuvata huomattavasti pienemmällä bittimäärällä kuin sekvenssin alkuperäinen esitys aakkoston $\{A, C, G, T\}$ merkkijonona.

Nollahypoteesi ja vaihtoehtoinen hypoteesi ovat siis kilpailevia datan selittämiseen käytettäviä malleja. Todennäköisyydet $p_0(t)$ ja $p_A(t)$ ovat sekvenssin t todennäköisyydet nollahypoteesin ja vaihtoehtoisen hypoteesin perusteella. Hypoteesit määritellään tavallisesti todennäköisyysjakaumina, mutta menetelmää rakentaessaan Milosavljević [Mil99] käyttää koodausmenetelmiä ensin vaihtoehtoisen hypoteesin ja laajennetussa versiossa myös nollahypoteesin määrittelemiseen. Tässä aliluvussa tarkastellaan kuitenkin vain sellaisia hypoteeseja, joiden avulla voidaan laskea suoraan todennäköisyydet $p_0(t)$ ja $p_A(t)$ kullekin merkkijonolle t eli hypoteesit P_0 ja P_A oletetaan esitetyiksi todennäköisyysjakaumina.

Tietylle sekvenssille t voidaan määritellä *uskottavuusosamäärä* (likelihood ratio) kaavalla $[p_A(t)/p_0(t)]$. Uskottavuusosamäärä kuvaa siis sekvenssin vaihtoehtoisen hypoteesin mukaisen ja nollahypoteesin mukaisen todennäköisyyden suhdetta. Jos sekvenssin todennäköisyys on suurempi vaihtoehtoisen hypoteesin kuin nollahypoteesin perusteella, niin myös uskottavuusosamäärä on suuri. Jos suuri uskottavuusosamäärä on nollahypoteesin perusteella epätodennäköinen niin nollahypoteesi ei selitä dataa hyvin ja se hylätään. Lemma 1 osoittaa, että suuret uskottavuusosamäärät ovat epätodennäköisiä nollahypoteesin perusteella.

Lemma 1 (Lemma 1.1, [Mil99]) *Kaikille nollahypoteeseille P_0 joille $p_0(t) > 0$, kaikille t ja kaikille vaihtoehtoisille hypoteeseille P_A,*

$$P_0\{\log \frac{p_A(t)}{p_0(t)} \geq d\} \leq 2^{-d}.$$

Tämä tulos pätee kaikille d. Jos siis logaritminen uskottavuusosamäärä voidaan rajoittaa alhaalta arvolla d niin datassa on havaittu ilmiö, jonka todennäköisyys nollahypoteesiin perustuen on korkeintaan 2^{-d}. Lemman rajaa 2^{-d} kutsutaan *merkitsevyystasoksi* (significance level).

Tyypillisesti biologisissa sovelluksissa vertaillaan kahta hypoteesia siten, että merkitsevyystaso on ennalta kiinnitetty. Tällöin lasketaan ensin uskottavuusosamäärä sekvenssin ja hypoteesien perusteella. Tämän jälkeen käytetään lemmaa 1 ja lasketaan todennäköisyys sille, että uskottavuusosamäärä on vähintään saadun arvon suuruinen. Jos tämä laskettu todennäköisyys on pienempi kuin ennalta määritelty merkitsevyystaso niin tulos on merkitsevä. Tällöin siis on havaittu datassa ilmiö, jonka todennäköisyys on nollahypoteesin perusteella pieni. Havaintoa on kuitenkin todellinen, joten ristiriita voidaan ratkaista hylkäämällä nollahypoteesi, joka siis ei selittänyt dataa riittävän hyvin.

2.3 Algoritmisen merkitsevyyden menetelmä

Algoritmisen merkitsevyyden menetelmän ideana on tarkastella sekvenssien koodaamiseen tarvittavien bittien määrää ja päätellä näistä jotain sekvensseissä esiintyvien hahmojen merkitsevyyksistä. Jos esimerkiksi yksi sekvenssi voidaan koodata huomattavasti pienemmällä bittimäärällä kuin mitä sen oletusarvoiseen kirjain kirjaimelta koodaamiseen kuluisi, niin sekvenssin voidaan sanoa sisältävän merkitsevän hahmon.

