Sztochasztikus modellek a bioinformatikában

Miklós István, Rényi Intézet
2011 tavasz

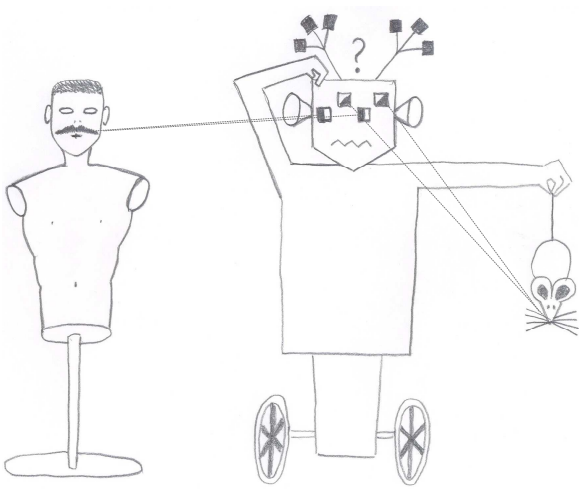
(készülő jegyzet, last update: 11/4/2013)
A jegyzet OTKA támogatásával készült (PD 84297)

1. fejezet

Bevezető

1.1. A marslakó és a modellek

Az alkalmazott matematika sajátossága, hogy a tiszta matematikát összekapcsolja egy másik tudományterülettel. Amíg a matematikában abszolút igazságok vannak, abszolút igazság más tudományterületeken nem létezik. Az alkalmazott matematika feladata az, hogy matematikai modelleket alkosson a világ leírására, és ezeken keresztül próbáljuk meg megérteni a világot. Azonban meg kell értenünk, hogy hogyan működnek a modellek, nehogy úgy járjunk, mint a marslakó az alábbi mesében.



Egy marslakó vizsgálni szeretné a Földön élő embereket, azonban nem mer közvetlen kapcsolatba lépni velük. Hogy mégis tanulmányozhassa őket, modelleket kér. Két modellt kap, egyet egy gyógyszerkutatótól, aki egy fehér egeret küld, egy másikat meg egy divattervezőtől, aki egy viaszbábut küld.

Szegény marslakó ledöbben, hogy mennyire változatosak lehetnek az emberek. Az egér folyamatosan rohangászik, cincogó hangokat hallat, és meglehetősen kicsi, míg a viaszbábú sokkal nagyobb, ellenben teljesen mereven és szótlanul áll, ráadásul sokkal hidegebb, mint az egér. Végül a marslakó levonja a konklúziót a modellekből: a földi emberek nagyon változatosak, méretük néhány centimétertől közel két méterig terjed, viszont mindegyiknek van bajusza.

Természetesen a marslakó rossz következtetéseket vont le a modellekből. De vajon mit rontott el? Ott hibázott, hogy nem kérdezte meg a gyógyszerkutatótól és a divattervezőtől, hogy a modellük a földi embereknek mely aspektusát modellezeik. A gyógyszerkutató azért tud egy fehér egeret használni modellként, mert az egerek és az emberek biokémiai rendszerei között meglepően sok hasonlóság van, így ha egy gyógyszer hatásos egy egeren, jó eséllyel hatásos lesz embereken is. A divattervező azért tudja a viaszbábut modellként használni, mert a viaszbábu formája követi az emberi test alakját, ezért ha egy ruha jól áll egy viaszbábun, valószínűleg jól fog állni egy emberen is. Mivel a két modell az emberek más-más aspektusát modellezik, teljesen értelmetlen a két modell hasonlóságait keresni. Ehelyett meg kell érteni, hogy a modellek a valóság mely aspektusát modellezik, és akkor helyes következtetéseket vonhatunk le: a földi emberek külsőleg hasonlítanak egy viaszbábura, a belső biokémiai felépítésük viszont egy fehér egér biokémiai rendszeréhez hasonlít.

Fontos azt is megérteni, hogy a modellek is csak modellezik a valóság egy-egy aspektusát, de nem követik tökéletesen, ezért még a modell helyes értelmezése esetén is fennáll a tévedés lehetősége. Például, egy viaszbábu, ellentétben az emberekkel, nem mozog. Ha egy ruha jól áll egy statikus viaszbábun, nem biztos, hogy jól áll egy emberen, aki mozgás során összagyűrheti a ruhadarabot, vagy bizonyos testhelyzetekben teljesen másképpen viselheti a ruhadarabot, ahogy az a viaszbábun áll. Hasonlóan, ha egy gyógyszer hatékony az egereken, még nem jelenti feltétlenül azt, hogy az embereken is hatékony lesz.

Természetesen, a modelleken lehet javítani. A fenti példáknál maradva, létrehozhatunk egy olyan robotot, amely külsejében ugyanolyan jól modellezi az emberi külsőt, mint egy viaszbábut, ugyanakkor a viaszbábuval szemben az emberek mozgását is utánozni tudja. Hasonlóan, egy gyógyszert lehet teszteni emberhez közelebb álló fajokon, például csimpánzokon is. Fontos megjegyezni, hogy a modellek soha nem lesznek tökéletesek: pl. egy roboton sem tudjuk vizsgálni azt, hogy egy ruhadarabba egy ember mennyire izzad bele, hasonlóan, van különbség az ember és a majmok biokémiai rendszerei között is.

Milyen tanulságokat vonhatunk le a fenti meséből? Tudnunk kell, hogy a matematikai modellek, amelyekkel a kurzus során megismerkedünk, mely aspektusát modellezik a valóságnak. A matematikai eredményeket ennek megfelelően kell értelmezni. Másrészt a modell tökéletesítése során mindig a valódi világra kell koncentrálni. Ez a szemlélet nagyon eltér a tiszta matematikában megszokott szemlélettől. A tiszta matematikában kialakult az a konszenzus, hogy mely kérdések az érdekesek, egy kutatást milyen irányba érdemes elvinni. A kutatás irányát a matematika szépsége vezérli, azt pedig, hogy mely eredmények, tételek tekinthetőek szépnek, egy évszázadok során kialakult konszenzus mondja meg.

Az alkalmazott matematika kutatási irányait nem a tiszta matematika szépsége, hanem a modellek használhatósága határozza meg. A modell használhatósága nem csak a valóság modellezésének a jóságától függ, hanem a modellel való számolások hatékonyságától is, amint ezt a következő alfejezetben látni is fogjuk.

1.2. Bioinformatikai modellek

A bioinformatikában alkalmazott modellek abban különböznek a többi biológiában alkalmazott, ún. klasszikus biomatematikai modellektől, hogy nagyon erőteljesen használnak kombinatorikai, diszkrét matematikai struktúrákat. Mint látni fogjuk, ezen struktúrák használata nagyon természetes: a biológiai szekvenciákat véges ABC feletti stringekkel, az evolúciós leszármazási kapcsolatokat gyökeres bináris fákkal fogjuk modellezni. Egy következménye a kombinatorikai struktúrák alkalmazásának, hogy a bioinformatikai modellalkotásban a használhatóságba sokkal jobban beleszól a számolás hatékonysága, mint más modellek esetében. Valóban, nagyon könnyű olyan modelleket megadni, amelyek sokkal jobban leírják a biológiai valóságot, mint a kurrens modellek, azonban a kombinatorikai struktúrákból adódó ún. kombinatorikus robbanás kezelhetetlensége miatt ezen modellek egyáltalán nem használhatóak.

A bioinformatikai modellalkotást tehát alapvetően meghatározza a modelleken számoló algoritmusok futási ideje. A számítástudományban használt klasszikus megközelítést fogjuk alkalmazni, a futási idő nagyságrendjét a bemenő adatok méretének a függvényében mondjuk meg. Általában természetes lesz a bemenő adatok mérete: a szekvenciák hossza, a fajok száma, és így tovább. A kurzus során

először azon modellekkel ismerkedünk meg, amelyben minden fontos kérdés megválaszolására polinom futási idejű algoritmust tudunk megadni. Majd megismerkedünk néhány klasszikus NP-teljes problémával, a kurzus végén pedig sztochasztikus approximációkkal fogunk foglalkozni.

1.3. Paraméterbecslés

Nagyon gyakran egy modellt azért állítunk fel, hogy segítségével paramétereket becsüljünk. Legyen M egy modell, θ a modellben szereplő paraméterek valamely rögzített értékei, D pedig az észlelt biológiai adat. A központi függvény az ún. likelihood függvény:

$$P(D|\theta, M) \tag{1.1}$$

azaz, az adatok észlelésének a valószínűsége az adott modellben, az adott paraméterek mentén. Az (1.1) függvényt a θ paraméterek likelihood függvényének hívjuk. Nagyon gyakran egyértelmű lesz, hogy melyik modellről beszélünk, ezekben az esetekben a jelölésekben a modell jelölését elhagyjuk.

Az egyik leggyakoribb parméterbecslés a maximum likelihood (ML) becslés:

$$\hat{\theta}_{ML} := \arg\max_{\theta} \{P(D|\theta)\} \tag{1.2}$$

azaz azon paraméterek, amelyek maximalizálják a likelihoodot. Fontos megérteni, hogy ez nem a paraméterek valószínűsége az adatok észlelése mentén, hiszen azt a Bayes tétel segítségével adhatjuk meg:

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{\int_{\theta'} P(D|\theta')P(\theta')d\theta'} \tag{1.3}$$

ahol $P(\theta)$ a paramétereknek az ún. prior valószínűsége, azaz a paraméterek valószínűsége azelőtt, hogy az adatokat észlelnénk. Általában a Bayes tétel nevezőjében szereplő integrál nem számolható ki analitikusan, ezért nagyon gyakran a Bayes tételt így írjuk fel:

$$P(\theta|D) \propto P(D|\theta)P(\theta) \tag{1.4}$$

vagy szavakkal

$$\textit{poszterior} \propto \textit{likelihood} \times \textit{prior} \tag{1.5}$$

ahol $\propto$ jelöli az arányosságot. A Bayes tétel alapján írhatjuk fel a másik leggyakoribb paraméterbecslést, a *maximum a posteriori* (MAP) becslést

$$\hat{\theta}_{MAP} := \arg\max_{\theta} \{P(\theta|D)\} = \arg\max_{\theta} \{P(D|\theta)P(\theta)\} \tag{1.6}$$

Az egyenlőség egyszerűen abból következik, hogy a Bayes tétel nevezőjében szereplő integrál pozitív és konstans, így a tört akkor lesz maximális, ha a számlálója maximális.

Mint látható, mind a maximum likelihood, mind a maximum *a posteriori* becsléshez szükséges, hogy bármely paraméter értékek mentén a likelihoodot könnyen kiszámolhassuk. A legegyszerűbb modellek esetén léteznek szofisztikált eljárások a ML paraméterek megkeresésére, bonyolultabb modellek esetében pedig klasszikus numerikus módszereket alkalmazhatunk, mint pl. a gradiens módszer, hogy megtaláljuk a likelihood függvény (legalább lokális) maximumát.

Nagyon gyakran a likelihood ilyen formában írható fel:

$$P(D|\theta) = \sum_{Y} P(D,Y|\theta) \tag{1.7}$$

ahol Y a szemlélő által nem ismert, valamilyen rejtett adat. Tipikusan a szummában az összeadandó tagok száma exponenciálisan növekszik az ismert adatok méretével, így a definíció szerinti közvetlen számolás komputációsan nagyon költséges. Egyszerű modellek esetén dinamikus programozási rekurzió segítségével mégis meg lehet határozni az összeget polinom futási időben. A dinamikus programozási algoritmusok a leggyakrabban használt algoritmikai módszerek a bioinformatikában, a kurzus első felében ilyen algoritmusról fogunk tanulni.

Sok modellben azonban a likelihood közvetlenül nem számolható, mert nem ismerünk hatékony algoritmust az (1.7) egyenletben szereplő szumma kiszámolására. Azonban ha minden Y-ra $P(D,Y|\theta)$ számolható, akkor lehetőség van a likelihood függvény sztohasztikus összegzésére. Ilyen sztohasztikus algoritmusokkal (Monte Carlo módszerekkel) ismerkedünk meg a kurzus második felében.

2. fejezet
Dinamikus programozás

A kurzus hallgatóiról feltételezem, hogy találkoztak már legalább egy dinamikus programozási algoritmussal (pl. Dijkstra algoritmusa, Bellman-Ford algoritmus). Azonban a dinamikus programozás annyira fontos a kurzus első felében, hogy itt néhány klasszikus bioinformatikai példán átismételjük, gyakoroljuk a dinamikus programozást.

2.1. Szekvenciaillesztés lineáris résbüntetéssel

Az információt hordozó DNS molekulák a sejt kettéosztódása előtt replikálódnak, az eredeti molekulával megegyező két molekula jön létre. Biokémiai szabályozás hatására mindkét utódsejtbe egy-egy DNS szál kerül, így az utódsejtek mindegyike tartalmazza a teljes genetikai információt. Azonban a DNS replikálódása nem tökéletes, véletlen mutációk hatására a genetikai információ kissé megváltozhat. Így egy egyed utódai között variánsok, mutánsok állnak elő, amikből az évmilliók alatt új fajok alakulnak ki.

