Identifisering av ikkje-kodande RNA-gen i genomiske sekvensar ved hjelp av samvariansmodellar

Josef Thingnes

Hovudfagsoppgåve

Førord

Dette høve vil eg nytte til å takke dei personane som har gjort arbeidet med denne oppgåva mogleg. Eg vil fyrst og fremst takke hovudvegleiar Torbjørn Rognes for fagleg støtte og vegleiing. Eg vil særskilt nemne at tilhøva har vore lagt svært godt til rette for meg som hovudfagsstudent.

Internvegleiar Knut Liestøl ved Bioinformatikkgruppa på Institutt for informatikk skal også ha ei stor takk for tolmodig å ha lest gjennom oppgåva og rettleia meg vidare ved fleire høver.

Resten av bioinformatikkgruppa og heile Molekylærbiologisk seksjon på Rikshospitalet har vore eit inspirerande og godt miljø å vere i. Her vil eg rette ein særskild takk til Gard Thomassen som har jobba med sitt hovudfag parallelt med meg.

Til slutt vil eg sende ein varm takk til Rhona Thingnes, kona mi, som tolmodig har støtta meg då hovudfagsarbeidet har butta i mot.

Samandrag

Dei seinare åra har interessa for ikkje-kodande RNA-gen (ncRNA-gen) auka dramatisk. Dette er gen som vert transkribert til RNA, men som aldri vert translatert til protein. Dei er derimot funksjonelle stoff i seg sjølve. Det har synt seg vanskeleg å finne desse gena i genomiske sekvensar ved hjelp av dei eksisterande geneleitingsprogramma.

Ein velprøvd måte å søkje etter nye gen i genomiske sekvensar er det å søkje etter homologar. Homologar, som er gen av same evolusjonære opphav, har gjerne same eller tilnærma same funksjon i ulike organismar. Sjølv om strukturen til RNA-molekyla, som er produkta frå desse gena, er rimeleg tilsvarande mellom organismar, kan sjølve nukleotidesekvensen vere svært ulik. Det er difor avgjerande å identifisere dei strukturimpliserte samanhengane i sekvensen til eit ncRNA-gen, og nytt denne informasjonen når ein skal søkje etter homologar.

Det vert her gjort ein grundig studie av søk etter ncRNA-gen ved hjelp av stokastiske kontekstfrie grammatikkar implementert som samvariansmodellar. Det vert gjort greie for korleis desse modellane skildrar nettopp denne strukturen. I tillegg til at delar av modellen er implementert, er det testa ut eit program som gjer seg nytte av denne modellen.

Modellen er funne å vere ein god måte å søkje etter homologar til kjente ncRNA-gen. Den kan vise til gode søkeresultat, men er resurskrevjande. Ein slik modell bør inngå i eit generelt søkjeprogram som skal identifisere ncRNA-gen.

Innhald

1 Innleiing

1.1 Mål

Det overordna målet for prosjektet som denne oppgåva er ein del av er å lage eit søkeprogram som skal finne informasjon i ein tekststreng som i utgangspunktet er uforståeleg. Tekststrengen representerar ein del av arvematerialet frå ei organisme, til dømes ei bakterie. Informasjonen som skal finnast er ein spesiell type gen som har eit anna sluttprodukt enn dei fleste. Det vanlege er at informasjonen i eit gen vert oversatt til eit protein, via eit RNA-molekyl, medan dei gena som det handlar om her ikkje kodar for protein men stoggar som RNA-molekyl. Difor kallast denne klassen av gen ikkje-kodande RNA-gen (engelsk *non coding RNA*, ncRNA).

Ein av eigenskapane til ncRNA-gen som skil dei frå andre gen er at det RNA-molekylet som er produktet av dette genet, faldar seg og dannar ei stabil tredimensjonal form. Deler av strukturen til denne faldinga går ofte an å kjenne att i den flate tekststrengen. Målet med denne oppgåva er å sjå på dei metodane som fins for slike søk og prøve ut dei mest lovande.

Målet med oppgåva vert presisert (2.3.3) etter at elementær molekylærbiologi er gjennomgått og prosjektet som oppgåva er ein del av er presentert.

1.2 Problemstilling

Den opphavlege problemstillinga for denne oppgåva var å freiste å finne ut korleis ein kan finne ncRNA-gen i ein genomisk sekvens ved hjelp av metoden

sekvenssamanlikning (primærstruktursamanlikning), eventuelt også med bruk av sekundærstruktursamanlikning. Etter grundige studiar av feltet, har det vist seg vanskeleg å kjenne att ncRNA-gen berre på bakgrunn av likskapar i sekvensen (Eddy og Durbin 1994). Det vert difor her sett nærare på ein modell som tek omsyn til sekundærstrukturen i tillegg til sekvenslikskapen. Problemstillinga vart dermed endra og lyd no slik: kva er dei beste måtane å søke etter kjente ncRNA-gen, kva føremoner og ulemper har dei og korleis bør desse metodane innlemmast i eit generelt søkeprogram.

Ordet samvariansmodell (engelsk: *covariance model*) og forkortinga CM vil her verte nytta om ein annan. Dei står for ein datastruktur bygd over eit datasett, der samanhengar i datasettet dannar grunnlaget for oppbygginga av datastrukturen. CM kan og tyda heile modellen som omgrep, medrekna algoritmar og datastrukturar til bygging og nytting av slike modellar. Det vil i dette tilfelle verte referert til CM-pakken eller samvariansmodellpakken.

1.3 Konkrete resultat frå arbeidet med oppgåva

I tillegg til dette dokumentet har arbeidet med denne oppgåva resultert i ein poster som vart publisert på Biokjemisk Kontaktmøte 2004. Denne posteren ligg tilgjengeleg på http://www.cmbn.no/rognes/vm2004_gard_jo.pdf. Dette arbeidet vart òg presentert i eit foredrag på *Bioinformatics Forum for Young Scientists* på Vatnahalsen i mars 2003. Dette foredraget vart og presentert på MASTERBIO i februar 2004. MASTERBIO er ei intern foredragsrekke av og for hovudfagsstudentar ved Bioinformatikkgruppa ved Institutt for Informatikk, UiO.

1.4 Presentasjon av oppgåva

Det vert fyrst gjeve ei innføring i elementær molekylærbiologi, deretter kjem ein introduksjon til teorien kring grammatikkar. Modellen som vert granska i denne oppgåva, samvariansmodellen får ein grundig presentasjon i avsnitt 2.6. Det vert testa ut eit program som nyttar denne modellen. Presentasjon av det forsøket er gjort i avsnitt 3.1.Vidare er det laga eit treningssett som er nytta til å trene opp modellen på. Produksjonen av dette treningssettet er skildra i avsnitt 3.2. Sjølve programmet som implementerar deler av modellen er skildra i avsnitt 3.3. Det vert lagt fram ei endring til modellen i avsnitt 3.4. I kapittel 4 vert resultata frå dei ulike forsøka diskutert og sett på i lys av den framlagde teorien og andres arbeid på området. Vegen vidare er staka ut i kapittel 5.

1.5 Vedlegg

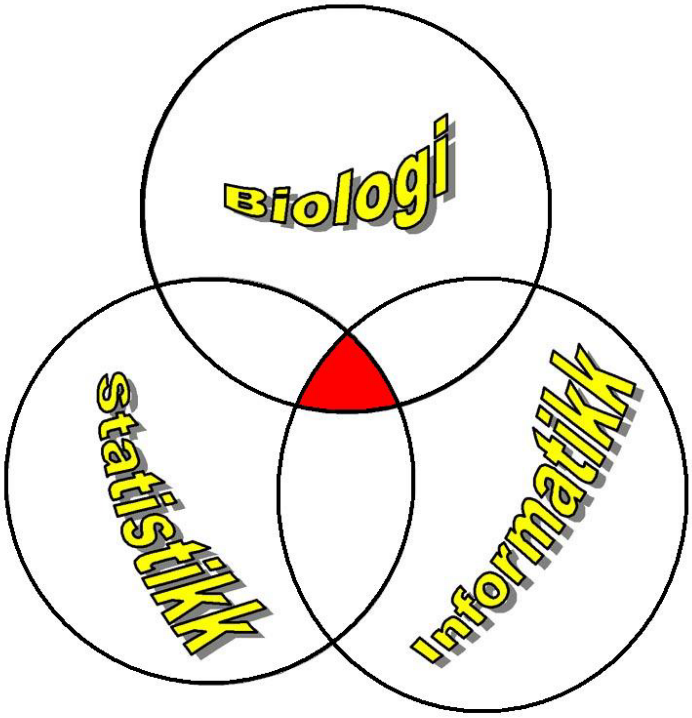
Vedlegga til denne oppgåva ligg på http://folk.uio.no/joseft/hovudfagsoppgaava/. Der fins den Java-koden som er skriven og dei resultatfilene som har danna grunnlaget for dei konklusjonane som er dregne i oppgåva.

2 Bakgrunn

I dette kapitlet vert det gjeve ei innføring i det bakgrunnsmaterialet som trengs for å forstå denne oppgåva. Kapitlet byrjar med ein presentasjon av fagfeltet bioinformatikk og held fram med ei innføring i elementær molekylærbiologi. Prosjektet som denne oppgåva er ein del av vert presentert og det vert gjeve ei innføring i homologisøk og dynamisk programmering. Det vert til slutt gjeve ei innføring i den matematiske bakgrunnen for samvariansmodellen, nemleg teorien kring grammatikkar, før sjølve modellen vert lagt fram.

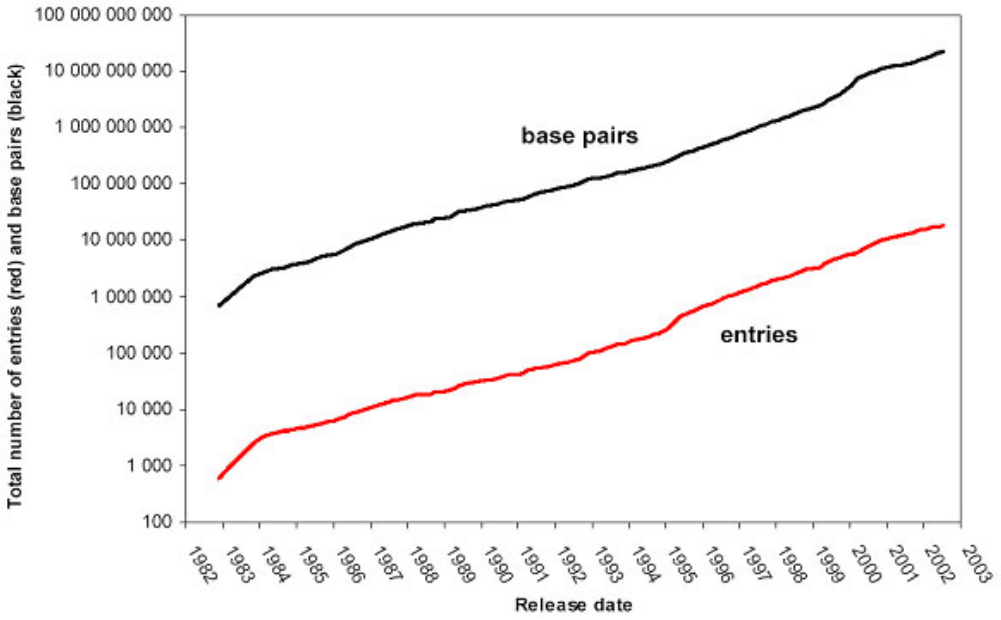
2.1 Bioinformatikk

Bioinformatikk er eit relativt ungt fagfelt som ligg i skjeringa mellom informatikk, statistikk og molekylærbiologi. Det er forskarar innanfor biologi og medisin som skapar behovet for, og har problemstillingane som bioinformatikken jobbar med (Figur 1). Statistikarane sitt bidrag er analysemetodar og teori for å dra slutningar basert på observerte data. Informatikarar kan hjelpe til med å programmere, gjere berekningar og finne dei mest høvelege algoritmane og datastrukturane.



Figur 1. Bioinformatikk ligg i skjeringa mellom biologi, statistikk og informatikk.

Analysering av store datamengder vert stadig viktigare innan biologi. Figur 2 syner utviklinga i talet på tilslag/nukleotidar i GenBank frå starten og fram til 2003. Det er ingen grunn til å tru at ikkje denne eksponensielle veksten kjem til å halde fram. Utstyret biologane nyttar for å framskaffe data med vert meir og meir raffinert og det er stor satsing på området over heile verda. Desse enorme datamengdene gir oss både moglegheiter og utfordringar. Samstundes som sjansen for nye oppdagingar aukar, vert kompleksiteten større og styrken på signala minkar i høve til støyen (Rognes 2003).



Figur 2. Det har vore sterk vekst i talet på tilslag/nukleotidar i verdas biologiske databasar. Denne grafen syner veksten i GenBank i USA. Grafen er laga av Torbjørn Rognes på grunnlag av data frå *National Center for Biotechnology Information* (NCBI).

2.2 Molekylærbiologi

I dette avsnittet vert det gjeve ei innføring i dei delane av molekylærbiologien som er viktig for denne oppgåva. Innleiingsvis kjem ei forklaring på kva DNA er og korleis proteinoppbygginga føregår. Deretter kjem ei innføring i kva ncRNA er, kvifor det er interessant å sjå etter desse og kva metodar ein nyttar.

2.2.1 Generell molekylærbiologi

Dette avsnittet er, dersom ikkje andre referansar er gjevne, henta frå Waterman (1995). Ein av dei basale tinga i biologi er å forstå arv. I 1865 laga Mendel ein abstrakt, nærmast matematisk modell for arv der gen var byggjesteinen. Mendels arbeid vart gløymt og ikkje før i byrjinga av 1900-talet vart det teke oppatt. Ikkje før i 1944 viste ein at genet var laga av DNA og fyrst i 1953 presenterte James Watson og Francis Crick den no berømte doble spiralstrukturen for DNA (Watson og Crick 1953). Denne doble spiralstrukturen har i seg eigenskapen av eigen reproduksjon.

Molekyla i cellene er delt inn i fleire klassar. I denne oppgåva er det dei største som er mest interessante. Desse vert kalla makromolekyl. Dei mest relevante typane makromolekyl er DNA, RNA og protein.

DNA, RNA og protein

DNA er basisen for arv og er ein polymer. Polymer tyder at det er eit molekyl som er samansatt av mindre einingar til ei lang rekke. Dei mindre molekyla som DNA er bygd opp av kallast nukleotidar. Nukleotidane er fire i talet og kan kjennast på dei fire basane adenin (A), cytosin (C), guanin (G) og tymin (T). For ei meir detaljert skildring av den kjemiske oppbygginga, sjå til dømes Waterman (1995), men til vårt føremål er DNA-molekyl eit ord over alfabetet $\Sigma = \{A,C,G,T\}$. Sjå 2.5 for ein definisjon av omgrepet alfabet som høver til vårt formål. DNA er den eine av to nukleinsyrer. Den andre heiter RNA er eit ord over eit anna firebokstavers alfabet $\Sigma = \{A,C,G,U\}$ der tymin er bytt ut med uracil (U). Desse molekyla har ein eintydig retning, og dei to endane vert kalla 5' og 3' (lesast ”femmerka” og ”tremerka”). Desse namna stammar frå organisk kjemi.

Eit protein er og ein polymer med ein eintydig retning men her er ordet laga frå eit alfabet av 20 aminosyrer. Det fins ein- og trebokstavskodar som er vanleg å nytte for kvar av aminosyrene.

Den doble spiralen

Den eigenskapen ved DNA som føreslo kopieringsmekanismen er dei komplementære basepara. Dette tyder at basane (nukleotidane) dannar par fordi A er komplementær til

T og G er komplementær til C. Denne sokalla paringa er hydrogenbindingar mellom to komplementære DNA-trådar. Ideen er at eit einskild ord av DNA, skrive i 5' til 3' retninga

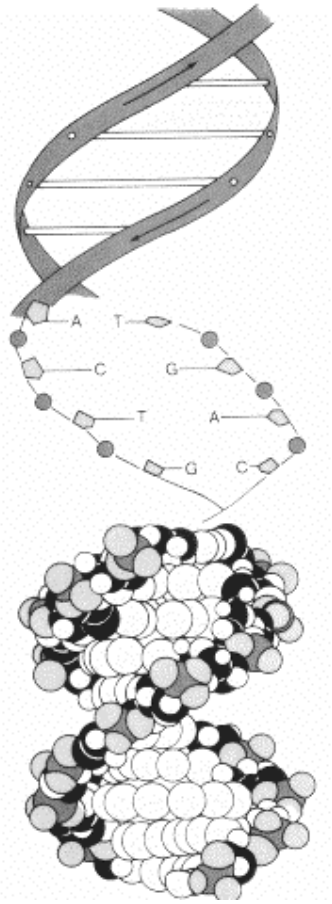
```
5' ACCTGAC 3'
```

er para med eit komplementært ord skrive den andre vegen:

```
5' ACCTGAC 3'
   |||||||
3' TGGACTG 5'
```

Det er sju basepar i denne illustrasjonen. Basepara A-T og C-G, som her er synt ved hjelp av ein loddrett strek, er forma av hydrogenbindingar. DNA førekjem oftast dobbeltråda (*double stranded*) og lengda er målt i kor mange basepar (bp) den inneheld. Lengda av ein enkeltråda nukleotidesekvens vert rekna i talet på nukleotidar (nt).

Den tredimensjonale strukturen til eit DNA-molekyl er ein dobbel spiral. Ein kan tenkje seg denne spiralen som ein vridd stige der trina er hydrogenbindingane mellom basepara og stammane til stigen er sukker-fosfat-kjeden som held nukleotidane saman til ei rekkje (Figur 3).



Figur 3. Tre ulike modellar for DNA-molekylet. Øvst framstår den vridde stigen, nedst er skildra ein 3D-modell med dei einskilde atoma i det organiske molekylet innteikna, medan i midten er den doble spiralen teikna slik det i denne samanhengen er mest teneleg å sjå han, nemleg som to komplementære strengar av nukleotidar, der basane er representert med kvar sin bokstav. I tilegg ser ein korleis sukkermolekylet (femkant) og fosfatmolekylet (runding) utgjer kjeda som dannar det som her vert kalla trådane (*strands*) i DNA-molekylet. Biletet er henta frå professor Rein Aasland sine internettsider: http://www.uib.no/aasland/

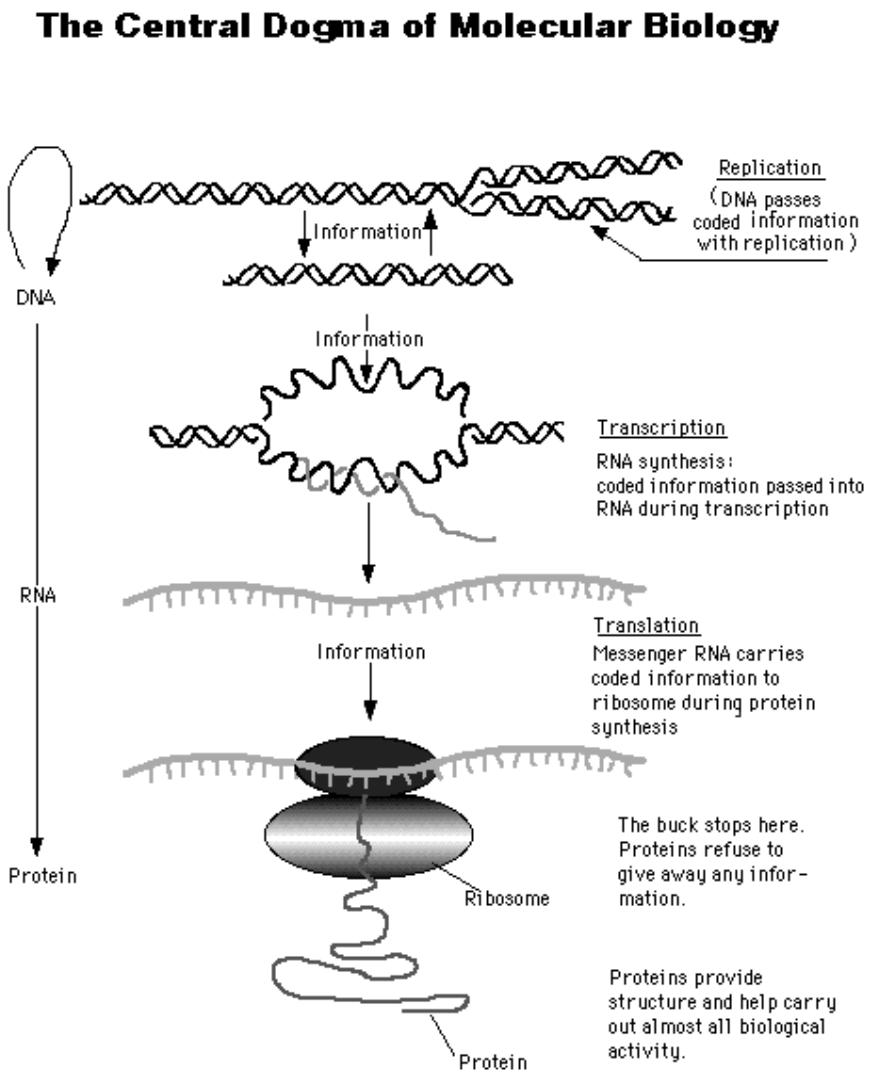
Det sentrale dogmet

DNA inneheld det genetiske materialet, det vil sei den informasjonen ein organisme treng for å fungere. (Det fins unntak i enkelte virus der det genetiske materialet er RNA.) DNA står og for overføringa av arvestoff til etterkommarane. I eukariote organismar (med cellekjerne) held DNA-molekylet seg inne i cellekjerna, medan proteina vert laga ute i cytoplasmaet utanfor kjernen. Det mellombelse molekylet som transporterar informasjonen ut av cellekjernen er RNA. Informasjonsflyten i biologi er oppsummert i det sentrale dogmet skrive av Francis Crick i 1958:

"The central dogma states that once 'information' has passed into protein it cannot get out again. The transfer of information from nucleic acid to nucleic acid, or from nucleic acid to protein, may be possible, but transfer from protein to protein, or from protein to nucleic acid, is impossible. Information means here the precise determination of sequence, either of bases in the nucleic acid or of amino acid residues in the protein."

F. Crick, 1958

På Figur 4 kan ein sjå det sentrale dogmet illustrert. Kvar overgong er produksjon av eit makromolekyl med utgangspunkt i sekvensen til eit eksisterande makromolekyl. Den generelle ideen er at eit makromolekyl kan brukast til å konstruere eit anna. Dei fascinerande detaljane av desse prosessane er grunnleggjande for alt liv. I dag er det sentrale dogmet utvida. Det fins eksempel på at RNA er mal for RNA. Og retrovirus kan kopiere sitt RNA-genom til DNA ved ei mekanisme kalla revers transkripsjon.

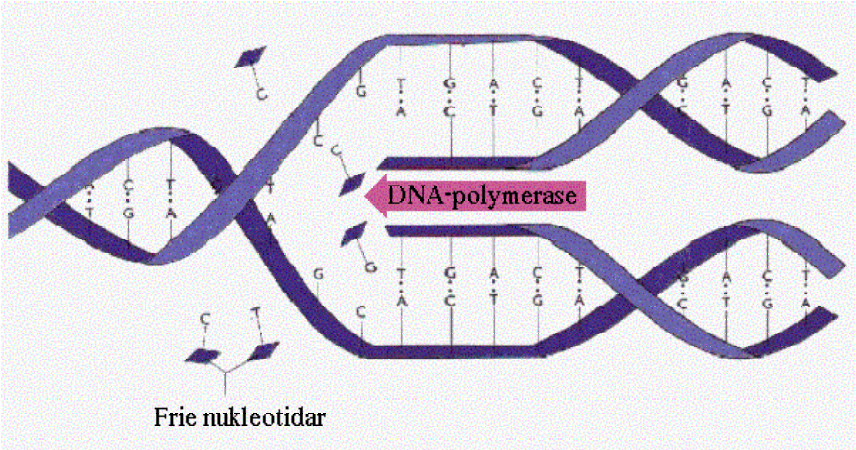


Figur 4. Øvst ser ein at DNA-molekylet kan reprodusere seg sjølv. Vidare kan DNA transkriberast til RNA. Nedst vert RNA translatert til protein. Biletet er henta frå http://www.cbs.dtu.dk/staff/dave/DNA_CenDog.html.

Det å lage nye molekyl kallast syntese. Syntesen av RNA, DNA og protein er svært kompliserte prosessar som her vert gjeve eit forenkla oversyn over.

Føremålet med DNA-syntesen er å lage ein kopi av alt arvestoffet i cella slik at ho kan dele seg og båe delane kan få eit sett kvar. DNA- syntesen byrjar med at den doble DNA-spiralen vert delt langs hydrogenbindingane av eit enzym som heiter DNA-polymerase (Figur 5). Dei to strengane av nukleotidar trekkjer, ved hjelp av

polymerasen, til seg enkle nukleotidar som bitt seg til sine komplementære motpartar på strengen. Det vert slik skapt to nye dobbeltrådar. Desse to dobbeltråda DNA-molekyla er identiske med det som fans i utgangspunktet.



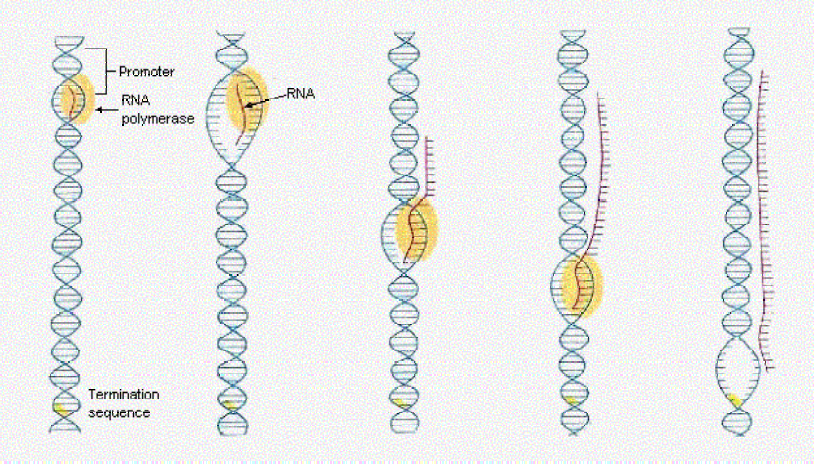
Figur 5. DNA-replikasjon. Dette bilete skildrar korleis enzymet DNA-polymerase skil hydrogenbindingane i eit DNA-molekyl og festar nye komplementære nukleotidar til dei to sidene. Både denne og neste figur er henta frå Eli-Anne Gjerde sine førelesingsnotat ved Universitetet i Bergen: http://www.uib.no/med/avd/miapr/arvid/MOD2_2001/Eli_Anne/4_dna_rna/

Under RNA-syntesen vert nukleotiden T i DNA bytta ut med U. Det vil sei at når DNA-molekylet vert delt og dei ulike frittsvevande nukleotidane skal feste seg for å danne ein komplementær RNA-streng, vert dei para slik det er gjeve i Tabell 1.