Milosavljević [Mil99] lähtee rakentamaan menetelmäänsä muuttamalla uskottavuusosamäärän tarkastelemisen sekvenssin koodattujen pituuksien vertailuksi. Uskottavuusosamäärän logaritmi, $\log[p_A(t)/p_0(t)]$ voidaan nimittäin kirjoittaa muodossa $-\log p_0(t) + \log p_A(t)$. Todennäköisyysjakaumalle P optimaalinen koodausmenetelmä koodaa sekvenssin t käyttäen $-\log p(t)$ bittiä. Näiden tietojen avulla lemmasta 1 voidaan muokata käyttökelpoinen versio, jossa vaihtoehtoinen hypoteesi kuvataan käytetyn koodausmenetelmän kautta.

Tarkastellaan vaihtoehtoista hypoteesia P_A. Olkoon A koodausmenetelmä, jonka avulla osataan rekonstruoida sekvenssi t, kun sekvenssistä t on olemassa lähdettä s käyttävä koodattu esitys. Tämän koodatun esityksen pituus on siis $I_A(t|s)$. Oletetaan lisäksi, että A tuottaa korkeintaan yhden tuloksen kullekin koodatulle esitykselle. Tällöin on olemassa vakio $b \geq 1$, jolla $\Sigma_t b2^{-I_A(t|s)} = 1$ (Milosavljević [Mil99] esittää tarkempia perusteluja). Tästä saadaan vaihtoehtoisen hypoteesin määritelmäksi todennäköisyysjakauma P_A, joka asettaa sekvenssille t todennäköisyyden $p_A(t|s) = b2^{-I_A(t|s)}$.

Nyt voidaan lemmassa 1 korvata $p_A(t)$ tuolla todennäköisyydellä $b2^{-I_A(t|s)}$. Kun vielä huomataan, että $\log b \geq 0$ ja että lähde s voidaan jättää huomiotta niin saavutetaan lause 1.

Lause 1 (Corollary 1.1, [Mil99]) *Mille tahansa todennäköisyysjakaumalle P_0 sekä koodausmenetelmälle A,*

$$P_0\{-\log p_0(t) - I_A(t) \geq d\} \leq 2^{-d}.$$

Nollahypoteesin P_0 perusteella on siis epätodennäköistä, että sekvenssin t koodatun esityksen pituus $I_A(t)$ on paljon lyhyempi kuin t:n nollahypoteesiin perustuva optimaalinen koodauspituus $-\log p_0(t)$. Koodauspituus ja todennäköisyys ovat suoraan riippuvaisia toisistaan: arvoa d suuremmasta koodauspituuksien erotuksesta voidaan päätellä nollahypoteesin mukaisen todennäköisyyden olevan korkeintaan 2^{-d}.

Tämä koodauspituuksien ja todennäköisyyden suhteen yhteys mahdollistaa yhden sekvenssin sisäisen hahmontunnistuksen, kuten esitetään aliluvussa 3.1, jossa tarkastellaan yksinkertaisia sekvenssejä ja niiden koodaamista.

Tarkastellaan seuraavaksi *algoritmista yhteisinformaatiota* (algorithmic mutual information), jota soveltamalla voidaan rakentaa *laajennettu* algoritmisen merkitsevyyden menetelmä. Tämän menetelmän avulla voidaan tarkastella kahden sekvenssin välisen riippuvuuden merkitsevyyttä. Menetelmää käytetään tässä raportissa esiteltävässä toisessa sovelluksessa, sekvenssien samankaltaisuuden tunnistamisessa (aliluvussa 3.2).

Algoritminen yhteisinformaatio $I(s;t)$ määritellään sekvenssin t tuottavan lyhyimmän ohjelman pituuden (universaalille Turingin koneelle A_0), $I_{A_0}(t)$ ja sekvenssin t sekvenssiä s käyttävän koodauksen pituuden $I_A(t|s)$ erotuksena: $I(s;t) = I_{A_0}(t) - I_A(t|s)$ [Mil99]. Tätä samankaltaisuuden mittaa ei kuitenkaan voida käytännössä laskea, joten sitä on tyydyttävä approksimoimaan sellaisia koodausmenetelmiä käyttäen, joiden tuottamien koodausten pituudet on helppo laskea.