Legyen adott két szekvencia, a kérdés az, hogy a két szekvencia mennyire rokon — azaz mennyi idő telt el a szétválásuk óta —, illetve milyen mutációk sorozatával lehet leírni a két szekvencia evolúciós történetét. Tegyük fel, hogy az egyes mutációk egymástól függetlenek, így egy mutáció sorozat valószínűsége az egyes mutációk valószínűségének a szorzata. Az egyes mutációkhoz súlyokat rendelünk, a nagyobb valószínűségű mutációk kisebb, a kisebb valószínűségű mutációk nagyobb súlyt kapnak. Egy nyilvánvalóan jó választás a valószínűség logaritmusának a mínusz egyszerese. Ekkor egy mutáció sorozat súlya az egyes mutációk súlyainak az összege. Feltételezzük, hogy egy mutáció és a megfordítottja — például egy timin szubsztitúciója adeninre és vissza — ugyanakkora valószínűséggel fordul elő. Így a két szekvencia közös ősből való leszármazása helyett azt kell vizsgálni, hogy hogyan alakulhatott ki az egyik szekvencia a másikból. A minimális evolúció elméletét feltételezve, azt a minimális súlyú mutáció sorozatot keressük, amely az egyik szekvenciát a másikba alakítja. Fontos kérdés, hogy hogyan lehet egy minimális súlyú mutáció sorozatot (lehet, hogy több minimális értékű sorozat van) hatékonyan megkeresni. A naiv algoritmus megkeresi az összes lehetséges mutáció sorozatot, és kiválasztja belőle a minimális súlyút. Könnyű belátni, hogy a lehetséges sorozatok száma legalább exponenciálisan nő a szekvenciák hosszával, ezért ez a naív algoritmus nyilvánvalóan nagyon lassú.

Az alábbiakban egy hatékony dinamikus programozási algoritmust mutatok be. Legyen Σ szimbólumok véges halmaza, Σ^* jelölje a Σ feletti véges hosszú szavak halmazát. Egy A szó első n betűjéből álló szót A_n jelöli, az n-edik karaktert pedig a_n. Egy szón különböző transzformációk hajthatók végre:

Egy *'a'* szimbólum beszúrása egy szóba egy adott pozíciónál. Ezt a transzformációt $- \rightarrow a$ jelöli.

Egy *'a'* szimbólum törlése egy szóból egy adott pozíciónál. Ezt a transzformációt $a \rightarrow -$ jelöli.

Egy *'a'* szimbólum cseréje egy *'b'* szimbólumra egy adott pozícióban. Ezt a transzformációt $a \rightarrow b$ jelöli.

A transzformációk kompozícióján azok egymás utáni végrehajtását értjük, és a ° szimbólummal fogjuk jelölni. A fenti három transzformáció, és ezek tetszőleges véges kompozícióinak a halmazát τ-val jelöljük. Azt, hogy egy $T \in \tau$ transzformáció az A szekvenciát B-vé transzformálja, $T(A)=B$-vel jelöljük.

Legyen w: $\tau \rightarrow R^+ \cup \{0\}$ egy olyan súlyfüggvény, amely bármely T_1, T_2 és S transzformációkra, ha

$$T_1 \circ T_2 = S \tag{2.1}$$

akkor

$$w(T_1)+w(T_2)=w(S) \tag{2.2}$$

Két szekvencia, A és B transzformációs távolságán az A-t B-be vivő minimális súlyú transzformációk súlyát értjük, azaz

$$\delta(A,B) = \min\{w(T)|T(A) = B\} \tag{2.3}$$

Ha w-ről feltesszük, hogy

$$w(a \rightarrow b) = w(b \rightarrow a) \tag{2.4}$$
$$w(a \rightarrow a) = 0 \tag{2.5}$$
$$w(a \rightarrow b) + w(b \rightarrow c) \leq w(a \rightarrow c) \tag{2.6}$$

bármely a, b, $c \in \Sigma \cup \{-\}$-re akkor könnyen belátható, hogy a $\delta(,)$ transzformációs távolság metrika Σ^*-on, így jogos a távolság elnevezés.

Fel szeretném hívni a figyelmet arra, hogy habár a súlyfüggvénnyel való számolást egy valószínűségelméleti okoskodás motiválta, nincsen egy egzakt és objektív származtatása a w transzformációs súlyoknak. Például, annak a valószínűsége, hogy egy karakter — aminosav vagy nukleotid — nem változott meg az evolúció során, nem 1, így ha transzformációs súlynak a valószínűség logaritmusának a mínusz egyszeresét választjuk, akkor $w(a \rightarrow a)$ semmiképpen sem lehet 0. Másfelől a mutációk valószínűségei függenek az eltelt időtől, de éppen ez az, amit szeretnénk meghatározni. Ennek ellenére a távolságalapú módszerek az egyszerűségük és gyorsaságuk miatt a mai napig elterjedtek.

Egy transzformáció sorozatot a szekvenciák illesztésével reprezentálunk. Konvenció alapján a felülre írt szekvencia az ősi szekvencia, az alulra írt a leszármazott. Például, az alábbi illesztés azt mutatja, hogy a harmadik és az ötödik pozícióban egy-egy szubsztitúció történt, az első pozícióban egy beszúrás, a nyolcadikban pedig egy törlés:

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

Az egyes pozíciókban az egymás alá írt szimbólumokat illesztett párnak hívjuk. A transzformáció sorozat súlya az egyes pozíciókban történt mutációk súlyainak az összege. Látható, hogy minden mutáció sorozathoz egyértelműen megadható egy illesztés, amely azt reprezentálja, és egy illesztésből a mutációk

sorrendjétől eltekintve egyértelműen megadhatóak azok a mutációk, amelyeket az illesztés reprezentál. Mivel az összeadás kommutatív, ezért a teljes súly független a mutációk sorrendjétől. Optimális illesztésnek nevezzük azt az illesztést, amelyhez rendelt mutáció sorozat súlya minimális. Jelöljük $\alpha^*(A_n,B_m)$-mel A_n és B_m optimális illesztéseinek a halmazát, $w(\alpha^*(A_n,B_m))$-mel jelöljük ezen illesztésekhez tartozó mutáció sorozatok súlyát.

A gyors algoritmus először az optimális illesztések súlyait számolja ki. Az ötlet az, hogy ha ismerjük $w(\alpha^*(A_{n-1},B_m))$-et, $w(\alpha^*(A_n,B_{m-1}))$-et, valamint $w(\alpha^*(A_{n-1},B_{m-1}))$-et, akkor ebből konstans idő alatt kiszámítható $w(\alpha^*(A_n,B_m))$. Ezt precízen a következő tétel mondja ki:

Tétel 2.1.

$$\begin{aligned} w(\alpha^*(A_n,B_m)) = \min\{ & w(\alpha^*(A_{n-1},B_m))+w(a_n \rightarrow -); \\ & w(\alpha^*(A_n,B_{m-1}))+w(- \rightarrow b_m); \\ & w(\alpha^*(A_{n-1},B_{m-1}))+w(a_n \rightarrow b_m)\} \end{aligned} \tag{2.7}$$

Bizonyítás. Belátjuk, hogy

$$\begin{aligned} w(\alpha^*(A_n,B_m)) \geq \min\{ & w(\alpha^*(A_{n-1},B_m))+w(a_n \rightarrow -); \\ & w(\alpha^*(A_n,B_{m-1}))+w(- \rightarrow b_m); \\ & w(\alpha^*(A_{n-1},B_{m-1}))+w(a_n \rightarrow b_m)\} \end{aligned} \tag{2.8}$$

és

$$\begin{aligned} w(\alpha^*(A_n,B_m)) \leq \min\{ & w(\alpha^*(A_{n-1},B_m))+w(a_n \rightarrow -); \\ & w(\alpha^*(A_n,B_{m-1}))+w(- \rightarrow b_m); \\ & w(\alpha^*(A_{n-1},B_{m-1}))+w(a_n \rightarrow b_m)\} \end{aligned} \tag{2.9}$$

ezért muszáj egyenlőségnek fennállnia. A (2.8) egyenlőtlenség bizonyításához vegyük A_n és B_m egy optimális illesztését. Ennek az utolsó oszlopában vagy a_n és b_n van összeillesztve, vagy a_n törlése, vagy b_n beszúrása látható. Ezt az oszlopot elhagyva rendre vagy A_{n-1} és B_{m-1}, vagy A_{n-1} és B_m, vagy A_n és B_{m-1} egy illesztését kapjuk. Ezen illesztés súlya nem lehet kisebb a legkisebb súlyú illesztés súlyánál, ehhez hozzávéve a most elvett oszlopot kapjuk, hogy A_n és B_m egy optimális illesztésének a súlya nem lehet kisebb vagy $w(\alpha^*(A_{n-1},B_m))+w(a_n \rightarrow -)$-nél, vagy $w(\alpha^*(A_n,B_{m-1}))+w(- \rightarrow b_m)$-nél, vagy $w(\alpha^*(A_{n-1},B_{m-1}))+w(a_n \rightarrow b_m)$-nél. Akár melyik eset áll fenn, $w(\alpha^*(A_n,B_m))$ nagyobb vagy egyenlő, mint a minimuma három olyan számnak, amelyik közül az egyikről tudjuk, hogy $w(\alpha^*(A_n,B_m))$-nél kisebb vagy egyenlő.

A (2.9) egyenlőtlenség belátásához rendre vegyük vagy A_{n-1} és B_m egy optimális illesztését, vagy A_n és B_{m-1} egy optimális illesztését, vagy A_{n-1} és B_{m-1} egy optimális illesztését, attól függően, hogy a minimumot rendre $w(\alpha^*(A_{n-1},B_m))+ w(a_n \rightarrow -)$ vagy $w(\alpha^*(A_n,B_{m-1}))+w(- \rightarrow b_m)$ vagy $w(\alpha^*(A_{n-1},B_{m-1}))+w(a_n \rightarrow b_m)$. A kiválaszott illesztéshez adjuk hozzá rendre vagy a_n törlését vagy b_n beszúrását vagy a_n és b_n összeillesztését. Így A_n és B_m egy illesztését kapjuk, amelynek a súlya a (2.9)

egyenlőtlenség jobboldalával egyezik meg. A_n és B_{m-1} bármely minimális súlyú illesztésének a súlya nem lehet ennél nagyobb, és pont ez az, amit be akartunk látni. □

Itt jegyezném meg, hogy a jegyzetben a továbbiakban nem adom meg a dinamikus programozási rekurziók bizonyítását, feltételezem, hogy a hallgatóknak nem okoz gondott az itt bemutatott bizonyítás alapján a bizonyításokat maguknak leírni.

Az optimális illesztéshez tartozó súlyokat egy ún. dinamikus programozási mátrix, **D** segítségével lehet megadni. Ez egy n és egy m hosszúságú szekvencia összehasonlítása esetén egy (n+1)x(m+1)-es táblázat, a sorok és oszlopok indexelése 0-tól n-ig illetve m-ig megy, $d_{i,j}$ definíció szerint $w(\alpha^*(A_i,B_j))$. A 0 hosszú prefix definíció szerint az üres szekvenciát jelenti. A kezdeti feltételek a nulladik sorra és oszlopra egyszerűen megadhatóak, mivel bármely szekvenciát az üres szekvenciához egyféleképpen lehet illeszteni:

$$\begin{aligned} d_{0,0}&=0 \\ d_{0,j}&=\sum_{l=1}^{j} w(-\rightarrow b_l) \\ d_{i,0}&=\sum_{l=1}^{i} w(a_l \rightarrow -) \end{aligned} \tag{2.10}$$

A táblázat belsejének a kitöltése a (2.7) képlet alapján történik.

A táblázat kitöltésének az ideje $O(nm)$. A táblázat segítségével megkereshetjük az összes optimális illesztést. Egy optimális illesztés megkereséséhez a jobb alsó sarokból kiindulva mindig a minimális értéket adó előző pozíciót választva — több lehetőség is adódhat — visszafelé haladunk a bal felső sarkig, minden egyes lépés az optimális illesztésnek egy illesztett párját adja meg. A $d_{i,j}$ pozícióból fölfelé lépés az adott pozícióban való törlést, a balra lépés az adott pozícióban való beszúrást jelent, az átlón való haladás pedig vagy az adott pozícióban való szubsztitúciót jelzi, vagy azt mutatja, hogy az adott pozícióban nem történt mutáció, attól függően, hogy a_i nem egyezik meg b_j-vel, vagy megegyezik. Az összes optimális illesztések száma exponenciálisan nőhet a szekvenciák hosszával, viszont polinomiális időben reprezentálható. Ehhez egy olyan irányított gráfot kell készíteni, amelynek a pontjai a dinamikus programozási táblázat elemei, és egy él megy egy v_1 pontból v_2-be, ha v_1 minimális értéket ad v_2-nek Ezen a gráfon minden irányított út $d_{n,m}$-ből $d_{0,0}$-ba egy optimális illesztést határoz meg.