DNA	RNA
A	U
T	A
G	C
C	G

Tabell 1. Når RNA-polymerase les av eit gen og produserer eit RNA-molekyl, vert det også gjort ved å kople komplementære basar til DNA-tråden. Ein skilnad er at den komplementære basen til adenine ikkje er tymine men urasil.

Enzymet RNA-polymerase går langs DNA-molekylet og genererer eit komplementært RNA-molekyl (Figur 6).



Figur 6. RNA-syntesen. Her ser ein korleis enzymet RNA-polymerase forflyttar seg langs DNA-molekylet og genererer eit komplementært RNA-molekyl.

Det fins fleire sortar RNA. Dei som er blitt mest forska på kallast mRNA. mRNA vert, etter transkripsjonen, fanga opp av eit stort molekylkompleks kalla ribosom. Ribosomet tolkar koden i mRNA og syntetiserer det proteinet som er eintydig bestemt av rekkjefylja på nukleotidane i mRNA.

Denne prosessen kallast translasjon. Tre og tre nukleotidar dannar eit kodon. Det fins $4^3 = 64$ ulike kombinasjonar av tre nukleotidar (sidan det er 4 ulike typar). Eit kodon kodar for ei eintydig aminosyre. Sidan det er berre 20 aminosyrer og 64 kombinasjonar av nukleotidar i eit kodon, er det ein del aminosyrer som har fleire ulike kodon tilordna seg. Denne tilordninga av kodon til aminosyrer kallast den genetiske koden. Viser til Waterman (1995) for tabell over tilordninga.

Andre RNA er i seg sjølv verkingsfulle stoff og kodar difor ikkje for protein. Desse kallast ikkje-kodande RNA, ncRNA. Det er denne typen RNA denne oppgåva skal handle om og den vert skildra i neste avsnitt. *The central dogma* seier at informasjonsflyten i biologien går frå DNA til RNA til protein. Dette har resultert i at ein har trudd at det er protein som utføre alle strukturelle, katalytiske og regulatoriske oppgåver i cella. I den seinare tid har det vist seg at ncRNA har fleire viktige roller i cella (Storz 2002, Mattick 2003a).

2.2.2 ncRNA

I dette avsnittet vert det gjeve ei skildring av ncRNA-gen, kva som er kjent ved desse gena og kva som er hovudutfordringa framover. Denne hovudfagoppgåva er ein del av prosjektet *"Computational methods for identifying functional non-coding RNA genes in genomic sequences"* ved Mikrobiologisk institutt, Rikshospitalet. Dette prosjektet vert skildra i neste avsnitt.

Kva er eit gen?

Ein kan sjå på eit gen som ein del av DNA-strengen som gir opphav til eit produkt i cella. Det vil sei ein region som vert transkribert til RNA. Dersom dette RNA-produktet er eit mRNA som vert translatert til protein (Figur 4), vert det kalla eit proteinkodande gen. Dei gena som produserer ncRNA kallar ein for ncRNA-gen.

Kva er ncRNA?

Dei fyrste ncRNA-gena vart oppdaga i bakteriar, men er no skildra i alle typar levande celler. Desse molekyla har mange ulike roller i cella, mellom anna genekspresjon (Argaman *et al.* 2001). Dei to mest skildra klassane av ncRNA er tRNA og rRNA som er med i proteinsyntesen (Hesthagen og Langangen 1990).

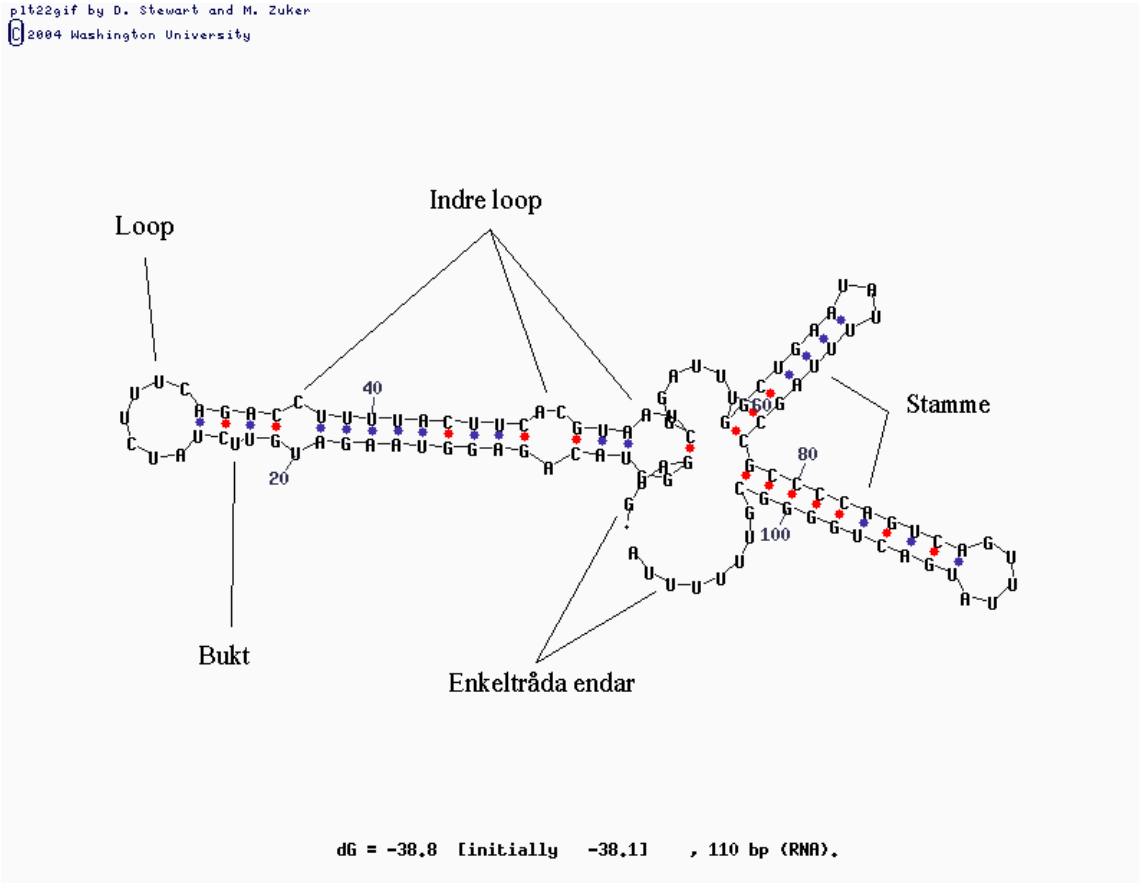
I Tabell 2 er det lista opp nokre ncRNA i lag med kva funksjon dei har i cella. For ein meir fyldig tabell vert det vist til Storz (2002), Hershberg (2003) eller http://www.sanger.ac.uk/Software/Rfam/browse/index.shtml.

ncRNA-type	Forkorting	Referanse	Funksjon
Transfer	tRNA	Hesthagen og Langangen (1990)	proteinsyntesen
Ribosomal	rRNA	Hesthagen og Langangen (1990)	proteinsyntesen
Small nucleolar	snoRNA	Dragon *et al*. (2002)	rRNA-syntese
Transfer-messenger	tmRNA	Lin-Chao *et al*. (1999)	Problemløysar i proteinsyntesen
Micro	miRNA	Storz (2002)	finjustering av utvikling

Tabell 2. Nokre ncRNA. Lista gjere eit lite oversyn over nokre av dei kjente ncRNA-gena med den vanlegaste forkortinga, referanse og kva funksjon det resulterande RNA-molekylet har i cella. I brødteksten er det vist til meir fyldige lister.

Sekundærstrukturen til RNA-molekyl

Funksjonen til eit ncRNA er gitt av den tredimensjonale strukturen den skapar ved å falde seg. Basisen for denne faldinga vert kalla sekundærstrukturen til RNA-molekylet, og kan skildrast i to dimensjonar ved å syne kva basepar som vert danna i denne faldinga. På Figur 7 ser ein RNA-molekylet *spf* som er eit ncRNA som er med på regulering av eit gen kalla *galK* (Polayes *et al.* 1988). Sekvensen til dette RNA-molekylet er nytta som input til programmet MFOLD (Zuker 2003) som har predikert ein sekundærstruktur.



Figur 7. Ein sekundærstruktur predikert av programmet MFOLD for RNA-molekylet *spf* som er med å regulerar genekspresjon for genet *galK*. MFOLD gir ein, utifrå sine kriteria, optimal og fleire suboptimale strukturar. Dette er den 3. mest optimale, og er valt for å syne dei ulike elementa i sekundærstrukturen til RNA. Figuren er generert av den nettbaserte versjonen av MFOLD (http://www.bioinfo.rpi.edu/applications/MFOLD) og deretter redigert i Paint.

Sjølve faldingsalgoritmen som vert nytta i MFOLD vart utvikla av Nussinov *et al*. (1978), medan scoringssystemet som byggjer på kjemisk energiminimering er utvikla av Zuker og Stiegler (1981) og vidareutvikla av mellom anna Mathews (1999). Dei raude og blå prikkane representerar basepar med ulik bindingsstyrke. Rekkjer av slike basepar kallast stamme (*stem*). Sirklane i endane på kvar stamme kallast ein loop. Ligg loopen mitt på ein stamme, er det ein indre loop (*internal loop*). Dersom loopen berre ligg på den eine sida kallast den ei bukt (*bulge*). Dersom ein loop har fleire stammar utifrå seg, kallast han ein fleirarma loop (*multi branched loop*).

Dei ulike basepara i RNA har ulik styrke. Når programmet MFOLD reknar ut den beste faldinga legg den ulik vekt på dei ulike basepara. Ein god men noko forenkla modell er å sei at desse basane parar seg:

```
A-U  G-U  G-C
```

Dette er ulikt DNA-basepar der berre A-T og C-G bitt seg til kvarandre. Med desse paringsreglane i bakhovudet er det lett å sjå at sekvensen:

```
U A G G A A A C U G
```

kan falde seg til ein stamme-loop-konstruksjon, òg kalla hårnål:

```
 U-G
 A-U
 G-C
G   A
 A A
```

Korleis ein kan representere denne strukturen på ein nytteleg måte vert skildra i avsnitta 2.5 og 2.6.

2.3 ncRNA-prosjektet ved Rikshospitalet

Ved bioinformatikkgruppa på Mikrobiologisk institutt på Rikshospitalet starta det eit ncRNA-prosjekt i 2001. Det overordna målet med prosjektet er å lage eit generelt søkeprogram som finn ncRNA-gen i ein DNA-sekvens. Her er det gjeve ei skildring av motivasjonen for prosjektet og metodane som er tenkt nytta. Dersom ikkje anna referanse er gjeven er stoffet henta frå prosjektskildringa (Lagesen og Rognes).

2.3.1 Kvifor ser ein etter ncRNA?

Dei siste åra har det våre lagt mykje energi i det å sekvensere genomet (finne rekkjefylja av alle nukleotidane i alle DNA-molekyla) til ulike artar. Sekvensseringa av det menneskelege genom (Lander *et al.* 2001, Venter *et al.* 2001) er det mest kjende prosjektet men absolutt ikkje det einaste. I skrivande stund (3. mars 2004) er det publisert 178 ferdig sekvenserte genom og det er 906 sekvenseiringsprosjekt på gong rundt om i verda. Desse tala er henta frå GOLD (*Genome Online Database*) (Kyrpides 1999, Bernal *et al.* 2001).

Men sjølv om ein organisme sitt genom er sekvensert, er det mykje av DNA-et som enno ikkje har kjent funksjon. Desse regionane utgjer store delar av genoma. I menneske er ca 1,5% av genomet proteinkodande (Mattick 2003b). Heilgenomstudiar av dei ferdig sekvenserte organismane har synt at talet på ulike proteinkodande gen hjå høgare organismar er lågare enn venta. Genomet til høgare organismar verkar mindre komplekse enn kva ein skulle tru. Noko av forklaringa kan finnast i ncRNA. Til dømes trudde ein at menneske hadde kring 100 000 proteinkodande gen, medan det viser seg å vere kring 30 000 (Mattick 2003b). Mattick hevdar også i denne artikkelen at så mykje som ¾ av det transkriberte materialet i menneske kan vere ncRNA.

Dei fleste ncRNA som er kjent er funne medan ein har leita etter andre ting og ikkje gjennom systematiske søk. Talet på kjente protein er auka dramatisk gjennom systematiske søk. Det er utvikla fleire program for å gjere slike søk, til dømes WEIL (Henderson *et al.* 1997) og GENSCAN (Burge og Karlin 1997).

Dette prosjektet har som mål å utvikle tilsvarande reisskap til leiting etter ncRNA-gen i genomiske sekvensar. Eit generelt leiteverkty for ncRNA-gen vil hjelpe på i prosessen med å forstå desse gena.

2.3.2 Korleis ser ein etter ncRNA?

Målet for prosjektet er å utforske ulike metodar for å lokalisere ncRNA-gen i genomiske sekvensar. Fokuset vil i byrjinga bli på å finne metodar for lokalisering av ncRNA-gen i bakteriar. Bakteriegenom er oftast betre dokumentert enn andre, noko som vil hjelpe på vegen med nokre av strategiane. Målet lyt likevel vere å utvikle metodane slik at dei kan nyttast på andre genom også. Det bør verte mogleg for andre forskarar å nytte programmet for å leite etter ncRNA i deira dokumenterte genom. Her kjem ei skildring av nokre av metodane som vil verte granska i dette prosjektet.

Sekvenslikskap

Ein kan leite etter likskap mellom kjente ncRNA-gen og dokumenterte genom. Til dette kan ein nytte velprøvde program for lokal samanstilling av søkesekvens mot ein database til dømes ParAlign (Rognes 2001), FASTA (Pearson og Lipman 1988) og BLAST (Altschul *et al.* 1990). Ein kan òg lage ein profil av samanstilte representantar av ein klasse ncRNA. Dette kan gjerast med ein teknikk kalla *Hidden Markov Model* (HMM)(Durbin *et al.* 1998). Ei ulempe med søk etter sekvenslikskap er at metoden manglar generalitet og kan ikkje vere med å oppdage nye klassar av ncRNA. Derimot kan den nyttast til å finne ut om nye ncRNA-gen, funne på andre måtar, er av ein ny klasse eller ein som er kjent frå før. Denne metoden vil og vere ein nyttig veg å gå når ein arbeider med nydokumenterte genom for å finne ut om dei inneheld kjente ncRNA-gen.

Strukturlikskap

Det fins måtar å skildre sekundærstrukturen til kjente ncRNA. På same måte som med sekvenslikskap kan ein sjå etter likskap til denne strukturen (Klein og Eddy 2003). Sjølv om sekvensen har endra seg so mykje gjennom evolusjon at han ikkje er gjenkjenneleg frå ein art til ein anna, kan den to- og tredimensjonale strukturen vere bevart. Det går og an å lage ein profil av sekundærstrukturen til ei multippel

samanstilling av ein klasse ncRNA. Denne teknikken kallast samvariansmodell (CM) (Eddy og Durbin 1994). Ulempa og føremonene som er diskutert for sekvenslikskap gjeld og for strukturlikskap, nemleg at ein berre kan sjå etter kjente ncRNA.

Statistisk analyse

Statistisk analyse av genomiske sekvensar kan og brukast for å skilje ncRNA-gen frå tilfeldig DNA. Ein metode som har vore nytta er å sjå etter hyppigheita av dei etterfyljande nukleotidane CG. Dette har ført til funn av nye ncRNA i *Methanococcus jannaschii* (Schattner 2002) (Klein *et al.* 2002). Det er mogleg at andre slike variasjonar og kan vere med å hjelpe til i leitinga etter nye ncRNA-gen.

Transkripsjonssignal

Transkripsjonssignal er sekvensar som ligg inntil eit gen og styrer transkripsjonen frå DNA til RNA. Blant desse fins det og sekvensar som hjelper til med reguleringa av translasjonen frå RNA til protein. Desse sekvensane skal ikkje vere til stades ved ncRNA-gen. Ein kan difor leite etter sekvensar med transkripsjonssignal men ikkje translasjonssignal. Grunna variasjon i desse signala mellom organismar må denne metoden vere fleksibel på kva for sekvensar den skal sjå etter. Denne metoden har mellom anna vore brukt for å finne ncRNA-gen i *Escherichia coli* (Argaman *et al.* 2001).

Genomsamanlikning.

Sekvensar som kodar for liknande protein i ulike organismar, er ofte konserverte. Dette vil og vere tilfelle for ncRNA-gen. Ein kan difor leite etter slike gen ved å samanlikne regionar som ligg mellom kjente proteinkodande gen (intergenetiske områder) og sjå etter subsekvensar som er meir konserverte enn andre. Fleire prosjekt av genomsamanlikning har synt metoden vellukka (Rivas og Eddy 2001, Rivas *et al.* 2001). Denne metoden vil bli meir og meir attraktiv etter som talet på sekvenserte genom stig.

2.3.3 Målet med denne oppgåva

I denne oppgåva vert det sett nærare på strukturlikskap og sekvenslikskap. Denne oppgåva skal

- gjere greie for kva metodar som kan nyttast på området
- gje ei grundig innføring i den viktigaste metoden
- prøve ut eksisterande programvare
- legge fram og prøve ut endringar til den mest aktuelle modellpakken
- gjere greie for korleis desse metodane best kan nyttast i dette prosjektet

I neste avsnitt vert det gjort greie for tidlegare studiar på dette området. I dei påfyljande avsnitta vert den viktigaste teknikken på området, CM-pakken, presentert. Fyrst den teoretiske bakgrunnen og dernest korleis ein kan implementere denne teknikken. I avsnitt 3.1 vert det lagt fram eit forsøk som er gjort med eit program som nyttar ein CM. Det vert diskutert i kapittel 5 korleis denne modellen best kan nyttast i dette prosjektet.

2.4 Homologisøk, RNA-folding og dynamisk programmering

I dette avsnittet vert det skildra kva som er gjort innan søking i genomiske sekvensar etter homologar til kjente ncRNA-molekyl. Ein seier at to sekvensar er homologe, eller at ein sekvens er homolog til ein annan, dersom dei er av same evolusjonære opphav, utan at sekvensane treng å vere heilt like kvarandre av den grunn. Når ein søker i ein database etter homologar er ein nødt til å definere eit likskapskriterium. Det er dette kriteriet som skil dei ulike søkemetodane.

2.4.1 Samanstilling

Alle metodar som seier noko om likskapen mellom sekvensar nyttar ei form for samanstilling (*alignment*). Her er vist eit døme på Needelman/Wunsch- samanstilling (Needleman og Wunsch 1970):

Kor like er strengane s og t og kva er likskapen mellom dei:

```
s = AACC     t = AGTCGC
```

Needelman /Wunsch (NW) er ei global samanstillingsalgoritme. Det vil sei at den finn den beste samanstillinga (dei beste samanstillingane) for heile dei to strengane. Dette i motsetnad til lokal samanstilling som plukkar ut områder der strengane liknar kvarandre og gjev dei score etter likskapsgrad. NW er ei algoritme av typen dynamisk programmering. I dynamisk programmering ser ein på delproblem av heile problemet og løyser delproblema kvar for seg. Ein føresetnad for å kunne nytte denne typen algoritme er at løysinga på alle delproblema til saman er ei løysing på heile problemet. I NW-algoritmen deler ein opp problemet ved å sjå på samanstilling av alle prefiks i kvar streng.

Algoritmen går ut på å fylle ei todimensjonal matrise frå øvst i venstre hjørnet til nedst i høgre, der den totale scoren til samanstillinga vil ligge til slutt. Fyrst av alt lyt ein definere nokre scoringsverdiar. Dersom teiknet i dei to strengane er like har vi ein match. Ein match gir i dette dømet score $f(i,j)_{Match} = 5$. Dersom dei er ulike er det ein mismatch som gir $f(i,j)_{Mismatch} = -1$. For å sei dette med ord: Funksjonen av i og j er 5 dersom teikn i i streng s er lik teikn j i streng t medan dersom dei er ulike er funksjonen av i og j lik -1. Dersom ein er nøydt til å justere lengda på den eine strengen ved å setje inn gap, straffar ein det med $g = -3$.

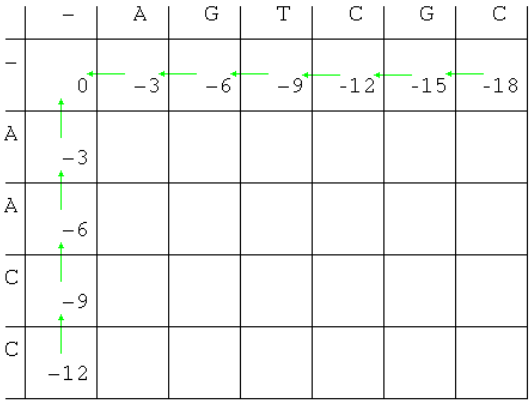
Algoritmen kan framstilast slik:

- Lag ei matrise der den eine strengen står langs overkanten og den andre nedover langs venstrekanten. Strengane skal vere forlenga med eit gap-teikn i starten. Her er dette teiknet '-' (Figur 8).

	–	A	G	T	C	G	C
–							
A							
A							
C							
C							

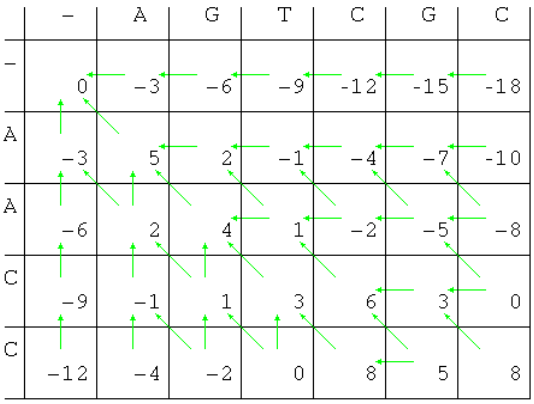
Figur 8. Dynamisk programmering-matrise. Strengen s står langs venstrekanten medan strengen t står langs overkanten. Ein kan tenkje seg at strengane er indeksert få 0 med høvesvis indeksane *i* og *j*. Denne og dei tre neste figurane er laga i Paint med utgangspunkt i utskrift frå ein Java-implementasjon av Needelman/Wunsch-algoritmen.

- Set verdien i øvste venstre hjørnet til $w_{0,0} = 0$.
- For dei andre rutene i den venstre kollonna, set rute i til $w_{i,0} = w_{i-1,0} + g$. Teikn ei pil til ruta over.
- For dei andre rutene i øvste rada, set rute j til $w_{0,j} = w_{0,j-1} + g$. Teikn ei pil til ruta til venstre.
- No er matrisa initsialisert (Figur 9).



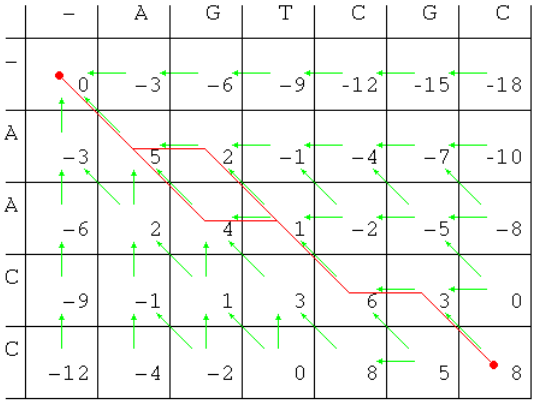
Figur 9. NW-matrisa er initsialisert. No kan ein fylle ut matrisa ved å byrje i rute (1,1). Dette vert gjort ved å maksimere over tre tal: 1. verdien i ruta over pluss gapstraff (-3 + (-3) = -6) 2. verdien i ruta til venstre pluss gapstraff (-3 + (-3) = -6) 3. verdien i ruta over til venstre bluss matchscore (0 + 5 = 5). Det skal altso stå 5 i denne ruta. Sidan vi fekk den maksimale verdien frå ruta over til venstre, skal det stå ei pil dit.

- For alle andre ruter, set verdien i rute i, j til $w_{i,j} = \max\begin{cases} w_{i,j-1} + g \\ w_{i-1,j} + g \\ w_{i-1,j-1} + f(i,j) \end{cases}$. Teikn ei pil i alle dei retningane som gav maksimal verdi (Figur 10).



Figur 10. NW-matrisa er fylt og den totale scoren er i ruta nedst i høgre hjørnet. Verdien i kvar rute svarar til scoren for å samanstille det tilhøyrande prefikset i kvar streng til kvarandre.

Når alle rutene er fylt ligg den totale scoren for samanstillinga i ruta nedst til høgre (her 8). For å finne sjølve samanstillinga må ein starte nedst i høgre hjørnet og fylje pilene attende (Figur 11). Dersom det er fleire alternative vegar, vert dette ulike samanstillingar som alle har maksimal score. Her er det to vegar attende og dette gjev dei to samanstillingane som er vist under figuren.



Figur 11. NW-matrise. Her er vegen attende merka med raudt. Det er to moglege veger attende langs pilene. Dette svarar til dei to ulike globale samanstillingane for desse to strengane.

```
A-AC-C
|  | |
AGTCGC

AA-C-C
|  | |
AGTCGC
```

Her er det nytta ei lineær gapstraff. Det vil sei at det "kostar" det same å innføre ein gap som å utvide den. Det er vanleg å skilje mellom *gap-open* og *gap-extend* der det er mindre gunstig å innføre ein gap enn kva det er å utvide den.

Dette dømet syner prinsippet i samanstilling og det vert her ikkje gjort greie for korleis dei ulike metodane nyttar dei ulike samanstillingsalgoritmane. I lokale samanstillingar vert det funne delstrengar med høg score i staden for å samanstille heile dei to

strengane. Den mest nytta algoritmen for lokal samanstilling kallast Smith/Waterman (Smith og Waterman 1981).