Milosavljević [Mil99] käyttää approksimaatiota

$$I(s;t) = I(t) - I(t|s)$$

yhteisinformaation määrittämiseksi.

Tässä pituudet $I(t)$ ja $I(t|s)$ ovat laskettavissa seuraavassa luvussa esitettyjen koodausmenetelmien perusteella lineaarisessa ajassa. Koodausmenetelmät ovat sellaisia, että niiden avulla tosiaan saadaan selville datassa esiintyviä hahmoja (yhdessä sekvenssissä toistuvia rakenteita tai kahden sekvenssin yhteisiä rakenteita), joten niitä voidaan pitää heuristisina menetelminä, joiden avulla universaalikoodausten I_{A_0} ja $I_A(t|s)$ pituuksia voidaan approksimoida.

Nyt sekvenssin t sisäinen tiivistettävissä oleva rakenne pienentää t:n koodauspituutta. Jos myös s sisäinen rakenne on samankaltainen niin $I(t|s)$ vähenee myöskin. Tällöin yhteisinformaatio $I(t;s)$ säilyy samansuuruisena. Sekvenssien samankaltainen sisäinen rakenne ei siis vaikuta sekvenssien yhteisinformaatioon.

Algoritmiseen yhteisinformaatioon perustuvan *laajennetun* algoritmisen merkitsevyyden menetelmän peruslause on seuraava.

Lause 2 (Theorem 1.2, [Mil99])

$$P_0\{I(s;t) \geq d\} \leq 2^{-d+O(1)}.$$

Tämä lause tarjoaa keinon arvioida kahden sekvenssin välisen samankaltaisuuden merkittävyyt. Lause voidaan tulkita siten, että d bittiä yhteisinformaatiota tarkoittaa sekvenssien olevan riippuvaisia toisistaan merkitsevyystasolla $2^{-d+O(1)}$.

3 Sovelluksia

Tässä luvussa esitellään kaksi algoritmisen merkitsevyyden menetelmän sovellusta: yksinkertaisten DNA–sekvenssien tunnistaminen sekä sekvenssien samankaltaisuuden tunnistaminen. Milosavljević esittää menetelmälleen myös muita sovelluksia, kuten sekvenssien tunnistus hybridisaation avulla [Mil99].

Tässä esiteltäviin sovelluksiin liittyy molempiin omat koodausmenetelmänsä, jotka käsitellään lyhyesti. Yksinkertaisia sekvenssejä etsittäessä koodaus tapahtuu korvaamalla usein esiintyviä sanoja osoittimilla sekvenssin kohtaan, jossa sana edellisen kerran esiintyi. Kahta sekvenssiä vertailtaessa puolestaan toinen sekvenssi koodataan toisen avulla, jolloin voidaan havaita sekvenssien välisiä yhtäläisyyksiä.

Koodatun esityksen minimaalinen pituus voidaan kumpaakin koodausmenetelmää käytettäessä laskea lineaarisessa ajassa. Kun esitysten minimaaliset pituudet ovat tiedossa niin lauseita 1 ja 2 soveltamalla saadaan laskettua sekvenssien sisältämien hahmojen merkitsevyystasot.

3.1 Yksinkertaisten DNA–sekvenssien tunnistaminen

Suuri osa DNA:sta koostuu toistuvista hahmoista, joiden koko vaihtelee muutamasta muutamiin satoihin nukleotideihin [BeW94]. On arvioitu, että jopa 30 - 50% ihmisen genomista koostuu eri tyyppisistä toistoista.

Toistojen tarkka merkitys on useissa tapauksissa vielä selvittämättä, mutta toistoja sisältävien alueiden tarkastelu voi johtaa uusiin oivalluksiin DNA:mme toimintaan ja syntyhistoriaan liittyen. Esimerkiksi joidenkin sairauksien kohdalla on havaittu yhteyksiä tiettyihin toistoja sisältäviin DNA–jaksoihin.