2.2. Szekvenciaillesztés tetszőleges és affin résbüntetéssel

A beszúrásokat és törléseket együttes nevükön réseknek (angolul gap-nek) hívjuk. Módosíthatjuk a szekvenciaillesztés értékelését úgy, hogy azonos típusú réseknek egy hosszú sorozatát együttesen büntetjük, egy k hosszú résnek a büntetése ne függjön attól, hogy mely karaktereket töröltük ki vagy szúrtuk be. Erre a módosításra azért van szükség, mert tudjuk, hogy egyetlen lépésben több karakter is kitörlődhet illetve beszúródhat, ezért egy hosszú beszúrást vagy törlést egyetlen eseményként akarjuk kezelni, és egyetlen értéket akarunk adni neki.

Egy k hosszú rés büntetését g_k-val jelöljük. Ha g_k lineáris függvény, akkor könnyen belátható, hogy továbbra is igaz a (2.7) képletben megadott dinamikus programozási rekurzió. Azonban más résbüntetési függvényekre más rekurziót kell

megadni. Ha $w(\alpha^*(A_i,B_j))$ rövidítésére továbbra is $d_{i,j}$ -t használjuk, akkor tetszőleges résbüntetési föggvény esetén a dinamikus programozási rekurziót a következő tétel adja meg

Tétel 2.2. Tetszőleges résbüntetési függvényre fennáll, hogy

$$d_{i,j}= \min\{d_{i-1,j-1}+w(a_n \rightarrow b_m); p_{i,j}; q_{i,j}\} \tag{2.11}$$

ahol

$$p_{i,j}= \min_{1\le k< i} \{d_{n-k,m} +g_k\} \tag{2.12}$$

és

$$q_{i,j}= \min_{0\le k< j} \{d_{n,m-k} +g_k\}\} \tag{2.13}$$

A tétel bizonyítása a **2.1** tételhez hasonlóan megy, és a leírása gyakorlásként javasolt minden hallgató számára. Könnyen belátható, hogy az algoritmus futási ideje $O(nm(n+m))$, ami egy nagyságrendel nagyobb, mint a (2.7) képlettel megadott rekurzió. Ha a résbüntetésre speciális függvényeket adunk, akkor a futási idő csökkenthető, erre adunk egy példát az alábbiakban.

Egy résbüntetési függvényt affin résbüntetési függvénynek hívunk, ha

$$g_k=uk+v,\ u\ge 0,\ v\ge 0 \tag{2.14}$$

Itt $u+v$-t résnyitási büntetésnek, u-t pedig réskiterjesztési büntetésnek hívjuk. Ilyen résbüntetési függvény esetén létezik olyan algoritmus, amelynek a futási ideje $O(nm)$. Az algoritmus ötlete a következő átindexelésben rejlik

$$\begin{aligned} p_{i,j} &= \min\{d_{i-1,j}+g_1,\ \min_{2\le k< i}\{d_{n-k,m} +g_k\}\} \\ &= \min\{d_{i-1,j}+g_1,\ \min_{1\le k< i-1}\{d_{n-1-k,m} +g_{k+1}\}\} \\ &= \min\{d_{i-1,j}+g_1,\ \min_{1\le k< i-1}\{d_{n-1-k,m} +g_k\}+u\} \\ &= \min\{d_{i-1,j}+g_1, p_{i-1,j}+u\} \end{aligned} \tag{2.15}$$

És hasonlóan

$$q_{i,j}= \min\{d_{i,j-1}+g_1, q_{i,j-1}+u\} \tag{2.16}$$

Így $p_{i,j}$ és $q_{i,j}$ konstans idő alatt kiszámítható, ezekből pedig $d_{i,j}$ is konstans idő alatt kiszámítható, a táblázat minden elemére. Így az algoritmus futási ideje O(nm) marad, és csak egy konstans faktorral lesz lassabb, mint a (2.7) képlettel megadott dinamikus programozási algoritmus.

3. fejezet

Sztochasztikus transzformációs nyelvtanok I. Sztochasztikus reguláris nyelvtanok és rejtett Markov modellek

"*Colorless green ideas sleep furiously*" -- "*Színtelen zöld ötletek alszanak őrjöngve*". Gondolom, minden olvasó egyetért azzal, hogy ez a mondat teljesen értelmetlen. Ugyanakkor, minden olvasó meg tudja állapítani, hogy a fenti mondat nyelvtanilag helyes, annak ellenére, hogy teljesen értelmetlen. Sőt, annak ellenére meg tudja állapítani, hogy ez a mondat nyelvtanilag helyes, hogy soha nem hallotta ezt a mondatot (hacsak nem hallott már Chomsky híres alapcikkéről, amiből ez a mondat származik), sőt, nagy valószínűséggel nem hallotta még ennek a mondatnak bármely két konszekutív szavát se egymás után: színtlen zöld, zöld ötletek, ötletek alszanak, alszanak őrjöngve.

A fenti példa rávilágít, hogy egy mondat nyelvtani helyessége független a mondat értelmétől. Hogyan lehet akkor a nyelvtani helyességet definiálni, illetve azt leellenőrizni? Ezt a kérdést tette fel Noam Chomsky a múlt század ötvenes éveiben, és alkotta meg az elméleti nyelvészet matematikai alapjait. Az általa kialakított elmélet továbblépett az elméleti nyelvészet határain, és mára számos alkalmazási területe lett, többek között számítógépes fordítóprogramok készítésekor is használják. A kilencevenes évek elején jöttek rá a bioinformatikusok, hogy Chomsky elméletét lehet alkalmazni biológiai szerkezetpredikcióban is. Ugyanis ami egy biológiai szekvenciának a szerkezete, az gyakorlatilag ugyanaz, mint egy valamely nyelven írt mondatot tartalmazó szekvencia nyelvtana.

Ebben a fejezetben megismerkedünk Chomsky elméletével és annak kiterjesztésével, a sztochasztikus transzformációs nyelvtanokkal. A következő fejezetben tárgyaljuk a szubsztitúciós modelleket, és utána mutatjuk be, hogy a szubsztitúciós modellek és transzformációs nyelvtanok összekapcsolásával hogyan lehet biológiai szerkezeteket prediktálni.

3.1. A transzformációs nyelvtanok Chomsky hierarchiája

Egy transzformációs nyelvtan egy $\{T,N,S,R\}$ négyes, ahol T szimbólumok egy véges halmaza, N szimbólumok egy másik véges halmaza, $T \cap N$ az üres halmaz, $S \in N$ egy kitüntetett elem, R pedig $\alpha \to \beta$ alakban írt átírási szabályoknak egy véges halmaza, ahol $\alpha \in (T \cup N)^* \setminus T^*$, $\beta \in (T \cup N)^*$, X^* jelöli az X ABC feletti véges hosszú szavakat, beleértve az üres stringet is. T-t a terminálisok halmazának hívjuk, N-et a nemterminálisok halmazának. S a kezdő nemterminális, néhány iskola ezt axiómának is hívja.

Azt mondjuk, hogy a nyelvtan egy $A \in T^*$ szekvenciát le tudja vezetni, ha létezik A_1, A_2, ... A_n szekvenciáknak egy véges hosszú sorozata a következő tulajdonságokkal. $A_1 = S$, $A_n = A$, és minden $1 \le i \le n$-1-re, létezik olyan $\alpha \to \beta$ szabály, valamint B és C stringek, hogy $A_i = B\alpha C$ és $A_{i+1} = B\beta C$.

A nyelvtan által generált nyelv T^* azon részhalmaza, amely a nyelvtanban levezethető. A levezethetőségi probléma annak eldöntése, hogy egy adott A string egy adott $\{T,N,S,R\}$ nyelvtan által generált nyelv eleme-e. Ez a feladat általánosságban nagyon nehéz, ahogy a következő tétel ezt kimondja.

Tétel 3.1. Nem létezik olyan determinisztikus Turing gép, amely minden A string és $\{T,N,S,R\}$ nyelvtanra véges időben eldönti, hogy az A string a $\{T,N,S,R\}$ nyelvtan által generált nyelv eleme.

A bizonyítástól itt eltekintünk, de megemlítjük, hogy a bizonyítás azon alapul, hogy a levezetési probléma ekvivalens a Turing gépek megállási problémájával, és a megállási probléma eldönthetetlenségéből adódik a levezetési probléma eldönthetetlensége is.

Ha az átírási szabályokra megszorításokat teszünk, akkor az így kapott speciális osztályokra a levezetési probléma már könnyebb lesz. Chomsky a transzformációs nyelvtanok következő hierarchiáját adta meg:

- A **megszorítás nélküli (unrestricted)** nyelvtanok halmaza az összes lehetséges nyelvtant tartalmazza
- A **környezetfüggő (context dependent)** nyelvtanok halmaza azokat a nyelvtanokat tartalmazza, amelyeknek az átírási szabályai
  $$\gamma_1 W \gamma_2 \rightarrow \gamma_1 \beta \gamma_2$$
  alakú, ahol γ_1 és γ_2 eleme $(T \cup N)^*$-nek, W egy nemterminális szimbólum, β pedig szintén $(T \cup N)^*$ eleme, de nem az üres string.
- A **környezetfüggetlen (context free)** nyelvtanok halmaza azokat a nyelvtanokat tartalmazza, amelyeknek az átírási szabályai
  $$W \rightarrow \beta$$
  alakúak, ahol W egy nemterminális szimbólum, β pedig szintén $(T \cup N)^*$ eleme, de nem az üres string.
- A **reguláris** nyelvtanok halmaza azokat a nyelvtanokat tartalmazza, amelyeknek az átírási szabályai
  $$W \rightarrow aW'$$
  vagy
  $$W \rightarrow a$$
  alakúak, ahol W és W' nemterminális szimbólumok, a pedig terminális szimbólum.

Könnyen belátható, hogy a nyelvtanok a fenti lista sorrendjében egymásba ágyazott halmazokat adnak, és minden tartalmazás valódi részhalmazt jelöl. A levezetési problémák komplexitására az alábbi tételek adhatóak meg.

Tétel 3.2. Környezetfüggő nyelvtanokra a levezetési probléma eldönthető.

Bizonyítás Megadunk egy algoritmust, amely minden $\{T,N,S,R\}$ környezetfüggő nyelvtanra és A stringre eldönti, hogy A a $\{T,N,S,R\}$ környezetfüggő nyelvtan által generált nyelv része-e.

Legyen n az A string hossza. Készítsünk egy irányított gráfot, amelynek a pontjai a $(T \cup N)$ halmaz feletti legfeljebb n hosszú stringek. A gráf minden (u,v) pontpárjára döntsük el, hogy van-e olyan $\gamma_1 W \gamma_2 \rightarrow \gamma_1 \beta \gamma_2$ szabály valamint B és C stringek, hogy az u ponthoz tartalmazó string felírható $B\gamma_1 W \gamma_2 C$ alakba, a v ponthoz tartozó string pedig $B\gamma_1 \beta \gamma_2 C$ alakba. Ez véges időben eldönthető, hiszen véges sok pontpár van, véges sok szabály van és mindegyik string véges hosszú.

Az A string pontosan akkor része a nyelvtannak, ha létezik irányított út a kezdő nemterminálist, mint 1 hosszú stringet reprezentáló pont és az A stringet

reprezentáló pont között. Az, hogy ilyen út létezik, véges időben eldönthető, hiszen a gráf véges. Minden ilyen út egy levezetést reprezentál, és minden levezetés egy ilyen úton van. Valóban, az út során a köztes stringek hossza monoton növekszik, így nem lehet hosszabb az A string hosszánál. Az összes ilyen string reprezentálva van a megkonstruált gráfban, tehát minden lehetséges levezetés reprezentációját tartalmazza a gráf.

Ugyanakkor, a fent vázolt algoritmus futási ideje exponenciálisan nő az A szekvencia hosszával. Azonban ennél sokkal jobbra nem is számíthatunk, ahogy ezt a következő tétel is kimondja:

Tétel 3.3. A környezettfüggő nyelvtanok levezetési problémája NP-nehéz.

A tétel bizonyításától itt eltekintünk. Ezzel szemben a környezetfüggetlen nyelvtanok levezetési problémája már könnyű, P-beli. Ezt a következő alfejezetekben be is fogjuk látni.