Søkemetodane BLAST (Altschul *et al.* 1990) og FASTA (Pearson og Lipman 1988) nyttar båe heurestikkar for raskt å gjere lokale samanstillingar mellom ein søkesekvens og sekvensane som ligg i den aktuelle databasen. Dei er ganske selektive men det er deira raske køyretid som er deira store styrke. Programmet ParAlign (Rognes 2001) kan samanliknast med Smith/Waterman i selektivitet og med FASTA i køyretid og er dermed ei klar forbetring.

Som vert vist seinare i denne oppgåva er dynamisk programmering ein kraftfull tilnærming til mange optimaliseringsproblem. Det å finne den optimale faldinga for eit RNA-molekyl er eit slikt problem. Nussinov *et al*. (1978) skildra ei algoritme som optimaliserte denne faldinga ved å maksimere talet på basepar. Variantar av denne algoritmen vert nytta av CM-pakken.

2.4.2 Profilsøk

Neste steg på vegen kan ein sei er det å nytte profilar eller sokalla *Hidden Markov Models* (HMM). HMM kan ein nytte når ein vil søke etter ein ny homolog til ei mengde av homologe sekvensar. I Figur 12 er det laga ei samanstilling av tolv sekvensar. For å kunne søke i ein database etter sekvensar som liknar på denne, må ein lage ein profil. Ein enkel profil er sokalla *position specific scoring matrices* (PSSM). Desse består av sannsyner for kvar av nukleotidane i kvar posisjon. Denne sannsyna kan ein estimere ved hjelp av dei observerte frekvensane (Figur 13).

pos	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
	T	T	A	T	A	T	G	G	G	A	T	A	C	T	A	G	T	G	C	C	G	G	C	C	A	A	A	G	A	T	T	T
	G	G	G	C	C	A	T	C	A	T	G	G	C	C	C	A	T	T	G	C	G	C	A	T	A	C	A	G	A	T	T	T
	A	A	T	G	C	G	T	A	C	C	G	C	C	A	T	C	T	A	T	G	C	A	T	G	A	T	A	G	A	T	T	T
	C	C	A	T	G	A	G	A	T	T	C	A	A	T	G	A	T	C	A	G	C	T	G	T	A	G	A	G	A	T	T	T
	T	A	C	A	G	C	T	G	A	G	C	T	A	G	T	T	T	T	C	G	C	G	A	A	A	A	A	G	A	T	T	T
	G	T	T	C	G	G	A	C	C	C	C	C	G	T	A	A	C	T	T	T	T	A	A	A	G	A	C	A	G	A	T	T	T
	A	C	T	T	T	T	T	T	T	A	A	A	T	A	G	A	T	C	T	A	T	A	G	T	A	T	A	G	A	T	T	T
	C	A	C	T	A	C	T	G	G	G	T	A	G	G	T	C	T	A	C	A	T	G	T	G	A	G	A	G	A	T	T	T
	T	C	A	C	A	C	A	C	A	G	T	G	G	T	G	T	T	A	T	A	T	A	T	A	A	A	A	G	A	T	T	T
	G	C	C	C	C	C	C	G	G	G	G	G	C	G	G	G	T	G	G	G	C	C	C	C	A	C	A	G	A	T	T	T
	A	T	T	T	T	T	T	T	T	A	A	A	C	A	A	T	T	T	T	T	T	A	A	A	A	A	T	A	G	A	T	T	T
	C	C	G	T	A	A	C	G	A	G	C	A	A	T	T	G	T	A	C	C	G	G	T	C	A	G	A	G	A	T	T	T

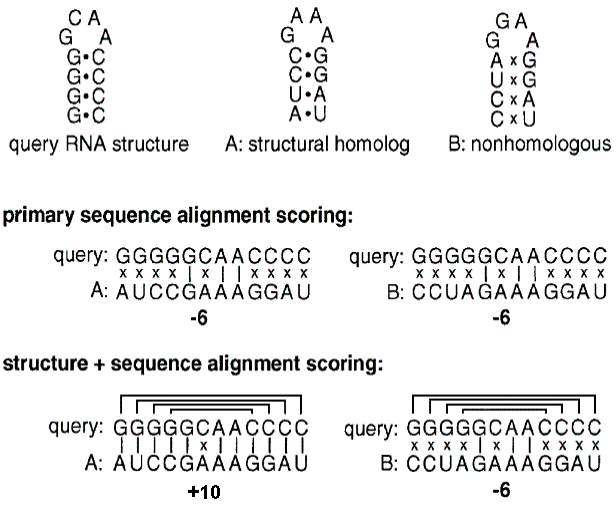
Figur 12. Ei multippel samanstilling av 12 dømesekvensar. Desse sekvensane er ikkje henta frå verklege data. Dei er konstruert som eit fyljeeksempel som vil verte referert til seinare i oppgåva. Både denne og neste figur er utskrifter frå Java-programmet som implementerar delar av CM-pakken og som er laga i høve denne oppgåva.

	A	T	C	G	-
pos 0	0.25	0.25	0.25	0.25	0.0
pos 1	0.25	0.25	0.4166	0.0833	0.0
pos 2	0.25	0.3333	0.25	0.1666	0.0
pos 3	0.0833	0.5	0.3333	0.0833	0.0
pos 4	0.3333	0.1666	0.25	0.25	0.0
pos 5	0.25	0.25	0.3333	0.1666	0.0
pos 6	0.1666	0.5	0.0833	0.25	0.0
pos 7	0.1666	0.1666	0.25	0.4166	0.0
pos 8	0.3333	0.25	0.1666	0.25	0.0
pos 9	0.25	0.1666	0.1666	0.4166	0.0
pos 10	0.1666	0.25	0.3333	0.25	0.0
pos 11	0.5	0.0833	0.0833	0.3333	0.0
pos 12	0.25	0.1666	0.4166	0.1666	0.0
pos 13	0.3333	0.3333	0.0833	0.25	0.0
pos 14	0.25	0.3333	0.0833	0.3333	0.0
pos 15	0.25	0.25	0.25	0.25	0.0
pos 16	0.0	1.0	0.0	0.0	0.0
pos 17	0.3333	0.3333	0.1666	0.1666	0.0
pos 18	0.0833	0.4166	0.3333	0.1666	0.0
pos 19	0.25	0.1666	0.25	0.3333	0.0
pos 20	0.1666	0.25	0.3333	0.25	0.0
pos 21	0.4166	0.0833	0.1666	0.3333	0.0
pos 22	0.3333	0.3333	0.1666	0.1666	0.0
pos 23	0.25	0.25	0.25	0.25	0.0
pos 24	1.0	0.0	0.0	0.0	0.0
pos 25	0.25	0.25	0.25	0.25	0.0
pos 26	1.0	0.0	0.0	0.0	0.0
pos 27	0.0	0.0	0.0	1.0	0.0
pos 28	1.0	0.0	0.0	0.0	0.0
pos 29	0.0	1.0	0.0	0.0	0.0
pos 30	0.0	1.0	0.0	0.0	0.0
pos 31	0.0	1.0	0.0	0.0	0.0

Figur 13. Frekvensane til kvar av nukleotidane i kvar av posisjonane i samanstillinga i førre figur.

Dei fleste ncRNA har konservert bestemte basepar i sekundærstrukturen meir enn sjølve sekvensen av nukleotidar. RNA-sekundærstruktur induserer sterke parvise samanhengar i sekvensen, oftast som komplementære basepar som nemnt i avsnitt

2.2.2. Skal ein søkje etter ncRNA-gen i databasar er det altso ikkje tilstrekkeleg med berre ein profil av sekvensen, ein er og avhengig av å kunne skildre desse samanhengane (Eddy og Durbin 1994). I Figur 14 er det synt korleis ei samanstilling som tek omsyn til sekundærstrukturen gir ein rettare score enn den som berre ser på sekvensen.



Figur 14. Samanstilling. Kvar samanstilling har fått ein score som seier noko om kor like sekvensane er kvarandre. I midten kan ein sjå samanstilling utan omsyn til sekundærstruktur. Her er det gjeve eit poeng for match og eit minuspoeng for kvar mismatch. Sekvensane A og B får i dette tilfellet like høg score. Nedst er det teke omsyn til sekundærstrukturen ved i tillegg å gje to poeng for kvart basepar som høver i strukturen. No vert det stor skilnad mellom samanstillingane. Figuren er henta frå Eddy (2002).

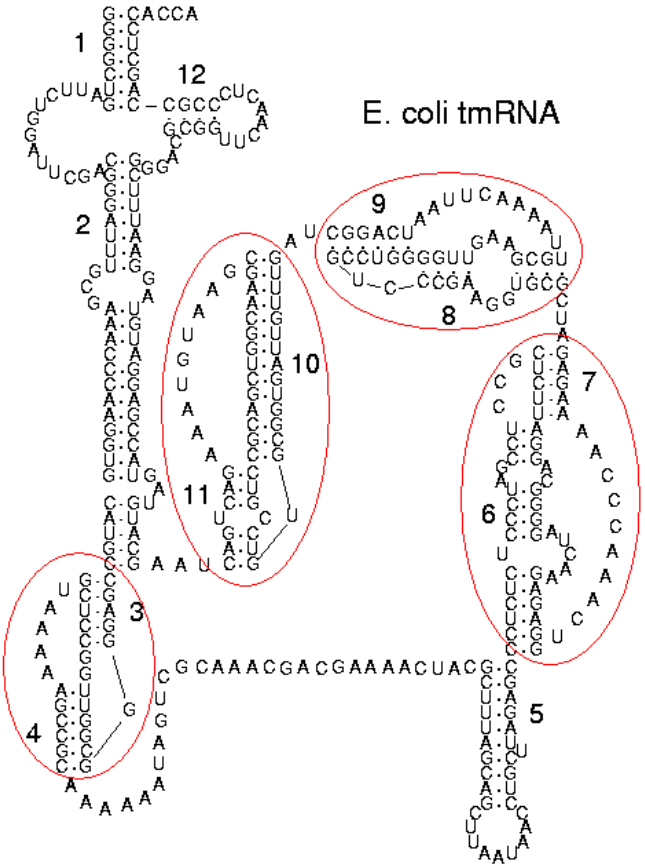
Neste steg er å lage profilar av sekundærstrukturen til ei multippel samanstilling. Det er dette som kallast ein samvariansmodell (CM). Ein CM er nettopp ein modell av sekundærstrukturen til ei multippel samanstilling. I avsnitt 2.5 er teorien for denne teknikken lagt fram og i 2.6 vert samvariansmodellen referert. tRNAscan-SE (Lowe og Eddy 1997) er eit program som implementerar CM-pakken og nyttar den til å leite etter tRNA-gen i genomisk materiale. I denne oppgåva er tRNAscan-SE prøvd ut på bakterien *E. Coli* og dette forsøket er skildra i avsnitt 3.1.

2.4.3 Andre innfallsvinklar til homologisøk

Profilar av sekundærstruktur implementert som CM-ar ser ut til å vere den beste måten å skildre ein klasse av kjente ncRNA-gen. I dette avsnittet vert det gjort greie for andre studie på området.

Pseudoknutar

Det er utvikla utvidingar til samvariansmodellen som innlemmar den ikkjenøsta strukturen pseudoknutar (Figur 15). Dette arbeidet vart presentert i to artiklar der den fyrste (Rivas og Eddy 1999) skildrar algoritmane og den andre (Rivas og Eddy 2000b) skildra teorien. I det generelle tilfellet er pseudoknutar i ein meir kompleks bereknbarheitsklasse der algoritmane vert eksponensielle. Deira løysing på dette problemet er å lage små polynomielle utvidingar til samvariansmodellen som fangar opp desse strukturane. Dette har resultert i attkjenningsalgoritmar som har ein tidskompleksitet på $O(L^6)$, og ein minnebruk på $O(L^4)$ (Rivas og Eddy 2000b), der L er lengda på sekvensen ein freistar å tilpasse til modellen. Sjølv om dette er polynomielt vert kompleksiteten for høg i høve til det ein vinn på å ta med pseudoknutar i strukturskildringa. Minst halvparten av basepara i ein psudoknute kan skildrast som nøsta samanhengar. Dette vert nærare kommentert etter at teorien for CM er gjennomgått.



Figur 15. Sekundærstrukturen til *ssrA* (small stable RNA) frå *E. coli* (også kalla tmRNA på grunn av sine kombinerte tRNA og mRNA-eigenskapar). Strukturane som er ringa inn er sokalla pseudoknutar. Desse hårnålene grip inni kvarandre på eit ikkjenøsta vis. Biletet er henta frå RNA-databasen Rfam: http://www.sanger.ac.uk/Software/Rfam/gifs/families/RF00023.jpg

Neurale nett

Neurale nett som tek omsyn til mellom anna sekundærstruktur i kjente ncRNA-molekyl er prøvd ut (Carter *et al.* 2001). Neurale nett er ein modell som kan trenast opp på eit treningssett av kjente ncRNA-gen og leite i genomiske sekvensar etter vindauge med dei same ”eigenskapane”. Denne metoden har synt seg brukbar og har resultert i funn av fleire nye ncRNA i m.a. ulike bakteriar.

Søk etter sekundærstrukturrelevante homologar til eit einskilt ncRNA

RSEARCH er eit søkeprogram som søkjer etter homologar til eit enkelt ncRNA-molekyl ved å sjå både etter sekvenslikskap og likskap med sekundærstrukturen til molekylet (Klein og Eddy 2003). Dette programmet bygger i store trekk på

samvariansmodellen sin måte å skildre sekundærstruktur på og nyttar dei same algoritmane for søking. Skilnaden er at RSEARCH ikkje nyttar ein profil av fleire samanstilte sekvensar men derimot eit einskilt ncRNA-gen.

I tabell 3 er det satt opp ei lita oversikt over nokon av dei ulike programma og metodane som er nemnt over.

	Sekvenslikskap	Sekundærstruktur
Enkel sekvens	FASTA, BLAST, ParAlign	RSEARCH
Profil av multippel samanstilling	HMM, PROSITE	CM, tRNAscan-SE, Rfam

Tabell 3. Ei systematisk inndeling av ulike søkemetodar. Proteinprofildatabasen PROSITE vert omtala i eit seinare kapittel.

2.4.4 Andre former for innhaldssøk

Samleomgrepet innhaldssøk omfattar alle metodar som freistar å skilje ut gen frå ein genomsekvens ved hjelp av det som faktisk skil sjølve genet frå resten av strengen. På den andre sida står signalsøk som ser etter det som ligg før og etter genet.

Når ein ser etter nye ncRNA-gen i ein genomisk sekvens ynskjer ein at det skal vere eigenskapar ved den delen av strengen som er eit ncRNA-gen som kan skilje det frå resten av strengen. Ein slik eigenskap kan vere at desse områda har lettare for å forme stabile sekundærstrukturar generelt. Dette har Rivas og Eddy (2000a) gjort eit forsøk på og funne at sekundærstruktur aleine ikkje er statistisk signifikant i det generelle tilfellet. Dei leita med eit fast vindauge gjennom strengen og freista å finne optimale faldingar. Alle faldingar får ein score som seier noko om kor god den er. Denne scoren var ikkje merkbart betre i dei områda der det fans kjente ncRNA-gen enn utanfor.

I 2003 lagde Lee P. Lim eit program som leitar etter ncRNA-gen av ein klasse han kallar microRNA (Lim *et al.* 2003a, Lim *et al.* 2003b). Programmet (MiRscan) er eit spesialbygd program for å finne nettopp denne typen gen. Det nyttegjer seg av m.a. ein heilt spesiell sekundærstruktur som vert skapa mellom genet og ei forlenging av

strengen i 5'-retningen. MiRscan er ein typisk representant for ei mengde program som nyttar sekundærstruktur men i spesialbygde ufleksible modellar. Ein nyare representant for denne kategorien er ARAGORN (Laslett og Canback 2004), eit program som ser etter tRNA og tmRNA-gen.

2.5 Grammatikkar

Før samvariansmodellen kan forståast, må ein ha på plass ein del teori. Her vert teorien om grammatikkar presentert. I neste avsnitt vert denne teorien gjeve eit innhald ved at den vert implementert i samvariansmodellen.

Dette avsnittet er stort sett henta frå bøkene "*Biological sequence analysis*" (Durbin *et al.* 1998) og "*Elements of the theory of computation*" (Lewis og Paradimitriou 1998). Notasjon og grammatikkteori er henta frå sistnemnte medan dømer og nytting på biologiske data er henta frå den fyrste.

2.5.1 Generelt

Alfabet

Eit alfabet er ei endeleg mengde teikn. Det er vanleg å kalle denne mengda Σ. Mengda av alle strengar (også kalla tekststrengar, ord eller setningar) som kan lagast over eit alfabet Σ, kallar ein Σ^*. Eit lite døme vil illustrere poenget: Gitt alfabetet $\Sigma = \{a, b\}$. Då definerer ein $\Sigma^* = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, ...\}$, der ε er den tomme strengen. Mengda Σ^* er altso mengda av alle strengar, av alle lengder over alfabetet Σ.

Språk

Eit språk er ei delmengde av alle strengar over eit alfabet. Gitt alfabetet Σ frå førre avsnitt kan vi definere språk både som ei endeleg mengde, som språket $L = \{abba, babba, a, aa\}$, eller ei uendeleg mengde som språket $U =$ {alle strengar x

som er slik at alle x inneheld to etterfyljande a-ar}. I det siste tilfellet vil til dømes strengane $aa, baa, aaa, bbaabba$ vere med i språket U medan a og $abba$ ikkje er med.

Reglar

Eit språk kan definerast ved hjelp av ein grammatikk. Ein grammatikk har ei endeleg mengde grammatiske reglar. Denne mengda er ofte kalla R. Til dømes kan språket U frå førre avsnitt skildrast med reglane:

$$R = \{$$
$$S \rightarrow TaaT,$$
$$T \rightarrow Ta \mid Tb \mid \varepsilon\}$$

der S og T er hjelpesymbol kalla ikkje-terminalar. Fyrste regelen seier at ikkje-terminalen S kan skrivast om til $TaaT$ der T er ein ny ikkje-terminal og aa er teikn frå alfabetet som strengane er over, gjerne kalla terminalar. Andre regelen seier at ikkje-terminalen T kan skrivast om til Ta, Tb eller den tomme strengen. Logikken i namna ”terminal” og ”ikkje-terminal” er at reglane i ein grammatikk ”produserer” ein streng. Denne produksjonen held fram so lenge det fins fleire ikkje-terminalar i strengen men terminerer når det berre fins terminalar. Det er vanleg å skrive ikkje-terminalane som store bokstavar og terminalane som små. Blant ikkje-terminalane er det alltid eit startsymbol, som i fylje konvensjonen oftast er S. Dersom ein startar med dette startsymbolet og nyttar reglane i grammatikken kan ein utleie alle strengar som er med i språket. På denne måten seier ein at språket er definert av grammatikken.

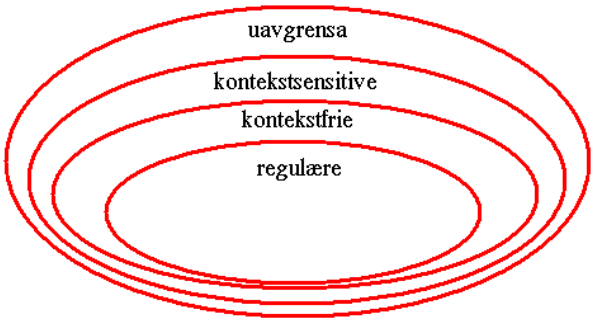
Formell definisjon av grammatikk

Ein grammatikk er eit firtuppel: $G = (V, \Sigma, R, S)$, der V er ei endeleg mengda av alle terminalar og ikkje-terminalar, Σ er mengda av terminalar (ei delmengde av V), R er mengda av omskrivingsreglar og S er startsymbolet (eit element frå $V - \Sigma$.

Med denne teorien på plass kan språket U, definert over, definerast på nytt ved hjelp av ein grammatikk: språket U er gitt ved grammatikken $G = (V, \Sigma, R, S)$, der $V = \{a, b, S, T\}$, $\Sigma = \{a, b\}$, $R = \{S \rightarrow TaaT, T \rightarrow Ta \mid Tb \mid \varepsilon\}$ og S er startsymbolet.

Eit hierarki av grammatikkar

Chomsky (1959) skildra fire ulike typar restriksjonar på grammatiske omskrivingsreglar. Dei resulterande fire klassane av omskrivingsgrammatikkar fell inn i eit hierarki kjent som Chomsky's hierarki for omskrivingsgrammatikkar (Figur 16). Desse klassane vert skildra i dei fyljande avsnitta. I desse avsnitta er W nytta for å representere ein vilkårleg ikkje-terminal og a for å representere ein vilkårleg terminal.



Figur 16. Chomskys hierarki av omskrivingsgrammatikkar, nøsta i tråd med dei aukande restriksjonane som vert lagde på produksjonsreglane i grammatikkane. Når ein ser på kva produksjonsreglar som er lov er regulære grammatikkar dei enklaste og mest avgrensa og er difor enklast å parse. Regulære grammatikkar er også dei som har minst høve til å skildre strukturelle samanhengar i strengar. Figuren er laga i Paint med utgangspunkt i ein figur i Durbin *et al.* (1998).

2.5.2 Regulære grammatikkar

Ein regulær grammatikk inneheld berre reglar på forma $W \to aW$ og $W \to a$. I tillegg er reglar på forma $W \to \varepsilon$ med for å terminere strengen. Essensielt produserar ein regulær grammatikk sine strengar frå venstre mot høgre. Regulære grammatikkar kan ikkje skildre samanheng mellom fjerntliggande terminalar i ein streng. Ein kan sei at regulære grammatikkar modellerar berre sekvensen eller primærstrukturen.

Eit enkelt døme

Her vert språket som består av alle "strengar av a og b som inneheld eit odde tal av a ar" skildra. Dette språket kan definerast ved hjelp av grammatikken $G = (V, \Sigma, R, S)$, der

$V = \{a, b, S, T\}$
$\Sigma = \{a, b\}$
$R = \{$
$S \to aT \mid bS$
$T \to aS \mid bT \mid \varepsilon\}$
S er startsymbolet.

Dersom ein streng inneheld eit odde tal av a ar er utleiinga i ikkje-terminal T og dersom den inneheld eit partal av a ar er den i ikkje-terminal S. Sidan den berre kan terminere frå T sikrar ein at ein berre produserar strengar med odde tal av a ar.

Eit døme på DNA

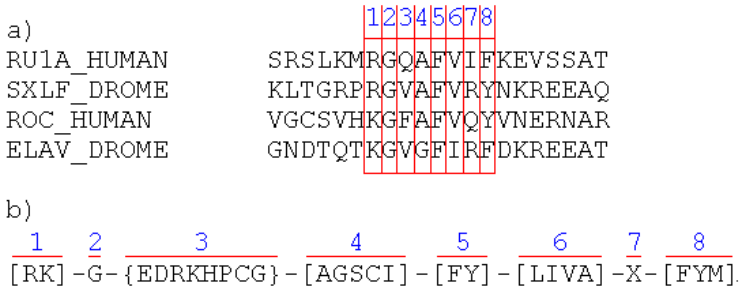
Det humane FMR-1-genet har ein sekvens som inneheld ein triplettrepetisjonsregion der sekvensen cgg er repetert mange gonger. Talet på triplettar er sterkt varierande mellom individ og eit auka tal av kopiar er assosiert med *fragile X syndrom*, ein genetisk sjukdom som orsakar mental tilbakesetjing og andre symptom hjå eit av 2000 barn. Denne regionen er starta med sekvensen gcg og avslutta med sekvensen ctg. Det som skal lagast er altso ein grammatikk som godtek språket med strengar på denne forma: $gcg[cgg]^i ctg$, der det som stå i klammeparentes kan takast oppatt ein eller fleire gonger ($i \geq 1$). Denne grammatikken kan skildrast slik:

$G = (V, \Sigma, R, S)$ der
$V = \{a, c, t, g, W_1, W_2, W_3, W_4, W_5, W_6, W_7, W_8, S\}$
$\Sigma = \{a, t, c, g\}$
$R = \{$
$S \to gW_1$
$W_1 \to cW_2$
$W_2 \to gW_3$
$W_3 \to cW_4$
$W_4 \to gW_5$
$W_5 \to gW_6$
$W_6 \to cW_7 \mid cW_4$
$W_7 \to tW_8$
$W_8 \to g\}$
S er startsymbolet.

Eit døme på protein

Et godt døme på biologisk nytting av regulære grammatikkar er PROSITE-databasen (Bucher og Bairoch 1994, Sigrist *et al.* 2002). Ein PROSITE førekomst inneheld eit mønster for sterkt konserverte signatursekvensar innehaldt i alle, eller nesten alle, proteina i ein proteinfamilie. Ulikt metodar som gir ei samanstilling av sekvensar ei vekt, passar eit PROSITE-mønster til ein sekvens, eller ikkje. Den er ein regulær grammatikk som sjekkar om ein streng er med i språket den definerar.

Eit PROSITE-mønster er ein streng som inneheld mønsterelement separert av bindestrekar og terminert av eit punktum (Figur 17b). I eit mønsterelement indikerar ein bokstav einbokstavskoden til ei av aminosyrene, hakeparentesar indikerer at ei vilkårleg av dei oppramsa aminosyrene kan førekomme, krøllparentesar indikerer at ei vilkårleg aminosyre utanom dei oppramsa kan førekomme og X indikerer at ei heilt vilkårleg aminosyre kan førekomme på denne plassen.



Figur 17. a) Ei multippel samanstilling av sekvensar som viser det sterkt konserverte "RNP-1" - mønsteret i ein stor og viktig RNA-bindande proteinfamilie. b) RNP-1 PROSITE-mønsteret PS00030, som er mønsteret som skildrar den konserverte delen i samanstillinga. Syntaksen til PROSITE-mønster er nær standarden for regulære utrykk. Tala og dei raude strekane er satt på for å syne samanhengen. Figuren er laga i Paint med utgangspunkt i Durbin *et al.* (1998).