Yksinkertaisia toistuvia hahmoja sisältävien alueiden tunnistaminen on myös DNA–sekvenssianalyysin eräs tärkeä osa. Tietyllä alueella sijaitsevien toistojen määrä vaihtelee yleensä eri henkilöiden kesken, joten toistoja sisältäviä alueita voidaan käyttää esimerkiksi geenien paikantamiseen. Näitä alueita voidaan myös käyttää apuna silloin, kun halutaan selvittää todennäköisyys sille, että kaksi DNA näytettä on peräisin samalta henkilöltä.

Sekvenssi t on *yksinkertainen*, jos se voidaan koodata lyhyesti korvaamalla joidenkin sanojen esiintymiä osoittimilla sanojen edellisiin esiintymiin sekvenssissä. Tämä määritelmä, kuten myös tässä tarkasteltavan sovelluksen kuvaus on peräisin Milosavljevićin ja Jurkan artikkelista [MiJ93a], joka esittelee parannellun version heidän alkuperäisestä menetelmästään yksinkertaisten sekvenssien tunnistamiseen.

Tässä sovelluksessa sekvenssi t koodataan siten, että siitä tehty koodattu esitys ei ole riippuvainen mistään lähdesekvenssistä s. Kun käytössä on koodausmenetelmä A niin sekvenssin t koodaamiseen tarvittavien bittien määrää esittää $I_A(t)$. Merkitsevyyksien tarkastelussa voidaan nyt soveltaa lausetta 1.

Koodausmenetelmä A on tässä tapauksessa sellainen, että se korvaa jotkut sekvenssissä t esiintyvät sanat osoittimella kyseisen sanan edelliseen esiintymään samassa sekvenssissä. Osoitin koostuu kahdesta kokonaisluvusta: ensimmäinen ilmoittaa sanan edellisen esiintymän alkamiskohdan t:ssä ja toinen kertoo sanan pituuden. Eri osoittimien osoittamat sanat voivat sisältää päällekkäisyyksiä, tästä ei aiheudu ongelmia koodausta purettaessa.

Koska tavoitteena on koodatun esityksen pituuden minimointi niin sana kannattaa korvata osoittimella vain, jos sanan esitys vaatii enemmän bittejä kuin osoittimen esitys. Osoittimen esityksen pituus riippuu luonnollisesti sekvenssin pituudesta, sillä osoitin ilmoittaa sanan sijainnin suhteessa sekvenssin alkuun. Tällöin myös osoittimella korvattavan sanan minimipituus riippuu sekvenssin pituudesta. Korvattavan sanan minimipituus kasvaa logaritmisesti suhteessa sekvenssin pituuden kasvuun.

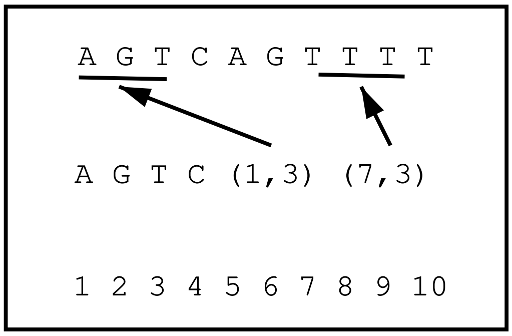
Kun sekvenssi koodataan tällaista koodausmenetelmää käyttäen niin samalla saadaan selville siinä esiintyvät toistuvat hahmot.

Kuvassa 1 on esitetty eräs yksinkertainen sekvenssi ja sen osoittimia käyttävä koodattu esitys. Koodattu esitys koostuu sekvenssin kirjaimista ja osoittimista siinä aiemmin esiintyneisiin sanoihin. Esimerkki on peräisin Milosavljevićin artikkelista [Mil99].

Milosavljević ja Jurka [MiJ93a] sovelsivat menetelmäänsä erään geenin tarkasteluun (Human Tissue Plasminogen Activator Gene, HUMPTA). Geenin 36594 merkin pituisesta sekvenssistä etsittiin yksinkertaisia alisekvenssejä tarkastelemalla sekvenssiä yksi ikkuna kerrallaan.