3.2. Sztochasztikus reguláris nyelvtanok algoritmusai: Viterbi, Forward, Backward

Egy sztochasztikus reguláris nyelvtan egy $\{T,N,S,R,P\}$ ötös, melyből $\{T,N,S,R\}$ egy reguláris nyelvtan, P pedig R-ről a pozitív valós számok halmazába képező függvény, melyre igaz, hogy minden W nemterminálisra

$$\sum_{a\in T|\exists W'\ \ W\to aW'\in R}\ \sum_{W'\in T|W\to aW'\in R} P(W\to aW') + \sum_{a\in T|W\to a\in R} P(W\to a) = 1 \qquad (3.1)$$

P-t az átírás valószínűségének nevezzük, és valóban, minden W-re P egy eloszlást definiál azon az átírási szabályokon, amelyeknek a baloldala W. Egy levezetés valószínűségét a benne levő átírási szabályok valószínűségeinek a szorzataként definiáljuk. A reguláris nyelvtan által generált nyelv egy A szekvenciájának a teljes valószínűsége a lehetséges levezetései valószínűségeinek az összege. Ezt $P(A)$-val fogjuk jelölni. Azt az eseményt, hogy az A szekvencia a_k karakterét egy $r\in R$ szabály vezeti le, $r\sim a_k$ fogja jelölni. Egy adott sztochasztikus reguláris nyelvtanra és egy adott n hosszú A szekvenciára a következő kérdéseket akarjuk feltenni:

- Mi az A szekvencia (egyik) legvalószínűbb levezetése? Mi ennek a valószínűsége? Ezt $P_{\max}(A)$-val fogjuk jelölni.
- Mi az A szekvencia levezetésének teljes valószínűsége?
- Mi $r\sim a_k$ valószínűsége, feltéve, hogy a nyelvtan az A szekvenciát generálta? Ezt $P(r\sim a_k\,|A)$-val fogjuk jelölni.

Az első kérdésre a Viterbi algoritmussal adhatunk választ. A Viterbi algoritmus minden W nemterminálisra és k-ra megkérdezi az

$$a_1\ a_2\ \ldots\ a_k W$$

közti string legvalószínűbb levezetésének valószínűségét. Jelölje $v(W,k)$ ezt a valószínűséget. Mivel a levezetés mindig a kezdő nemterminálisból indul,

$$v(S,0)=1 \qquad (3.2)$$

és

$$v(W,0)=0 \quad \forall W \neq S \tag{3.3}$$

A további értékeket dinamikus programozási rekurzióval tudjuk megadni:

$$v(W,k) = \max_{W'|W' \to a_k W \in R} \{v(W',k-1)P(W' \to a_k W)\} \tag{3.4}$$

A rekurzió terminációja pedig

$$P_{\max}(A) = \max_{W|W \to a_n \in R} \{v(W,n-1)P(W \to a_n)\} \tag{3.5}$$

A (3.4) és (3.5) képletk helyességének a bizonyítása a 2.1. tétel bizonyításával analóg, és a bizonyítást az olvasókra hagyjuk gyakorlásként. $P_{\max}(A)$ kiszámolása után lehet egy legvalószínűbb levezetést megkonstruálni. A megkonstruálás analóg a már ismertetett dinamikus programozási algoritmusok trace-back fázisával: először megkeressük azt a $W \to a_n$ levezetési szabályt, amely a (3.5) egyenletben a maximumot adta, a prefix hosszat n-1-re állítjuk, majd amíg nem értünk el a 0 hosszú prefixig, ha a kurrens nemterminális W, megkeressük azt a W' nemterminálist, amely a (3.4) képletben a maximumot adja, erre állítjuk át a kurrens nemterminálist, és a prefixhosszat eggyel rövidítjük.

A teljes valószínűséget a Forward algoritmussal számolhatjuk ki. Jelölje $f(W,k)$ az

$$a_1 \; a_2 \; ... \; a_k W$$

közti string teljes levezetési valószínűségét. Ezeket az értékeket megint dinamikus programozással számolhatjuk ki. Az inicializálás a Viterbi algoritmushoz hasonlóan:

$$f(S,0)=1 \tag{3.6}$$

és

$$f(W,0)=0 \quad \forall W \neq S \tag{3.7}$$

a rekurzió

$$f(W,k) = \sum_{W'|W' \to a_k W \in R} v(W',k-1)P(W' \to a_k W) \tag{3.8}$$

a termináció pedig

$$P(A) = \sum_{W|W \to a_n \in R} v(W,n-1)P(W \to a_n) \tag{3.9}$$

A (3.8) és (3.9) egyenletek helyességének a bizonyítása analóg a (3.4) és (3.5) képletek helyességének a bizonyításával, és az olvasóra hagyjuk a bizonyítást.

Az utolsó kérdés, $P(r{\sim}a_k\,|A)$ kiszámolásához először a Backward algoritmust ismertetjük és azután adjuk meg, hogy hogyan kell $P(r{\sim}a_k\,|A)$-t kiszámolni. A Backward algoritmus minden W-re és A^k suffixre kiszámolja annak a valószínűségét, hogy W nemterminálisból indulva a nyelvtan az A^k suffixet vezeti le, ezt $b(W,k)$ fogja jelölni. Az A^k suffixet definíció szerint úgy kapjuk meg, hogy az A szekvenciából kivesszük az A_k prefixet, azaz a suffix az a_{k+1} karakterrel kezdődik. A dinamikus programozás visszafelé halad (innen is az algoritmus neve), az inicializálás minden W-re:

$$b(W,n-1)=\begin{cases}P(W\to a_n) & ha\ W\to a_n\in R\\ 0 & egyébként\end{cases} \tag{3.10}$$

A dinamikus programozási rekurzió:

$$b(W,k)=\sum_{W'|W\to a_{k+1}W'\in R} b(W',k+1)P(W\to a_{k+1}W') \tag{3.11}$$

ammenyiben $P(A)$-t ki akarjuk számolni a Backward rekurzióval, a termináció:

$$P(A)=b(S,0) \tag{3.12}$$

Így a Backward algoritmussal ellenőrizhető a Forward algoritmus, de ennél sokkal fontosabb, hogy $P(r{\sim}a_k\,|A)$-t ki tudjuk számolni.

Tétel 3.4. Legyen $r = W\to a_kW$'. Ekkor

$$P(r\sim a_k|A)=\frac{f(W,k-1)P(W\to a_kW')b(W',k)}{P(A)} \tag{3.13}$$

Bizonyítás Mivel a feltételes valószínűségre teljesül a

$$P(r\sim a_k\wedge A)=P(r\sim a_k|A)P(A) \tag{3.14}$$

egyenlőség, ezért csak azt kell belátni, hogy

$$P(r\sim a_k\wedge A)=f(W,k-1)P(W\to a_kW')b(W',k) \tag{3.15}$$

$P(r\sim a_k\wedge A)$ annak a valószínűsége, hogy a sztochasztikus reguláris nyelvtan az A szekvenciát vezeti le, úgy, hogy az a_k karaktert a $W\to a_kW$' szabály generálja le. Ez nem más, mint az összes olyan levezetés valószínűségének az összege, amelyben a $W\to a_kW$' szabály generálja le az a_k karaktert. Az ilyen levezetések úgy néznek ki, hogy először levezetjük az $A_{k-1}W$ közti szekvenciát, majd alkalmazzuk a $W\to a_kW$' szabályt, majd W'-ből levezetjük az A^k suffixet. $A_{k-1}W$ bármely levezetése folytatható a $W\to a_kW$' szabállyal és az A^k suffix bármilyen levezetésével W'-ből. Az

$$f(W,k-1)P(W\to a_kW')b(W',k) \tag{3.16}$$

szorzat első és utolsó szorzótényezője egy soktagú összeg, az összeadandók éppen az előbb említett levezetések valószínűségei. Így a szorzat kifejtésével kapott összeg tagjai pontosan a lehetséges utak valószínűségei.

□

$P(r\sim a_k|A)$-t poszterior valószínűségnek hívjuk. Egyrészt az Expectation Maximization algoritmusban (3.4. alfejezet) fogjuk használni, másrészt szokás a Viterbi levezetés minden egyes átírására megadni a poszterior valószínűséget. A Viterbi levezetésben a poszterior valószínűségek az adott pontban a becslés megbízhatóságát jelenti. A magas poszterior valószínűség azt jelenti, hogy az olyan utak valószínűsége, amelyek az adott pontban más átírástt használnak, nagyon kicsi. Az alacsony poszterior valószínűség azt jelenti, hogy bár az adott pontban az adott átírást használta a Viterbi levezetés, azon alternatív utak valószínűségeinek az összege, amelyekben az adott pontban más átírás történik, jelentős.

3.3. Rejtett Markov modellek

Egy Rejtett Markov modell (angolul Hidden Markov Model, HMM) egy $\{G(E,V),\Sigma,T,e\}$ négyes, ahol $G(E,V)$ egy irányított gráf, Σ karakterek véges halmaza, T egy E-ről a pozitív valós számokra képező függvény, e pedig (Σ,V)-ről a nemnegatív valós számokra képező függvény, a következő tulajdonságokkal:

- A G gráfnak van két kitüntetett pontja, a *START* és az *END*. A *START* pont befoka 0, az *END* pont kifoka szintén 0. A gráfban megengedünk hurkokat, de többszörös éleket nem. A gráf pontjait állapotoknak is fogjuk hívni.
- Minden kiindulási pontra T értékeinek az összege 1, azaz minden pont kifokain definiálunk egy eloszlást. T-t fogjuk $T(v|u)$ alakban is használni, $T(v|u) = \mathrm{T}(f)$, ahol f az (u,v) irányított él.
- A *START* és az *END* pontok kivételével minden pontra az e föggvény értékeinek az összege 1. Azaz minden pontra definiálunk a karaktereknek egy eloszlását. e-t fogjuk $e(\sigma|v)$ alakban is használni.

Egy rejtett Markov modellen egy kibocsájtási út egy random séta a *START* állapotból az *END* állapotig, a T által megadott valószínűségek alapján, úgy, hogy a séta minden egyes pontjában (a *START* és az *END* kivételével) az adott v pont kibocsájt egy random karaktert az $e(\cdot|v)$ feltételes eloszlás alapján. A kibocsájtási út valószínűsége az úton használt összes T és e valószínűség szorzata.

Tétel 3.5 Minden olyan Rejtett Markov modell, amelyben nem vezet él a *START* állapotból az *END* állapotba, ekvivalens egy sztochasztikus reguláris nyelvtannal, abban az értelemben, hogy létezik egy valószínűségtartó szürjekció a Rejtett Markov modell kibocsájtási útjairól a sztochasztikus reguláris nyelvtan levezetéseire. A valószínűségtartó szürjekción azt értjük, hogy minden levezetés valószínűsége az ősképei valószínűségeinek az összege. Továbbá, minden út képe ugyanazt a szekvenciát vezeti le, mint amit az út kibocsájt.

Bizonyítás Egy $\{G(E,V),\Sigma,T,e\}$ rejtett Markov modellhez megkonstruálunk egy $\{T,N,S,R,P\}$ sztochasztikus reguláris nyelvtant, megadjuk a bijekciót és bebizonyítjuk azt. A sztochasztikus reguláris nyelvtanban $T = \Sigma$, $N = V$, $S = START$. Mivel a transzformációs nyelvtanokban konvenció szerint a nemterminálisokat nagy

betűvel jelöljük, a gráfokban úgyszintén konvenció alapján a pontokat kisbetűvel jelöljük, a továbbiakban a gráf egy u pontját W_u fogja jelölni a nyelvtanban. Az átírási szabályok a következők:

Minden $f=(u,v)$ $v \neq END$ élre, valamint minden a karakterre amelyre $e(a|v) \neq 0$ és v-ből nem csak az END állapotba lehet ugrani, létrehozunk egy $W_u \rightarrow aW_v$ szabályt, valamint minden u pontra és a karakterre, amelyre létezik olyan $f = (u,v)$ él, hogy $e(a|v) \neq 0$ és $T(END|v) \neq 0$, létrehozunk egy $W \rightarrow a$ szabályt. A szabályok valószínűségeit a következőképpen definiáljuk:

$$P(W_u \rightarrow aW_v) = \frac{T(v|u)e(a|v)(1-T(END|v))}{1-T(END|u)} \tag{3.17}$$

$$P(W_u \rightarrow a) = \frac{\sum_v T(v|u)e(a|v)T(END|v)}{1-T(END|u)} \tag{3.18}$$

Először azt látjuk be, hogy ezek a valószínűségek tényleg egy eloszlást generálnak minden W_u nemterminálisra, azaz nemnegatívak (valójában az is igaz, hogy pozitívak) és az összegük egy. A pozitivitást onnan lehet látni, hogy sem a (3.17)-es, sem a (3.18)-as képletben sem $T(END|v)$ sem $T(END|u)$ nem 1, így minden tag pozitív, így a szorzatuk is az, valamint minden összeadandó pozitív, így az összegük is az.

