Rakenteiden etsintä proteiinitopologioista

Kari Laasonen
Kari.Laasonen@cs.helsinki.fi

Tiedon louhinta biomolekyyliaineistoista
Helsingin yliopisto, tietojenkäsittelytieteen laitos
Raportti C–2003–52, s. 40–50, marraskuu 2003

Tiivistelmä

Proteiinin sekundäärirakenne kuvaa sen aminohappojen muodostamia kolmiulotteisia rakenteita. Tätä voidaan mallintaa verkkoteoreettisesti. Sekundäärirakenteesta muodostetun mallin avulla on mahdollista verrata useita rakenteita keskenään tai etsiä jo tunnettua motiivia rakennetietokannoista.

Raportissa kuvaillaan, kuinka proteiinin rakenne esitetään topologiaverkkona. Tehtävänä on etsiä kahdesta tällaisesta verkosta suurimmat mahdolliset aliverkot, jotka ovat isomorfisia. Tämä tehdään muodostamalla kaarituloverkko, jonka maksimaaliset klikit vastaavat yhteisiä aliverkkoja. Tämän jälkeen ongelmaksi jää annetun verkon klikkien etsiminen. Etsintä voidaan tehdä Bron-Kerbosch -algoritmilla tai jollain sen muunnoksella.

Myös TOPS-kaavioita voi käyttää proteiinirakenteen mallinnukseen. Raportissa esitellään hahmonsovitusalgoritmi, joka etsii annettua hahmoa joukosta TOPS-kaavioita.

1 Johdanto

Proteiinit ovat makromolekyylejä, jotka rakentuvat aminohapoista. Aminohapoilla tarkoitetaan molekyylejä, jotka sisältävät sekä aminoryhmän (NH_2) että karboksyyliryhmän (CO_2H). Kaikissa proteiineissa esiintyvissä ns. α-aminohapoissa nämä ovat liittyneet samaan hiiliatomiin. Tähän hiiliatomiin liittyy myös *sivuketju* (side chain), joka on erilainen eri aminohapoilla. Amino- ja karboksyyliryhmät liittyvät toisiinsa muodostaen proteiinin *pääketjun* (backbone) [FF90].

Proteiinisynteesissä RNA:n kolmen emäksen jaksot (kodonit) määrittelevät yhden aminohapon. Näitä aminohappoja on 20 erilaista, ja jokaisella on oma kirjainsymbolinsa. Proteiinin *primäärirakenne* on aminohappojen ketju, joka voidaan siis esittää merkkijonona. *Sekundäärirakenne* kertoo, kuinka proteiinin pääketju etenee avaruudessa. Ketju muodostaa paikallisesti rakenteita, joita ovat esimerkiksi *alfa-kierre* (α-helix), *beta-säie* (β-strand) ja *käännös* (turn). Sekundäärirakenteet koostuvat yleensä useista aminohapoista, ja niitä pitävät yllä esimerkiksi vetysillat aminohappojen sivuketjujen välillä. Proteiinin *tertiäärirakenne* kuvaa sen laskostumista (folding) eli atomien sijoittumista kolmiulotteisessa avaruudessa. Rakennetietoa löytyy useista proteiinitietokannoista, esimerkiksi PDB:stä [Ber00].

Motiivi on abstraktio jossain joukossa usein esiintyvistä piirteistä, ja motiiviesiintymät voivatkin erota toisistaan pienissä yksityiskohdissa. Motiiveista saadaan sekvenssien ja rakennetekijöiden rinnalle uusia rakennuspaloja ja käsitteitä, joilla kuvata ja selittää proteiinien toimintaa. Kun jonkin motiivin vaikutus yhdessä proteiinissa tunnetaan, voidaan tätä tietoa hyödyntää tutkittaessa muita proteiineja. Tämä on tärkeää, koska käytettävissä olevan proteiinidatan määrä on suuri ja kasvaa edelleen.

Koska proteiinien rakennetta voidaan kuvata usealla tavalla, on proteiinimotiivejakin eri tyyppisiä [GSF99]. *Sekvenssimotiivit* ovat aminohappojonoja. Niitä voidaan ajatella esimerkiksi säännöllisinä lausekkeina, joita sovitetaan aminohapposekvenssiin. *Rakennemotiivit* ovat kolmiulotteisia malleja proteiinin pääketjusta. Proteiineilla, joilla on samankaltainen kolmiulotteinen rakenne, voi olla suuriakin eroja primäärirakenteessa. Samantyyppinen laskostuminen voi kertoa esimerkiksi sukulaissuhteesta [CHK+99]. Näiden kahden motiivityypin lisäksi voidaan tutkia yhdistelmiä: sekvenssi-rakennemotiivissa on sekvenssimotiivia täydennetty tiedolla vastaavasta sekundäärirakenteesta ja rakenne-sekvenssimotiivissa liitetään aminohapposymboleja rakennemotiivin "luurangon" päälle. Rakenteisiin liittyviä motiiveja voidaan etsiä erilaisten klusterointimenetelmien avulla [GSF99, Con95].

Tässä raportissa käsitellään lähinnä proteiinien rakennemotiiveja. Tilannetta yksinkertaistetaan kuitenkin hieman ylläolevasta. Täysin kolmiulotteisen rakennemotiivin asemesta tarkastellaan rakenneosien jonoja. Osa tertiäärirakenteesta säilytetään asettamalla rajoitteita jonon elementtien välille. Tällaisia proteiinin topologiaa kuvaavia olioita on luonteva mallintaa verkoilla. Luvussa 2 määritellään sekundäärirakennetta vastaava *topologiaverkko*, jota käyttämällä voidaan löytää proteiineista yhteisiä topologisia rakenteita. Luvussa 3 tarkastellaan hyvin samantapaista ongelmaa, jossa topologiaa kuvaava verkko määritellään hieman toisin. Verkkoa käytetään sen jälkeen eräänlaiseen hahmonsovitukseen. Muita mallinnusmenetelmiä käsitellään lyhyesti luvussa 4. Luku 5 on lyhyt yhteenveto.