Ikkunoiden pituus oli 128 emästä ja seuraava ikkuna sisälsi aina myös puolet edellisestä ikkunasta, eli ikkunaa siirrettiin eteenpäin 64 emästä kerrallaan. Tätä menetelmää kutsutaan liukuvan ikkunan menetelmäksi (sliding window), tässä tapauksessa ikkunan koolla 128 ja liukuaskeleen koolla 64.

Kullekin ikkunalle laskettiin koodausmenetelmän perusteella minimaalinen koodattu pituus ja verrattiin sitä nollahypoteesin perusteella odotettavissa olleeseen pituuteen. Kukin



Kuva 1: Yksinkertainen sekvenssi ja sen koodattu esitys. Sekvenssin osa viidennestä seitsemänteen kirjaimeen *AGT* voidaan korvata osoittimella saman sanan esiintymään sekvenssin alussa. Samoin sekvenssin lopun jakso *TTT* voidaan korvata osoittimella kyseisen sanan edelliseen esiintymään sekvenssissä, joka alkaa positiosta 7. Osoitin koostuu sanan alkamiskohdasta sekvenssin sisällä sekä sanan pituudesta. Sekvenssin koodattu esitys on *AGTC* (1,3) (7,3).

ikkuna siis käsiteltiin toisistaan riippumattomasti. Ikkuna todettiin yksinkertaiseksi, jos siihen oli lauseen 1 mukaisesti aihetta eli jos nollahypoteesin mukainen todennäköisyys alitti asetetun merkitsevyystason. Merkitsevyystasoksi asetettiin yleisesti käytetty arvo $0,01$.

Kokeissa löytyi neljä yksinkertaisen jakson sisältävää ikkunaa. Näistä kaksi oli löydetty jo aikaisemmin. Myöhemmin samasta geenistä on löydetty vielä ainakin kuusi uutta yksinkertaista jaksoa [Mil99] eli menetelmä ei löytänyt kaikkia sekvenssissä esiintyviä yksinkertaisia jaksoja. Milosavljević perustelee kuitenkin alkuperäisten kokeiden heikkoa tulosta sillä, että valitut ikkunan ja liukuaskeleen koot eivät olleet parhaita mahdollisia.

Myöhemmissä kokeissa menetelmää sovellettiin ikkunan koolla 32 ja liukuaskeleen koolla 8. Tällöin löydettiin kaikki muidenkin löytämät jaksot ja vielä yksi uusi yksinkertainen jakso. Menetelmä siis osoittautui kilpailijoitaan paremmaksi, kunhan parametrit asetettiin sopivalla tavalla.

Menetelmään liittyvä perustavan laatuinen ongelmana on mielestäni parametrien asettamisen vaikeus, jota Milosavljević ei kuitenkaan kirjoituksessaan tarkastele. Sopivan ikkunan koon valinnan luulisi riippuvan toistuvien hahmojen koosta, sillä ainakaan ikkunan pituutta pidempiä hahmoja ei tällä menetelmällä pystytä havaitsemaan. Menetelmä ei aseta rajoituksia etsittävien sanojen koolle, mutta ikkunan koon valinta sisältää implisiittisesti myös sanojen pituuden ylärajan asettamisen.

Menetelmällä on kuitenkin muitakin etuja kuin pelkästään tarkasteltavien sanojen rajoittamaton pituus [MiJ93a]. Menetelmän soveltamiseksi kehitetyt koodausalgoritmit ovat nimittäin myös tehokkaita mielivaltaisen pitkille sanoille ja ikkunoille. Laskenta-aika on kaikissa tapauksissa lineaarinen suhteessa sekvenssin pituuteen.

Menetelmässä käytetty samankaltaisuuden mitta, minimaalinen koodauspituus, on laajalti tutkittu ja hyväksi todettu, toisin kuin jotkut muiden menetelmien käyttämät sovelluskohtaiset kompleksisuuden ja samankaltaisuuden mittarit. Merkitsevyystasojen käyttö tarjoaa myös yksinkertaisen ja yleisen menetelmän merkitsevyyden arviointiin.