I reglane nedanfor er notasjonen endra litt for at dette eksemplet skal vere lettare å lese. Her tyer: $[abc]W_1$ det same som $aW_1 \mid bW_1 \mid cW_1$. Med denne modifikasjonen på den tidlegare nytta grammatikknotasjonen kan vi definere grammatikken G slik at han skildrar PROSITE-mønsteret gitt over:

$G = (V, \Sigma, R, S)$ der
$V = \{a, c, d, e, f, g, h, i, k, l, m, n, p, q, r, s, t, v, w, y, W_1, W_2, W_3, W_4, W_5, W_6, W_7, S\}$
$\Sigma = \{a, c, d, e, f, g, h, i, k, l, m, n, p, q, r, s, t, v, w, y\}$
$R = \{$
$S \rightarrow rW_1 \mid kW_1$
$W_1 \rightarrow gW_2$
$W_2 \rightarrow [afilmnqstvwy]W_3$
$W_3 \rightarrow [agsci]W_4$
$W_4 \rightarrow [fy]W_5$
$W_5 \rightarrow [liva]W_6$
$W_6 \rightarrow [acdefghiklmnpqrstvwy]W_7$
$W_7 \rightarrow [fym]\}$
S er startsymbolet

No kan den konserverte sekvensen til den øvste strengen over utleiast frå grammatikken slik:

$$S \Rightarrow rW_1 \Rightarrow rgW_2 \Rightarrow rgqW_3 \Rightarrow rgqaW_4 \Rightarrow rgqafW_5 \Rightarrow rgqafvW_6 \Rightarrow rgqafviW_7 \Rightarrow rgqafvif$$

Kva kan ein regulær grammatikk ikkje gjere?

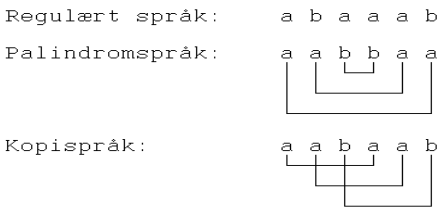
To klassiske eksempel av språk L som regulære grammatikkar ikkje kan skildre får vi når:

- L inneheld alle strengar på forma $aa, abba, babbab$, og så bortetter, som kan lesast både framlengs og baklengs (palindromspråk).
- L inneheld alle strengar på forma $aa, abab, bbabba$, og så bortetter, der andre halva er ei oppattaking av den fyrste (eit kopispråk).

Regulære grammatikkar kan skildre språk som inneheld palindrom. Poenget er at ingen regulær grammatikk kan generere berre palindrom, og kan difor ikkje sikkert skilje eit palindrom frå eit ikkje-palindrom. For å skildre desse meir kompliserte grammatiske samanhengane i ein streng må vi ty til meir komplekse grammatikkar.

Ulikt regulære språk har strengar frå palindrom- og kopispråk samanhengar mellom fjerntliggande posisjonar. På Figur 18 er det liner som understrekar samanhengen mellom desse posisjonane. Ein kan sjå at i palindromspråket er samanhengane nøsta,

det vil sei at linene ikkje kryssar kvarandre. I kopispråket derimot ser vi at linene kryssar kvarandre. Dette er eit viktig skilje for å avgjere kva type grammatikk som avgjer språket.



Figur 18. I eit ord i eit regulært språk er det ingen fjerntliggande samanhengar mellom teikna. I palindromspråka er alle samanhengar nøsta og det er difor mogleg å teikne samanhengane utan at strekane kryssar kvarandre. Figuren er laga i Paint med utgangspunkt i Durbin *et al.* (1998).

2.5.3 Kontekstfrie grammatikkar

Palindromspråka vert tekne hand om av neste nivå i Chomskys hierarki, dei kontekstfrie grammatikkane (engelsk *context-free grammar,* CFG). Grunnen til at det er vert å sjå nærare på kontekstfrie grammatikkar er at RNA sin sekundærstruktur er ein variant av eit palindromspråk. Sidan sekundærstrukturen til eit RNA-molekyl er meir konservert enn primærstrukturen, er det samanhengen mellom fjerntliggande, nøsta basepar-relasjonar som lyt skildrast i grammatikken som skal kjenne dei att.

Dei kontekstfrie grammatikkane tillet reglar som gjer dei i stand til å skildre nøsta, fjerntliggande, parvise samanhengar mellom terminalsymbol. Venstresida i ein regel lyt framleis vere ein enkel ikkje-terminal, men høgresida kan vere ein vilkårleg streng av terminalar og ikkje-terminalar. Ei høgreside kan altso generere eit par av terminalar, som har ein samanheng, i motsetning til dei regulære grammatikkane, som berre kan generere par av terminalar gjennom ulike og uavhengige ikkje-terminalar. Eit døme på ein CFG som kan generere palindromspråk er grammatikken med berre denne regelen:

$$S \rightarrow aSa \mid bSb \mid aa \mid bb$$

Ei utleiing av palindromet $aabaabaa$ frå denne grammatikken er:

$$S \Rightarrow aSa \Rightarrow aaSaa \Rightarrow aabSbaa \Rightarrow aabaabaa$$

Medan regulære grammatikkar genererar strengar frå venstre mot høgre, genererar CFG strengar utafrå og inn. Berre nøsta samanhengar kan skildrast med denne "utanfrå og inn"-genereringa. Dei kryssande samanhengane i kopispråka (Figur 18) strir mot dette kravet til nøsting, so kopispråk er ikkje kontekstfrie språk.

Med denne teorien på plass kan ein skildre grammatikken som produserer strukturen i dømet på side 16. Denne grammatikken skildrar berre strukturen utan å seie noko om kva basepar som skal utgjere stammen i hårnåla:

$G = (V, \Sigma, R, S)$ der
$V = \{a, u, c, g, W_1, W_2, W_3\}$
$\Sigma = \{a, u, c, g\}$
$R = \{$
$S \rightarrow cW_1g \mid gW_1c \mid aW_1u \mid uW_1a \mid uW_1g \mid gW_1u$
$W_1 \rightarrow cW_2g \mid gW_2c \mid aW_2u \mid uW_2a \mid uW_2g \mid gW_2u$
$W_2 \rightarrow cW_3g \mid gW_3c \mid aW_3u \mid uW_3a \mid uW_3g \mid gW_3u$
$W_3 \rightarrow gaaa\}$
S er starsymbolet.

Denne grammatikken kan skildre alle hårnåler med ein loop av $gaaa$ og ein stamme av tre basepar. Ei utleiing av strengen frå dømet nemnt over ser slik ut:

$$S \Rightarrow uW_1g \Rightarrow uaW_2ug \Rightarrow uagW_3cug \Rightarrow uaggaaacug$$

2.5.4 Kontekstsensitive og uavgrensa grammatikkar

Sjølv om kopispråka ved fyrste augnekast ikkje ser meir komplekse ut enn palindromspråka, så er dei ikkje kontekstfrie. For å skildre eit kopispråk lyt ein ha ein kontekstsensitiv grammatikk. Her vil verken kontekstsensitive eller uavgrensa grammatikkar verte skildra av di dei ikkje vert nytta i dette studiet. Det fins mykje litteratur på området, mellom anna kan ein lesa Durbin *et al.* (1998) for introduksjon til språk og biologisk nytteverdi, Lewis og Papadimitriou (1998) for ein grundig og matematisk innføring i verda av språk og Garey og Johnson (1979) for vidare og djupare teori kring kompleksitet og kva som er berekneleg.

2.5.5 Stokastiske grammatikkar

Dersom vi granskar dømet vårt som omhandla PROSITE-mønster ser vi kvifor enkle deterministiske grammatikkar ikkje eignar seg so godt til bioinformatikk. Ettersom fleire sekvensar vert dokumentert og klassen av protein aukar, vert det vanskelegare og vanskelegare å generere ein spesiell sekvens. Unnatak til den generelle regelen kan førekomme i alle posisjonar. Til dømes RNP-1-mønsteret i eit anna RNA-bindande protein, SRP55-proteinet SR55_DROME som er involvert i mRNA-spleising i bananfluger, har sekvensen

```
NGYGFVEF
```

Den fyrste N-en passar ikkje inn i PROSITE-mønsteret sidan det krev ein R eller ein K i denne posisjonen. Mønsteret lyt modifiserast slik at det tillet N i fyrste posisjon. Etter som unntak akkumulerast i mønsteret og det vert mindre spesifikt, kan mønsteret passe for ikkje-relaterte tilfeldige sekvensar. For nokre proteinfamiliar har det synt seg umogleg å skape eit diskriminerande PROSITE-mønster. Den beste løysinga er å tillate unntaka men tillegge dei mindre vekt enn det som høver betre med hovudregelen i mønsteret. Denne ideen leiar oss til stokastiske (sannsynsbaserte) regulære grammatikkar. *Hidden Markov Models* (HMM) er ein mykje nytta modell i bioinformatikk og er ein stokastisk regulær grammatikk.

Alle nivåa i Chomskys hierarki kan nyttast i stokastisk form som basis for sannsynsbasert modellering av sekvensar. Ein stokastisk grammatikkmodell θ genererar ulike strengar x med sannsyn $P(x \mid \theta)$, medan ikkje-stokastiske grammatikkar anten produserar ein streng x, eller ikkje.

I ein stokastisk regulær grammatikk eller ein stokastisk kontekstfri grammatikk er summen av sannsynene for alle moglege produksjonar frå ein vilkårleg gitt ikkje-terminal lik 1,0. Den resulterande stokastiske grammatikken definerar ei sannsynsfordeling over sekvensar x der $\sum_x P(x \mid \theta) = 1$. Til dømes kan vi sjå på fyrste regelen frå PROSITE-dømet, $S \to rW_1 \mid kW_1$. Ein stokastisk regulær grammatikk ville tilskrive ei sannsyn til kvar av alternativa:

$$S \underset{(0,5)}{\rightarrow} rW_1 \qquad S \underset{(0,5)}{\rightarrow} kW_1$$

Den stokastiske grammatikken kan no tillate avvik utan å redusere tyngda til det mest brukte mønsteret i vesentleg grad ved å gje unntaket ei låg sannsyn $P > 0$. Til dømes kan N-en i fyrste posisjon i RNP-1-mønsteret til SR55_DROME modellerast slik:

$$S \underset{(0,45)}{\rightarrow} rW_1 \qquad S \underset{(0,45)}{\rightarrow} kW_1 \qquad S \underset{(0,1)}{\rightarrow} nW_1$$

Dersom omskrivingsreglane tillet ei sannsyn for kvart av dei moglege symbola (her, alle aminosyrene) og grammatikken er slik at den kan generere strengar av vilkårleg lengde, så inneheld språket til ein stokastisk grammatikk alle moglege strengar, ikkje berre ei delmengde av dei. Ein stokastisk grammatikk kan difor nyttast til å spesifisere ei sannsynsfordeling over ei uendeleg mengde av strengar.

2.5.6 Stokastiske kontekstfrie grammatikkar for sekvensmodellering

No kan vi skrive ned stokastiske kontekstfrie grammatikkar (engelsk *stochastic context-free grammar*, SCFG) som modellar for sekvensfamiliar. Men det å skrive ned ein stokastisk grammatikk er berre det fyrste steget i arbeidet med å skape eit brukbart sannsynsbasert modelleringssystem for sekvensanalysar. Vi treng og algoritmar som kan handsame desse tre problema:

1. Rekne ut ei optimal samanstilling av ein sekvens til ein stokastisk grammatikk. (Samanstillingsproblemet.)
2. Rekne ut sannsyna for at ein sekvens vert generert av ein gitt stokastisk grammatikk. (Scoringsproblemet.)

3. Gitt eit sett av sekvensar/strukturar, estimer optimale sannsyner til kvar av omskrivingsreglane i ein stokastisk grammatikk. (Treningsproblemet.)

Det fins algoritmar av typen dynamisk programmering til kvart av desse problema. Desse algoritmane vert gjennomgått i det fyljande. Denne gjennomgangen startar med definisjon av ei normalform for SCFG-ar.

Ei normalform for stokastiske kontekstfrie grammatikkar

Ein SCFG har ingen avgrensing i kva for strengar den kan ha på høgresida i omskrivingsreglane. For å utrykke generelle SCFG-parsealgoritmar er det nyttig med ei normalform som avgrensar korleis høgresidene i reglane kan sjå ut, utan å avgrense uttrykkrafta. Chomskys normalform (engelsk *Chomsky's normal form*, CNF) er ei slik normalform. CNF krev at omskrivingsreglane lyt vere på forma $W_v \rightarrow W_y W_z$ eller $W_v \rightarrow a$. Einkvar SCFG kan transformerast til å vere på CNF ved å ekspandere ein regel som ikkje er på CNF til ei rekkje omskrivingsreglar på CNF via nye ikkje-terminalar. Til dømes kan omskrivingsregelen $S \rightarrow aSa$ frå palindromdømet, ekspanderast til $S \rightarrow W_1 W_2$, $W_1 \rightarrow a$ og $W_2 \rightarrow SW_1$ som er på CNF og har den same tydinga. Ei parsealgoritme som fungerar på ein SCFG på CNF er soleis generelt nytteleg på alle SCFG-ar.

Innvendig-utvendig-algoritmen

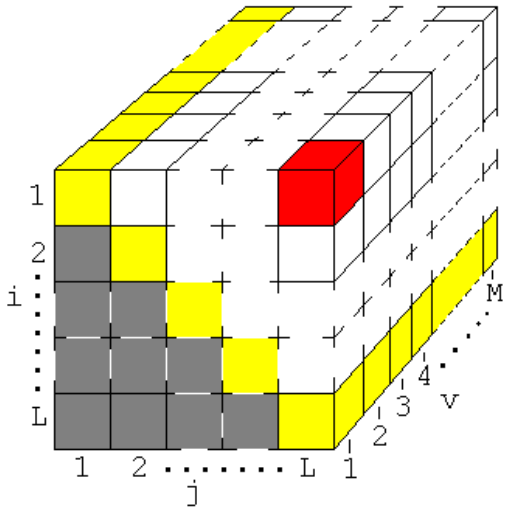
Innvendig-utvendig-algoritmen (engelsk *inside-outside algorithm*) for SCFG-ar på CNF er det naturlege motstykket til den meir kjende *forward-backward*-algoritmen som vert nytta i HMM-ar. Innvendigalgoritmen reknar ut sannsyna (scoren) til ein sekvens gitt ein SCFG. Ein "beste veg" -variant av innvendigalgoritmen, Cocke-Younger-Kasami (CYK) -algoritmen, finn den maksimalt sannsynlege samanstillinga av SCFG-en til sekvensen. Alle desse algoritmane er rekursive dynamiske programmering-algoritmar.

Her kjem ein del definisjonar som trengs til å forklare algoritmane: Ta ein SCFG på CNF med M ulike ikkje-terminalar $W = W_1, ..., W_M$. Startsymbolet er W_1. La v, y og z være indeksar av ikkje-terminalane W_v, W_y og W_z. Omskrivingsreglar er på CNF og kan altso skrivast på forma $W_v \rightarrow W_y W_z$ og $W_v \rightarrow a$, der a er eit symbol i

terminalalfabetet. La dei tilskrivne sannsynsparametrane vere høvesvis $t_v(y,z)$ og $e_v(a)$ (engelsk forkorting for *transition* og *emission*). $t_v(y,z)$ er altso sannsyna for at overgangsregelen $W_v \rightarrow W_y W_z$ skal nyttast når ikkje-terminalen W_v skal "velje regel" å transformerast etter (det talet vi skreiv under pila i dømet over). Sekvensen x har L symbol: $x_1,...,x_L$. La i, j og k vere indeksar for terminalar x_i, x_j og x_k i sekvensen x.

Innvendig

Innvendigalgoritmen reknar ut sannsyna $\alpha(i,j,v)$ av ei parsegrein med rot i ikkje-terminalen W_v for delsekvensen $x_i,...,x_j$ for alle i, j og v. Algoritmen krev ein tredimensjonal matrise på storleik $L \times L \times M$ (Figur 19). Utrekninga startar med sekvensar av lengde 1, det vil sei $i = j$, deretter tek den for seg alle delsekvensar av lengde 2, og slik jobbar den seg utover på lenger og lenger delsekvensar til ei sannsyn er utrekna for det komplette parsetreet med rot i startsymbolet. Ei formell framstilling av algoritmen er gjeve under figuren.



Figur 19. 3-dimensjonal dynamisk programmering-matrise. Kvar rute *(i,j,v)* svarar til ein substreng frå *i* til *j* utleia frå ikkje-terminalen W_v, altso variabelen $\alpha(i,j,v)$. Den grå delen svarar til substrengar av negativ lengde og er difor ikkje i bruk. Den gule diagonalen svarar til substrengar av lengde 1. Resultatet som svarar til heile strengen utleia frå startsymbolet ligg i den raude ruta *(1,L,1)*

Initiering:
For($i = 1$ til L, $v = 1$ til M){
(Dette er dei gule rutene på figuren)
$\alpha(i,i,v) = e_v(x_i)$
}

Repetisjon:
For($i = 1$ til $L-1$, $j = i+1$ til L, $v = 1$ til M){

$$\alpha(i,j,v) = \sum_{y=1}^{M}\sum_{z=1}^{M}\sum_{k=i}^{j-1}\alpha(i,k,y)\alpha(k+1,j,z)t_v(y,z)$$

}

Returverdi: $P(x \mid \theta) = \alpha(1,L,1)$ (Dette er den raude ruta på figuren)

Innvendigalgoritmen reknar altso ut sannsyna for (scoren til) ein sekvens gitt ein SCFG. Minnebehovet til algoritmen er $O(L^2M)$, som kjem av dei tre indeksane i α. Tidsbruken er $O(L^3M^3)$, som kjem av dei tre indeksane i, j og k som går gjennom strengen og dei tre indeksane v, y og z som går gjennom ikkje-terminalane i grammatikken.

Utvendig

Utvendigalgoritmen reknar ut ei sannsyn kalla $\beta(i,j,v)$ for eit komplett parsetre med rot i startsymbolet for heile sekvensen x, bortsett frå alle parsetre for delsekvensen $x_i,...,x_j$ med rot i ikkje-terminalen W_v for alle i, j og v. Som i innvendigalgoritmen er utrekninga gjort i ein $L \times L \times M$ matrise. Det å rekne ut sannsyna $\beta(i,j,v)$ krev at ein på førehand har rekna ut $\alpha(i,j,v)$ ved hjelp av innvendigalgoritmen.
Utvendigalgoritmen startar med den største ekskluderte delsekvensen $x_1,...,x_L$ og arbeider seg innover. Formelt kan ein skildre algoritmen slik:

Initiering:
$\beta(1, L, 1) = 1$
For($v = 2$ til M) $\beta(1, L, v) = 0$

Repetisjon:
For($i = 1$ til $L-1$, $j = i+1$ til L, $v = 1$ til M){

$$\beta(i,j,v) = \sum_{y,z} \sum_{k=1}^{i-1} \alpha(k, i-1, z)\beta(k, j, y)t_y(z, v)$$

$$+ \sum_{y,z} \sum_{k=j+1}^{L} \alpha(j+1, k, z)\beta(i, k, y)t_y(z, v)$$

}

Returverdi: $P(x \mid \theta) = \sum_{v=1}^{M} \beta(i, i, v)e_v(x_i)$ for vilkårleg i.

Innvendig- utvendigalgoritmen gir oss løysinga på vektingsproblemet ved å tilordne ei sannsyn P til ein streng x gitt ein SCFG θ. Dersom vi legg til grunn noko som kallast Viterbi- føresetnaden som seier at sannsyna for at ein stokastisk grammatikk genererer ein sekvens er tilnærma lik sannsyna for det mest sannsynlege parsetreet, kan vi nytte beste veg varianten av utvendig- innvendigalgoritmen, nemleg CYK-algoritmen. Det er denne algoritmen som er implementert i CM-pakken.

Ei algoritme for samanstilling: CYK

Det å finne eit optimalt parsetre (samanstilling av ein streng til grammatikken) vert løyst ved Cocke-Younger-Kasami-algoritmen (CYK), ein variant av innvendigalgoritmen med maxoprasjonar i staden for summane. CYK si motsvarigheit i HMM-ar er Viterbi-algoritmen. CYK reknar ut ein variabel $\gamma(i, j, v)$ som ideelt fører til logaritmen til $P(x, \hat{\pi} \mid \theta)$, der $\hat{\pi}$ er det mest sannsynlege parsetreet. Det er denne sannsyna som i fylje Viterbi- føresetnaden kan estimere $P(x \mid \theta)$. Vi tek òg vare på ein sporingsvariabel $\tau(i, j, v)$ som er ein triplett (y, z, k) som vi treng for å spore oss attende gjennom den tredimensjonale matrisa og etablere den optimale samanstillinga. Sporingsvariabelen spelar her same rolla som pilene som vart teikna i Needelman/Wunch-algoritmen i (Figur 11). CYK-algoritmen sin matrisefyllingsdel kan formelt skildrast slik:

Initiering:
For($i=1$ til L, $v=1$ til M){
$\gamma(i,i,v) = \log e_v(x_i)$
$\tau(i,i,v) = (0,0,0)$
}
Repetisjon:
For($i=1$ til $L-1$, $j=i+1$ til L, $v=1$ til M){
$\gamma(i,j,v) = \max_{y,z} \max_{k=i\ldots j-1} \{\gamma(i,k,y) + \gamma(k+1,j,z) + \log t_v(y,z)\}$
$\tau(i,j,v) = \arg\max_{(y,z,k),k=i\ldots j-1} \{\gamma(i,k,y) + \gamma(k+1,j,z) + \log t_v(y,z)\}$
}
Returverdi: $\log P(x,\hat{\pi} \mid \theta) = \gamma(1,L,1)$

Dette er etterfylgt av ei attendesporing for å etablere den beste samanstillinga, nett slik det vart gjort i Needelman/Wunsch-algoritmen (Figur 11). Dette er gjort ved å ”pushe” og ”poppe” triplettar (ruter i matrisa) på og av ein stakk:

Initiering:
Push $(1,L,1)$
So lenge det fins noko på stakken{
Pop (i,j,v)
$(y,z,k) = \tau(i,j,v)$
Dersom $\tau(i,j,v) = (0,0,0)$ ($\Rightarrow i=j$), set x_i som eit barn av v.
Elles{
Set y,z som barn av v
Push $(k+1,j,z)$
Push (i,k,y)
}
}

Det å sette y,z som barn av v tyder at ein her skal nytte omskrivingsregelen $W_v \rightarrow W_y W_z$ og det å sette x_i som eit barn av v at ein skal bruke regelen $W_v \rightarrow x_i$. Det er skildra på denne måten fordi ein då ender opp med eit tre der W_1 er rot og terminalsymbola i strengen x er bladnodane.

Parameterreestimering ved forventningsmaksimering

Innvendigvariabelen α og utvendigvariabelen β kan nyttast til å reestimere parametrane (sannsynene som er tilskrivne kvar omskrivingsregel) i ein SCFG ved

forventningsmaksimering. Det forventa talet gonger ein ikkje-terminal W_v er nytta i ei uleiing er $c(v)$ og kan reknast ut ved sannsynsrekning. Ein kan og finne det forventa talet på gonger $c(v \to yz)$ som regelen $W_v \to W_y W_z$ er nytta i ei utleiing. Då kan vi estimere overgangssannsyna til

$$\hat{t}_v(y,z) = \frac{c(v \to yz)}{c(v)}$$

Liknande likningar held for dei andre omskrivingsreglane $W_v \to a$:

$$\hat{e}_v(a) = \frac{\sum_{i|x_i=a} \beta(i,i,v) e_v(a)}{c(v)}$$

Utviding av desse likningane frå å gjelde ein enkel streng til tilfellet då vi har mange uavhengige observerte sekvensar er rimeleg rett fram. Forventa tal er ganske enkelt summert over alle strengane. Desse likningane gjer oss løysinga på treningsproblemet. Det vert vist til Durbin (1998) for ei meir detaljert gjennomgong av denne estimeringa.

Men også her kan CYK nyttast som ei tilnærming. I staden for å rekne ut talet kvar einskild omskrivingsregel vert nytta ved sannsynsrekning, reknar vi ut optimale CYK samanstillingar for treningssekvensane og tel deretter opp kor mange gonger dei ulike omskrivingsreglane er nytta i desse samanstillingane. Dette vert nærare vist nedanfor då samvariansmodellpakken vert gjennomgått.

Oppsummering av SCFG-algoritmar

Ved hjelp av innvendig- utvendig- og CYK-algoritmane, kan SCFG-ar nyttast som fullverdige sannsynsbaserte modelleringssystem. I Tabell 4 vert det vist kva algoritmar som er løysing på dei ulike problema vi nemnde i starten.

Optimal samanstilling (samanstillingsproblemet)	CYK
$P(x \mid \theta)$ (scoringsproblemet)	Innvendig
Parameterestimering (treningsproblemet)	Innvendig-utvendig

Tabell 4. Eit oversyn over kva algoritme som løyser dei ulike optimaliseringsproblema ein støyter på når ein nyttar ein SCFG til å skildre RNA-sekundærstruktur.

Både innvendig-, utvendig- og CYK-algoritmen har ein minnekompleksitet på $O(L^2M)$ og ein tidsbruk på $O(L^3M^3)$, der L er lengda på sekvensen som skal parsast og M er talet på ikkje-terminalar. Denne tidsbruken ser nok i utgangspunktet noko skremmande ut men kjem i store trekk av at SCFG-ar er generelle verkty. I neste avsnitt vert CM-pakken skildra. CM-pakken, som er ei implementering av denne teorien og er spesialbygd for RNA-analyse, har tidsbruk $O(L^3M)$ som heller ikkje er spesielt imponerande men mykje betre enn $O(L^3M^3)$.

2.6 Skildring av samvariansmodellen

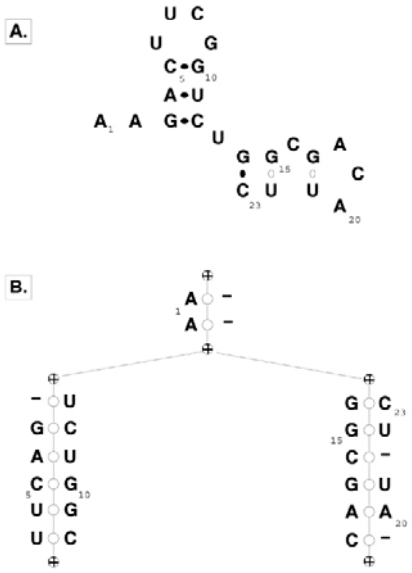
I dette avsnittet vert samvariansmodellen skildra. Det vert synt at denne modellen er ei implementering av ein stokastisk kontekstfri grammatikk (SCFG). Algoritmane for samanstilling, scoring og trening vert skildra. Mykje av innhaldet i dette kapittelet er, der ikkje andre referansar er gjevne, henta frå Eddy og Durbin (1994).