2 Yhteisten rakenteiden etsintä

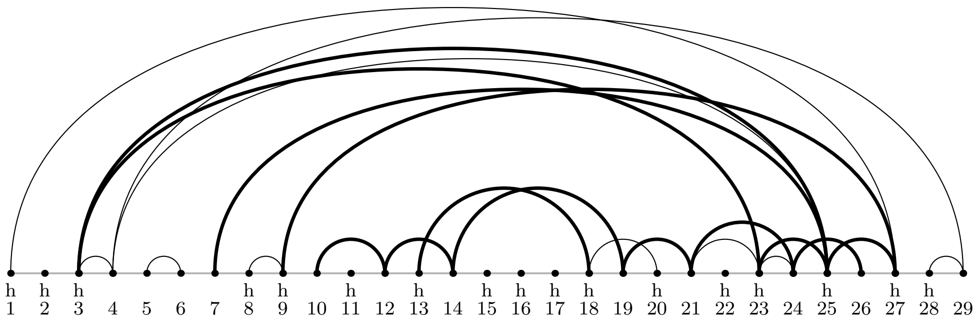
Rakennemotiiveja voidaan etsiä mallintamalla proteiinin sekundäärirakennetta verkolla. Seuraavassa menetelmässä [KLW96] rakennetaan kahdesta proteiinista tai proteiinista ja oletetusta motiivista verkko, jotka yhdistetään sopivasti. Osoittautuu, että yhdisteessä olevat klikit vastaavat yhteistä rakennetopologiaa.

2.1 Topologiaverkko

Tarkastellaan suuntaamatonta verkkoa $G = (V, E)$, missä V on verkon solmujen ja E kaarten joukko. Verkon G solmu- ja kaarijoukoille käytetään myös merkintöjä V_G ja E_G. Topologiaverkko G muodostetaan proteiinin sekundäärirakenteen perusteella seuraavasti. Jos S on rakenteen alfa-säikeiden ja H beta-kierteiden joukko, verkon solmujoukko on $V_G = S \cup H$. (Koska vain nämä piirteet sekundäärirakenteesta huomioidaan, käytetään topologiaverkosta myös nimitystä α-β -verkko.)

Verkossa G on kahden solmun välillä kaari, jos solmuja vastaavat rakenneosat ovat "lähellä toisiaan". Tätä voidaan hieman tarkentaa. Olkoon u jokin G:n solmu. Sitä vastaa joukko primäärirakenteen aminohappoja, joista edelleen jokainen koostuu joukosta atomeja. Yksinkertaistaen voidaan ajatella, että jokainen atomi täyttää avaruudesta pallonmuotoisen tilan. Tämän pallon sädettä sanotaan *van der Waalsin säteeksi*, ja se on muutaman ångströmin (10^{-10} m) luokkaa. Olkoon $T(u)$ kaikkien solmun u sisältämien atomien van der Waalsin säteen kokoisten pallojen joukko. Kahden solmun u ja v välillä on *kontakti*, jos on olemassa toisiaan leikkaavat pallot $x \in T(u)$ ja $y \in T(v)$.

Topologiaverkkoon lisätään kaari, jos tällaisia atomien välisiä kontakteja on riittävästi. Osan kontakteista tulee olla pääketjujen välisiä, osan sivuketjujen välisiä ja osan pää- ja sivu-



Kuva 1: Proteiinin `1GOX` topologiaverkko [KLW96].

ketjujen välillä. Kontaktien täsmällinen lukumäärä riippuu sovelluksesta. Kaariin liitetään lisäksi *tyyppi* eli tieto siitä, ovatko vierekkäiset rakenneosat samansuuntaisia, vastakkaissuuntaisia vai eivät kumpaakaan.

Seuraavassa on alku glykolaatti-oksidaasiproteiinin, jonka PDB-koodi on `1GOX`, 370 aminohapon primääri- ja sekundäärirakenteesta [Ber00]. Ylemmällä rivillä on primäärirakenteen aminohapposekvenssi, alemmalla taas rakennetieto:

```
XMEITNVNEY EAIAKQKLPK MVYDYYASGA EDQWTLAENR NAFSRILFRP
.....SSHHH HHHHHHHTTH HHHHHHH..S TT.HHHHHHH HGGGGEEE..

RILIDVTNID MTTTILGFKI SMPIMIAPTA MQKMAHPEGE YATARAASAA
..S...S..B ..EEETTEEE SSSEEE...S .GGGTTTTHH HHHHHHHHHH
```

Koska sekundäärirakenteesta huomioidaan topologiaverkon muodostuksessa vain kierteet (`H`) ja säikeet (`E`), solmulistan alku on ”`HHHEEEEH`”, sillä kaikki samojen merkkien jonot tiivistetään yhdeksi rakenne-elementiksi. Koko verkko on kuvan 1 mukainen. Kuvan ohuet kaaret ovat vastakkaissuuntaisia, paksummat kaaret samansuuntaisia. Kierteitä on merkitty kirjaimella ’h’, muut solmut ovat säikeitä.

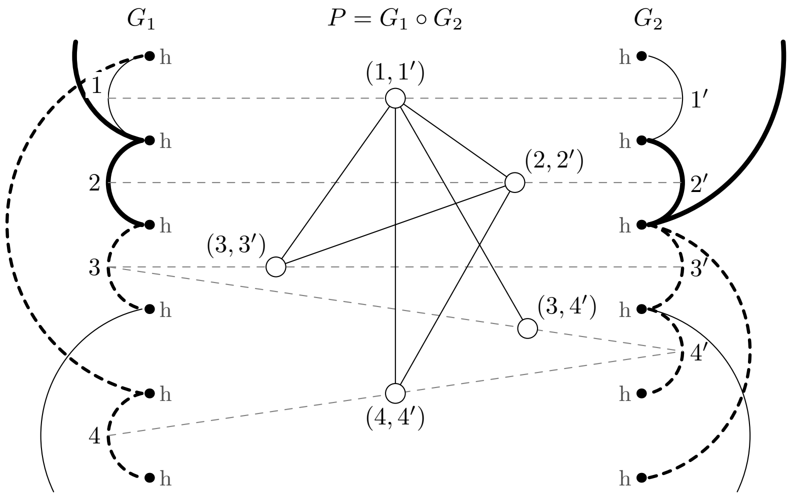
2.2 Kaarituloverkko

Motiiveja etsittäessä käytössä voi olla proteiinin topologiaverkko, jota verrataan tunnettuun hahmoon. Toinen mahdollisuus on verrata kahta proteiinia keskenään ja yrittää löytää niistä yhteisiä rakenteita. Hahmonsovitus onkin puettavissa verkkoteoreettiseksi ongelmaksi. Tätä tarkoitusta varten määritellään muutamia käsitteitä.