Most belátjuk, hogy minden W_u -ra azon átírások valószínűségeinek az összege, amelyben W_u az átírási szabály bal oldalán áll, 1. Azaz, minden W_u-ra

$$\sum_v \sum_a P(W_u \rightarrow aW_v) + \sum_a P(W_u \rightarrow a) = 1 \tag{3.19}$$

A (3.19) egyenletbe beírva a (3.17) és (3.18) egyenleteket, és a törteket közös nevezőre hozva kapjuk, hogy

$$\frac{\sum_v \sum_a T(v|u)e(a|v)(1-T(END|v)) + \sum_v \sum_a T(v|u)e(a|v)T(END|v)}{1-T(END|u)} = 1 \tag{3.20}$$

A két szummát összevonva kapjuk, hogy

$$\frac{\sum_v \sum_a T(v|u)e(a|v)}{1-T(END|u)} = 1 \tag{3.21}$$

Mivel $e(\cdot|v)$ eloszlás minden v-re, az a-kon való összegzés 1 lesz, minden v-re. Ebből kapjuk, hogy

$$\frac{\sum_v T(v|u)}{1-T(END|u)} = 1 \tag{3.22}$$

de ez meg igaz, hiszen a szumma az összes kifokon megy, kivéve az *END* pontot, és $T(\cdot|u)$ is eloszlás minden *u*-ra. A megadott *P* függvény tehát teljesíti a (3.1) képletet, így tényleg egy sztochasztikus reguláris nyelvtant definiáltunk.

Most megadjuk a szürjekciót, és bizonyítjuk annak a valószínűségtartását. Legyen π egy kibocsájtási út, amely a $v_0 = START$, v_1, v_2, ... v_n, $v_{n+1} = END$ állapotokon megy, és az $A = a_1a_2 \ldots a_n$ szekvenciát bocsájtja ki. Ehhez az úthoz a

$$S \to a_1 W_{v_1}, W_{v_1} \to a_2 W_{w_2}, \ldots W_{v_{n-1}} \to a_n$$

levezetést rendeljük. Ez a levezetés létezik, mert definíció szerint minden átírási szabálya létezik. A leképezés szürjektív, mert minden

$$S \to a_1 W_{v_1}, W_{v_1} \to a_2 W_{w_2}, \ldots W_{v_{n-1}} \to a_n \tag{3.23}$$

levezetéshez legalább egy utat hozzá lehet rendelni, amely a $v_0 = START$, v_1, v_2, ... v_n, $v_{n+1} = END$ állapotokon megy, és az $A = a_1a_2 \ldots a_n$ szekvenciát bocsájtja ki. A (3.23) levezetés valószínűsége

$$\begin{aligned} &T(v_1|START)e(a_1|v_1) \times T(v_2|v_1)e(a_2|v_2) \times \ldots \times \\ &\times T(v_{n-1}|v_{n-2})e(a_{n-1}|v_{n-1}) \times \sum_{v_n} T(v_n|v_{n-1})e(a_n|v_n)T(END|v_n) \end{aligned} \tag{3.24}$$

mivel az $(1 - T(END|v_i))$ szorzótényezők és nevezők kiesnek. Valóban, a *START*-ból az *END* állapotba nem lehet ugorni, így az első átírási szabály valószínűségében a nevező 1, a második átírási szabálytól kezdve pedig a nevező az előző átírási szabály valószínűségében levő $(1 - T(END|v_i))$ szorzótényezővel egyezik meg. A (3.24) képlet viszont megegyezik azon kibocsájtási utak valószínűségeinek az összegével, amelyek a a $v_0 = START$, v_1, v_2, ... v_n, $v_{n+1} = END$ állapotokon mennek át, és az $A = a_1a_2 \ldots a_n$ szekvenciát bocsájtják ki. Pontosan ezek azok a kibocsájtási utak, amelyek képe az adott levezetés, így a szürjekció valóban valószínűségtartó.

□

A tétel visszafelé nem igaz, könnyű belátni, hogy van olyan sztochasztikus reguláris nyelvtan, amely nem ekvivalens semmilyen Rejtett Markov Modellel. Ennek belátásához elég annyit belátni, hogy egy *W* nemterminálishoz megadható $W \to aW'$ levezetési szabályoknak egy olyan eloszlása, amely nem dekomponálható ugrási és kibocsájtási valószínűségek szorzataként.

Rejtett Markov Modellekre is megadható a Viterbi, Forward és Backward algoritmusok a következőképpen. A Viterbi algoritmusban $v(u,k)$ jelöli annak az útnak a valószínűségét, amely az A_k prefixet bocsájtotta ki, jelenleg az *u* állapotban van, az *u* állapot bocsájtott már ki karaktert. Az algoritmus inicializációja:

$$v(START,0) = 1 \tag{3.25}$$

$$v(u,0) = 0 \quad u \neq START \tag{3.26}$$

A fő rekurzió:

$$v(u,k) = \max_w \{v(w,k-1)T(u|w)e(a_k|u)\} \tag{3.27}$$

A termináció pedig

$$P_{\max}(A) = \max_u \{v(u,n)T(END|u)\} \tag{3.28}$$

A Forward algoritmusban $f(u,k)$ jelöli azon utak valószínűségeinek az összegét, amelyek az A_k prefixet bocsájtották ki, jelenleg az u állapotban vannak, az u állapot bocsájtott már ki karaktert. Az algoritmus inicializációja:

$$f(START,0) = 1 \tag{3.29}$$

$$f(u,0) = 0 \quad u \neq START \tag{3.30}$$

A fő rekurzió:

$$f(u,k) = \sum_w v(w,k-1)T(u|w)e(a_k|u) \tag{3.31}$$

A termináció pedig

$$P(A) = \sum_u v(u,n)T(END|u) \tag{3.32}$$

A Backward algoritmusban $b(u,k)$ jelöli azon utak valószínűségeinek az összegét, amelyek az A^k suffixet bocsájtják ki az u állapotból indulva, az u állapot bocsájtott már ki karaktert. Az algoritmus inicializációja:

$$b(u,n) = T(END|u) \tag{3.33}$$

A fő rekurzió:

$$b(u,k) = \sum_w b(w,k+1)T(w|u)e(a_{k+1}|w) \tag{3.34}$$

A termináció pedig

$$P(A) = b(START,0) \tag{3.35}$$

A Forward és Backward algoritmusból származó értékekkel adhatjuk meg a poszterior valószínűségeket Rejtett Markov Modellekre. Jelölje $v \sim a_k$ azt az eseményt, hogy a_k-t, v bocsájtotta ki.

Tétel 3.6.

$$P(v \sim a_k|A) = \frac{f(v,k)b(v,k)}{P(A)} \tag{3.36}$$

Bizonyítás A bizonyítás teljesen analóg a 3.4. tétel bizonyításával.

3.4. Expectation Maximization

Az előző alfejezetekben megismerkedtünk azokkal az algoritmusokkal, amelyek a legvalószínűbb levezetéseket, illetve kibocsájtási útvonalakat tudják meghatározni, egy adott paraméterezésű sztochasztikus reguláris nyelvtan, illetve rejtett Markov model esetén. Ha a levezetési valószínűségek, illetve ugrási és kibocsájtási valószínűségek nincsenek meghatározva, ezeket megbecsülhetjük az rendelkezésre álló adatokból. Azon paramétereket szeretnénk meghatározni, amelyek maximalizálják a likelihoodot. Az Expectation Maximization metódus egy rekurzív módszer, amely valamilyen θ_0 kezdeti paraméterekből egy θ_1 becslést ad meg, ebből egy θ_2 becslést, és így tovább. A paraméterekre fennáll, hogy

$$P\left(D\middle|\theta_i\right) \leq P\left(D\middle|\theta_{i+1}\right) \quad \forall i = 0,1,... \tag{3.37}$$

azaz a likelihood monoton növekszik rekurzió során. A metódus a nevét onnan kapta, hogy bizonyos várható értékeket számolunk ki, és ezek alapján állítjuk be az új paramétereket. Először az EM definíció szerinti képletét adjuk meg, utána azt mutatjuk meg, hogy a szóban forgó várható értékeket hogyan lehet gyorsan kiszámolni, majd bebizonyítjuk, hogy a likelihood monoton növekszik az iteráció során.

Az EM algoritmusban sztochasztikus reguláris nyelvtanokra kettő, rejtett Markov modellekre három feltételes várható értéket számolunk ki. Sztochasztikus reguláris nyelvtanokra az első várható érték egy nemterminális használatának a várható értéke, feltéve, hogy a nyelvtan az adott szekvenciát generálta. Jelölje $n(W,d)$ azt, hogy a d levezetésben hányszor használtuk a W nemterminálist. Ekkor a várható érték definíció szerint:

$$E\left(W\middle|A\right) := \frac{\sum_{d|d\sim A} P(d)n(W,d)}{P(A)} \tag{3.38}$$

ahol a szumma azon d levezetésekre megy, amelyek az A szekvenciát vezetik le. A mások feltételes várható érték egy r átírási szabály használatának a várható értéke, feltéve, hogy a nyelvtan az A szekvenciát vezeti le. A fentiekhez hasonlóan jelölje $n(r,d)$ azt, hogy egy d levezetésben az r szabályt hányszor használtuk. Ekkor a várható érték definíció szerint

$$E\left(r\middle|A\right) := \frac{\sum_{d|d\sim A} P(d)n(r,d)}{P(A)} \tag{3.39}$$

Rejtett Markov modellek esetében a három feltételes várható érték egy adott állapot használatának a várható értéke, egy adott átmenet használatának a várható értéke, illetve egy adott karakternek egy adott állapot általi kibocsájtásának a várható értéke, mindegyik esetben azon feltétel mellett, hogy az A szekvenciát bocsájtja ki a modell. Hasonlóan a fentiekhez jelölje $n(W,p)$, $n(W_1 \rightarrow W_2,p)$ és $n(W\sim a,p)$ rendre W használatának a számát, W_1-ből W_2-be ugrások számát, illetve azt, hogy W hányszor bocsájtott ki egy a karaktert a p útban. A három feltételes várható érték definíció szerint:

$$E(W|A) := \frac{\sum_{p|p\sim A} P(p)n(W,p)}{P(A)} \tag{3.40}$$

$$E(W_1 \to W_2|A) := \frac{\sum_{p|p\sim A} P(p)n(W_1 \to W_2,p)}{P(A)} \tag{3.41}$$

$$E(W \sim a|A) := \frac{\sum_{p|p\sim A} P(p)n(W \sim a,p)}{P(A)} \tag{3.42}$$

Ezeket a várható értékeket komputációsan költséges definíció szerint számolni, hiszen az utak vagy levezetések száma exponenciálisan nőhet a levezetett/kibocsájtott szekvencia hosszával.A Forward és a Backward algoritmusok által kiszámolt értékek segítségével azonban polinomiális időben ki tudjuk számolni, a következő tétellel:

Tétel 3.7. A fenti várható értékek a következő képletekkel is kiszámolhatóak:

$$E(W|A) = \frac{\sum_{k=0}^{|A|} f(W,k)b(W,k)}{P(A)} \tag{3.43}$$

$$E(W_1 \to aW_2|A) = \frac{\sum_{1\le k\le|A| \| a_k=a} f(W_1,k-1)P(W_1 \to aW_2)b(W_2,k)}{P(A)} \tag{3.44}$$

$$E(W \sim a|A) = \frac{\sum_{1\le k\le|A| \| a_k=a} f(W,k)b(W,k)}{P(A)} \tag{3.45}$$

$$E(W_1 \to W_2|A) = \frac{\sum_{k=0}^{|A|-1} f(W_1,k)T(W_2|W_1)e(a_{k+1}|W_2)b(W_2,k+1)}{P(A)} \tag{3.46}$$

Bizonyítás: A bizonyítást a 3.43 képletre adjuk meg, a többi bizonyítása ezzel analóg történik. Rejtett Markov modellekre a bizonyítás a következő. Ha *W* nem a *START* állapot, akkor vegyük észre, hogy a 3.40 képletben a szummát kiszámolhatjuk úgy is, hogy minden *k* pozícióra összegezzük azon utak valószínűségét, amelyekben az a_k karaktert a *W* állapot bocsájtotta ki. Azaz

$$\sum_{p|p\sim A} P(p)n(W,p) = \sum_{k=1}^{|A|} \sum_{p|p\sim A, W\sim a_k} P(p) \tag{3.47}$$

hiszen a jobboldalon minden út valószínűsége annyiszor szerepel az összegzésben, ahányszor a *W* állapotot használtuk. De a jobboldal belső szummája nem más, mint $f(W,k)b(W,k)$, így kapjuk, hogy

$$\sum_{p|p\sim A} P(p)n(W,p) = \sum_{k=1}^{|A|} f(W,k)b(W,k) = \sum_{k=0}^{|A|} f(W,k)b(W,k) \tag{3.48}$$

A második egyenlőség azért áll fenn, mert $f(W,0)=0$ minden $W \neq START$ állapotra. Ha W a $START$ állapot, akkor $f(W,0)= b(W,0)=1$, és minden $k\neq 0$-ra $f(W,k)= f(W,k)=0$. Ebben az esetben is helyes a képlet, hiszen a $START$ állapotot használatának a várható értéke 1.