2.6.1 Eit oversyn over CM-pakken

Samvariansmodellen vart fyrste gong presentert av Eddy og Durbin (1994). I 1997 publiserte Todd M. Lowe saman med Sean R. Eddy programmet tRNAscan-SE (1997, 2001) som nyttar samvariansmodellen. Dette programmet er skildra i kapittel 3.1.

Ein RNA samvariansmodell er basert på eit ordna tre. Eit tre kan fange alle parvise samanhengar i eit RNA-molekyl so lenge dei er nøsta. Samanhengar som ikkje er

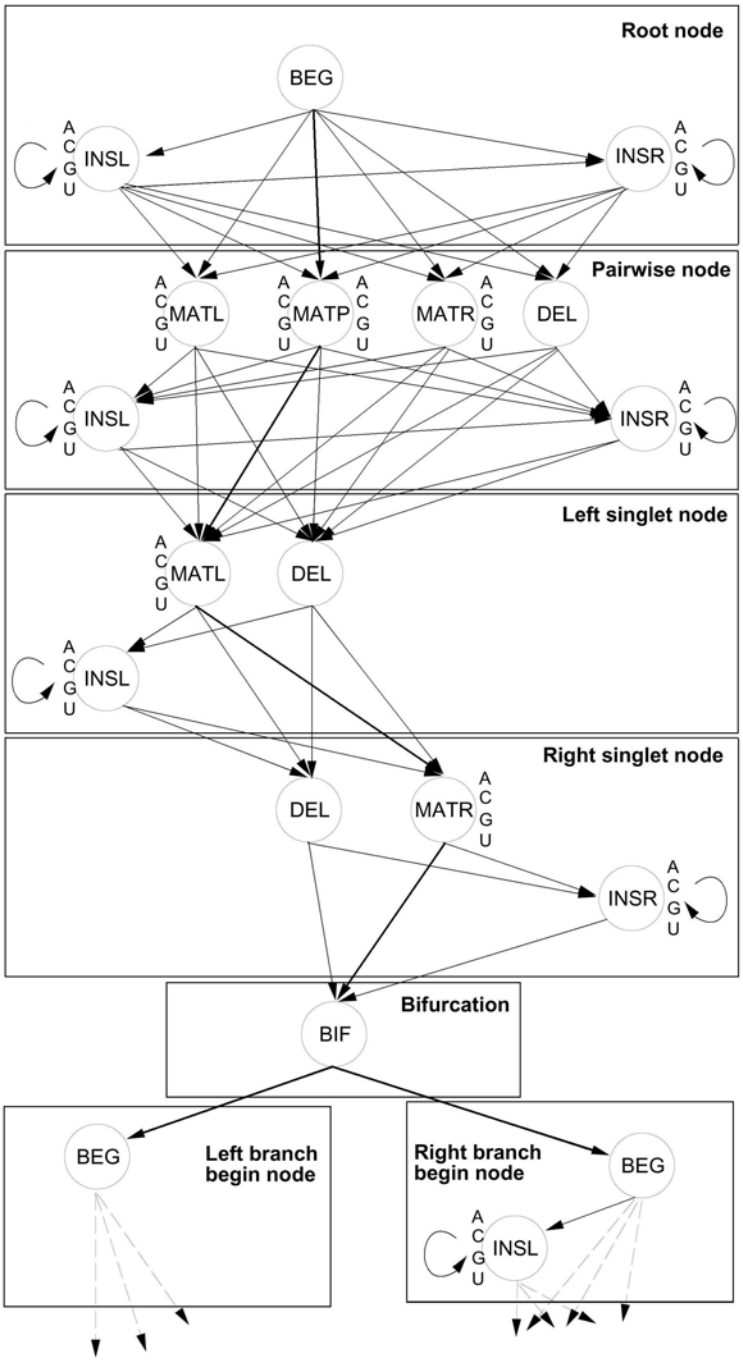
parvise, som basetrippel, eller ikkje er nøsta, som pseudoknutar, kan derimot ikkje skildrast av eit tre. Figur 20 B. syner ein trerepresentasjon av sekundærstrukturen til eit enkelt RNA-molekyl. Mange slike tre kan representere den same sekvensen, men i denne samanheng er det "beste" treet det som har parvise nodar der det er basepar i sekundærstrukturen og einsidige nodar der det er enkelttråda RNA. Både sekvensen (primærstrukturen) og sekundærstrukturen er skildra i dette treet. Sekvensen kan ein rekonstruere ved å traversere kvar grein i treet ned på venstresida og oppatt på høgresida.



Figur 20. A. Eksempel på RNA-struktur. B. Ein trestruktur som syner den same sekundærstrukturen. Treet inneheld nodar for Start, End og Biforcation (forgriningsnodar) i tillegg til enkle og parvise nodar som representerar strengen. Figuren er henta frå Eddy og Durbin (1994).

Dette treet er ikkje fleksibelt og skildrar berre eit enkelt RNA-molekyl. For å kunne skildre strukturen til ei familie av RNA-molekyl, må vi kunne skildre set-inn (*insert*), slett (*delete*) og mismatch i høve til ein konsensus struktur og sekvens. Difor tenkjer vi oss at kvar node er tilskrive kollonnar i ei multippel samanstilling i staden for enkle basar i eit einskild RNA-molekyl. Dei spesifikke basane byter vi ut med emisjonssannsyner (*emission probabilities*) tilskrive til dei 16 moglege basepara i parvise nodar og dei 4 moglege basane i einskildnodar. For å kunne ta opp i seg

mindre variasjonar i høve til konsensus som set-inn, slett og mismatch, legg vi til ei mengde tilstandar med særskilde eigenskapar i kvar node (Figur 21). Kvar node (runding) i Figur 20 B. svarar til ein av nodetypane (firkantane) i Figur 21. Matchtilstandar (MATP, MATL, MATR) tilsvarar konsensusvegen gjennom modellen. Denne vegen er markert i figuren med ein litt tjukkare strek. Dei andre emitterande tilstandane, set-inn-tilstandane (INSL, INSR) svarar til det å setja inn nukleotidar i høve til konsensusstrengen. Slettilstandar emitterar ingen ting og gjer det mogleg med slettingar i høve til konsensusmodellen. Tilstandar som emitterar einskildnukleontidar (MATL, MATR) er inkludert i dei parvise nodane for å tillate at den eine nukleotiden i eit konsensus basepar er sletta for å skape ei bukt. Tilstandane er knytte saman med overgangar (*transitions*) som kvar er tilskrive ei overgangssannsyn (*transition probability*). Denne overgangssannsyna er scoren for å gå over i ein av fleire moglege nye tilstandar. På Figur 21 ser vi til dømes at dersom ein har hamna i ein set-inn-tilstand fins det ei overganssannsyn for å gå attende til den same tilstanden. Dette tilsvarar på eit vis *gap-open* og *gap-extend* som er kjende omgrep frå sekvenssamanstilling og HMM. Spesielle ikkje-emitterande tilstandar skildrar trestrukturen i seg sjølv. Det er forgreiningstilstanden BIF, starttilstanden BEG som startar kvar grein i treet og ”bladtilstanden” END. Det er viktig å hugse at END ikkje skildrar enden på strengen men ei hårnålsvending i sekundærstrukturen. Overgangssannsynene vil favorisere hovudlina gjennom modellen, den som skildrar konsensusstrukturen. Kvar av tilstandane i ein CM kan sjåast på som ein ikkje-terminal i ein SCFG. Overgangsreglane i grammatikken svarar til overgangane mellom tilstandane i samvariansmodellen.



Figur 21. Dei sju ulike nodetypane i samvariansmodelltreet vert utvida med ei avgrensa mengde tilstandar. Det er i alt sju ulike tilstandar: Forgreining BIF (biforcation), start BEG (begin), set-inn-venstre INSL (insert-left), set-inn-høgre INSR (insert-right), match-parvis MATP (match-pairwise), match-venstre MATL (match-left), match-høgre MATR (match-right) og slett DEL (delete). Tilstandsovergongar er markert med piler. Dei emiterande tilstandane er markert med "ACGU" ved sida av (høgre, venstre eller båe) ringen som representerar tilasstanden. Figuren er henta frå Eddy og Durbin (1994).

Den sannsynsbaserte modellen som er resultatet av denne omlegginga er det som vert kalla ein samvariansmodell (CM). Den ferdige modellen består av ei mengde tilstandar M, emisjonssannsyner P og overgangssannsyner T. Ein CM kan sjåast på som ei sannsynsbasert maskin som genererer representantar for ei RNA-familie. Den skildrar ei multippel samanstilling både med tanke på sekvenskonsensus og med tanke på den parvise samvariansen som skuldast den underliggande sekundærstrukturen. Ein CM er ei generalisering ein HMM. Ein HMM er eit spesialtilfelle av ein CM utan forgreiningar og utan parvise tilstandar som skildrar samvarians. Dei algoritmane som her vert skildra for CM-ar har sine motstykkje i HMM-ar.

2.6.2 Modelltrening

Gitt ei multippel samanstilling av RNA-sekvensar frå same RNA-familie (Figur 12), er vi interessert i å finne den beste CM-en for denne samanstillinga. Det vil sei den CM-en som med størst sannsyn kan produsere strengane i den multiple samanstillinga. Dette kallast modelltrening.

Dette er eit globalt optimeringsproblem utan eit klart fasitsvar. I røynda har vi med to problem å gjere: For det fyrste lyt ein finne strukturen til treet, det vil sei finne ut kor mange nodar og tilstandar vi skal ha og korleis treet skal forgreine seg. For det andre må ein finne dei mest optimale emisjonssannsynene og overgangssannsynene gitt den aktuelle strukturen. Det siste problemet kan løysast med ein teknikk kalla *expectation maximization* (EM), som finn gode lokale optima for parameterverdiar. Men for å kunne nytte EM må vi fyrst løyse det fyrste problemet, altso å finne den optimale strukturen.

Det er skildra ulike heurestikkar for å predikere sekundærstruktur utifrå multiple samanstilling (Chiu og Kolodziejczak 1991, Han og Kim 1993), men CM-pakken nyttar i staden ei dynamisk programmering-algoritme for konsensus sekundærstrukturprediksjon.

Samvarians

Algoritmen nyttar verdiar for det som på engelsk heiter *mutual information content* for alle par av kolonnar i den multiple samanstillinga. Dette talet skildrar kor mykje desse

to kolonnane samvarierer. Her vert denne verdien kalla for samvariansverdi. I ei multippel samanstilling er samvariansverdien for kolonne i og kolonne j, der x_i varierar over dei fire ulike symbola, f_{x_i} er symbolfrekvensen for kvart symbol i kolonne i og $f_{x_i x_j}$ er den samla frekvensen for kvart par av symbol i kolonnane i og j er:

$$M_{i,j} = \sum_{x_i, x_j} f_{x_i x_j} \log_2 \frac{f_{x_i x_j}}{f_{x_i} f_{x_j}}$$

Denne algoritmen er i høve denne oppgåva implementert og modellen er trena på dataa i Figur 12. I dette avsnittet går indeksar frå 0 til N-1 i staden for frå 1 til N som var gjort i avsnittet om grammatikkar. Figur 22 syner ei utskrift frå matrisa M. Det er berre den øvste, høgre delen av matrisa som er nytta.

pos	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	2.0	0.4	0.9	1.0	1.0	0.8	0.5	0.9	0.7	0.8	1.0	1.0	0.8	1.0	0.4	0.5	0.0	0.6	1.0	0.3	0.3	1.0	0.6	0.5	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.0	1.8	0.6	0.5	0.3	0.8	0.5	0.4	0.5	0.8	0.3	0.5	0.4	0.8	1.5	0.7	0.0	0.8	0.5	0.8	0.8	0.5	0.8	0.7	0.0	0.4	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	1.9	0.3	0.6	1.2	0.8	0.8	1.0	0.9	0.6	0.3	0.5	1.6	0.7	0.5	0.0	0.4	0.9	0.8	0.8	0.9	0.4	0.5	0.0	0.9	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	1.6	0.7	0.7	0.5	0.8	0.7	0.6	0.6	1.6	0.4	0.4	0.5	0.3	0.0	0.4	0.7	0.3	0.3	0.7	0.4	0.3	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	1.9	0.6	0.4	0.8	0.8	0.8	1.6	0.7	0.9	0.7	0.3	0.5	0.0	0.8	0.8	1.0	1.0	0.8	0.8	0.5	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
5	0.0	0.0	0.0	0.0	0.0	1.9	0.4	0.8	1.1	1.6	0.8	0.7	0.7	1.1	0.7	0.8	0.0	0.4	0.6	0.8	0.8	0.6	0.4	0.8	0.0	0.8	0.0	0.0	0.0	0.0	0.0	0.0
6	0.0	0.0	0.0	0.0	0.0	0.0	1.7	0.6	0.6	0.3	0.4	0.5	0.5	0.7	0.5	0.8	0.0	0.7	0.6	0.6	0.6	0.6	0.7	0.8	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0
7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.8	1.1	1.0	0.7	0.8	0.5	0.8	0.5	0.6	0.0	0.7	1.0	0.7	0.7	1.0	0.7	0.6	0.0	0.9	0.0	0.0	0.0	0.0	0.0	0.0
8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.9	1.1	0.8	0.7	0.5	0.8	0.5	1.0	0.0	0.9	0.8	0.4	0.4	0.8	0.9	1.0	0.0	0.7	0.0	0.0	0.0	0.0	0.0	0.0
9	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.8	0.7	0.6	0.6	1.0	0.8	0.9	0.0	0.6	0.8	0.5	0.5	0.8	0.6	0.9	0.0	0.8	0.0	0.0	0.0	0.0	0.0	0.0
10	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.9	0.6	1.2	0.7	0.3	0.3	0.0	0.6	0.8	0.8	0.8	0.8	0.6	0.3	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
11	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.6	0.4	0.4	0.5	0.3	0.0	0.4	0.7	0.3	0.3	0.7	0.4	0.3	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.8	0.6	0.4	0.4	0.0	0.7	0.7	0.9	0.9	0.7	0.7	0.4	0.0	0.8	0.0	0.0	0.0	0.0	0.0	0.0
13	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.8	0.6	0.6	0.0	0.5	1.0	0.7	0.7	1.0	0.5	0.6	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
14	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.8	0.6	0.0	0.9	0.7	0.8	0.8	0.7	0.9	0.6	0.0	0.4	0.0	0.0	0.0	0.0	0.0	0.0
15	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	1.0	0.6	0.5	0.5	0.6	1.0	2.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0
16	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
17	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.9	0.6	0.6	0.6	0.6	1.9	1.0	0.0	0.6	0.0	0.0	0.0	0.0	0.0	0.0
18	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.7	0.6	0.6	1.7	0.6	0.6	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
19	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.9	1.9	0.6	0.6	0.5	0.0	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0
20	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.9	0.6	0.6	0.5	0.0	0.3	0.0	0.0	0.0	0.0	0.0	0.0	0.0
21	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.7	0.6	0.6	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
22	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.9	1.0	0.0	0.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0
23	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0
24	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
25	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0
26	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
27	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
28	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
29	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
30	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
31	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Figur 22. Matrisa M. I kvar posisjon er oppført samvariansverdien mellom kolonne *i* (vertikalt) og *j* (horisontalt) i den multiple samanstillinga frå Figur 12. Her er av omsyn til plassen berre gjeve to siffer. I programmet er verdien lagra som *float* (32 bit). Denne og dei to neste figurane er laga i Paint med utgangspunkt i utskrifter frå Java-programmet som er laga i høve denne oppgåva.

$M_{i,j}$ varierar mellom 0 og 2 bit. 0 bit tilsvarar ingen samvarians og 2 bit tilsvarar full samvarians utan bevaring av primærstruktur. Til dømes kan ein sjå at kolonne 0 i Figur

12 har ein høg bitscore i høve til kolonne 25. Dette talet finn ein altso i posisjon $i = 0$ (loddrett), $j = 25$ i Figur 22. Posisjon 16 er derimot ikkje samvariert med noka anna kolonne. Ved å studere Figur 12 kan ein sjå at kolonne 0 og 25 har basepar i alle sekvensane. Frå frekvenstabellen i Figur 13 ser vi òg at alle basane i desse to kolonnane er jamt fordelt med ein frekvens på 0,25. Det er nett desse kreteria som vert favorisert i likninga til M . Kolonne 16 derimot, har T i alle sekvensane. Sidan denne kolonnen ikkje varierar, kan den fyljeleg ikkje samvariere med andre kolonnar.

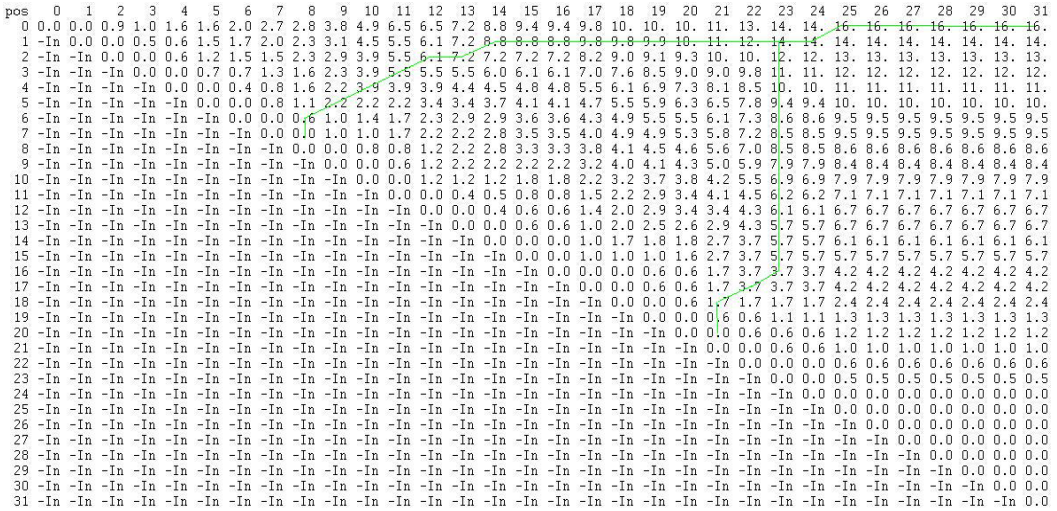
Måleeininga bit kjem frå informasjonsteori som ein kan lese meir om til dømes i Durbin *et al*. (1998). Det er informasjonsmengde ein måler i bit og ein kan tolke $M_{i,j}$ som "kor mykje meir veit vi om kva teikn som fins i kolonne j dersom vi veit kva teikn som fins i kolonne i". Sidan det er fire ulike teikn (nukleotidar), er den maksimale informasjonen vi kan oppnå 2 bit. Samvariansverdien representerar den forventa auke i score for å tilskrive kolonnane i og j til ein parvis node i staden for til to enkeltnodar.

Treet

Treet som representerar konsensus sekundærstruktur er det treet som inneheld mest samla samvariansverdi. Dette treet reknar ein ut ved hjelp av dynamisk programmering og samvariansverdiane som ligg i matrisa M . Dersom vi startar på diagonalen $i = j$ og arbeider oss mot hjørnet der $i = 0$ og $j = N - 1$ kan vi fylle inn ei ny matrise S der

$$S_{i,j} = \max[S_{i+1,j}, S_{i,j-1}, S_{i+1,j-1} + M_{i,j}, \max_{i<mid<j}[S_{i,mid} + S_{mid+1,j}]]$$

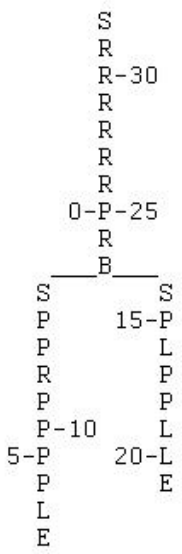
I Figur 23 er synt ei utskrift frå matrisa S . Her kan ein sjå korleis samvarians vert akkumulert oppover til høgre i matrisa. Øvst til høgre, i posisjon (0,31) ligg den totale samvariansen i denne samanstillinga.



Figur 23. Matrisa S. Denne matrisa inneheld den summerte og maksimerte samvariansen. Total samla samvarians ser ein øvst i høgre hjørnet. Den grøne linja skildrar attendesporinga. Også her er berre to siffer synt sjølv om verdien er lagra som 32 bits flyttal.

Dette er eigentleg den algoritmen Nussinov *et al*. (1978) skildra for falding av RNA-molekyl. Skilnaden er at grunnlaget her er samvariansverdiar frå ei multippel samanstilling, i staden for talet av basepar.

Kompleksiteten til denne algoritmen er $O(N^2)$, men det er ganske fordringslaust i høve til samanstillingsalgoritmen som er skildra lenger ned. $S_{0,N-1}$ inneheld den maksimale summen av samvarians $M_{i,j}$ som er mogleg å få med i eit tre, og ei attendesporing på vanleg dynamisk programmering-vis, gjennom scorematrisa vil etablere strukturen til dette optimale treet (Figur 24). I denne implementasjonen er det ikkje teke vare på kva rute/ruter som har gjeve opphavet til summen i kvar rute, slik som vart gjort med pilene i NeedelmanWunsch-algoritmen og lagring av koordinatar i CYK-algoritmen. Her vert vegen berekna etter kvart som ein sporar seg attende ved å samanlikne verdien i ei rute med dei verdiane som kan ha gjeve grunnlaget for den.



Figur 24. Tre av samvariansmodellnodar. Tala syner kva kolonne i den multiple samanstillinga som er tilskrive den aktuelle noden. Treet er generert av ein implementasjon av samvariansmodellen. Ein P står for ein parvis node, ein R for ein enkeltnode som emitterer til høgre og L for ein enkeltnode som emitterer til venstre. Strukturen er bygd opp av S- (start), E- (end) og B- (biforkation) nodar som ikkje emitterer.

Dersom ei kolonne inneheld mykje gap, vert det eit avvegingsspørsmål om ein skal tilskrive den til matchnodar eller til set-inn-tilstndar til nabonodar. Eddy og Durbin (1994) har sett denne grensa til 50%, men understrekar at dette er ein verdi som er valt meir eller mindre tilfeldig.

Per definisjon er den nye modellen samanstilt med alle sekvensane i den multiple samanstillinga, so emisjonssannsyner og overgangssannsyner kan no bereknast ved hjelp av EM. Basert på frekvensen dei ulike overgangane og emisjonane er nytta, bereknar vi verdiar for P og T:

$$P(x \mid y) = \frac{n(x \mid y) + R(x \mid y)}{n(y) + \sum_{x'} R(x' \mid y)}$$

$$T(y_{next} \mid y) = \frac{n(y_{next} \mid y) + R(y_{next} \mid y)}{n(y) + \sum_{y_{next}} R(y_{next} \mid y)}$$

For å forstå desse likningane kan vi i fyrste omgang sjå vekk i frå R-en. Då representerar det som står att $P(x \mid y) =$ talet på gonger ein tilstand y har emittert ein viss base/basepar x (x står her både for base og basepar alt etter kva tilstand y er), delt på talet på gonger den har emittert i det heile. For overganssannsynene vert resultatet $T(y_{next} \mid y) =$ talet på gonger ein overgong frå ein tilstand y til y_{next} er nytta delt på det totale talet gonger tilstanden y er nytta. Dersom vi set $R = 1$ tilsvarar det standard korreksjon av målte frekvensar for små datasett. R Kan og nyttast til å favorisere ting vi vil skal føretrekkast av modellen (*expert Bayesian prior*).

2.6.3 Samanstilling

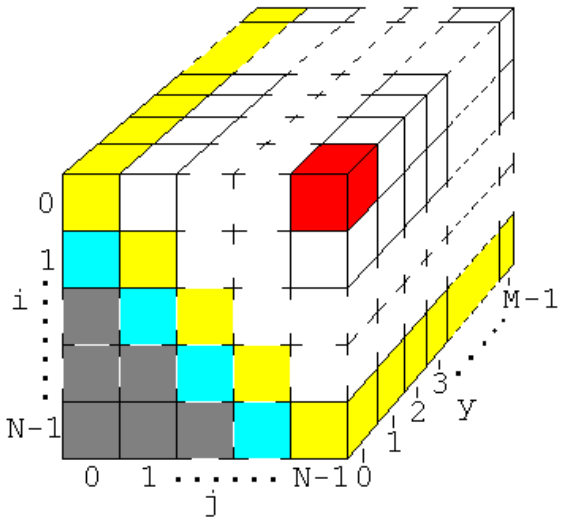
Den mest grunnleggjande operasjonen ein gjer med ein CM er å samanstille ein RNA sekvens til ein CM og å rekne ut ein sannsynsscore. Multiple sekvenssamanstillingar produserar ein ved å samanstille einskilde sekvensar til ein CM. Trening av modellen skjer ved at ein optimaliserar parametrane og strukturen til modellen slik at ei mengde treningssekvensar oppnår høgast mogleg score. Søking i databasar skjer ved at ein ser etter høgtscorande samanstillingar mellom CM-en og subsekvensar av vilkårleg lange sekvensar.

Den optimale samanstillinga av ein RNA-sekvens til ein CM og den tilhøyrande sannsynsscoren vert rekna ut ved hjelp av ei tredimensjonal dynamisk programmering-algoritme. Ideen er å starte med å samanstille dei minste subtrea med dei minste subsekvensane (dei tomme strengane og enkeltsymbol) og å bruke desse mindre samanstillingane til å rekursivt rekne ut den optimale samanstillinga for gradvis større delsekvensar til større subtre, heilt til ei global optimal samanstilling er berekna.

Denne algoritma tilsvarar CYK-algoritmen for SCFG.