Olkoon G ja H (mielivaltaisia) verkkoja. Verkko H on verkon G *aliverkko*, jos $V_H \subseteq V_G$ ja $E_H \subseteq E_G$. Tällöin merkitään $H \subseteq G$. Verkko G on täydellinen (eli *klikki*), jos jokainen sen solmuista on yhteydessä kaikkiin muihin solmuihin[1]. Klikki $H \subseteq G$ on maksimaalinen, jos verkossa G ei ole enemmän solmuja sisältävää klikkiä J. Jos pätee $F \subseteq E_G$, on F:n *indusoima aliverkko* verkko, jonka kaarijoukko on F ja solmuina ovat kaikki ne verkon G solmut, jotka ovat jonkin kaaren $f \in F$ päätepisteitä. Verkkojen G ja H välinen *isomorfismikuvaus* on bijektio $V_G \rightarrow V_H$, joka säilyttää solmujen naapuruussuhteet. Jos G ja H ovat isomorfisia, merkitään $G \equiv H$.

Tarkoitus on löytää kaikki verkkojen G ja H maksimaaliset yhteiset aliverkot. Toisin sanoen etsitään pareja (G', H'), jotka ovat isomorfisia ($G' \equiv H'$) ja joille pätee $G' \subseteq G$ ja $H' \subseteq H$. Lisäksi näiden verkkojen tulee olla suurimpia mahdollisia. Etsintään käytetään

[1] Tässä seurataan artikkelia [KLW96]. Joskus termi ”klikki” tarkoittaa vain *maksimaalista* täydellistä aliverkkoa.



Kuva 2: Osa kahdesta topologiaverkosta G_1 ja G_2 ja niiden tuloverkosta P [Koc01].

kaarituloverkkoa (edge product graph). Verkkojen G ja H kaarituloverkko $P = G \circ H$ määritellään seuraavasti [Koc01]:

1. Solmujoukko on $V(P) \subseteq E_G \times E_H$. Tuloverkon solmut vastaavat siis kaaripareja. Vain sellaiset parit (e, f) hyväksytään, joissa e ja f ovat samaa tyyppiä.

2. Tuloverkon solmujen (e, f) ja (e', f') välillä on kaari, jos $e \neq e'$ ja $f \neq f'$ sekä toinen seuraavista ehdoista pätee:

   (a) Solmu, joka yhdistää kaaria e ja e' verkossa G, on samaa tyyppiä kuin solmu, joka yhdistää kaaria f ja f' verkossa H.

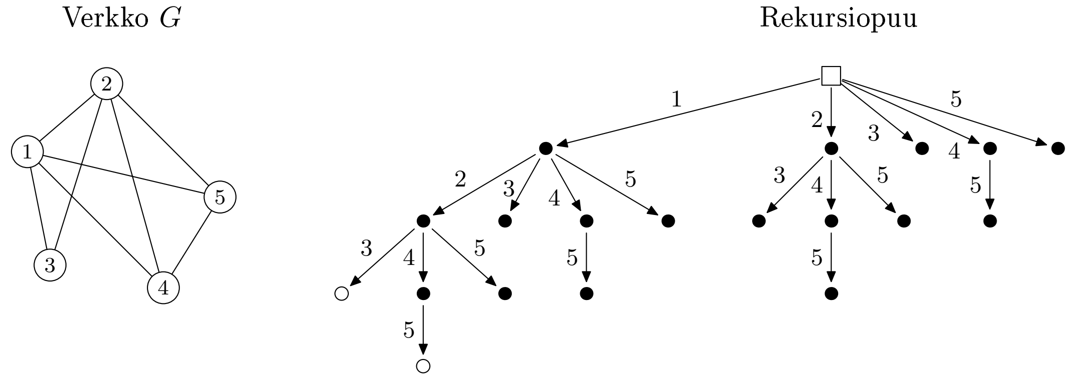
   (b) Kaaret e ja e' eivät ole naapureita, eivätkä myöskään kaaret f ja f'.

Määritelmää havainnollistaa kuva 2. Siinä on osa kahdesta topologiaverkosta G_1 ja G_2. Keskellä on osa kaarituloverkkoa P. Verkon P solmut ovat yhteensopivia kaaripareja, esimerkiksi $(1, 1')$ ja $(3, 4')$. Solmua $(1, 2')$ ei ole, koska nämä kaaret ovat eri tyyppiä. Tuloverkon kaaret muodostuvat nekin yhteensopivista kaaripareista. Esimerkiksi $(2, 2')$ ja $(3, 3')$ ovat yhteensopivia, koska kaarilla 2 ja 3 on yhteinen solmu, joka on samaa tyyppiä kuin kaarten $2'$ ja $3'$ yhteinen solmu. Vastaavasti $(1, 1')$ ja $(4, 4')$ ovat yhteensopivia, koska 1 ja 4 eivät ole naapureita verkossa G_1, eivätkä $1'$ ja $4'$ naapureita verkossa G_2. Esimerkkinä epäyhteensopivuudesta ovat parit $(2, 2')$ ja $(3, 4')$: kaaret 2 ja 3 ovat naapureita, mutta $2'$ ja $4'$ eivät.