3.2 Sekvenssien samankaltaisuuden tunnistaminen

DNA– ja proteiinisekvenssien vertailuun liittyy eräs keskeinen ongelma: jos sekvenssit vaikuttavat samankaltaisilta niin perustuuko se siihen, että niiden sisäiset rakenteet ovat samankaltaisia (ne sisältävät esimerkiksi samanlaisia toistuvia jaksoja) vai siihen, että niillä on yhteinen alkuperä [MiJ93b].

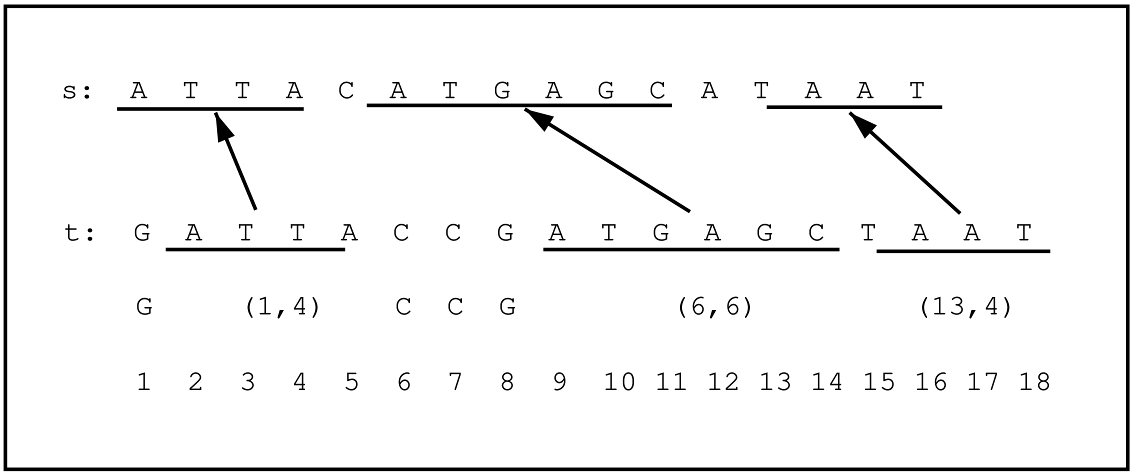
Sekvenssien vertailussa voidaan joskus haluta jättää rakenteelliset samankaltaisuudet kokonaan huomioimatta ja keskittyä yhteisen alkuperän omaavien sekvenssien tunnistamiseen. Tällöin voidaan soveltaa laajennettua algoritmisen merkitsevyyden menetelmää, joka nimenomaan pyrkii minimoimaan sekvenssien sisäisten rakenteiden samankaltaisuuksien vaikutukset sekvenssien sisältämää yhteisinformaatiota laskettaessa.

Sekvenssien t ja s algoritmista yhteisinformaatiota voidaankin approksimoidaan aliluvussa 2.3 esitetyllä kaavalla $I(s;t) = I(t) - I(t|s)$ [Mil99] [2]. Nyt $I(t)$ lasketaan edellisessä aliluvussa esitetyn yksinkertaisen koodausmenetelmän mukaisesti. Koodauspituuden $I(t|s)$ laskemiseksi sekvenssi t koodataan lähdesekvenssin s avulla siten, että jotkut sekvenssissä t esiintyvät sanat korvataan osoittimilla sekvenssissä s esiintyviin samoihin sanoihin. Merkitsevyyksien tarkastelussa voidaan nyt soveltaa lausetta 2.

Sekvenssissä t esiintyvä sisäinen rakenne johtaisi $I(t)$:n vähenemiseen ja siten myös yhteisinformaation vähenemiseen. Jos sama sisäinen rakenne esiintyy myös sekvenssissä s niin $I(t|s)$ vähenee myös, jolloin yhteisinformaation säilyy ennallaan. Laajennettu menetelmä pyrkii näin jättämään sekvenssien sisäisen rakenteen samankaltaisuudet huomiotta ja keskittyy muilla tavoin samankaltaisten sekvenssien tunnistamiseen.