Heilt konkret skjer dette ved at ei tredimensjonal matrise vert berekna. Denne matrisa inneheld scorar $S_{i,j,y}$ som er logaritmen til sannsyna for samanstilling av subsekvensen $i..j$ ($0 \le i \le j \le N-1$) til subtreet med rot i tilstand y ($0 \le y \le M-1$) der N er tal på basar i strengen og M er tal på tilstandar i modellen. For å visualisere dette kan ein tenkje på i og j som kollonnar og rader og y som nivå (Figur 25). Tilstandane i treet

er nummerert frå rota slik at barn av ein tilstand y_{next} alltid har høgare indeks enn foreldretilstanden y. Det vil sei prefiks traversering (Weiss 1999). $T(y_{next} \mid y)$ er overgangssannsyna frå ein tilstand y til ein mogleg etterfyljande tilstand y_{next}. $P(x_i, x_j \mid y)$ er sannsyna for at tilstand y emitterar symbolet x_i til venstre og x_j til høgre. $S_{i,j,y}$ finn vi ved å rekne ut scoren til dei opptil seks ulike tilstandane y_{next} som y har overgangar til, og beheld den høgaste. Kvar mogleg $S_{i,j,y}$ er ein sum av tre tal:



Figur 25. 3-dimensjonal matrise for samanstilling av RNA-sekvens til CM.

- logaritmen til emisjonssannsyna for x_i og/eller x_j (eller ingen av dei) avhengig av kva type tilstand y er.
- logaritmen til overgangssannsyna frå y til y_{next}
- Scoren $S_{i',j',y_{next}}$ som allereie er rekna ut i tidlegare steg i rekursjonen. Både i' og j' avheng av kva type tilstand y er. Sidan y kan emittere x_i , x_j, båe eller ingen av dei, kan i' vere i eller $i+1$ og j' vere j eller $j-1$.

Utrekninga startar med å opprette og initsialisere ei delvis kube der vi har N kolonnar med j som indeks, $j+2$ rader med i som indeks og M nivå med y som indeks. Ei

mengde scorar $S_{j+1,j,y}$ som ligg like under diagonalen tek seg av grensetilfella der tilstanden y er samanstilt med ein tom streng. På Figur 25 er desse rutene lyseblå. Scoren til END tilstandar i denne mengda set vi til 0,0. Alle andre scorar er initsialisert til $-\infty$. No startar sjølve berekninga og vi startar med dei tomme strengane og held fram med delstrengar av lengde ein og så bortetter. I den tredimensjonale matrisa vil det sei at vi startar med $i = j + 1$ (dei blå rutene) og arbeider oss utover mot hjørnet der $i = 0$ og $j = \mathrm{N} - 1$. For kvar delstreng går vi gjennom alle tilstandane frå $y = M - 1$ til $y = 0$. Alt etter kva type tilstand y er, vel vi ei av dei fyljande likningane:

$$S_{i,j,y}(y = MATP) = \max_{y_{next}}[S_{i+1,j-1,y_{next}} + \log T(y_{next} \mid y) + \log P(x_i, x_j \mid y)]$$
$$S_{i,j,y}(y = MATL, INSL) = \max_{y_{next}}[S_{i+1,j,y_{next}} + \log T(y_{next} \mid y) + \log P(x_i \mid y)]$$
$$S_{i,j,y}(y = MATR, INSR) = \max_{y_{next}}[S_{i,j-1,y_{next}} + \log T(y_{next} \mid y) + \log P(x_j \mid y)]$$
$$S_{i,j,y}(y = DEL) = \max_{y_{next}}[S_{i,j,y_{next}} + \log T(y_{next} \mid y)]$$
$$S_{i,j,y}(y = BIF) = \max_{i-1<=mid<=j}[S_{i,mid,y_{left}} + S_{mid+1,j,y_{right}}]$$

Til slutt er scoren til den globale samanstillinga i posisjon $S_{0,N-1,0}$ (den raude ruta). For å finne sjølve samanstillinga må vi starte i posisjon $S_{0,N-1,0}$ og fylje maksimumsvegen attende slik som er vanleg for dynamisk programmering.

Sidan vi her har ei tredimensjonal matrise på storleik $\mathrm{M} \times \mathrm{N} \times \mathrm{N}$ er minnebruken $O(N^2 M)$. Det å gå gjennom alle substrengar av ein streng har ein tidskompleksitet på $O(N^2)$. Sidan ein i kvar tilstand av typen BIF lyt gå gjennom alle indeksane mellom i og j, må ein legge på ein N til. Vi går og gjennom alle tilstandane for kvar av substrengane og kjem dermed ut med tidsbruken $O(N^3 M)$. Talet på tilstandar i CM-en M, er lineært i høve til lengda på den multiple samanstillinga som vert nytta som treningssett.

Her er det interessant å samanlikne tidsbruksanalysen med CYK-algoritmen til den generelle SCFGen som er $O(N^3 M^3)$. Grunnen til denne skilnaden er at vi har eit

konstant tal av tilstandar som er interessant å drøfte i staden for alle par av ikkje-terminalar som ein lyt gjere i det generelle tilfellet.

Eit problem med denne scoren er at den er svært lengdeavhengig. Skal ein nytte CM-en til å søkje i databasar, må denne avhengigheita vekk. Dette kan ein gjere ved å rekne ut sokalla "log odds" i staden for logaritmen til kvar emisjonssannsyn i modellen, før ein tek til på sjølve samanstillinga. Log odds kan ein finne ved å subtrahere logaritmen til sannsyna for at denne sekvensen vart generert som ein tilfeldig sekvens med same basefordeling frå logaritmen til sannsyna til samanstillinga. For å oppnå dette byter ein ut siste lekkja i fyrste likninga over med

$$\log[\mathrm{P}(x_i, x_j \mid y) / f_{x_i} f_{x_j}]$$

der f_{x_i} er frekvensen til basen som står i posisjon i i heile strengen. Tilsvarande for dei neste to likningane. Denne korreksjonen gjer søking i databasar mogleg og forenklar tolkinga av scorane. Scorar over null er ein meir sannsynleg match til modellen enn til ein tilfeldig sekvens, og dess meir positiv dess betre.

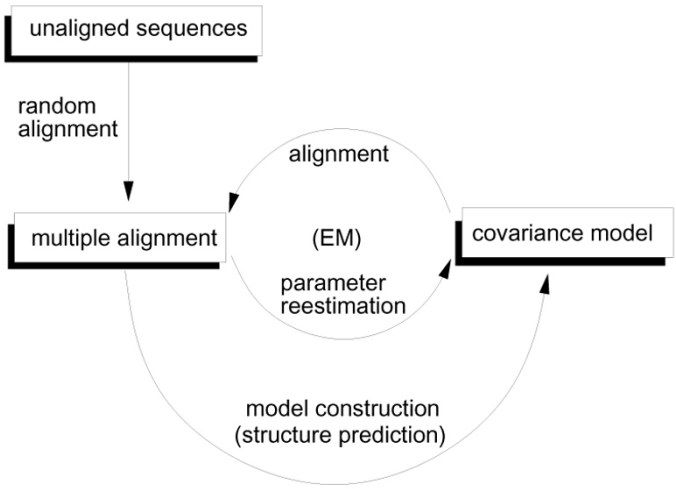
I tillegg til CYK-algoritmen frå teorien kring SCFG, liknar denne algoritmen både på den som er skildra over for predikering av sekundærstruktur og Needelman/Wunsch-algoritmen (Needleman og Wunsch 1970) for samanstilling av to sekvensar (utan omsyn til struktur) som vart skildra i 2.4.1.

Parameter reestimering

No er verdiar for P og T estimert og ei samanstillingsalgoritme er skildra. Ved å nytte denne algoritmen til å samanstille kvar einskild streng i den multiple samanstillinga til modellen, vil resultatet verte ei enno betre multippel samanstilling. Dette vil i sin tur danne grunnlag for å estimere betre parametrar.

Samanstilling og reestimeringsprosessen held fram til verdiane for P og T konvergerer. Denne prosessen er garantert å konvergere til eit lokalt optimum (Eddy og Durbin 1994).

Ein full modelltreningsprosedyre kan altso oppsummerast slik (Figur 26): Ein førebels modell vert bygd av ei (gjerne tilfeldig) samanstilling av treningssekvensar ved hjelp av modellbyggingsalgoritmen. Overgangs- og emisjonssannsynene vert iterativt reestimerte ved hjelp av ei EM algoritme. Når parametrane konvergerar, vert det bygd ein ny modell utifrå den gjeldande multiple samanstillinga. Dette vert oppatteke til verken modellen eller parametrane endrar seg nemneverdig. Denne treningsprosessen er analog til den intuitive manuelle samanliknande analysen av sekvensar: Ei opphavleg samanstilling av sekvensar vert gradvis raffinert ved at ein erkjenner fleire parvise samanhengar og konserverte posisjonar. Modelltrening fungerer sjølvsagt best om ein har ei god samanstilling som ein startar med, men Eddy og Durbin (1994) hevdar at det går òg bra å nytte tilfeldig samanstilte treningssekvensar som utgangspunkt.



Figur 26. Den iterative prosessen som leier fram til den mest optimale samvariansmodellen (CM) for ei multippel samanstilling av RNA-sekvensar. Det er best å ha ei sekundærstrukturrelevant multippel samanstilling i utgangspunktet, men denne prosessen kan sjølv finne ei bra samanstilling frå tilfeldig samanstilte sekvensar. Figuren er henta frå Eddy og Durbin (1994).

2.6.4 Søking

Det å søke med ein CM i ein database for å finne ncRNA av same familie, er med litt endringar, algoritmar som er presentert tidlegare i denne oppgåva. Det som er målet er å finne høgtscorande delsekvensar av ein lang sekvens, til dømes eit heilt genom. Til dette vert det nytta ei algoritme som er nesten lik samanstillingsalgoritmen. Scoringsmatrisa indekserast med distanse frå diagonalen d, j, og y i staden for i, j, y. Søk gjennom lange sekvensar oppnår ein ved å legge til ei ny rad j for neste sekvensposisjon, rekne ut scoren ved å starte med diagonalen $d = 0$ og samanlikne dei endelege scorane i $y = 0$ for kvar d. Dette er scorane til alle delsekvensar som endar i posisjon x_j. Startposisjonen til ein match er kjent ($i = j - d$) utan at ein treng å gjera noko attendesporing gjennom matrisa. Det er neppe interessant å analysere delsekvensar som er veldig mykje lenger enn treningssekvensane og ein kan difor tenke seg ei maksimumslengde w som avgrensar lengda på samanstillingsdelsekvensen ($d <= w$). Vi kan no indeksere søkematrisa med

$$j' = j \bmod w$$

i staden for j, og oppnår ei konstant storleik på scoreutrekningsmatrisa uavhengig av lengda på søkesekvensen.

Sidan søking nyttar same algoritmen som samanstilling, vert tidsbruksanalysen om lag den same. Lengda på strengen som skal samanstillast med modellen er w, men grunna den nye indekseringsmåten vinn vi ein gjennomgang. Den resulterande analysen vert då $O(Mw^2N)$. Tidsbruken til søk er lineært i høve til lengda av databasesekvensen.

3 Empiriske studium

Med det mål for auge å utforske CM-pakken, er eit program som nytta denne modellen prøvd ut. Det er og utarbeidd treningssett for to ulike ncRNA-familiar. Deler av samvariansmodellpakken er implementert og eit framlegg til endring av treningsalgoritmen vert lagd fram og prøvd ut. Ei skildring av dette arbeidet er lagt fram i dette kapittelet medan dei ulike resultata fyrst vert diskutert i neste kapittel.

3.1 Skildring og test av eit eksisterande program

Ein programpakke, Cove, er laga av Sean R. Eddy og Todd Lowe (1997). Denne implementerar algoritmane i CM-pakken. Denne programpakken er innlemma i eit program, tRNAscan-SE (Lowe og Eddy 1997, Lowe 2001), som søkjer etter tRNA-gen i genomiske sekvensar.

Programmet tRNAscan-SE består av tre delar kalla tRNAscan, EufidtRNA og Cove. Dette er tre ulike tRNA-deteksjonsprogram som kvar for seg har svakheitar, men i lag er dei eit effektivt og bra verktøy for å finne tRNA i ein genomisk sekvens. Når dei to fyrste delprogramma vert køyrt, lagar dei ei liste av tRNA-kandidatar. Det tredje delprogrammet går gjennom kandidatane og vel ut dei beste.

Det fyrste delprogrammet, tRNAscan, finn tRNA ved fyrst å sjå etter vel bevarte promotersekvensar. Når eit bestemt tal nukleotidar passar med konsensus promoteren (definert av ei valfri grense), freistar programmet å identifisere dei ulike sekundærstrukturane i eit typisk tRNA. Sekvensen får ein score etter kor godt den passar inn i strukturmodellen. Dersom scoren overstig ei empirisk framskaffa grense, vert plasseringa og tRNA-typen lagra.

EufindtRNA nyttar derimot berre sekvenslikskap i si leiting etter promoterar. Ei stegvis algoritme nyttar ei log odds scorematrise til å identifisere promoterelement og gje dei ein score. Grensa for kva score som skal reknast som eit treff, er satt noko mindre restriktivt i tRNAscan-SE enn i den opphavlege algoritmen. Dette reduserar ikkje tRNAscan-SE sin totale selektivitet av di det er det siste delprogrammet, Cove, sin jobb å eliminere falske positivar.

Sensitiviteten til EufindtRNA er samanlikneleg med den til tRNAscan, men det ser ut som dei er komplementære, dvs at båe finn tRNA som den andre ikkje finn. Dette nyttar tRNAscan-SE til sin fordel ved å ta unionen av dei to programma sine kandidatmengder, og lage ei liste av tRNA-kandidatar. Køyretida til båe desse programma er lineær, $O(N)$.

Cove er ein implementasjon av samvariansmodellen. I tRNAscan-SE vert Cove trena på eit treningssett av 1415 tRNA som er sekundærstruktursignifikant multippelt samanstilte. Søkealgoritmen vert nytta på alle kandidatane frå dei to føregåande delprogramma. Cove lagar òg ein rein HMM for samanstillinga utan informasjon om sekundærstrukturen. Med denne modellen kan kvar enkelt sekvens få ein score som berre kjem frå sekvenslikskap. For å finne sekundærstrukturscoren vert HMM-scoren trekt ifrå den totale CM-scoren.

Ved hjelp av desse delscorane freistar tRNAscan-SE å skilje ekte tRNA-gen frå pseudogen. Pseudogen er ein sekvens som har våre eit gen for mange generasjonar sidan men ikkje er i bruk lenger. Heurestikken som vert nytta gjer dette ved å skilje ut dei som får svært dårleg score enten for strukturlikskap eller for sekvenslikskap. Lowe og Eddy (1997) hevdar at dette er bra kriteria for å skilje ekte tRNA-gen frå pseudogen.

3.1.1 Føremål

I høve denne oppgåva vart tRNAscan-SE prøvd ut for å få eit inntrykk av verkingsgrada til CM-pakken. I test av eit deteksjonsprogram treng ein to kvalitetskriterium. Det fyrste kriteriet kallast selektivitet og kan målast med høvetalet ”falske positivar” per million basar. Ein falsk positiv er ein posisjon som programmet

peikar ut til å vere eit gen av den aktuelle typen, men som ikkje er det. Det andre kriteriet er sensitivitet og kan målast som deteksjonsrate. Deteksjonsrate er kor mange som er funne av dei gena som verkeleg fins i den strengen det er søkt gjennom. Eit estimat for dette høvetalet i ein test er prosenten av dei kjente gena som vert utpeika av programmet som gen.

3.1.2 Metode

I dette forsøket vart tRNAscan-SE køyrt på ei fil som innehelt heile genomet til *E. Coli*. Denne fila er lasta ned frå ftp://ftp.ncbi.nih.gov/genomes/Bacteria/Escherichia_coli_K12/ og er datert 20. mars 2003.

tRNAscan-SE vart køyrt med opsjonane:

- -B for søk etter bakterielle tRNA. Denne opsjonen gjer at det vert nytta ein covariansmodell som er trena på bakterielle tRNA. Den gjer og at nokre innstillingar i EufindtRNA vert sett slik at dette programmet finn fleire bakterielle tRNA.
- -H for å vise både primær og sekundærstrukturdelen av scoren frå samvariansmodellen. Ved å vise desse to tala er det mogeleg å gå inn og sjå kvifor eit gen er merka som pseudogen.
- -m for å få ut litt statistikk om køyringa av programmet
- -f for å få med data om sekundærstrukturen, predikert av Cove, til utfila.

3.1.3 Resultat

Køyringa av tRNAscan-SE gav 88 tRNA. Eit av desse er merka som pseudogen fordi scoren for sekundærstruktur er for låg. Dette resultatet er samanlikna med ei fil som inneheld alle annoterte tRNA i *E. coli*. 80 av dei tRNA-gena som vart plukka ut ved

hjelp av tRNAscan-SE er heilt identiske med dei kjende. 6 stykkje er like men er starta eller avslutta litt ulikt, opp til 4 nukleotidar. Det tRNA-genet som tRNAscan-SE foreslo å vere eit pseudogen, er ikkje med på lista over kjente tRNA-gen. Dette er oppsummert i Tabell 5.

Kjente tRNA	86
Funne med tRNAscan-SE	88
Like gen	80
Nesten like gen	6
Pseudogen	1
Tidlegare ukjente tRNA-gen	1

Tabell 5. Ei samanlikning mellom tidlegare kjente tRNA-gen i *E. coli* og resultatet frå søk med tRNAscan-SE. Dei 86 verifiserte tRNA-gena ert plukka ut med stor nøyaktigheit. I tillegg vert det plukka ut eit som programmet reknar med er eit pseudogen og eit gen som ikkje tidlegare er skildra.

tRNAscan-SE har altso funne alle tRNA-gena på lista over kjente tRNA-gen i *E. coli*, men i tillegg to gen der det eine er sagt å vere eit pseudogen. I Tabell 6 er det gjeve data frå resultatfila til tRNAscan-SE. Her kan ein sjå korleis desse to gena er scora i høve til gjennomsnittet.

	Primærstrukturscore	Sekundærstrukturscore	Total
Gjennomsnitt	57,5	24,9	82,4
Grenseverdi	10,0	5,0	20,0
tRNA-pseudo	7,32	21,1	28,5
TRNA-ukjent	34,9	7,43	42,4

Tabell 6. Resultat frå søk gjennom genomet til *E. coli* med tRNA-søkeprogrammet tRNAscan-SE. Det vart funne to gen av tRNAscan-SE som ikkje er med på lista over kjente tRNA-gen i E. coli. Det eine er sagt å vere eit pseudogen av di primærstrukturscoren 7,32 er under grenseverdien 10,0. Ein kan sjå at båe desse gena er langt under gjennomsnittet for totalscore. Resultatfilene kan sjåast på http://folk.uio.no/joseft/hovudfagsoppgaava/vedlegg/trna/.

Som ein kan sjå i Tabell 6 har båe desse gena ganske låge poengsummar i høve til gjennomsnittet (det er dei to lågaste poengsummane). Grenseverdien for totalscoren er

satt til 20,0. For å sei om eit gen er eit pseudogen er grensa for primærstrukturscoren satt til 10,0 og for sekundærstrukturscoren til 5,0. Sidan tRNA-psedo har primærstrukturscore under grenseverdien, er den tolka som eit pseudogen.

3.2 Utarbeiding av treningssett for samvariansmodellar

Det er mange aspekt som må takast omsyn til når ein skal lage eit treningssett. CM-pakken vart i utgangspunktet testa på det treningssettet av 1415 tRNA som er brukt i tRNAscan-SE (Eddy og Durbin 1994). Det ideelle for å lage ein god CM er å finne store familiar av ncRNA som har lik sekundærstruktur og ulik primærstruktur. Det store tRNA-settet fyller desse krava. Men det er få ncRNA-familiar som kan vise til like store tal av dokumenterte gen.

3.2.1 Føremål

Det overordna målet med RNA-prosjektet er å lage eit generelt søkeprogram for ncRNA-gen. For at CM-pakken skal kunne nyttast i eit slikt program må ein ha eit treningssett for kvar av alle ncRNA-familiar. For å kunne bygge ein CM for ei ncRNA-familie drengs det ei multippel samanstilling av representantar frå denne familien. Difor er det eit overordna, langsiktig mål å lage gode treningssett for alle kjente familiar av ncRNA. Målet for dette forsøket er å finne ein metode for å lage slike treningssett.

3.2.2 Metode

Det vart i august 2003 laga to treningssett. Hershberg (2003) har sett på dei 55 til då kjente ncRNA-gena i *E. coli*. I denne artikkelen er det lista opp kva andre beslekta organismar som har kvar av desse ncRNA-gena. Det er nokre gen som ser ut til å vere konservert i fleire organismar enn andre. Med dette som avgjerdsgrunnlag vart det plukka ut to gen: *spf* (Figur 7) og *ssrA* (Figur 15). Desse to genfamiliane har ulike eigenskapar som gjer dei interessante som prøvetreningssett. Genet *spf* er kort og tek difor lite tid å falde og parse under uttesting av algoritmane. Det andre, *ssrA* er langt men er konservert i mange fleire organismar (Tabell 7).

nrRNA	Lengde	(Hershberg *et al.* 2003)	Funne med ParAlign		
			Totalt	Unike gen	E-verdi
Spf	109 bp	12 andre organismar	16	7	<0,001
ssrA	466 bp	21 andre organismar	80	54	<0,015

Tabell 7. Talet på homologar i andre organismar til gena *spf* og *ssrA* frå *E. coli*. Blant Hershberg *et al*. (2003) sine homologar er det fleire gen som er heilt like. Til eit treningssett for ein samvarians-modell treng ein gen som er ulike i primærstruktur men med konservert sekundærstruktur. Med fare for å få med sekvensar som ikkje er biologisk relevante, er kravet til E-verdien senka. I tillegg til E-verdi var lengda på det konserverte området med i vurderinga av kva treff som skulle med i treningssettet.

Hershberg *et al*. (2003) har ikkje presentert noka liste over sekvensane eller posisjonane til dei konserverte områda. Det vart difor gjort eit homologisøk for å finne dei tilsvarande gena i andre organismar. Det vart gjort med programmet ParAlign (Rognes 2001). I ParAlign er parametrane satt slik at mest mogleg vert med. Det vil sei at ei alternativ scorematrise og gapstraff er nytta (viser til vedlegg på Internett for utdjuping: http://folk.uio.no/joseft/hovudfagsoppgaava/vedlegg/parAlign/). På same måten som Hershberg (2003), vart det her nytta sekvensen til gena *spf* og *ssrA* i *E. coli* som basis for søket. Det vart leita etter homologar i ein database som var laga den 7. juli 2003 av data frå NCBI (ftp://ftp.ncbi.nih.gov/genomes/Bacteria). Denne inneheld genomet til dei 124 bakteriane som var tilgjengelege på det tidspunktet. Resultatet frå søket er ei lang liste av samanstillingar (Figur 27), med den beste samanstillinga øvst. Denne resultatlista lyt arbeidast med på ulike måtar før vi har eit treningssett.

```
>gnl|BL_ORD_ID|38434 gi|16120353|ref|NC_003143.1|_3 [20001..31000] Yersinia pes
 tis strain CO92, complete genome
Length: 11000 nt
Matches on same strands.

Score: 467, Expect: 1e-29
Identical: 103/109 (94%)Indels: 3/109 (2%), Gaps: 2

Q:       1 GTAGGGTACAGAGGTAAGATGTTCTATCTTTCAGACCTTTTACTTCACGTAATCGGATTT 60
           ||||||||||||||||||||||||||||||||||||||||||||||||||||||||||||
D:    5878 GTAGGGTACAGAGGTAAGATGTTCTATCTTTCAGACCTTTTACTTCACGTAATCGGATTT 5937

Q:      61 GGCTGAATATTTTAGCCGCCCCAGTCAGTAAT-GACTGGGGCGTTTTTT 108
           |||||  ||| ||||||||||||||| | || ||||||||||||||||
D:    5938 GGCTG--TATATTAGCCGCCCCAGTCATTTATTGACTGGGGCGTTTTTT 5984
```

Figur 27. Utskrift frå søkeprogrammet ParAlign. Figuren syner eit av treffa i utfila. Øvste bolken gir opplysingar om kva organisme treffet er frå og identifikasjon av fila. "*Matches on same strands*" seier noko om kva retning treffet har i høve til databasesekvensen. *Score* og *Expect* seier noko om kor godt dette treffet er. Scoren kjem frå Smith/Waterman-samanstillinga og varierar etter kva parametrar ein gir programmet. *Expect*- eller E-verdien derimot, er eit samanliknbart tal mellom metodar og ulike program. Denne verdien seier noko om forventa tal av treff med like god Score eller betre ved søk med ein tilfeldig sekvens av same lengde i denne databasen. Det vil i dette eksempelet sei at forventa tal av treff med Score 467 eller betre for ein tilfeldig streng av lengde 109 i denne databasen er $1{,}0 \cdot 10^{-29}$. Nedst ser vi sjølve samanstilinga. Strengen merka *Q:* er søkestrengen (den som ein leitar etter homologar til) og den som er merka *D:* er frå databasen. Tala angir relativ posisjon i søkestrengen og databasestrengen. Heile resultatfila frå båe søka ligg tilgjengeleg på http://folk.uio.no/joseft/hovudfagsoppgaava/vedlegg/parAlign/.

Duplikat vart eliminert. Ein kan argumentere med at dersom det fins duplikat, skal dei vere med i treningssettet for at modellen skal trenast på det som faktisk finst av representantar for dette genet. Men ein CM er ein modell av samvariansen mellom ulike posisjonar i genet og vinn difor ingen ting på å ha fleire like sekvensar i treningssettet (like sekvensar syner ingen varians, og dermed ingen detekterleg samvarians).

I resultatfila frå ParAlign er ikkje nødvendigvis heile sekvensen med. Det er berre den delen av genet som er konservert som er teke med. I ein god del av organismane var størsteparten av genet konservert men ikkje heile. Dermed var det nødvendig å leite fram sekvensen i databasefila for å finne resten. Dette vart ein kombinasjon av manuelt arbeid og små spesialbygde sekvenshandteringsprogram. Det vart til dømes implementert eit lite program som returnerar den reverskomplementære strengen til ein inputstreng, noko som er til stor hjelp når genet ligg på den motsette tråden av den som ligg i databasen.

Ein lyt og avvege kor mange treff ein skal ta med. Då lyt ein vurdere kor godt konservert den konserverte delen er og kor stor den er i høve til resten av genet. Målet er sjølvsagt å ta med dei sekvensane som faktisk er eit tilsvarande gen i den aktuelle organismen og dermed har ein ekvivalent sekundærstruktur, og å utelate dei som er tilfeldige treff. Sidan dette er noko ein ikkje veit *a priori*, vert det kvalifisert gjeting om kor grensa bør gå. E-verdiar som er over 0,5 vart i denne samanhengen ikkje rekna som relevante. Eit anna aspekt som må takast omsyn til i vurderinga er at dersom endane ikkje er med i samanstillinga, er det vanskeleg å slå fast kvar enden er.