Syy siihen, miksi tuloverkkoa tarkastellaan, on seuraava lause. Jos $(e_1, f_1), \ldots, (e_k, f_k)$ on klikki tuloverkossa $G \circ H$, ja jos kaarten $\{e_1, \ldots, e_k\}$ indusoima aliverkko $G' \subseteq G$ ja kaarten $\{f_1, \ldots, f_k\}$ indusoima aliverkko $H' \subseteq H$ ovat yhtenäisiä ja sisältävät vähintään neljä kaarta, niin G' ja H' ovat isomorfisia. Kuvassa 2 on kaksi kolmen solmun klikkiä, jotka vastaavat isomorfisia aliverkkoja, esimerkkinä $(1, 2, 4)$ ja $(1', 2', 4')$. (Tämä ei kuitenkaan ole taattua yleisessä tapauksessa, sillä kolmiota vastaavat aliverkot eivät välttämättä ole isomorfisia.) On myös osoitettu, että lause pätee myös siinä tapauksessa, että indusoidut aliverkot eivät olekaan yhtenäisiä [Koc01, §2].

2.3 Klikkien etsintä

Yhtenevien osarakenteiden etsintä on nyt muunnettu (maksimaalisten) klikkien etsinnäksi tuloverkosta. Tämä on kuitenkin tunnetusti vaikea ongelma; päätösongelma CLIQUE on



Kuva 3: Algoritmin 1 toiminta verkossa G [Koc01].

NP-täydellinen. Seuraavat algoritmitkin vaativat pahimmassa tapauksessa eksponentiaalisen ajan. Mutta muunnoksesta on myös hyötyä: klikkien etsintä on tunnettu ongelma, jonka ratkaisuun löytyy valmiita algoritmeja. Lisäksi näitä algoritmeja voidaan helposti muokata ottamaan huomioon kaarten ja solmujen tyyppirajoitteita.

Bron ja Kerbosch [BK73] esittivät *branch-and-bound* -tyyppisen algoritmin 1, joka etsii verkon $G = (V, E)$ klikkejä. Algoritmi ylläpitää kolmea joukkoa. Joukossa C ovat mahdolliseen klikkiin kuuluvat solmut, joukossa P solmut, joita voidaan lisätä klikkiin C, ja joukossa S algoritmin jo käyttämät solmut. Aluksi kaikki solmut ovat käytettävissä, joten algoritmi käynnistetään kutsulla $\textsc{Search}(\emptyset, V_G, \emptyset)$. Joukko C on siis aluksi tyhjä. Sitä kasvatetaan ottamalla joukosta P solmuja, jotka ovat kaikkien joukon C solmujen naapureita. Klikin kasvattamiseen käytetyt solmut siirretään joukkoon S. Ei ole kovin vaikeata osoittaa, että BK-algoritmin toimii oikein [Koc01, lause 3.3].

$\textsc{Search}(C, P, S)$
▷ Tulostaa kaikki verkon $G = (V, E)$ sisältämät klikit
C: solmut, jotka kuuluvat käsiteltävään klikkiin
P: solmut, jotka voidaan lisätä klikkiin C
S: solmut, joita ei saa lisätä joukkoon C

Olkoon joukon P alkiot $\{u_1, \ldots, u_k\}$
if $P = \emptyset$ **and** $S = \emptyset$ **then**
    Tulosta C
else
    for $i \leftarrow 1$ **to** k **do**
        $P \leftarrow P \setminus \{u_i\}$
        $N \leftarrow \{v \in V \mid \{u_i, v\} \in E\}$
        $\textsc{Search}(C \cup \{u_i\}, P \cap N, S \cap N)$
        $S \leftarrow S \cup \{u_i\}$
    end for
end if

Algoritmi 1: BK-algoritmi klikkien etsintään.

Algoritmin toimintaa havainnollistaa kuva 3. Kuvassa vasemmalla on verkko G ja oikealla syntyvä rekursiopuu. Puun kaariin on liitetty tieto siitä, mikä solmu u_i kulloinkin liitetään joukkoon C. Kunkin solmun lehdet vastaavat joukkoa P. Kukin polku juuresta valkeaan lehteen määrittelee yhden maksimaalisen klikin verkossa G, ja algoritmi tulostaa juuri nämä klikit. Muut lehdet vastaavat myös klikkejä, mutta niitä voidaan kasvattaa joukon S

solmuilla.

Rekursiopuussa esiintyy samoja alipuita useaan kertaan. Käsittelyä voidaankin nopeuttaa karsimalla pois alipuita, joita on jo tarkasteltu. Eräs yksinkertainen keino on valita ennen rivin 5 **for**-silmukkaa jokin $u_t \in P$; tavallisesti valitaan se solmu, jolla on eniten naapureita joukossa P. Tämän jälkeen solmu u_i käsitellään silmukassa vain jos se ei ole solmun u_t naapuri. Jos nimittäin u_i on solmun u_t naapuri ja $t < i \leq k$, käsiteltäessä solmua u_i on $u_t \in S$, eikä sitä naapuruussuhteen vuoksi poisteta joukosta S rekursiossa. Kuvan 3 rekursiopuusta jää tällöin jäljelle vain vasemmanpuoleinen haara polun $\{1, 2\}$ alla.

Algoritmi nopeutuu tästä muutoksesta huomattavasti, muttei vielä riittävästi. Solmun u_t valinta voidaan tehdä toisinkin, esimerkiksi voidaan valita ensimmäinen solmu joukosta S, mutta tämä ei oleellisesti paranna tuloksia. Artikkelissa [KLW96] mainitaan tuloverkon kooksi n. 300 solmua ja 10^5 kaarta. BK-algoritmi ei eksponentiaalisen aikavaatimuksensa takia pysty käsittelemään näin suuria verkkoja.

2.4 Yhtenäisten aliverkkojen etsintä

Kaarituloverkosta P löydetty klikki vastaa paria maksimaalisia yhteisiä aliverkkoja. Tämä klikki määrittelee myös isomorfismin tämän parin välillä. On mahdollista, että eri klikit esittävät samaa aliverkkoparia. Jos parin automorfismiryhmän koko (toisin sanoen, kuinka monella tavalla verkko voidaan permutoida sitä muuttamatta) on suuri, se tarkoittaa useiden "turhien" klikkien etsimistä.