Sztochasztikus reguláris nyelvtanokra a bizonyítás hasonlóan történik, itt is az összegzés megtehető úgy is, hogy minden egyes k pozícióra összegezzük azon utak valószínűségeit, amelyekben W az a_k karaktert vezeti le. Azaz,

$$\sum_{d|d\sim A} P(d)n(W,d) = \sum_{k=1}^{|A|} \sum_{d|d\sim A, W\sim a_k} P(d) \tag{3.49}$$

A 3.49 képlet jobboldalán a belső szumma $f(W,k\text{-}1)b(W,k\text{-}1)$. Ezt beírva, és átindexelve kapjuk, hogy

$$\sum_{d|d\sim A} P(d)n(W,d) = \sum_{k=0}^{|A|-1} f(W,k)b(W,k) = \sum_{k=0}^{|A|} f(W,k)b(W,k) \tag{3.50}$$

A második egyenlőtlenség onnan adódik, hogy $f(W,|A|)= b(W, |A|)=0$ minden nemterminálisra.

□

Az Expectation Maximization a következő becslést adja a az r: $W_1 \to aW_2$ levezetési szabály valószínűségére:

$$\hat{P}(r) = \frac{E(r|A)}{E(W_1|A)} \tag{3.51}$$

A $T(W_2|W_1)$ átmeneti valószínűségre a becslés

$$\hat{T}(W_2|W_1) = \frac{E(W_1 \to W_2|A)}{E(W_1|A)} \tag{3.52}$$

Végül az $e(a|W)$ emissziós valószínűségre a becslés

$$\hat{e}(a|W) = \frac{E(W \sim a|A)}{E(W|A)} \tag{3.53}$$

Tétel 3.8. Legyen $\{T,N,S,R,P\}$ egy sztochasztikus reguláris nyelvtan, melyben a levezetési valószínűségek halmazát jelölje θ_1. Jelölje θ_2 azon levezetési valószínűségek halmazát, amelyeket a 3.51 képlettel kaphatunk meg. Ekkor

$$P(A|\theta_1) \leq P(A|\theta_2) \tag{3.54}$$

és egyenlőség csak akkor áll fenn, ha $\theta_1 = \theta_2$.

Hasonlóan, legyen $\{G(E,V),\Sigma,T,e\}$ egy rejtett Markov modell, amelyben az ugrási és kibocsájtási valószínűségek halmazát jelölje θ_1. Jelölje θ_2 azon ugrási és kibocsájtási valószínűségek halmazát, amelyeket a 3.52 és 3.52 képletekkel kaphatunk meg. Ekkor

$$P\left(A|\theta_1\right) \leq P\left(A|\theta_2\right) \tag{3.55}$$

és egyenlőség csak akkor áll fenn, ha $\theta_1 = \theta_2$.

Mielőtt ezt a tételt bizonyítanánk, be kell vezetnünk a relatív entrópiát, és arra bizonyítunk egy lemmát.

Definíció. Legyen p és π két diszkrét eloszlás a megszámlálható X halmazon. Ekkor p-nek a π-re vonatkozó relatív entrópiája vagy más néven Kullback-Leiber divergenciája

$$H\left(p\|\pi\right) = \sum_{x\in X} p(x)\log\frac{p(x)}{\pi(x)} \tag{3.56}$$

Lemma 3.9. Legyen p és π két diszkrét eloszlás a megszámlálható X halmazon. Ekkor

$$H\left(p\|\pi\right) \geq 0 \tag{3.57}$$

és egyenlőség akkor és csak akkor áll fenn, ha minden $x \in X$-re $p(x) = \pi(x)$.

Bizonyítás Elemi függvénytani ismeretkből adódik, hogy

$$\log(x) \leq x - 1 \tag{3.58}$$

és egyenlőség akkor és csak akkor áll fenn, ha x=1. A relatív entrópia -1-szeresére felírható, hogy

$$-H\left(p\|\pi\right) = \sum_{x\in X} p(x)\log\frac{\pi(x)}{p(x)} \leq \sum_{x\in X} p(x)\left(\frac{\pi(x)}{p(x)} - 1\right) = 0 \tag{3.59}$$

Ezért a relatív entrópia -1-szerese mindig kisebb vagy egyenlő, mint 0, és egyenlőség akkor és csak akkor áll fenn, ha minden x-re $\frac{\pi(x)}{p(x)} = 1$, azaz $p(x) = \pi(x)$.

□

Ezek után már belefoghatunk a 3.8 tétel bizonyításába.

Bizonyítás (3.8 tételé): A bizonyítást rejtett Markov modellekre adjuk meg, sztochasztikus reguláris nyelvtanokra a bizonyítás teljesen analóg. Mivel a logaritmus függvény szigorúan monoton növekszik (1-nál nagyobb alap esetén), ezért a likelihood maximuma ott van, ahol a logaritmusának a maximuma van. A likelihood-ot általában valamely rejtett változók menti összegzés mentén kapjuk meg -- például rejtett Markov modellek esetén a rejtett utak likelihoodjainak az összegeként --, így a likelihood logaritmusa felírható ilyen alakba:

$$\log P(x|\theta) = \log \sum_y P(x,y|\theta) \tag{3.60}$$

Jelöljük θ^t-vel az EM algoritmus t-edik iterációjában a paraméterek halmazát. Olyan θ paraméterhalmazt keresünk, amelyre a likelihood nem csökken. Kiindulva x és y együttes valószínűségére felírt feltételes valószínűség tételéből:

$$P(x,y|\theta) = P(y|x,\theta)P(x|\theta) \tag{3.61}$$

mindkét oldal logaritmusát véve és átrendezve kapjuk, hogy

$$\log P(x|\theta) = \log P(x,y|\theta) - \log P(y|x,\theta) \tag{3.62}$$

Mindkét oldalt megszorozva $P(y|x,\theta^t)$-vel, és minden y-ra összegezve kapjuk, hogy

$$\log P(x|\theta) = \sum_y P(y|x,\theta^t) \log P(x,y|\theta) - \sum_y P(y|x,\theta^t) \log P(y|x,\theta) \tag{3.63}$$

Definiáljuk $Q(\theta|\theta^t)$-t mint

$$Q(\theta|\theta^t) := \sum_y P(y|x,\theta^t) \log P(x,y|\theta) \tag{3.64}$$

Olyan θ paramétereket keresünk, amelyekre $\log(x|\theta)$ nem kisebb, mint $\log(x|\theta^t)$, azaz különbségük nem negatív. A különbség (3.63) és (3.64) segítségével felírható a következő alakban:

$$\log P(x|\theta) - \log P(x|\theta^t) = Q(\theta|\theta^t) - Q(\theta^t|\theta^t) + \sum_y P(y|x,\theta^t) \log \frac{P(y|x,\theta^t)}{P(y|x,\theta)} \tag{3.65}$$

Vegyük észre, hogy a szumma a (3.65) képlet jobb oldalán nem más, mint $P(y|x,\theta^t)$-nek a $P(y|x,\theta)$ eloszlásra vonatkozó relatív entrópiája, így nemnegatív a 3.9 lemma alapján. Ha az új paramétereket a

$$\theta^{t+1} := \arg\max_\theta Q(\theta|\theta^t) \tag{3.66}$$

egyenlettel definiáljuk, akkor a likelihood nem csökkenhet, hiszen $Q(\theta|\theta^t)$ értéke a maximumhelyén nem lehet kisebb, mint egy partikulásris helyen felvett $Q(\theta^t|\theta^t)$ érték, valamint a relatív entrópia is csak akkor 0, ha a két eloszlás minden helyen egyenlő egymással. A következőkben megmutatjuk, hogy a (3.52)-es és (3.53)-as képletekkel megadott paraméterek maximalizálják $Q(\theta|\theta^t)$-t.

Emlékeztetőül, rejtett Markov modellekben a rejtett adat a kibocsájtási utak, így a likelihood logaritmusát a

$$\log P(A|\theta) = \log \sum_p P(A,p|\theta) \tag{3.67}$$

alakba írhatjuk. Ennek megfelelően $Q(\theta|\theta^t)$ a

$$Q(\theta|\theta^t) = \sum_p P(p|A,\theta^t) \log P(A,p|\theta) \tag{3.68}$$

alakba írható. Egy út valószínűsége a benne levő ugrások és kibocsájtások valószínűségeinek a szorzata, mindegyik valószínűség annyiadik hatványon, ahányszor a kérdéses ugrás, illetve kibocsájtás szerepel az útban. Azaz,

$$P(A,p|\theta) = \prod_W \prod_a e(a|W)^{n(W\sim a,p)} \prod_{W_1} \prod_{W_2} T(W_2|W_1)^{n(W_1 \to W_2,p)} \tag{3.69}$$

(3.69)-et beírva (3.68)-ba beírva kapjuk, hogy

$$\begin{aligned} Q(\theta|\theta^t) &= \sum_p P(p|A,\theta^t) \times \\ &\times \left(\sum_W \sum_a n(W \sim a,p) \log e(a|W) + \sum_{W_1} \sum_{W_2} n(W_1 \to W_2,p) \log T(W_2|W_1) \right) \end{aligned} \tag{3.70}$$

Cseréljük fel (3.70)-ben az utakra vett szummát a kibocsájtásokra, illetve ugrásokra vett dupla szummákkal! Ekkor látjuk, hogy pont a (3.41) és (3.42) képletek jobb oldalán szereplő törtek számlálóit kapjuk, azaz

$$\begin{aligned} Q(\theta|\theta^t) &= P(A|\theta^t) \times \\ &\times \left(\sum_W \sum_a E(W \sim a|A) \log e(a|W) + \sum_{W_1} \sum_{W_2} E(W_1 \to W_2|A) \log T(W_2|W_1) \right) \end{aligned} \tag{3.71}$$

Ebben a formában már megmutatható, hogy a (3.52) és (3.53) képletekben megadott paraméterbecslések maximalizálják $Q(\theta|\theta)$-t. Jelölje ugyanis ezen partikuláris értékek melett a Q függvény értékét $Q(\hat{\theta}|\theta^t)$ Ekkor

$$\begin{aligned} Q(\hat{\theta}|\theta^t) - Q(\theta|\theta^t) &= P(A|\theta^t) \times \\ &\times \left(\sum_W \sum_a E(W \sim a|A) \log \frac{\hat{e}(a|W)}{e(a|W)} + \sum_{W_1} \sum_{W_2} E(W_1 \to W_2|A) \log \frac{\hat{T}(W_2|W_1)}{T(W_2|W_1)} \right) \end{aligned} \tag{3.72}$$

amit átírhatunk

$$Q\left(\hat{\theta}\middle|\theta^t\right)-Q\left(\theta\middle|\theta^t\right)=P\left(A\middle|\theta^t\right)\times$$
$$\times\left(\sum_W E\left(W\middle|A\right)\sum_a \frac{E\left(W\sim a\middle|A\right)}{E\left(W\middle|A\right)}\log\frac{\frac{E\left(W\sim a\middle|A\right)}{E\left(W\middle|A\right)}}{e\left(a\middle|W\right)}+\right. \tag{3.73}$$
$$\left.+\sum_{W_1} E\left(W_1\middle|A\right)\sum_{W_2}\frac{E\left(W_1\to W_2\middle|A\right)}{E\left(W_1\middle|A\right)}\log\frac{\frac{E\left(W_1\to W_2\middle|A\right)}{E\left(W_1\middle|A\right)}}{T\left(W_2\middle|W_1\right)}\right)$$

alakba. Viszont vegyük észre, hogy a belső szummák éppen az $\hat{e}\left(\cdot\middle|W\right)$ és $\hat{T}\left(\cdot\middle|W_1\right)$ feltételes eloszlásoknak rendre az $e\left(\cdot\middle|W\right)$ és $T\left(\cdot\middle|W_1\right)$ eloszlásokra vett relatív entrópiája, így nem lehetnek negatívok, és csak akkor 0-k, ha az eloszlások minden pontban egyenlőek egymással.

4. fejezet

Sztochasztikus transzformációs nyelvtanok II. Sztochasztikus környezetfüggetlen nyelvtanok

4.1. A környezetfüggetlen nyelvtanok erösebbek a reguláris nyelvtanoknál

A Chomsky-hierarchiában a környezetfüggetlen nyelvtanok a regulárisak felett helyezkednek el. Megmutatjuk, hogy a környezetfüggetlen nyelvtanok által generálható nyelvek halmaza bővebb, mint a reguláris nyelvtanok által generált nyelvek halmaza.

Tétel 4.1. Nincs olyan reguláris nyelv, amely által generált nyelv az $\underbrace{aa...a}_{k}\underbrace{bb..b}_{k}$ k=1,2,3... szekvenciákat tartalmazza és csak azokat.

Bizonyítás Indirekt bizonyítunk, tegyük fel, hogy egy reguláris nyelv a tételben megadott nyelvet generálja. Legyen a nyelvtan nemterminálisainak a száma n. Vegyük az ab, $aabb$, ... $\underbrace{aa...a}_{n+1}\underbrace{bb..b}_{n+1}$ egy-egy generálását, és tekintsük azokat a közti szekvenciákat, amelyekben az összes a-t már generálta a nyelvtan, de még egyetlen b karaktert sem generált. Mivel csak n nemterminális van, létezik olyan W nemterminális, amelyik szerepel két közti szekvenciában is, legyen ez $\underbrace{aa...a}_{i}W$ és $\underbrace{aa...a}_{j}W$. De ekkor a nyelvtan által generált nyelv tartalmazza a $\underbrace{aa...a}_{i}\underbrace{bb..b}_{j}$ és $\underbrace{aa...a}_{j}\underbrace{bb..b}_{i}$ szekvenciákat is, $i \neq j$, ami ellentmond annak, hogy a nyelvtan által generált nyelv csak a $\underbrace{aa...a}_{k}\underbrace{bb..b}_{k}$ k=1,2,3... szekvenciákat tartalmazza.