For å nytte dei to datasetta til å trene kvar sin CM er det nødvendig å gjere ei multippel samanstilling. Det ideelle er å ha ei sekundærstrukturrelevant multippel samanstilling. Dette kan anten lagast for hand av folk med stor biologisk kunnskap eller ein kan nytte algoritmiske tilnærmingar. Den delen av CM-pakken som inneheld algoritmar for multippel samanstilling, er ikkje implementert i denne oppgåva. Difor vart her i staden nytta ei anna mykje brukt algoritme kalla *CLUSTAL W* (Thompson *et al.* 1994). Denne metoden tek berre omsyn til sekvensen og ikkje strukturen når den samanstiller.

3.2.3 Resultat

Det er her prøvd ut ein metode for generering av treningssett for CM-ar som tek utgangspunkt i eit gen i ein organisme og deretter søkjer etter homologar i andre organismar med eit konvensjonelt og raskt søkjeprogram. Det vart funne 52 unike homologar til genet *ssrA* (tmRNA) og 6 unike homologar til genet *spf* (*spot42*).

Resultatet er to treningssett der det eine er eit sett av 53 som består av sekvensen til genet *ssrA* frå *E. coli* og 52 homologar funne med ParAlign. Kor vidt dette er eit biologisk relevant treningssett vert diskutert vidare i det avsnittet som skildrar utprøvinga av den implementerte delen av CM-pakken på dette treningssettet. Vidare er det laga eit treningssett av 7 sekvensar der den eine er genet *spf* frå *E. coli* og dei andre 6 er homologar funne i andre organismar ved hjelp av ParAlign. Desse sekvensane varierer svært lite seg imellom. Dette gjer at ein kan vere rimeleg trygg på den biologiske signifikansen, men det gjer også at det er svært vanskeleg å trene ein CM. Dette kjem fram i neste avsnitt der den implementerte delen av CM-pakken vert

testa på desse datasetta. Det fyrstnemnde treningssettet er derimot stort nok og kan, dersom det er biologisk signifikant, syne samvarians nok til å predikere sekundærstrukturen.

3.3 Implementering av deler av CM-algoritmane

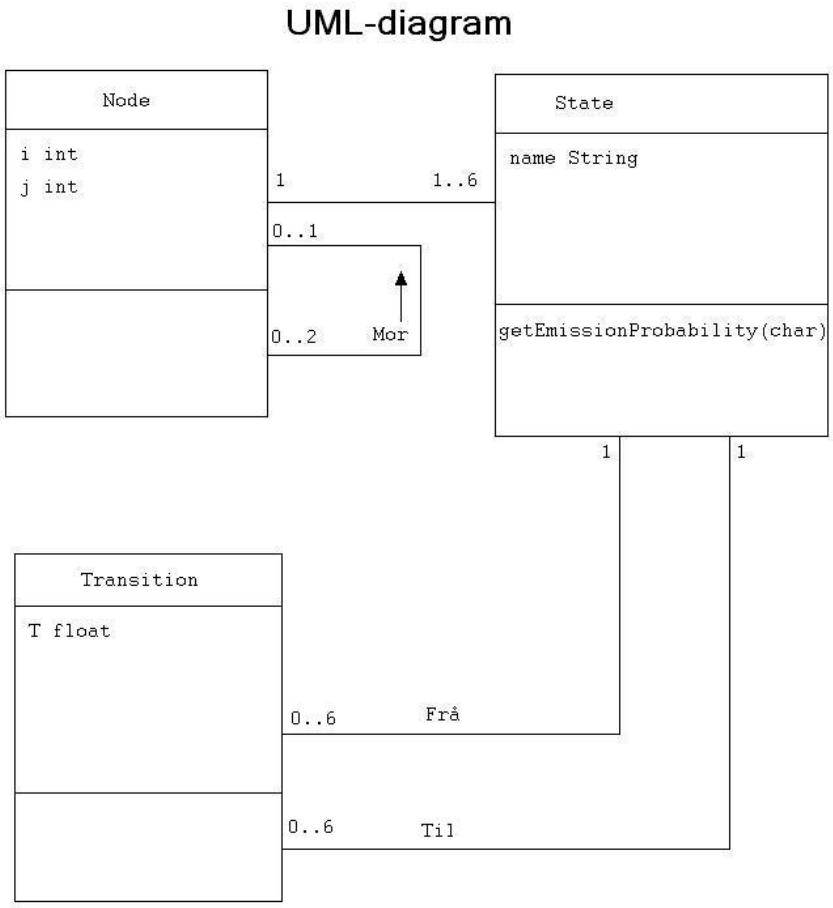
I høve denne oppgåva er delar av CM-pakken implementert. Programmet er skrive i Java. I dette avsnittet vert dette arbeidet skildra saman med ein test av programmet på datasettet skildra i førre avsnitt.

3.3.1 Føremål

Som ledd i fordjupinga i CM-pakken er deler av pakken implementert. Det er fleire mål med denne jobben. For det fyrste fører ei implementering i seg sjølv til ei djupare forståing av modellen og algoritmane den nyttar seg av. Dernest er det mogleg å prøve ut modellen når ein har eit program. Ei viktig rolle dette programmet har hatt i denne oppgåva er å lage utskrifter av strukturen i modellen. Figur 13, Figur 22, Figur 23 og Figur 24 er utskrifter frå dette programmet.

3.3.2 Metode

Datastrukturen til ein CM består av nodar, tilstandar og overgangar (Figur 28). Nodane utgjer sjølve treet som skildrar konsensus sekundærstrukturen (sjå Figur 21 for tilhøvet mellom nodar og tilstandar). I tillegg til sjølve datastrukturen som skildrar samanhengane i treningssettet, er det implementert fleire klassar som kvar står for sin del av oppbygginga av modellen.



Figur 28. UML-diagram for CM. Dette er ein noko forenkla modell som syner hovudtrekka i datastrukturen.

Eit mykje brukt filformat for sekvensar er FASTA-format (Figur 29). Dette er eit svært enkelt men veldig nyttig filformat. Filene består av par av skildringar og sekvensar. Det einaste formelle kravet til skildringa er at den skal starte med teiknet '>' og at den berre er på ei linje. Det er vanleg at det fyrste ordet i skildringa er identifiserande for denne sekvensen på denne fila. Sekvensen kjem på dei påfyljande linjene og kan vere sekvensar av DNA, RNA eller protein. Sekvensen er som regel stykka opp i 60 eller 70 teikn på kvar linje. Alle sekvensfilene som vert nytta av programmet er på FASTA-format.

```
>Salmonella_enterica_subsp._enterica_serovar_Typhi |NC_003198.1|:c3736271-3736162
GTAGGGTACAGAGGTAAGATGTTCTATCTTTCAGACCTTTTACTTCACGTAATCGGATTT
GGCTGAATATTTTAGCCGCCCCAGTCAGTTTATGACTGGGGCGTTTTTTA
>Yersinia_pestis_strain_CO92 |NC_003143.1|:25878-25984
GTAGGGTACAGAGGTAAGATGTTCTATCTTTCAGACCTTTTACTTCACGTAATCGGATTT
GGCTGTATATTAGCCGCCCCAGTCATTTATTGACTGGGGCGTTTTTT
```

Figur 29. Døme på fil i FASTA-format. Dette er to av førekomstane i treningssettet for genet spf. Skildringslinja startar med teiknet '>' og det fyrste påfyljande ordet er namnet på organismen. På dei påfyljande linjene kjem sekvensen. Det er ingen avgrensing i kor lange eller mange sekvensane kan vere, men det er tilrådt at kvar linje i fila ikkje skal vere lenger enn 80 teikn.

For å auke gjenbruksverdien til koden er den skilt i to pakkar. Den eine inneheld dei klassane som utgjer sjølve CM-en, medan den andre er ei verktøykasse som inneheld klassar som er av meir generell interesse. Denne pakken heiter bioTools. Eit døme er klassen FASTAIO som kan lese og skrive filer på FASTA-format. I tillegg er programmet dokumentert ved hjelp av programmet javadoc. Javadoc les programkode, tolkar kommentarar og genererer HTML-sider. Desse sidene har standard java API form.

Brukargrensesnittet er linjebasert på typisk UNIX-vis. Ein startar programmet med ei multippel samanstilling på FASTA-format som parameter. Deretter kan ein få utskrifter av den predikerte sekundærstrukturen på ulike måtar. Funksjonar for søking og reestimering av parametrane er ikkje implementert.

Programmet vart prøvd ut både på det konstruerte datasettet som er synt i Figur 12 og dei to ”ekte” treningssetta som var skildra i førre avsnittet.

3.3.3 Resultat

Ved køyring av programmet på testdata som datasettet i Figur 12, vert sekundærstrukturen predikert riktig. Det viser seg at dei kolonnane som er sett saman i parvise nodar i Figur 24, faktisk er dei som inneheld basepar i kvar sekvens. Dette konfirmerar at samvarians kan nyttast som grunnlag for prediksjon av sekundærstruktur.

Når programmet vert køyrt på treningssettet av sekvensar med utgangspunkt i genet *spf*, vert berre eit av 30 basepar i den riktige sekundærstrukturen predikert. I det store

datasettet av 53 sekvensar var deteksjonsandelen mykje høgare. Her vart 48 av 73 basepar i den riktige sekundærstrukturen predikert. I neste avsnitt vert det lagt fram eit forslag til endring av grunnlaget for sekundærstrukturprediksjon. Resultatet frå denne testen vert der diskutert vidare og definisjonen av ”riktige basepar” og ”riktig prediksjon” vert drøfta.

3.4 Eit framlegg til forbetring av algoritmen for sekundærstrukturprediksjon

Nussinov *et al*. (1978) utvikla ei dynamisk programmering-algoritme som folda RNA-molekyl ved å maksimere talet på basepar i strukturen. Det var denne algoritmen som låg til grunn då Zuker og Stiegler (1981) utvikla si meir sofistikerte foldingsalgoritme. I botnen er det denne algoritmen som vert nytta av CM-pakken også. Det er grunnlaget for foldinga som er ulikt i dei tre metodane. Det er Nussinovs teljing av basepar som er fundamentet for utvidinga som er gjort her.

3.4.1 Føremål

Resultatet frå testen i førre avsnitt syner at det er turvande med eit anna grunnlag for berekning av sekundærstrukturen til små datasett.

I små datasett bestående av gen frå artar i nær slekt, vil fleire av kolonnane vere bevarte med same basen gjennom heile settet sjølv om den er med i eit basepar. Til dømes inneheld treningssettet for *spf* berre sju sekvensar. I den multiple samanstillinga av dette settet er dei fleste kolonnane (75 av 106) bevarte gjennom alle sekvensane. Når det ikkje er variasjon i ei kolonne kan det heller ikkje førekomme samvariasjon med andre kolonnar. Dermed klarer ikkje samvariansmodellen å oppfatte kva kolonnar som skal tilordnast parvise nodar.

Sidan dei aller fleste familiar av ncRNA-gen er små og dei skildra gena ofte er rimeleg like, er det eit poeng å kunne predikere sekundærstrukturen til desse også.

3.4.2 Metode

Det fins informasjon i strengane som samvariansmodellen ikkje tek omsyn til. Det er kjent kva basar som dannar par. Det er i denne testen laga ei svært enkel likning som tek med denne informasjonen. CM-algoritmen for prediksjon av sekundærstruktur nyttar samvarians mellom kolonnar i ei multippel samanstilling som grunnlag (2.6.2). Den alternative likninga for fylling av den fyrste matrisa ser slik ut:

$$M'_{i,j} = M_{i,j} + A_{i,j}$$

$M_{i,j}$ er samvariansverdien slik den er skildra i 2.6.2. $A_{i,j}$ skal gje høg score mellom kolonnane i og j dersom dei har mange basepar. I dette forsøket er $A_{i,j}$ sat til talet på basepar i høve til talet på sekvensar:

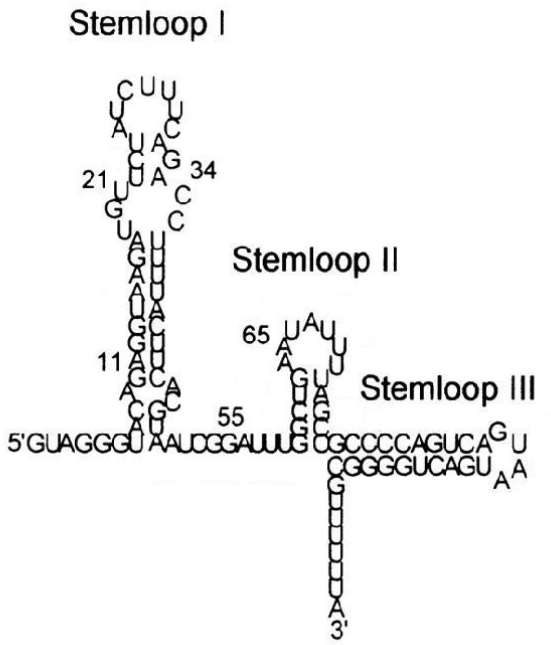
$$A_{i,j} = \frac{1}{N(x_i, x_j)} \sum_{x_1, x_j} B_{x_i, x_j}$$

B_{x_i, x_j} er 1 dersom x_i og x_j dannar eit basepar og 0 elles. $N(x_i, x_j)$ er talet på sekvensar som har basar i både kolonne i og j. Som basepar er rekna AU, UA, CG, GC, GU og UG.

Det er mogleg å køyre programmet med ein opsjon, *–adjust*, som gjer at sekundærstrukturen vert berekna på grunnlag av M' i staden for M. Programmet er køyrt ein gong med denne opsjonen, og ein gong utan, for kvar av dei to treningssetta.

For å kunne teste kva for ein metode som er best må ein ha ein riktig sekundærstruktur, ein fasit. For *ssrA* (Figur 15) sin del er fasiten henta i Rfam-databasen (Griffiths-Jones

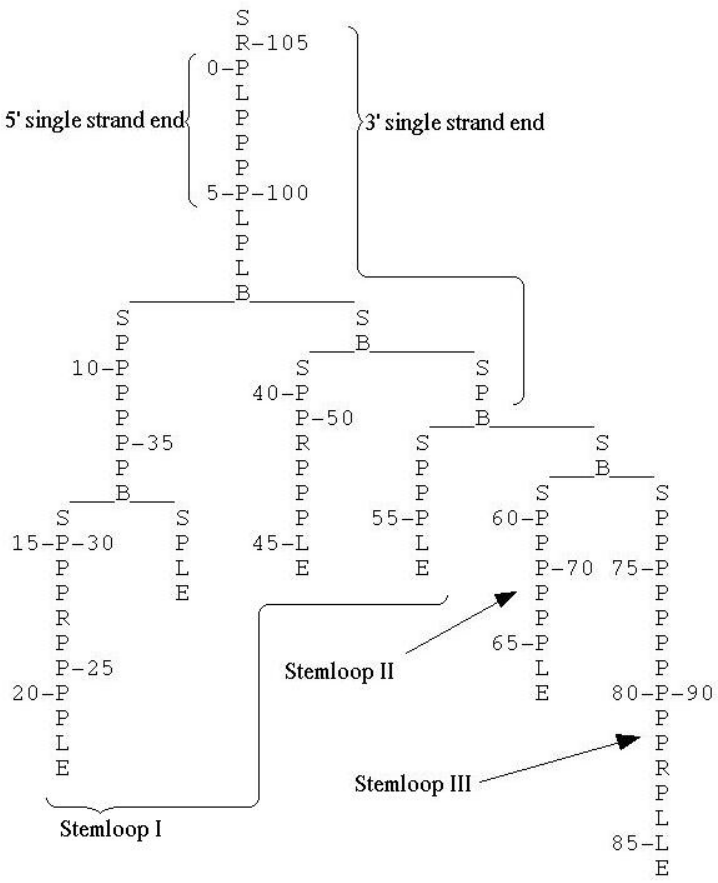
et al. 2003), medan sekundærstrukturen til *spf* (Figur 30) er skildra av Møller (2002). Desse er skildringar av sekundærstrukturen til dette genet i *E. coli.*



Figur 30. Ein verifisert sekundærstruktur for spf. Denne sekundærstrukturen er fyrst predikert med MFOLD og deretter er den predikerte strukturen underbygd med ulike laboratorieforsøk (Møller *et al*. 2002).

3.4.3 Resultat

For genet spf var forbetringa enorm. Figur 31 syner den predikterte strukturen. Resultata frå forsøket er oppsummert i Tabell 8.



Figur 31. Sekundærstrukturen til *spf* prediktert på bakgrunn av den justerte formelen *M'*. Her er det skrive på figuren kva område som høyrer til dei ulike områda på figuren som syner den antatt riktige sekundærstrukturen til *spf* i *E. coli*. *Stemloop* II og *stemloop* III er heilt korrekt predikert medan *stemloop* I har masse forgreiningar og har ingen korrekte basepar. Dei to endane som eigentleg skal vere enkelttråda er delvis para saman. Figuren er laga i Paint på grunnlag av utskrift frå implementasjon av CM-pakken.

	Nøsta basepar	Basepar predikert	% riktige av predikerte	% predikerte av riktige
ssrA	76	140	34,3 %	63,2 %
ssrA justert	76	164	42,1 %	90,8 %
spf	30	12	8,3 %	3,3 %
spf justert	30	46	37,0 %	56,7 %

Tabell 8. Resultat frå prediksjon av sekundærstrukturen til dei to treningssetta. Prediksjonen med berre samvarians som grunnlag er samanlikna med falding på grunnlag av ei likning som i tillegg tek omsyn til baseparfrekvens (sjå teksten for definisjon).

Det vart i høve denne testen definert eit kvalitetsmål på ein predikert sekundærstruktur. Fyrst vart talet på nøsta basepar i dei verifiserte sekundærstrukturane talt opp. I *spf* er dette alle basepara, medan i *ssrA*, som har pseudoknutar, kan ikkje alle basepara skildrast som nøsta samanhengar. Det maksimale talet basepar som kan predikerast på denne måten er funne ved å telle opp talet på basepar totalt og deretter er talet på basepar i den kortaste stammen i kvar pseudoknute trekt i frå (stammane 3, 11, 9 og 7 i Figur 15). Dette talet er gjeve i fyrste kolonnen i Tabell 8.

Kor mange av desse basepara ein kan predikere med dei ulike metodane, er presumptivt eit godt kvalitetskriterium. I den siste kolonnen er dette høvetalet gjeve i prosent. Det kjem her tydleg fram at M' er ei mykje betre matrise å ha som grunnlag for berekning av sekundærstruktur enn M. Dette gjeld mest for det minste datasettet.

Som riktige prediksjonar vart her rekna snittet ($\cap$) av dei predikterte basepara og dei riktige basepara. I tillegg er dei basepara som har eine basen forskyve med ein eller to posisjonar i høve til den riktige også teke med. Dette er forskyvingar som ville verte retta opp dersom ein køyrde den iterative delen av treningsprosedyren.

4 Diskusjon

Denne oppgåva kan sjåast som eit forprosjekt der målet er å finne ut kor godt samvariansmodellpakken høver som del av eit generelt deteksjonsprogram som skal leite etter ncRNA-gen i genomiske sekvensar. Dei empiriske undersøkingane som er gjort og teorien som er presentert set fokus på ulike sider ved denne modellpakken. Resultata skal vere med å danne grunnlaget for ei avgjerd om eit eventuelt program som implementerar CM-algoritmane skal innlemmast i et slikt generelt program. Dette vil her verte diskutert i tillegg til kva tilknyting CM-pakken i so fall bør ha til resten av programmet.

Dei ulike resultata frå førre kapittel vil verte tekne opp og sett i lyset av kvarandre og resultat som andre har komme fram til.

4.1 Prediksjon av sekundærstruktur på grunnlag av samvarians og baseparfrekvens

Ein forbetra måte å predikere sekundærstruktur der ein tek omsyn til talet på basepar i tillegg til samvarians er laga og prøvd ut. Tanken bak denne endringa var å lage eit grunnlag for sekundærstrukturprediksjon som også fungerar på små datasett. For det minste datasettet sin del auka delen av dei riktige basepara, som vart predikert av algoritmen, frå 3% til 57% ved innføring av det nye berekningsgrunnlaget for sekundærstruktur.

Ein anna måte å sjå dette resultata på er å sjå på kor mange av stammane i den korrekte sekundærstrukturen som vart predikert i dei to forsøka. For det minste datasettet var

auken her frå ingen til to. På Figur 30 er den sekundærstrukturen som her er nytta som referanse gjeven. Dei to stammane som på figuren heiter *stemloop* II og *stemloop* III vart med den nye formelen predikert korrekt. Den lengste stammen vart derimot delt opp i fleire greiner (Figur 31).

Dette er ei klar forbetring men ikkje tilfredstillande. Det er mogleg at dette ville sjå annleis ut dersom den iterative delen av CM-pakken var implementert, men dette er lite truleg. Det som endrar seg for kvar iterasjon er kvar einskild sekvens si samanstilling til modellen (og dei andre sekvensane). I dette datasettet er problemet at sekvensane i settet er so like at dei ikkje syner nokon varians. Ein kan av same grunn rekne med at den opphavlege samanstillinga gjort med *clustal W* er korrekt. Det er meir truleg at formelen kan forbetrast.

Formelen M' som grunnlag for prediksjon av sekundærstruktur

Matrisa M' er fylt opp ved hjelp av ein *ad hoc* formel som berre er meint å syne at det er mogleg å ta omsyn til denne informasjonen og nytte han i utrekninga av sekundærstrukturen til ei multippel samanstilling. Samvariansen $M_{i,j}$ er eit tal som varierer frå 0,0 til 2,0 og har eininga bit. $A_{i,j}$ er eit høvetal som seier kor stor del av para (x_i, x_j) som er typiske basepar, altso eit tal som varierar mellom 0,0 og 1,0 og har inga eining. Dette gjer det matematisk ukorrekt å addere desse to tala. På den andre sida kan ein sjå det som to tal som er i same storleiksorden og som seier noko om kor godt to kolonnar i ei multippel samanstilling høver som basepar. Summen vert eit tal som varierer mellom 0,0 og 3,0 og fungerar bra som grunnlag til å berekne sekundærstruktur med.

Men det er endringar i M' som kan gjerast for å få den betre. For det fyrste bør sjølve justeringa, A, ta inn i seg all den informasjonen som er presentert av Zuker (1981, 1999). Han vektar alle par etter kor sterke hydrogenbindingane er mellom basane. Han nyttar dessutan scoringsmatriser som gir dei ulike basepara score etter kva par som ligg føre i stammen. Denne scoren vert kalla *stacking energy*.

Eit anna aspekt som denne formelen her ikkje løyser optimalt er vektinga mellom M og A. Ein kan tenke seg at denne vektinga bør koplast til talet på strengar i samanstillinga slik at A vert nytta dersom det berre er ein streng og M vert nytta

dersom det er mange. Det er den glidande overgongen som er utfordrande å definere. Eit stort datasett kan ha dei same uheldige eigenskapane som *spf*- treningssettet i denne oppgåva hadde, nemleg for liten varians mellom dei ulike sekvensane. Difor bør vektinga koplast til total samvarians. Det vil sei at samvariansen vert vekta meir, i kvart par, dersom det er meir av han totalt. På denne måten kan ein sikre at sett som syner liten varians, og dermed liten samvarians, vert folda på grunnlag av basepar og *stacking energy*.

A har òg det problemet ved seg at den legg til ein konstant til samvariansen. Denne konstanten kjem av at det som regel vil vere nokre basepar i alle par av kolonnar. Gjennomsnittleg er denne konstanten 6/16 dersom ein legg til grunn at alle basane er like hyppig førekommande. Dette talet kjem frå dei 6 para som vert rekna som basepar av dei 16 moglege. Konsekvensen av det å legge til ein konstant er at modellen vert meir tilbøyeleg til å pare saman kolonnar generelt. For å oppvege for denne uheldige konsekvensen kan ein trekkje i frå denne konstanten. Konstanten bør bereknast utifrå basesamansetninga i dei aktuelle kolonnane.

Kjent sekundærstruktur

Dersom ein har ein kjent sekundærstruktur til ein eller fleire av RNA-sekvensane i den multiple samanstillinga, er det nok ei føremon å nytte denne kunnskapen. Ein laboratorieverifisert sekundærstruktur bør ein kunne nytte til å bygge ein samvariansmodell etter og deretter samanstille kvar av dei andre sekvensane i familien til denne.

4.2 Utvikling av treningssett

Dei to datasetta i denne oppgåva vart produsert med utgangspunkt i to gen, *spf* og *ssrA* frå *E. Coli.* Det vart søkt med programmet ParAlign (Rognes 2001) etter homologar i andre bakterielle genom. ParAlign gir lokale samanstillingar og nyttar berre sekvenslikskap utan å sjå på sekundærstrukturen. Treningssetta vart samanstilte med

eit nettbasert samanstillingsprogram (http://www.ebi.ac.uk/clustalw/) som nyttar algoritmen Clustal W (Thompson *et al.* 1994).

Eit treningssett har biologisk relevans dersom sekvensane som utgjer settet har tilnærma same funksjon i dei respektive organismane. For at eit treningssett skal vere godt til å trene ein CM, må sekvensane som utgjer det strukturelt sett vere rimeleg like. Det er ei vanleg holdning at strukturell likskap mellom genpodukt (ncRNA eller protein), impliserar funksjonell likskap (Durbin *et al.* 1998). Ein kan dermed utleie biologisk relevans utifrå strukturell likskap.

Ein kan sjå at *ssrA* er predikert 63% riktig når berre samvarians i treningssettet vart nytta som grunnlag. Grunnlaget for samvarians er det som er ulikt mellom sekvensane, altso det som ParAlign oppfattar som ”feil” og som der fører til ein dårlegare score. Det vil sei at dette datasettet, som er plukka ut på grunnlag av sekvenslikskap, viser ein strukturelt ”rimeleg riktig” varians. Det er difor grunn til å tru at datasettet stort sett består av biologisk relevante sekvensar, men at ein ikkje kan sei noko om den enkelte sekvens sin relevans. Eller sagt på ein annan måte, ein kan sei at treningssettet skildrar ei strukturell familie, men ein kan ikkje sei om alle sekvensane i settet er medlemmar av familien.