Automorfismiryhmän koko voi olla suuri esimerkiksi siinä tapauksessa, että aliverkot eivät ole yhtenäisiä. Siispä BK-algoritmia pyritään rajoittamaan siten, että se raportoisi vain yhtenäisiä aliverkkoja vastaavia klikkejä.

Tätä tarkoitusta varten jaetaan tuloverkon kaaret kahteen luokkaan, *yhtenäiskaariin* (connectivity edge) ja epäyhtenäiskaariin. Olkoon (e, f) ja (e', f') kaksi tuloverkon solmua; e ja muut ovat siis alkuperäisten verkkojen kaaria. Näiden solmujen välillä on yhtenäiskaari, jos ja vain jos kaaret e ja e' sekä kaaret f ja f' ovat naapureita eli niillä on yhteinen solmu. Jos solmujen välillä on yhtenäiskaari, sanotaan niiden olevan "y-naapureita". Tuloverkon määritelmästä itse asiassa seuraa, että joko molemmat parit ovat naapureita tai kummatkaan eivät ole. Tuloverkon klikki vastaa nyt yhtenäisiä aliverkkoja, jos ja vain jos minkä tahansa kahden sen solmun välillä on polku, joka koostuu pelkästään yhtenäiskaarista.

Yllä esitettyyn algoritmiin ei vaadita suuria muutoksia. Uusi versio on algoritmin 2 mukainen. Solmujoukko P on jaettu kahteen osaan: yhtenäisnaapureihin P ja muihin naapurisolmuihin D. Klikkiä C kasvatetaan vain joukon P solmuilla. Solmuja siirretään joukosta toiseen sen mukaan, mikä on kulloinkin käsiteltävä solmu u_i. Algoritmi vaatii kuitenkin enemmän alustusta kuin ensimmäinen versio, koska jokaisen $v \in P$ täytyy olla y-naapuri jonkin solmun $v' \in C$ kanssa. Alussa ei siis voida asettaa $C = \emptyset$, koska tällöin myös P olisi tyhjä.

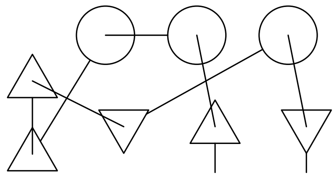
Ratkaisu on aloittaa kutsulla $\textsc{SearchConnected}(\{u\}, P, D, \emptyset)$, missä u on jokin verkon solmu, $P \cup D$ solmun u naapurien joukko, joka jaetaan y-naapureihin P ja muihin naapureihin D. Tätä toistetaan kaikille $u \in V_G$ huolehtien lisäksi siitä, ettei samaa klikkiä tulosteta useammin kuin kerran. Yksityiskohdat löytyvät kirjallisuudesta [KLW96, Koc01].

Tehty muutos pienentää hakuavaruutta huomattavasti, koska tutkittavia yhtenäiskaaria on vain pieni osa koko tuloverkon kaarista. Algoritmilla 2 pystytään käsittelemään vaadittun kokoisia tuloverkkoja. Siitä ei kuitenkaan ole hyötyä etsittäessä klikkejä muista kuin tuloverkoista, koska nopeutus perustuu juuri tuloverkon ominaisuuksiin.

SearchConnected(C, P, D, S)
▷ Tulostaa kaikki verkon sisältämät yhtenäisiä aliverkkoja vastaavat klikit
P: solmut, jotka voidaan lisätä klikkiin C ja jotka ovat y-naapureita
D: solmut, joita ei voida lisätä C:hen, koska ne eivät ole y-naapureita
C,S: kuten algoritmissa 1

Olkoon joukon P alkiot $\{u_1, \ldots, u_k\}$
if $P = \emptyset$ **and** $S = \emptyset$ **then**
  Tulosta C
else
  for $i \leftarrow 1$ **to** k **do**
    $P \leftarrow P \setminus \{u_i\}$
    $N \leftarrow \{v \in V \mid \{u_i, v\} \in E\}$
    $P' \leftarrow P; D' \leftarrow D$
    for all $v \in D'$ **do** ▷ Siirrä kelvolliset solmut joukkoon P
      if v ja u_i ovat y-naapureita **then**
        $P' \leftarrow P' \cup \{v\}; D' \leftarrow D' \setminus \{v\}$
      end if
    end for
    SearchConnected$(C \cup \{u_i\}, P' \cap N, D' \cap N, S \cap N)$
    $S \leftarrow S \cup \{u_i\}$
  end for
end if

Algoritmi 2: Yhtenäisiä aliverkkoja vastaavien klikkien etsintä.



Kuva 4: TOPS-piirros proteiinille `2BOP` [GWN+99]. Ympyrät kuvaavat α-kierteitä ja kolmiot β-säikeitä.

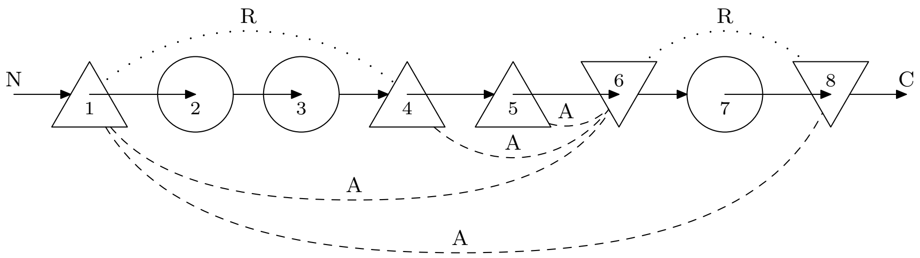
3 TOPS-kaaviot

TOPS-kaavioilla esitetään proteiinin sekundäärirakennetta, joka yksinkertaistetaan sarjaksi säikeitä ja kierteitä [GWN+99]. Kaavioita voi käyttää motiivien etsinnän ja hahmonsovituksen lisäksi myös yksinkertaisena kaksiulotteisena mallina proteiinin kolmeulotteisesta rakenteesta. TOPS-palvelussa [TPS03] voi katsella TOPS-kaavioita antamalla proteiinin PDB-koodin tai hakea proteiineja, joissa esiintyy käyttäjän antama TOPS-motiivi.