□

Ezzel szemben triviális belátni, hogy az

$$S \to ab|aSb \tag{4.1}$$

nyelvtan által generált nyelv az $\underbrace{aa...a}_{k}\underbrace{bb..b}_{k}$ k=1,2,3... szekvenciákat tartalmazza és csak azokat. A környezetfüggetlen nyelvtanok tehát gazdagabb struktúrákat tudnak leírni, mint a reguláris nyelvtanok, és valóban, a bioinformatikában az RNS-ek álcsomómentes másodlagos térszerkezeteit le lehet írni környezetfüggetlen nyelvtanokkal. Ezt az alábbiakban bemutatjuk, és részletesen a 6. fejezetben térünk vissza rá.

Definíció Egy RNS térszerkezet egy N pozitív egész szám és pozitív egészek párjaiból álló rendezetlen pároknak a véges halmaza, amelyben minden egész szám legfeljebb egyszer szerepel, és minden szám kisebb vagy egyenlő N-nél.

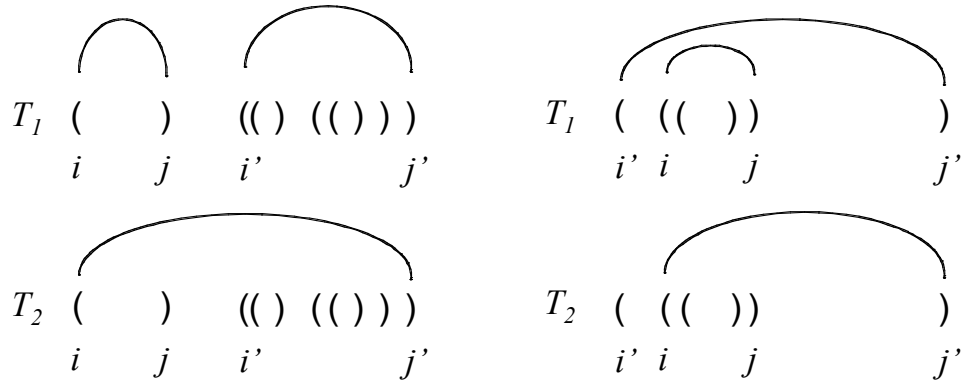
A párokat (i,j) formában fogjuk írni, és a konvenció az, hogy $i < j$. Az RNS szekvenciák a DNS-sel ellentétben egyszálú nukleinsav szekvenciák az $\{A, C, G, U\}$ ABC felett, és az RNS szekvencia visszakanyarodva önmagával képezhet nukleinsav-párokat. A térszerkezet azt írja le, hogy melyik pozícióban levő nukleinsav melyik pozícióban levő nukleinsavval párosodik.

Definíció Egy RNS térszerkezet álcsomó-mentes, ha minden (i,j) és (i',j') párjára, ha $i < i'$, akkor vagy $j < i'$ vagy $j' < j$. Azaz egy álcsomó-mentes térszerkezetben bármely két pár által kifeszített intervallumok nem lehetnek valódi metszők: vagy diszjunktak, vagy az egyik valódi része a másiknak.

Lemma 4.2. Minden álcsomómentes RNS térszerkezet egyértelműen leírható a {., (,)} feletti ABC N hosszú szekvenciájával, amelyikben minden (i, j) párra az i-edik pozícióban egy „(” áll, a j-edik pozícióban „)”, és minden olyan pozícióban, és amelyik pozíció nem vesz részt egyik párban sem, „.” áll.

Bizonyítás Nyílván minden RNS térszerkezet felírható egy ilyen stringgel, az egyértelműséget kell bizonyítani. Indirekten bizonyítunk, tegyük fel, hogy két RNS térszerkezethez tartozik ugyanaz a szekvencia. Ekkor az egyik térszerkezetben van egy (i,j) pár, a másikban pedig egy (i,j') pár, $j \neq j'$, az általánosság megsértése nélkül feltehetjük, hogy $j < j'$, és hívjuk T_1-nek azt a térszerkezetet, amely az (i,j) párt tartalmazza, és T_2-nek azt a térszerkezetet, amely az (i,j') párt tartalmazza (ld. 4.1. ábra). T_1-ben kell lennie egy (i',j') párnak, mivel a j' pozícióban is bezárójel áll, és $j < i'$ vagy $i' < i$, mivel T_1 álcsomómentes. Ha $j < i'$, akkor legyen az i'+1 pozíciótól a j'-1 pozícióig terjedő részstringben levő nyitójelek száma k. Ugyanebben a részstringben a bezárójelek száma is k kell, hogy legyen, különben T_1 nem lenne álcsomómentes. De ekkor az i' pozíciótól a j'-1 pozícióig terjedő részstringben k+1 nyitójel van, de csak k bezárójel. Így a k+1 nyitójelhez tartozó párok közül legalább egynek a záró párja j'-n felül van T_2-ben, ellentmondva annak, hogy T_2 álcsomómentes.

Ha viszont $i' < i$, akkor legyen az i+1 és j-1 pozíciók közötti nyitójelek száma k. Ugyanezen intervallumon a bezárójelek száma is k, hiszen T_1 álcsomó-mentes. De ekkor az i+1 és j pozíciók közötti zárójelek száma k+1, viszont csak k nyitójel van. Tehát T_2-ben a kérdéses k+1 zárójelhez tartozó pár közül legalább az egyiknek a nyitó párja i előtt van, megint csak ellentmondva annak, hogy T_2 álcsomó mentes.



4.1. ábra A 4.2 lemma bizonyításához.

Az álcsomómentes RNS térszerkezetek leírhatóak környezetfüggetlen nyelvtanokkal, amint azt a következő tétel is mutatja.

Tétel 4.3. A következő, ún. Knudsen-Hein nyelvtan által generált nyelv pontosan azokból az álcsomómentes RNS térszerkezetekből áll, amelyben minden (i,j) párra $j - i$ legalább 3:

$$\begin{aligned} S &\to L \,|\, LS \\ L &\to . \,|\, (F) \\ F &\to (F) \,|\, LS \end{aligned} \tag{4.2}$$

Bizonyítás Először megmutatjuk, hogy minden levezetett szekvencia az adott tulajdonságot teljesítő álcsomómentes térszerkezet, majd azt, hogy minden adott tulajdonságú álcsomómentes térszerkezet levezethető a nyelvtanban.

A levezetett szekvenciában a párokat vagy egy L vagy egy F nemterminális generálja. Minden levezetett párra igaz, hogy a további nemterminálisokat vagy közbezárják, vagy az általuk kifeszített intervallumon kívül vannak, így a generált párok nem tudnak álcsomót alkotni. Minden levezetett páron belül egy F nemterminális van, ez vagy további párokat generál vagy egy LS-t. Mivel a köztes szekvenciák hossza nem csökkenhet, így a legenerált szekvenciában minden (i,j) párra $j - i$ legalább 3.

Ezek után megmutatjuk, hogy minden adott térszerkezetet le tud vezetni a nyelvtan. Ha a térszerkezet nem tartalmaz párokat, akkor az $S \to L \,|\, LS$ és $L \to .$ szabályok ismétlésével vezethetjük le a térszerkezetet reprezentáló stringet. Ha van pár, akkor bevezetünk a párokon egy részbenrendezést, $(i,j) \prec (i',j')$, ha $i < i'$ és $j' < j$. Vegyük a maximális párokat az adott térszerkezetre, legyenek ezek (i_1,j_1), (i_2,j_2), ... (i_k,j_k), $i_1 < i_2 < ... < i_k$. Ekkor egyrészt az i_1+1-től j_1-1-ig, i_2+1-től j_2-1-ig, stb. részstringek $N \geq 2$ hosszú álcsomómentes RNS térszerkezetek, másfelől az $S \to L \,|\, LS$ valamint a $L \to . \,|\, (F)$ szabályok ismétlésével le tudjuk vezetni a

$$\underbrace{...}_{i_1-1}(F)\underbrace{.....}_{i_2-j_1-1}(F) \quad \cdots \quad (F)\underbrace{.....}_{N-j_k} \tag{4.3}$$

közti szekvenciát, ahol bármelyik pontokból álló intervallum hossza lehet 0. Így elég annyit megmutatni, hogy az F-ből elindulva le tudjuk vezetni bármelyik álcsomómentes RNS térszerkezetet, amelyre N legalább 2, és minden (i,j) párra $j - i$ legalább 3. Ehhez viszont elég annyit belátni, hogy minden közti szekvenciát le lehet vezetni F-ből, amelyik a (4.3) képletben szereplő string alakú, és legalább 2 hosszú. De ez megtehető, hiszen vagy $i_1 = 1$ és $j_1 = N$, ekkor az $F \to (F)$ szabályt alkalmazzuk, vagy pedig az $F \to LS$ szabály alkalmazása után ugyanúgy járunk el, mint fent.

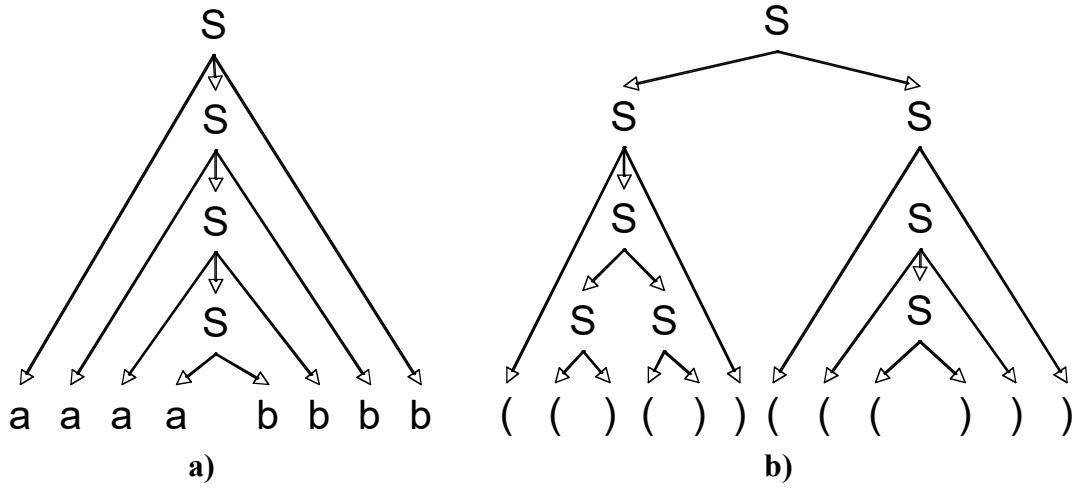
□

4.2. Sztochasztikus környezetfüggetlen nyelvtanok Chomsky normál formája

Definíció Egy sztochasztikus környezetfüggetlen nyelvtan egy $\{T,N,S,R,P\}$ ötös, melyből $\{T,N,S,R\}$ egy környezetfüggetlen nyelvtan, P pedig R-ről a pozitív valós számok halmazába képező függvény, melyre igaz, hogy minden W nemterminálisra

$$\sum_{\beta|W\to\beta\in R} P(W\to\beta)=1 \tag{4.4}$$

Egy környezetfüggetlen nyelvtan által generálható köztes szekvenciákban a nemterminálisok száma több, mint 1 lehet, azonban nem akarunk megkülönböztetni két levezetést, amely ugyanazokat a levezetési lépéseket tartalmazzák, csak más sorrendben. Ezért bevezetjük a levezetési fát (angolul parse tree), amely leírja a levezetési lépéseket. A levezetési fa egy gyökeres fa, a gyökér a kezdő nemterminálissal van felcimkézve, a levelek terminális karakterekkel, és minden belső pont fokszáma a hozzá tartozó levezetésben szereplő β string hossza, β minden egyes karakterére van egy kimenő él, a végén levő pont az adott karakterrel van felcimkézve. Levezetési fákra mutat példákat a 4.2. ábra.



4.2. ábra a) Egy lehetséges levezetés az $S\to ab|aSb$ nyelvtan. **b)** Egy lehetséges levezetés az $S\to SS|(S)|()$ nyelvtanban.

Egy levezetési fa valószínűsége a benne szereplő levezetések valószínűségének a szorzatai, egy A szekvencia levezetésének a valószínűsége az összes olyan levezetési fa valószínűségének az összege, amely A-t vezeti le. Hasonlóan a sztochasztikus reguláris nyelvtanokhoz, megkérdezhetjük egy adott szekvencia levezetésének a valószínűségét illetve feladatként kaphatjuk meg egy adott szekvencia egyik legvalószínűbb levezetési fájának a megkonstruálását. Ahhoz, hogy ezeket a feladatokat hatékonyan tudjuk megoldani, át kell írni a nyelvtant ún. Chomsky normál formába.