4.3 Databasen Rfam

Forsking på ncRNA er eit fagfelt i rivande utvikling og mange grupper rundt om i verda publiserer på området. Etter at det omtala treningssettet var laga, har det dukka opp ein database med treningssett for alle kjente ncRNA-familiar (Griffiths-Jones *et al.* 2003). Denne databasen, Rfam, ligg tilgjengeleg på http://Rfam.wustl.edu/. Her ligg multiple samanstillingar, konsensus sekundærstruktur og ferdige samvariansmodellar til 176 ulike ncRNA-familiar (april 2004), mellom anna dei to gena *ssrA* og *spf*.

Lage treningssett krev biologisk kunnskap

Det er klart at biologisk kunnskap er viktig for å lage eit godt og relevant treningssett. Tilnærminga brukt her kan nok raskt leie fram til nye kandidatar, men det er fare for at

feil og unøyaktigheiter akkumulerast. I Rfam skil dei mellom *seed* og *full* samanstilling. Seed samanstillinga er den opphavlege samanstillinga der alle sekvensane er annotert med sekundærstrukturinformasjon. Annotert tyder at den er skildra og lagt inn i ein søkbar database med alle kjende opplysingar og kopla til dei artiklane som har publisert opplysingane. Det er denne samanstillinga som er nytta for å trene CM-en. Sekvensane som er med i den fulle samanstillinga er homologe sekvensar som er funne i andre genom ved hjelp av CM-en. Dvs at alle sekvensane i treningssetta er samanstilt med biologisk kunnskap som utgangspunkt.

Mål

Bakgrunnen for å lage Rfam var å lage eit tilsvarande verkty som Pfam (Bateman *et al.* 2000) er for protein, til forskarar som ser etter ncRNA. Målet er å integrere alle multiple samanstillingar av eksisterande og framtidige ncRNA-familiar i eit felles strukturannotert format. Vidare skal det nyttast samvariansmodellar av desse familiane til å søkje dei stadig veksande sekvensdatabasane og oppretthalde automatisk genererte multiple samanstillingar av alle funne homologar til kvar familie. Til slutt skal Rfam utvikle og stille til rådvelde for ålmenta eit system for å analysere og annotere sekvensdata (som til dømes heile genom) med homologar til kjente ncRNA.

Lengda til *ssrA*

I Rfam sitt treningssett av *ssrA* (dei nyttar namnet tmRNA), er det nytta ein noko kortare versjon av genet. Grunnen til dei ulike lengdene er at RNA-molekylet vert prosessert etter at det er transkribert. Dette vert gjort av eit enzym kalla *RNase E* (Lin-Chao *et al.* 1999). Dvs at heile den delen som i denne oppgåva vart nytta til å lage treningssett for *ssrA*, vert transkribert til RNA, medan det funksjonelle RNA-molekylet er den kortare varianten som er resultatet av prosesseringa til *RNas E*. Det er interessant å finne ut om den delen som vert kappa vekk òg har ein konservert sekundærstruktur. I so fall bør den vere med i treningssettet for at denne informasjonen skal takast opp i modellen.

4.4 Køyretid og minnebruk

Det er køyretidskompleksiteten og minnebruken til CM-søkealgoritmen som er største problemet med CM. Her vert dette problemet diskutert frå ulike vinklar. Problema med køyretidskompleksitet og minnebruk er polynomiske i høve til lengda på treningssekvensane og lineære i høve til lengda på genoma ein vil søkje gjennom.

For å vurdere om denne kompleksiteten vert overkomeleg i framtida er det fleire variablar ein lyt ta omsyn til:

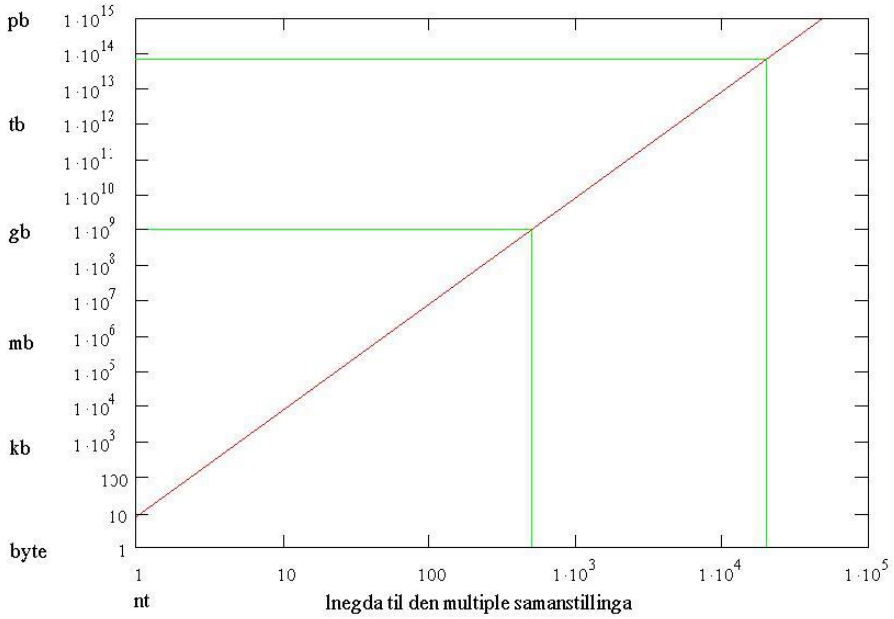
- Lengda på ncRNA-gen
- Talet på ncRNA-familiar
- Maskiners rekne- og minnekapasitet
- Talet på sekvenserte genom

Lengda på ncRNA-gen

Lengda på ncRNA-gen kan ein rekne med vil halde seg konstant eller endre seg minimalt. Storz (2002) nemner lengder frå 20 nt til over 10 000 nt. I Rfam (Griffiths-Jones *et al.* 2003) varierer lengdene mellom 26 nt og 474 nt og gjennomsnittet er 125 nt. Erdmann (2000) snakkar om ncRNA som regulerer utrykket frå X-kromsom hjå kvinner som er opp til 17 000 basepar lange. Talet på tilstandar i CM-en er ca 3,5 gonger lengda på den multiple samanstillinga. Søketida auka kubisk i høve til lengda på sekvensane i treningssettet. Det er klart at dette ikkje er noko problem for treningssett med opp til 500 kolonnar og at det er eit problem for treningssett med 17000 kolonnar. Kor grensa bør gå vil variere mellom maskiner og implementasjonar og er vanskeleg å sei noko generelt om. Minnebruken er noko enklare å gripe teoretisk, og vert difor diskutert litt meir i detalj lenger ned. Når Rfam søkjer gjennom nysekvenserte genom etter homologar, nyttar dei BLAST som ein rask fyrste gjennomgang for deretter å søkje gjennom resultata derifrå med CM-søkjealgoritmen på same måte som tRNAscan-SE nyttar ein heurestikk til å plukke ut kandidatar.

Den dynamisk programmering-matrisa som vert nytta under søking er i storleiksorden $O(w^2M)$ der w er lengda på vindauga vi søkjer med og M er talet på tilstandar. Det

kjem fram av Figur 32 at det lengste treningssettet i Rfam som er 474 nt, kjem akkurat innanfor storleiken av minnet til ei typisk skrivebordsdatamaskin i dag (storleiksorden 1 gb), medan dei lengste ncRNA-molekyla som er skildra krev opp mot 100 terrabyte minne.



Figur 32. Logaritmisk plot av minnebruken til søkealgoritmen til CM-pakken mot gjennomsnittslengda av sekvensane i den multiple samanstillinga. Her er w satt 20% lenger enn gjennomsnittslengda av treningssekvensane, talet på tilstandar er rekna som 3,5 gonger treningssekvensanes lengde og kvar celle i matrisa får to byte til å lagre scoren. Plotet er laga i Matlab og omarbeida i Paint.

Talet på ncRNA-familiar

Talet på ncRNA-familiar er eit spørsmål som har vore diskutert ei stund utan at noko klart svar har dukka opp. John S. Mattick (2003a) seier at det i *E. Coli* truleg er kring 200 ulike ncRNA-gen, kanskje fleire. Han reknar det for sannsynleg at alt transkribert materiale i eukariote organismar som ikkje er mRNA, er ncRNA-gen. Dvs fleire titals tusen gen. I Rfam (Griffiths-Jones *et al.* 2003) som er den mest komplette samlinga av ncRNA-gen, er det skildra 173 ulike klassar av gen frå både prokaryote og eukaryote organismar. Denne samlinga er laga for å lage CM-ar og lyt sjåast i det lyset. Dei har utelete ncRNA-gen som av ulike grunnar ikkje høver til å lage CM-ar av (Griffiths-Jones *et al.* 2003). Storz (2002) snakkar om 50 til 500 ncRNA i *E. coli* og at det kan

vere frå nokre hundre til nokre tusen miRNA i *Caenorhabditis elegans*. MiroRNA (miRNA) er ein klasse av ncRNA som Rfam har delt inn i 14 ulike familiar.

Eit søkeprogram som trenar ein CM for kvar av alle kjente ncRNA-klassar vil ha ein tidsbruk som aukar lineært med talet på slike klassar, sidan søkealgoritmen lyt nyttast ein gong for kvar klasse.

Maskiners minne og reknekapasitet

Maskiners reknekraft har auka eksponensielt i fleire tiår og ser ut til å gjere det ei stund til. Doblingstida er rekna for å vere kring 18 månader (Moors lov). Ei typisk ny skrivebordsmaskin har i dag (i storleiksorden) 1 GB minne og ein klokkefrekvens på 3 GHz.

Talet på sekvenserte genom

Talet på sekvenserte genom har òg auka eksponensielt dei siste tiåra (Figur 2). Denne kurva vil kanskje flate ut etter kvart som dei mest interessante (vitskapleg og økonomisk) genoma vert ferdigsekvensert. Eller kanskje ikkje sidan utstyret for å sekvensere med vert betre, billigare og fleire med tida.

Det er altso berre talet på sekvenserte genom som har eksponensiell auke med tida. Dette er i og for seg ikkje eit problem for kvar einskild brukar av eit program som skal nytte det for å finne ncRNA i "sitt" genom. Det er difor nærliggande å konkludere at dette er ein metode som har framtida føre seg og vert betre og betre etter kvart som reknekrafta i maskiner vert betre og fleire klassar av ncRNA vert skildra.

Heurestikkar som løysing på køyretidskompleksitet

I tRNAscan-SE er det nytta to heuristiske program som raskt finn kandidatar til tRNA-gen. Deretter er det nytta ein CM til å verifisere om desse kandidatane faktisk er tRNA-gen. Det er nokolunde same oppbygginga som er nytta i Rfam der det fyrst vert plukka ut kandidatar ved hjelp av det heuristiske søkeprogrammet BLAST. Alle treff med ein E-verdi < 10 vert rekna som kandidatar. For å verre sikker på å få med seg nok vert kandidaten utvida litt i båe retningane før den vert søkt gjennom av ein CM (Griffiths-Jones *et al.* 2003).

Denne oppbygginga verkar som ein god og fornuftig måte å komme seg rundt kompleksiteten til CM sin søkealgoritme. Det viktige er at den heurestikken som plukkar ut kandidatsettet har høg deteksjonrate, helst 100%.

4.5 tRNAscan-SE

Eddy og Durbin (1994) skildra to forsøk der ein samvariansmodell er samanlikna med det, til då, beste spesialbygde programmet. Det spesialbygde programmet TRNASCAN er laga for å finne tRNA. CM-en vart trena på eit datasett av alle kjente førekomstar av tRNA. Både denne CM-en og TRNASCAN er no delar av programmet tRNAscan-SE.

Resultata frå desse forsøka er gjeve i Tabell 9. Fyrste forsøket var på ein 2,2 Mb lang genomisk sekvens frå *C. elegans* og det andre på heile det mitokondriske genom til *Podospora anserina*.

	C. elegans		*Podospora anserina*	
Kvalitetskriterium	TRNASCAN	CM	TRNASCAN	CM
Deteksjons prosent	97,5%	>99,98%	67%	100%
Falske positivar	0,37/Mb	<0,2/Mb	0	0

Tabell 9. Resultat frå testar gjort samvariansmodellen. Ein lyt sjå desse resultata i lys av at det er forfattarane av modellen som har utført testane.

Desse tala må sjåast i lyset av at dei er presentert i den artikkelen som legg fram samvariansmodellen. Dei er likevel ein indikasjon på at ein CM er godt eigna som søkjeverkty. Tala frå køyringa av tRNAscan-SE på heile genomet til *E. coli* som vart gjort i høve denne oppgåva syner same gode deteksjonsraten (100%). Der vart det funne to gen som ikkje er verifisert til å vere tRNA, der programmet hevdar at det eine er eit pseudogen. Dersom den posisjonen som er hevda å vere eit gen, viser seg ikkje å vere eit gen, vil andelen falske positivar vere ca 0,2/Mb (ein falsk positiv i genomet til *E. coli* som er på 4,6 millionar basar).

Er det funne to nye tRNA i *E. coli*?

For å finne ut om dei to nye gena ligg i intergenetiske områder, er det søkt med både FASTA og BLAST etter annoteringar for dei respektive sekvensane. For den sekvensen som tRNAscan-SE tok for å vere eit tRNA, er det ingen ting som er annotert. Den sekvensen som programmet tok for å vere eit pseudogen, overlappar

med noko som er annotert som eit ”hypotetisk proteinkodande gen”. Det vil sei at programmet GeneMark (Borodovsky og McIninch 1993) har predikert denne posisjonen som eit proteinkodande gen men ingen har verifisert det.

Dette er interessante funn som absolutt vil sei noko om CM-pakkens evne til å plukke ut relevante sekvensar. Nærare bestemt vil ei avkrefting eller stadfesting av desse to potensielle gena hjelpe til med to ting. For det fyrste vil den sei noko både om selektivitet og sensitivitet for modellen slik den er nytta her. For det andre kan det vere til hjelp for å justere terskelverdiane. Det må eit laboratorieforsøk til for å avkrefte eller stadfeste om desse verkeleg vert uttrykt i *E. Coli* og den avgjerda er utanfor rekkevidda av denne oppgåva.

5 Oppsummering og Konklusjon

5.1 Kva føremoner og ulemper har CM-pakken?

Søkealgoritmen i CM-pakken er svært sensitiv. Det forsøket som er gjort her og dei som er skildra i litteraturen syner at stort sett alt som skal finnast vert funne (Eddy og Durbin 1994, Lowe og Eddy 1997) (Tabell 6). Det spesielle her er at det i tillegg vert funne svært få falske positivar. Dette er denne framgangsmåtens store styrke.

berekningskompleksiteten er ei ulempe med samvariansmodellen. Eit søk med denne teknikken er lineær i høve til lengda på databasesekvensen men er kubisk i høve til lengda på treningssekvensane (altso lengda på det ein ser etter).

Denne kompleksiteten er mogleg å komme seg rundt ved å nytte andre teknikkar som er raskare til å finne kandidatar (til dømes heuristiske homologisøk som i Rfam eller signalsøk som i tRNAscam-SE) og deretter nytte samvariansmodellar for alle kjente ncRNA-familiar til å luke ut falske positivar. Dette vil dramatisk minke talet på nukleotidar som CM-en er nøydt til å leite gjennom.

Som teknikk nytta i eit generelt søkeprogram som skal finne nye ncRNA-gen, har denne metoden enno ei svak side. Ein lyt vite kva den skal sjå etter. Ein kan altso ikkje finne noko anna i eit nysekvensert genom enn dei ncRNA-familiane ein har treningssett for. Dette er heilt klart ei lyte ved modellen, men den vert mindre og mindre signifikant etter kvart som fleire familiar av ncRNA-gen vert skildra, og dei familiane som er skildra vert større.

CM-pakken er som modellpakke ikkje triviell. Det tarvst relativt djup matematisk forståing for å forstå algoritmane i detalj, noko ein lyt for å kunne programmere dei. Dette er nok noko av grunnen til at denne teknikken ikkje er meir nytta enn kva den er sjølv om den vart skildra for ti år sidan.

5.2 Kan modellen forbetrast?

Det ser ut til at det i små datasett kan vere svært nyttig å sjå på både talet på basepar og *stacking energy* når ein skal predikere sekundærstrukturen til ei multippel samanstilling.

5.3 Vidare arbeid

Ein veg vidare er å ta tak i dei forbetringane som er skildra i avsnitt 3.4 og diskutert i avsnitt 4.1 og sjå vidare på dei. Det burde gå an å lage ei algoritme som kan trene ein CM på ein sekvens ved å falde den på same måte som programmet MFOLD og ha ein glidande overgong til samvarians etter som treningssetta vert større.

Dette er eit fag i rivande utvikling. Medan dette hovudfagsprosjektet har laga to treningssett, har det blitt publisert ein stor database av treningssett for 172 ulike ncRNA-gen, Rfam. Det er klart at vidare arbeid på treningssett lyt ta utgangspunkt i denne databasen.

Når eit generelt søkeprogram etter ncRNA-gen vert laga, bør det inngå eit søk etter kjente ncRNA-familiar ved hjelp av CM-ar for dei familiane ein kjenner til.

6 Bibliografi

Altschul SF, Gish W, Miller W, Meyers EW og Lipman DJ (1990). *Basic Local Alignment Search Tool*. Journal of Molecular Biology 215(3): 403-410.

Argaman L, Hershberg R, Vogel J, Bejerano G, Wagner EGH, Margalit H og Altuvia S (2001). *Novel small RNA-encoding genes in the intergenic regions of Escherichia coli*. Current Biology 11(12): 941-950.

Bateman A, Birney E, Durbin R, Eddy SR, Howe KL og Sonnhammer ELL (2000). *The Pfam Protein Families Database*. Nucleic Acids Research 28(1): 263-266.

Bernal A, Ear U og Kyrpides N (2001). *Genomes OnLine Database (GOLD): a monitor of genome projects world-wide*. Nucleic Acids Research 29(1): 126-127.

Borodovsky M og McIninch J (1993). *GeneMark: parallel gene recognition for both DNA strands*. Computers & Chemistry 17(19): 123-133.

Bucher P og Bairoch A (1994). *A generalized profile syntax for biomolecular sequence motifs and its function in automatic sequence interpretation*. Proceedings of the International Conference on Intelligent Systems for Molecular Biology 2: 53-61.

Burge C og Karlin S (1997). *Prediction of complete gene structures in human genomic DNA*. Journal of Molecular Biology 268(1): 78-94.

Carter RJ, Dubchak I og Holbrook SR (2001). *A computational approach to identify genes for functional RNAs in genomic sequences*. Nucleic Acids Research 29(19): 3928-3938.

Chiu D og Kolodziejczak T (1991). *Inferring consensus structure from nucleic acid sequences*. Bioinformatics 7(3): 347-352.

Chomsky N (1959). *On certain formal properties of grammars*. Information and Control 2: 137-167.

Dragon F*, et al.* (2002). *A large nucleolar U3 ribonucleoprotein required for 18S ribosomal RNA biogenesis.* Nature 417(6892): 967-970.

Durbin R, Eddy SR, Krogh A og Mitchison G (1998). *Biological sequence analysis*, Cambridge University Press, Cambridge.

Eddy SR og Durbin R (1994). *RNA sequence analysis using covariance models*. Nucleic Acids Research 22(11): 2079-2088.

Eddy SR (2002). *A memory-efficient dynamic programming algorithm for optimal alignment of a sequence to an RNA secondary structure*. BioMed Central Bioinformatics 3(1): 18-33

Erdmann VA, Szymanski M, Hochberg A, Groot Nd og Barciszewski J (2000). *Non-coding, mRNA-like RNAs database Y2K*. Nucleic Acids Research 28(1): 197-200.

Garey MR og Johnson DS (1979). *Computers and intractability*, W.H. Freeman and Company, New York.

Griffiths-Jones S, Bateman A, Marshall M, Khanna A og Eddy SR (2003). *Rfam: an RNA family database*. Nucleic Acids Research 31(1): 439-441.

Han K og Kim H (1993). *Prediction of common folding structures of homologous RNAs*. Nucleic Acids Research 21(5): 1251-1257.

Henderson J, Salzberg S og Fasman KH (1997). *Finding genes in DNA with a Hidden Markov Model*. Journal of Computational Biology 4(2): 127-141.

Hershberg R, Altuvia S og Margalit H (2003). *A survey of small RNA-encoding genes in Escherichia coli*. Nucleic Acids Research 31(7): 1813-1820.

Hesthagen IH og Langangen A (1990). *Biologi 3Bi*, Forlaget Fag og Kultur A/S, Oslo.

Klein RJ, Misulovin Z og Eddy SR (2002). *Noncoding RNA genes identified in AT-rich hyperthermophiles*. Proceedings of the National Academy of Sciences of the United States of America 99(11): 7542-7547.

Klein RJ og Eddy SR (2003). *RSEARCH: Finding homologs of single structured RNA sequences*. BioMed Central Bioinformatics 4(1): 44-59.

Kyrpides N (1999). *Genomes OnLine Database (GOLD 1.0): a monitor of complete and ongoing genome projects world-wide.* Bioinformatics 15(9): 773-774.

Lagesen K og Rognes T (2002). *Computational methods for identifying functional non-coding RNA genes in genomic sequences*. Prosjektskildring, Oslo

Lander ES*, et al.* (2001). *Initial sequencing and analysis of the human genome*. Nature 409(6822): 860-921.

Laslett D og Canback B (2004). *ARAGORN, a program to detect tRNA genes and tmRNA genes in nucleotide sequences*. Nucleic Acids Research 32(1): 11-16.

Lewis HR og Paradimitriou CH (1998). *Elements of the theory of computation*, Prentice Hall, New York.

Lim LP, Glasner ME, Yekta S, Burge CB og Bartel DP (2003a). *Vertebrate MicroRNA Genes*. Science 299(5612): 1540.

Lim LP, Lau NC, Weinstein EG, Abdelhakim A, Yekta S, Rhoades MW, Burge CB og Bartel DP (2003b). *The microRNAs of Caenorhabditis elegans*. Genes and Development 17(8): 991-1008.

Lin-Chao S, Wei C-L og Lin Y-T (1999). *RNase E is required for the maturation of ssrA RNA and normal ssrA RNA peptide-tagging activity*. Proceedings of the National Academy of Sciences of the United states of America 96(22): 12406-12411.

Lowe T og Eddy SR (1997). *tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence*. Nucleic Acids Research 25(5): 955-964.

Lowe T (2001). *tRNAscan-SE: a program for improved detection of transfer RNA genes in genomic sequence*. Brukarmanual til tRANscan-SE, Santa Cruz, CA

Mathews DH, Sabina J, Zuker M og Turner DH (1999). *Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure1, *1*. Journal of Molecular Biology 288(5): 911-940.

Mattick JS (2003a). *Challenging the dogma: the hidden layer of non-protein-coding RNAs in*. Bioessays 25(10): 930-939.

Mattick JS (2003b). *The human genome and the future of medicine*. Medical Journal of Australia 179(4): 212-216.
Møller T, Franch T, Udesen C, Gerdes K og Valentin-Hansen P (2002). *Spot 42 RNA mediates discoordinate expression of the E. coli galactose operon*. Genes and Development 16(13): 1696-1706.
Needleman SB og Wunsch CD (1970). *A General Method Applicable to the Search for Similarities in the Amino Acid Sequence of Two Proteins*. Journal of Molecular Biology 48: 443-453.
Nussinov R, Pieczenik G, Griggs JR og Kleitman D (1978). *Algorithms for loop matchings*. SIAM Journal of Applied Mathematics 35: 68-82.
Pearson WR og Lipman DJ (1988). *Improved tools for biological sequence comparison*. Proceedings of the National Academy of Sciences of the United States of America 85(8): 2444-2448.
Polayes DA, Rice PW og Dahlberg JE (1988). *DNA polymerase I activity in Escherichia coli is influenced by spot 42*. Journal of Bacteriology 170(5): 2083-2088.
Rivas E og Eddy SR (1999). *A dynamic programming algorithm for RNA structure prediction including pseudoknots*. Journal of Molecular Biology 285(5): 2053-2068.
Rivas E og Eddy SR (2000a). *Secondary structure alone is generally not statistically significant for the detection of noncoding RNAs.* Bioinformatics 16(7): 583-605.
Rivas E og Eddy SR (2000b). *The language of RNA: a formal grammar that includes pseudoknots*. Bioinformatics 16(4): 334-340.
Rivas E og Eddy SR (2001). *Noncoding RNA gene detection using comparative sequence analysis*. BMC Bioinformatics 2(1): 8-26.
Rivas E, Klein RJ, Jones TA og Eddy SR (2001). *Computational identification of noncoding RNAs in E. coli by comparative genomics*. Current Biology 11(17): 1369-1373.
Rognes T (2001). *ParAlign: a parallel sequence alignment algorithm for rapid and sensitive database searches*. Nucleic Acids Research 29(7): 1647-1652.
Rognes T (2003). *Databaser og databasesøk*. lecture notes, Oslo
Schattner P (2002). *Searching for RNA genes using base-composition statistics*. Nucleic Acids Research 30(9): 2076-2082.
Sigrist CJ, Cerutti L, Hulo N, Gattiker A, Falquet L, Pagni M, Bairoch A og Bucher P (2002). *PROSITE: a documented database using patterns and profiles as motif descriptors*. Briefings in Bioinformatics 3(3): 265-274.
Smith TF og Waterman MS (1981). *Identification of common molecular subsequences*. Journal of Molecular Biology 147(1): 195-197.
Storz G (2002). *An Expanding Universe of Noncoding RNAs*. Science 244: 1260 - 1263.
Thompson JD, Higgins DG og Gibson TJ (1994). *CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice*. Nucleic Acids Research 22(22): 4673-4680.
Venter JC*, et al.* (2001). *The Sequence of the Human Genome*. Science 291(5507): 1304-1351.

Waterman MS (1995). *Introduction to computational biology, Maps, sequences and genomes*, Chapman & Hall, Cambridge.
Watson DJ og Crick FHC (1953). *Genetical implications of the structure of deoxyribonucleo acid*. Nature 171: 964-967.
Weiss MA (1999). *Data Structures and Algorithm Analysis in Java*, Addison-Wesley Longman
Zuker M og Stiegler P (1981). *Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information*. Nucleic Acids Research 9(1): 133-148.
Zuker M (2003). *Mfold web server for nucleic acid folding and hybridization prediction*. Nucleic Acids Research 31(13): 3406-3415.