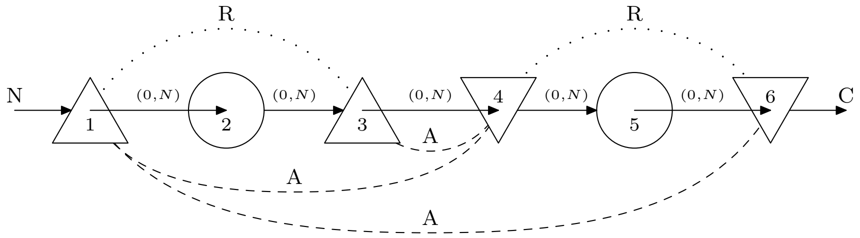
3.1 Piirrokset ja kaaviot

Kuvassa 4 on yksinkertainen TOPS-*piirros* (cartoon). Piirrokset eroavat kaavioista siinä, että ne sisältävät vain rakennesekvenssin (ja jonkin verran rakenteiden keskinäistä orientaatiota). Kaavioihin on lisätty rajoitteita, joita voidaan käyttää supistamaan hakuavaruuden kokoa sovitettaessa motiivihahmoa proteiineihin.

TOPS-piirroksen ympyrät kuvaavat alfa-kierteitä ja kolmiot beta-säikeitä. Viiva N-päästä



Kuva 5: Linearisoitu TOPS-kaavio proteiinille 2BOP.



Kuva 6: Linearisoitu TOPS-kaavio punosmotiiville.

C-päähän esittää proteiinin pääketjua. Rakenneosan keskelle päättyvä viiva on kyseisen osan ”yläpuolella”, osan reunaan päättyvä taas ”alapuolella”. Kolmion suunta kertoo myös vastaavan säikeen orientaation. Kuvan 4 säikeet muodostavat beta-levyn (sheet), mitä havainnollistaa niiden sijaitseminen vierekkäin. Seuraamalla pääketjua alusta loppuun nähdään levyn olevan vastakkaissuuntainen, sillä vierekkäisissä säikeissä ketju etenee aina eri suuntiin. Vasemmassa reunassa olevat kaksi säiettä ovat samassa kohtaa levyä. Vielä nähdään, että kaikki alfa-kierteet ovat levyn samalla puolella. Piirroksesta voidaan siis päätellä yhtä ja toista proteiinin kolmiulotteisesta rakenteesta.

TOPS-kaavion pohjana on rakenne-elementtien jono, jota voidaan käsitellä suunnattuna verkkona $\{S_1 \to S_2, S_2 \to S_3, \ldots, S_{k-1} \to S_k\}$ tai sekvenssinä $E = S_1 S_2 \ldots S_k$. Aakkostossa on neljä symbolia $\{\alpha_\pm, \beta_\pm\}$; esimerkiksi α_+ on ylöspäin suunnattu alfa-kierre. Vetysidoksia esiintyy kahden sellaisen säikeen välillä, jotka ovat osana beta-levyä. Kaaviossa sidokset esitetään kaarena solmujen S_i ja S_j välillä. Kaareen liitetään suuntatieto $\delta \in \{P, A\}$ sen mukaan, ovatko vierekkäiset säikeet samansuuntaisia (parallel) vai vastakkaissuuntaisia (antiparallel). Rakenteen *kiraliteetti* eli kätisyys $\chi = \{L, R\}$ sitoo samantyyppisiä elementtejä. Rajoitteet voidaan ilmoittaa kolmikkoina, esimerkiksi (S_i, χ, S_j). TOPS-kaavio (E, H, C) sisältää rakenneosien sekvenssin E, joukon vetysidosehtoja H ja kiraliteettirajoitteita C. Kuvassa 5 on edellä olleeseen 2BOP-proteiinin piirroksen lisätty rajoitteita ja kaavio on linearisoitu.

Motiivi on TOPS-kaavion yleistys. Ideana on, että motiivin rakenneosat löytyvät hakusekvenssistä motiivin määrittämässä järjestyksessä, mutta ei välttämättä yhtenäisenä palana. Jokaisen motiivin rakenneosan välissä onkin tieto siitä, kuinka monta osaa tähän väliin mahtuu. Tämä tieto annetaan lukuparina (n_i, m_i), mikä tarkoittaa, että väliin i täytyy sisältyä vähintään n_i ja enintään m_i osaa. Motiivin sovituksessa täytyy lisäksi huolehtia vetysidos- ja kiraliteettirajoitteiden voimassaolosta. Esimerkkinä motiivista on kuvassa 6 *punos* (plait). Tässä motiivissa ei käytännössä ole sisällytettävien osien määrään liittyviä rajoitteita: jokaiseen väliin voi sisältyä mielivaltainen määrä (”0–N kappaletta”) rakenneosia.

3.2 Motiivien etsintä TOPS-kaavioista

Tarkastelemalla kuvia 5 ja 6 näyttää ilmeiseltä, että edellisen kuvan kaavio sisältää jälkimmäisen kuvan punosmotiivin. Ongelma on oleellisesti sama kuin topologiaverkkojen tapauksessa: on annettu kaksi verkkoa, joista tulisi löytää yhteinen, isomorfinen aliverkko. Gilbert et al. [GWN+99] esittävät ratkaisuksi perääntymistä (backtracking) käyttävän hakualgoritmin. Hakuavaruutta pienentävät kaavioihin sisältyvät rajoitteet.

Mielenkiintoinen piirre algoritmissa on, että se perustuu rajoiteyhtälöihin. Voidaan ajatella, että rajoitettu muuttuja x saa arvoja äärellisestä arvojoukosta $S_x = \{v_1, \ldots, v_n\}$. Jokainen uusi ehto muuttujalle x pienentää tätä joukkoa. Jos se tyhjenee kokonaan, ongelmalla ei ole ratkaisua. Algoritmin tuottamat rajoitteet ovat lineaarisia, joten rajoiteyhtälöiden ja -epäyhtälöiden ratkaiseminen on kohtalaisen tehokasta.