Koodaaminen tapahtuu samoin kuin edellisen sovelluksen tapauksessa, sillä erotuksella että osoittimet osoittavat nyt sekvenssissä s esiintyviin sanoihin. Tällöin saadaan selville sekvenssin t sekvenssistä s riippuva koodauspituus.

Kuvassa 2 on esitetty eräs lähdesekvenssi ja sekvenssi, joka halutaan koodata kyseisen lähteen avulla (s ja t). Sekvenssin t koodattu esitys koostuu kirjaimista ja osoittimista lähdesekvenssin sanoihin. Tämäkin esimerkki on peräisin Milosavljevićin artikkelista [Mil99].



Kuva 2: Lähdesekvenssi s ja lähteen avulla koodattu sekvenssi t. Sekvenssissä t esiintyviä sanoja on korvattu osoittimilla saman sanan esiintymiin lähdesekvenssissä s. Osoitin koostuu sanan alkamiskohdasta sekvenssissä s sekä sanan pituudesta. Sekvenssin t koodattu esitys on G (1,4) CCG (6,6) (13,4).

[2]Lähteessä sivulla 16 kaava on kirjoitettu muotoon $I(t;s) = I(t) - I(t|s)$. Yhteisinformaatio $I(s;t)$ on saman lähteen sivulla 13 todettu symmetriseksi t:n ja s:n suhteen, joten kaava voidaan kirjoittaa myös tässä esitetyllä tavalla

Milosavljević [Mil95] suoritti kokeita samalle HUMTPA geenille, jota käytettiin myös edellisessä aliluvussa esitellyissä, yksinkertaisia jaksoja etsivissä kokeissa. Geenin runsaasta 36 000:sta emäksestä tarkasteltiin jaksoa 22 000 - 26 000. Kyseisen jakson tarkasteluun käytettiin liukuvan ikkunan menetelmää ikkunan koolla 200 ja liukuaskeleen koolla 100. Yksi ikkuna sisältää siis aina puolet edelliseen ikkunaan kuuluneista merkeistä.

Vertailussa käytiin läpi kaikki mahdolliset ikkunaparit, kuitenkin niin että peräkkäisiä ikkunoita (jotka siis sisälsivät osittain samoja merkkijonoja) ei vertailtu keskenään. Algoritmista yhteisinformaatiota $I(t;s)$ käytettiin samaa alkuperää olevien ikkunaparien tunnistamiseen. Merkitsevyystasoksi valitiin sama kuin yksinkertaisia jaksoja etsittäessä eli 0,01.

Kokeiden tarkoituksena oli varmistua menetelmän soveltuvuudesta havaitsemalla tarkastelujakson sisältämän kahden niin kutsutun *Alu*–sekvenssin samankaltaisuus. Alu–sekvenssit ovat lyhyitä toistollisia DNA–jaksoja, joista koostuu noin 5-10% ihmisen genomista [MiJ93b]. Näiden jaksojen tarkoitus ei ole vielä selvinnyt, mutta samaa alkuperää olevien jaksojen tunnistaminen voi auttaa niiden evolutionaarisen historian selvittämisessä, joka puolestaan saattaa paljastaa jotain kyseisten jaksojen merkityksestä.

Milosavljević [Mil95] onnistui löytämään geenissä piilevät Alu–sekvenssit, joiden peittämät ikkunat menetelmä tunnisti samankaltaisiksi. Kokeissa löytyi lisäksi yksi ikkunapari, jonka ikkunat kattoivat toistoja sisältävät jaksot, jotka eivät kuitenkaan olleet luokiteltuja yhteistä alkuperää oleviksi. Näiden ikkunoiden sisältämät jaksot sisälsivät kuitenkin muutakin yhteistä rakennetta kuin pelkästään saman yksinkertaisen sanan toiston. Menetelmä ei siis virheellisesti tunnistanut samoja yksinkertaisia toistoja sisältävän ikkunaparin vaan ikkunoiden jaksot olivat myös muilla tavoin samankaltaisia.