On siis annettu motiivi $P = (T, H_p, C_p)$ ja kaavio $Q = (S, H_d, C_d)$. Tässä S ja T ovat sekvenssejä, $H_{p,d}$ vetysidosehtoja ja $C_{p,d}$ kiraliteettiehtoja. Motiivisekvenssin T pituus olkoon k. Ensimmäiseksi alustetaan korrelaatiovektori $D = (d_1, \ldots, d_k)$, jonka alkio d_i kertoo, mikä kaavion elementti vastaa motiivin P elementtiä i. Tämän lisäksi alustetaan sisällytysvektori $I = (I_1, \ldots, I_{k-1})$. Tässä I_i on "välin" pituus motiivin elementin i jälkeen. Näille muuttujille asetetaan yksinkertaiset ehdot:

$$
\begin{aligned}
&\text{Määrittelyalue} && 0 \leq d_i \leq N && (1)\\
&\text{Rajoitus välien pituudelle} && n_i \leq I_i \leq m_i && (2)\\
&\text{Elementtien järjestys} && d_i + I_i + 1 = d_{i+1}\,. && (3)
\end{aligned}
$$

Tämän jälkeen hyödynnetään rajoitteita pienentämään hakuavaruutta:

1. (*H-sovitus*) Etsitään jokaista motiivin vetysidosta (T_i, δ, T_j) vastaava sidos (S_k, δ, S_l) kaaviosta. Vastaavuus tarkoittaa, että $T_i = S_k$ ja $T_j = S_l$, jonka lisäksi k:n täytyy olla muuttujan d_i sallittu arvo (ja l:n d_j:n sallittu arvo). Etsintä tehdään myös toiseen suuntaan: jokainen kaavion sidos, joka on vielä mukana eli ainakin toinen elementeistä on löytynyt, sovitetaan motiivin sidoksiin.

2. (*C-sovitus*) Tämä vaihe on sama kuin yllä, mutta vetysidosten asemesta tarkastellaan kiraliteettiehtoja.

3. (*S-sovitus*) Etsitään jokaista motiivin elementtiä $T_i \in T$ vastaava $S_k \in S$, sillä ehdolla, että i on d_k:n sallittu arvo.

Jos kaikki ehdot ovat toteutettavissa, lopputulos saadaan vektoreista D ja I.

Tarkastellaan tämän algoritmin toimintaa kuvien 5 ja 6 esimerkin tapauksessa. Ehtojen (1) ja (2) myötä kaikki muuttujat d_i ja I_i on rajattu välille $[0, N]$. Lisäksi saadaan ehdosta (3) joukko yhtälöitä näiden muuttujien välille. Ensimmäisessä vaiheessa tehdään H-sovitus. Vetysidosehdoista seuraa, että $d_1 = 1$, $d_4 = 6$ ja $d_6 = 8$. Yhdistämällä nämä rajoitteisiin (3) nähdään, että $I_4 = I_5 = 0$ ja $d_5 = 7$. Kun koko H-sovitus on tehty, tilanne on

$$
\begin{aligned}
D &= (1, d_2, d_3, 6, 7, 8), && 2 \leq d_2 \leq 3,\ 4 \leq d_3 \leq 5;\\
I &= (I_1, I_2, I_3, 0, 0), && 0 \leq I_1 \leq 1,\ 0 \leq I_2 \leq 2,\ 0 \leq I_3 \leq 1.
\end{aligned}
$$

C-sovituksessa d_3 saa arvon 4, ja tämä määrää myös välin pituuden $I_3 = 1$. Lopulta merkkijonosovituksen jälkeen saadaan kaksi ratkaisua

$$
\begin{aligned}
D_1 &= (1, 2, 4, 6, 7, 8), && I_1 = (0, 1, 1, 0, 0);\\
D_2 &= (1, 3, 4, 6, 7, 8), && I_2 = (1, 0, 1, 0, 0).
\end{aligned}
$$

Vaikka verkko on järjestetty, ongelma on edelleen NP-täydellinen. Pahimmassa tapauksessa aikaa kuluu $O(2^n)$, missä n on rajoitteiden määrä. Käytännössä tilanne ei kuitenkaan ole näin huono. Verkoissa ei ole juurikaan silmukoita, ja rajoitteet auttavat hakuavaruuden karsimisessa. Rakennesekvenssit sisältävät enintään n. 60 elementtiä, ja rajoitteita on suunnilleen yhtä paljon. Tyypillisen motiivin löytäminen usean tuhannen proteiinisekvenssin tietokannasta vie vain muutamia sekunteja [GWN+99].

Rajoitteiden käyttö tässä esitetyllä tavalla ei ole ainoa tapa etsiä motiiveja TOPS-kaavioista. Koska tässäkin on kyse isomorfisten aliverkkojen etsinnästä, yllä kuvattu topologiaverkkomenetelmä sopinee pienin muutoksin. Vīksna ja Gilbert [VG01] esittivät ratkaisun, jossa rajoiteyhtälöt on korvattu syvyyssuuntaisella verkkohaulla. Tämä ei juurikaan vaikuta tuloksiin. Tärkeämpi seikka lienee se, että TOPS-kaavioiden malli proteiinin sekundäärirakenteesta on loppujen lopuksi varsin yksinkertainen. Toisaalta juuri mallin yksinkertaisuus mahdollistaa riittävän tehokkaan motiivien etsinnän.

4 Muita menetelmiä

Kuvattaessa proteiinirakennetta verkkoteoreettisesti ongelmana on usein mallin määrittely. Tähän ei ole yhtä oikeaa ratkaisua, joka kertoisi, mitä rakenteen osaa verkon solmut ja kaaret vastaisivat. Näyttääkin siltä, että verkkomalliin otetaan rakennetietoa mukaan juuri sen verran, että käsillä olevaan tutkimusongelmaan saadaan vastaus. Erityisesti mallin valintaan vaikuttaa se, millaisia motiiveja sen avulla halutaan löytää. Koska eri menetelmien tavoitteet ovat erilaisia, on tulosten vertaaminen vaikeaa.

Proteiinin kolmiulotteinen rakenne esiintyy sekä TOPS-kaavioiden että topologiaverkon kaarissa. Molemmissa kahden rakenne-elementin läheisyys vastaa verkon kaarta. TOPS-kaavioissa verkon solmut ovat lisäksi järjestyksessä pääketjun mukaan. Mallia voidaan rikastaa lisäämällä tarkempaa etäisyystietoa. Esimerkiksi SUBDUE-menetelmässä [Coo01] kaariin liitetään tarkempaa tyyppitietoa, ja pääketjun kaarista hylätään ne, joissa etäisyys on suurempi kuin 6 Å. SUBDUE etsii topologiaverkosta *piirteitä*, joilla tarkoitetaan lähes identtisiä aliverkkoja. Piirteen hyvyyttä arvioidaan MDL-periaatteen mukaisesti selvittämällä, kuinka paljon verkko pienenee, jos piirteellä korvataan ne aliverkot, joissa tämä piirre havaittiin. Menetelmää voidaan käyttää kaikenlaisen rakenteisen tiedon käsittelyyn, ei pelkästään molekyylibiologisiin sovelluksiin.

Proteiinirakenteen mallinnuksen ei välttämättä tarvitse olla verkkoteoreettista. Kuten johdannossa mainittiin, rakennemotiivit voivat olla kolmiulotteisia geometrisia malleja. Seuraamalla proteiinin pääketjua saadaan avaruudessa etenevä murtoviiva. Chew et al. ovat kehittäneet menetelmän [CHK+99], joka etsii tällaisesta mallista geometrisia yhteneväisyyksiä. Etsittävät rakennemotiivit voivat myös koostua useasta palasta. Geometrisia algoritmeja motiivien etsintään on useita [GSF99, CHK+99]. Verkkoteoreettisiin malleihin verrattuna etuina ovat ainakin mallin selkeys ja pienempi, polynominen aikavaativuus.

5 Yhteenveto

Proteiinin kolmiulotteista rakennetta voidaan mallintaa (esimerkiksi) verkoilla, joissa kaaret esittävät sekä lineaarista pääketjua että muita rakenteiden välisiä yhteyksiä ja rajoitteita. Saatua verkkoa voidaan verrata muihin vastaaviin yhteisten rakenteiden löytämiseksi, tai voidaan etsiä tunnettua motiivia rakennetietokannasta.

Algoritmisesti ongelmat ovat vaikeita, usein jopa NP-täydellisiä. Tämä tarkoittaa, että on kehitettävä aineistoon sopivia heuristiikkoja. Raportissa esiteltiin *branch-and-bound*

-algoritmi klikkien löytämiseksi ja hahmonsovitusalgoritmi topologiaverkoille. Näiden molempien aikavaatimus on pahimmassa tapauksessa eksponentiaalinen, mutta todellisilla aineistoilla molemmat näyttävät, ainakin kehittelijöidensä mukaan, toimivan riittävän tehokkaasti.

Proteiinirakennemallien kehittäminen ei ole helppoa, eikä mikään verkkoteoreettinen menetelmä olekaan saanut yleistä hyväksyntää. Verkkomallien määrittely vaatii biologista asiantuntemusta, jonka avulla sekundäärirakenteesta saadaan poimittua tarkoituksenmukaiset piirteet. Geometriset rakenteet ovat ehkä helpommin ymmärrettävissä, eikä geometristen algoritmien polynominen aikavaativuuskaan haittaa. Avoin kysymys on, mikä on eri menetelmillä löydettyjen motiivien informaatiosisältö.

Viitteet

[BK73] C. Bron and J. Kerbosch. Algorithm 457—finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577, 1973.

[Ber00] H.M. Berman, et al. The protein data bank. *Nucleic Acids Research*, 28:235–242, 2000. `http://www.pdb.org`.

[CHK+99] L. P. Chew, D. Huttenlocher, K. Kedem, and J. Kleinberg. Fast detection of common geometric substructure in proteins. In *Proc. 3rd Annual International Conference on Computational Molecular Biology (RECOMB'99)*, pages 104–113. ACM Press, 1999.

[Coo01] D. J. Cook, et al. Structural mining of molecular biology data. *IEEE Engineering in Medicine and Biology*, 20(4):67–74, 2001.

[Con95] D. Conklin. Machine discovery of protein motifs. *Machine Learning*, 21:125–150, 1995.

[FF90] R. Fessenden and J. Fessenden. *Organic Chemistry*, chapter 24. Brooks/Cole Publishing Company, fourth edition, 1990.

[GSF99] J. Glasgow, E. Steeg, and S. Fortier. Motif discovery in protein structure databases. In J. T. Wang, B. A. Shapiro, and D. Shasha, editors, *Pattern Discovery in Biomolecular Data*, pages 77–96. Oxford University Press, 1999.

[GWN+99] D. Gilbert, D. Westhead, N. Nagano, and J. Thornton. Motif-based searching in TOPS protein topology databases. *Bioinformatics*, 15(4):317–326, 1999.

[KLW96] I. Koch, T. Lengauer, and E. Wanke. An algorithm for finding maximal common subtopologies in a set of protein structures. *Journal of Computational Biology*, 3(2):289–306, 1996.

[Koc01] I. Koch. Enumerating all connected maximal common subgraphs in two graphs. *Theoretical Computer Science*, 250:1–30, 2001.

[TPS03] Topology of protein structures (TOPS), 2003. `http://www.tops.leeds.ac.uk`.

[VG01] J. Vīksna and D. Gilbert. Pattern matching and pattern discovery algorithms for protein topologies. In *Algorithms in Bioinformatics: First International Workshop (WABI 2001)*, LNCS 2149, pages 98–111. Springer Verlag, January 2001.