Milosavljević [Mil95] suoritti samanlaisen ikkunoiden vertailun myös kahden eri DNA–jakson kesken, joista toinen oli sama HUMTPA geenin jakso kuin edellä kuvatussa kokeessa. Tässä kokeessa käytetty toinen jakso sisälsi yhden Alu–sekvenssin ja tutkittu HUMTPA:n osa siis sisälsi kaksi Alu–sekvenssiä. Kokeessa onnistuttiin tunnistamaan näiden jaksojen Alu–sekvenssit sisältäneet ikkunat samankaltaisiksi.

Jaksojen sisältämiä yksinkertaisista toistoista koostuvia alueita ei sen sijaan tunnistettu samankaltaisiksi. Tämä ei ollut tarkoituksenakaan, sillä alueet sisälsivät vain samankaltaisia yksinkertaisia toistuvia sanoja eivätkä muuta samankaltaisuutta. Kokeet siis onnistuivat osoittamaan sekä menetelmän kyvyn löytää samankaltaisia sekvenssejä että sen hyvän menestyksen, kun vertailtavien sekvenssien sisäinen rakenne halutaan jättää huomiotta.

4 Yhteenveto

Algoritmisen merkitsevyyden menetelmä on datan minimaaliseen koodauspituuteen perustuva menetelmä, jota voidaan soveltaa mielivaltaisen tyyppisten hahmojen etsinnässä. Kutakin sovellusta varten tulee kuitenkin määritellä oma koodausmenetelmänsä, joka määrää löydettyjen hahmojen tyypin.

Esimerkkisovelluksina tarkasteltiin yksinkertaisten DNA–sekvenssien etsimistä sekä samankaltaisten sekvenssien tunnistamista. Näissä sovelluksissa käytettiin yksinkertaisia koodausmenetelmiä, joiden tuottamien koodausten pituudet voidaan laskea lineaarisessa ajassa.

Kyseisiä ongelmia ratkaistaessa täytyi kuitenkin asettaa ikkunointiparametrien arvot: ikkunan ja liukuaskeleen koko tulee määrittää ennalta. Samoin hahmojen merkitsevyyden arvioimisessa käytettävä merkitsevyystaso tulee määritellä etukä teen. Näiden parametrien asettamiseen ei kuitenkaan ole olemassa yleispäteviä ohjeita. Yksinkertaisia sekvenssejä etsittäessä nähtiin miten ikkunointiparametrien arvot vaikuttivat oleellisesti löydettyjen sekvenssien lukumäärään.

Tätä menetelmää on käytetty myös muissa kuin tässä raportissa mainituissa sovelluksissa, kuten hybridisaatioon perustuvassa sekvenssintunnistuksessa ja Alu–sekvenssien evoluutiota kuvaavien mallien rakentamisessa [Mil99]. Uusien sovelluskohteiden haasteena on biologisesti merkityksellisten hahmojen ja hahmoja vastaavien koodausmenetelmien määritteleminen.

Viitteet

[BeW94] Benson, G., Waterman, M. S., A Method for Database Search for All k-nucleotide Repeats. *Nucleic Acids Research*, 22, 4828–4836, 1994.

[LiV97] Li, M., Vitányi, P., *An Introduction to Kolmogorov Complexity and Its Applications*. Springer, 1997.

[Mil95] Milosavljevíc, A., Discovering Dependencies via Algorithmic Mutual Information: A Case Study in DNA Sequence Comparisons. *Machine Learning*, 21, 35–50, 1995.

[Mil99] Milosavljević, A., Discovering Patterns in DNA Sequences by the Algorithmic Significance Method. In *Pattern Discovery in Biomolecular Data*. Oxford University press, 3–23, 1999.

[MiJ93a] Milosavljević, A., Jurka, J., Discovering Simple DNA Sequences by the Algorithmic Significance Method. *Computer Applications in the Biosciences*, 9(4), 407–411, 1993.

[MiJ93b] Milosavljević, A., Jurka, J., Discovery by Minimal Length Encoding: A Case Study in Molecular Evolution. *Machine Learning*, 12, 69–87, 1993.