Definíció Egy környezetfüggetlen nyelvtan Chomsky normál formában (CNF) van, ha minden átírási szabálya

$$W \to W_1 W_2 | a \tag{4.5}$$

formába írható fel.

Tétel 4.4. Minden $\{T,N,S,R,P\}$ sztochasztikus környezetfüggetlen nyelvtan ekvivalens egy Chomsky normál formában levő $\{T,N',S,R',P'\}$ sztochasztikus környezetfüggetlen nyelvtannal abban az értelemben, hogy létezik egy valószínűségtartó szürjekció $\{T,N,S,R,P\}$ lehetséges levezetési fáiról $\{T,N',S,R',P'\}$ lehetséges levezetési fára. A $\{T,N,S,R,P\}$ nyelvtan minden PT levezetési fájának a képe ugyanazt a szekvenciát vezeti le, mint PT. Továbbá, a $\{T,N',S,R',P'\}$ nyelvtanban a nemterminálisok száma $O(|T|+||R||)$, ahol $||R||$ jelöli a levezetési szabályok összhosszát.

Bizonyítás A $\{T,N,S,R,P\}$ nyelvet több lépésben alakítjuk át Chomsky normál formába, és minden lépésben bizonyítjuk a valószínűségtartó szürjekciót. A lépések három fázisra oszthatóak.

Az első fázis az átírásokban szereplő β stringek hosszainak a redukciója. Amíg van olyan $W \to \beta$ szabály, amelyben β hossza nagyobb, mint 2, vezessünk be két új nemterminálist, W_1-et és W_2-t, távolítsuk el a $W \to \beta$ szabályt az átírási szabályok halmazából, és vezessük be a következő szabályokat a következő valószínűségekkel:

$$P(W \to W_1 W_2) = P(W \to \beta)$$

$$P\left(W_1 \to \beta_{\left\lfloor \frac{|\beta|}{2} \right\rfloor}\right) = 1$$

$$P\left(W_2 \to \beta^{\left\lfloor \frac{|\beta|}{2} \right\rfloor}\right) = 1$$

Minen olyan levezetési fa képe, amelyben szerepel a $W \to \beta$ szabály legyen az a fa, amelyben a $W \to \beta$ szabály le van cserélve a $W \to W_1 W_2$ szabályra és utána a $W_1 \to \beta_{\left\lfloor \frac{|\beta|}{2} \right\rfloor}$ és $W_1 \to \beta^{\left\lceil \frac{|\beta|}{2} \right\rceil}$ szabályokra. A kép megtartja a valószínűséget, és a leképezés szürjekció lesz, hiszen W_1 és W_2 nem írható át más szabályok alapján, csak a fent megadott módon, így a módosított nyelvtan összes levezetési fája előáll képként.

Az első fázis végére csak olyan levezetési szabályok maradnak, amelyben β hossza vagy 1 vagy 2. Ezek közül a $W \to W_1 W_2$ és $W \to a$ alakú szabályok Chomky normálformában vannak. A második fázisban a $W \to W_1 a$ alakú szabályokat lecseréléséhez új W_2 nemterminálist vezetünk be, és az átírási szabályt lecseréljük a következő szabályokra a következő valószínűségekkel:

$$P(W \to W_1 W_2) = P(W \to W_1 a)$$
$$P(W_2 \to a) = 1$$

Hasonlóan az előbbi szabálycserékhez, itt is a levezetési fák leképezései valószínűségtartó szürjekciók lesznek. A $W \to aW_2$, illetve a $W \to a_1 a_2$ átírási szabályokkal hasonlóan járunk el.

Végül a harmadik fázisban a $W \to W_1$ alakú átírási szabályok esetén minden $W_1 \to \beta$ szabályra létrehozunk egy $W \to \beta$ szabályt $P(W \to W_1)$ x $P(W_1 \to \beta)$ valószínűséggel, ha még nem volt ilyen, illetve a $W \to \beta$ szabály valószínűségéhez hozzáadjuk a $P(W \to W_1)$ x $P(W_1 \to \beta)$ valószínűséget, amennyiben ilyen szabály már volt. Ezek után a $W \to W_1$ és $W_1 \to \beta$ szabályokat elmináljuk. Minden levezetési fában, amelyben szerepelt a $W \to W_1$ szabály lecseréljük a megfelelő $W \to \beta$ szabályra. Amennyiben keletkezik $W \to W$ alakú átírási szabály (valamely β egyenlő W-vel), akkor ezt a szabályt kivesszük, és a többi szabály valószínűségét megszorozzuk $1/(1-P(W \to W))$-vel.

Amennyiben a nyelvtan módosítása előtt volt már $W \to \beta$ szabály, több levezetési fának lesz ugyanaz a képe, de továbbra is valószínűségtartó lesz a szürjekció.

Végül belátjuk, hogy véges sok lépésben eljutunk Chomsky normál formába, és az új nemterminálisok száma tényleg $O(\|R\|)$. Az első fázisban minden lépésben valamely β-t megfeleztünk, így $O(|\beta|)$ új nemterminális bevezetésével eljuthatunk oda, hogy az új β-k hossza legfeljebb 2. A második fázisban minden szabályt 1 lépésben írunk át, a bevezetett új nemterminálisok száma maximum 2 minden lépésben, és 1 eredeti β-ból létrehozott új β-k száma $O(|\beta|)$. A harmadik fázisban nem hozunk már létre új nemterminálist, így csak azt kell belátni, hogy véges sok lépésben eljutunk a Chomsky normál formáig. De ez igaz, hiszen minden egyes lépésben eggyel kevesebb a $W \to W_1$ alakú átírási szabályok száma.

□

4.3. Sztochasztikus környezetfüggetlen nyelvtanok algoritmusai: CYK, Inside, Outside

A CYK algoritmus egy dinamikus programozási rekurzió, amely egy adott, Chomsky normál formában levő $\{T,N,S,R,P\}$ környezetfüggetlen nyelvtanra és A szekvenciára megadja A egyik legvalószínűbb levezetési fáját. Jelölje $c(W,i,j)$ a legvalószínűbb rész-levezetési fának a valószínűségét, amelyik a W nemterminálisból vezeti le az i-től j-edik pozícióig terjedő részstringet. Az inicializáció:

$$c(W,i,i) = P(W \to a_i) \tag{4.6}$$

A rekurzió:

$$c(W,i,j) = \max_{W_1} \max_{W_2} \max_{i \le k < j} \left\{ P(W \to W_1 W_2) c(W_1,i,k) c(W_2,k+1,j) \right\} \tag{4.7}$$

Végül a termináció

$$P_{\max}(A) = c(S,1,n) \tag{4.8}$$

A trace-back itt nem szekvenciális, hiszen nem egy utat keresünk, hanem egy fát, de ez rekurzív függvényekkel könnyen implementálható.

Az Inside algoritmus a teljes valószínűséget adja meg. Jelölje $I(W,i,j)$ a lehetséges rész-levezetési fák valószínűségeinek az összegét, amelyek a W nemterminálisból vezetik le az i-től j-edik pozícióig terjedő részstringet. Az inicializáció:

$$I(W,i,i) = P(W \to a_i) \tag{4.9}$$

A fő rekurzió

$$I(W,i,j) = \sum_{W_1}\sum_{W_2}\sum_{i \le k < j} P(W \to W_1 W_2) I(W_1,i,k) I(W_2,k+1,j) \tag{4.10}$$

A termináció:

$$P(A) = I(S,1,n) \tag{4.11}$$

Az Outside algoritmus kívülről befelé haladva számolja ki a teljes levezetési valószínűséget. Jelölje $O(W,i,j)$ az $a_1 a_2 ... a_{i-1} W a_{j+1} ... a_n$ közti string összes lehetséges levezetési fája falószínűségeinek az összegét. Az inicializáció

$$O(S,1,n) = 1 \tag{4.12}$$

$$O(W,1,n) = 0 \qquad \forall W \neq S \tag{4.13}$$

A fő rekurzió

$$\begin{aligned} O(W,i,j) = &\sum_{W_1}\sum_{W_2}\sum_{1 \le k < i} P(W_1 \to W_2 W) I(W_2,k,i-1) O(W_1,k,j) + \\ &+ \sum_{W_1}\sum_{W_2}\sum_{j < k \le n} P(W_1 \to W W_2) I(W_2,j+1k) O(W_1,i,k) \end{aligned} \tag{4.14}$$

Mint látható, az Outside algoritmus nem annyira egyszerűen viszonyul az Inside algoritmushoz, mint a Backward a Forwardhoz. Ennek az oka az, hogy egy belső pont, mint gyökér részfájánn kívüli rész a szülő, mint gyökér részfáján kívüli rész és a testvér részfája, és a testvér lehet bal, illetve jobb oldalon is. Ezért van a fő rekurzióban két összetett szumma, és azért van szükség az Outside algoritmus során az Inside értékek felhasználására. Így az Outside algoritmushoz mindeig le kell futtatni az Inside algoritmust, hogy az Inside értékek kiolvashatóak legyenek, amikor szükség van rájuk. (Ezzel szemben a Backward algoritmust le lehet futtatni a Forward algoritmus nélkül.)

Könnyen belátható, hogy mindhárom algoritmus futási ideje $O(|T|^3 n^3)$, míg a szükséges memória $O(|T|n^2)$.

4.4. Posterior decoding és Expectation Maximization környezetfüggetlen nyelvtanokra

Hasonlóan a sztochasztikus reguláris nyelvtanokhoz, sztochasztikus környezetfüggetlen nyelvtanokra is megadhatjuk a posterior decodingot, illetve csinálhatunk Expectation Maximizationt is. Az alábbi tételeket nem bizonyítjuk, mivel bizonyításuk teljesen analóg a sztochasztikus reguláris nyelvtanoknál tárgyalt megfelelő tételek bizonyításával.

Tétel 4.5. (Posterior decoding) Jelölje $W\sim(i,j)$ azt az eseményt, hogy a W nemterminális az i-edik pozíciótól a j-edik pozícióig terjedő részstringet vezeti le.

$$P\big(W \sim (i,j)\big|A\big) = \frac{I(W,i,j)O(W,i,j)}{P(A)} \tag{4.15}$$

Tétel 4.6. (Expectation Maximization) A következő paraméter update-ek során a likelihood nem csökken

$$\hat{P}\big(W \to a\big) := \frac{\sum_{i|a_i=a} P\big(W \to a\big)O(W,i,i)}{\sum_{i\le j} I(W,i,j)O(W,i,j)} \tag{4.16}$$

$$\hat{P}\big(W \to W_1W_2\big) := \frac{\sum_{i\le k<j} O(W,i,j)P\big(W \to W_1W_2\big)I(W_1,i,k)I\big(W_2,k+1,j\big)}{\sum_{i\le j} I(W,i,j)O(W,i,j)} \tag{4.17}$$

5. fejezet

Folytonos idejű Markov modellek

5.1. Szubsztitúciók modellezése

A szubsztitúciók modellezésére leggyakrabban a folytonos idejű Markov modelleket szokták használni. A Markov modell feltételezi, hogy egy *a* karakter *b*-re való cseréjének a valószínűsége csak *a*-tól és *b*-től függ, és attól nem, hogy az adott pozícióban az *a* karakter előtt milyen karakter volt. Ez a valóságban persze nem így van, hiszen például ha kis valószínűséggel egy aminosav radikálisan más tulajdonságú aminosavra cserélődik le (pl. egy kisméretű apoláros aminosav nagyméretű polárosra), akkor az nagyobb eséllyel cserélődik vissza valamilyen kisméretű apoláros aminosavra, mint egy másik nagyméretű polárosra. Azonban, mint már az előző fejezetekben is láttuk, a Markov tulajdonság esetén a modellekkel való számolás kevésbé számolásigényes, és ez mindig döntő jelentőségű a bioinformatikai modellekben.

Mivel a biológiai szekvenciák esetében véges sok karakter van (nukleinsavak esetében 4, aminosavak esetében 20), a Markov modell véges állapotú. Egy véges állapotú Markov modellt egy lineáris differenciálegyenlet-rendszerrel lehet megadni, mátrix-vektor formában

$$\frac{d\mathbf{x}}{dt} = \mathbf{Q}\mathbf{x} \tag{5.1}$$

ahol **Q** a modellt leíró rátamátrix, és **x** tartalmazza az egyes állapotok valószínűségeit. **Q**-ról ki kell kötni, hogy a főátló kivételével minden eleme nem negatív legyen, és minden egyes oszlop összegének 0-nak kell lennie (így a főátlóban nem pozitív számok állnak). Ez a feltétele annak, hogy a valószínűségek összege minden egyes időpontban 1 legyen. Kezdeti érték feltételnek azt a vektort választva, amelynek az *i*-edik sora 1 és az összes többi eleme 0, a differenciálegyenlet-rendszer megoldása megadja annak a valószínűségét, hogy az *i*-edik állapotból kiindulva *t* idő elteltével melyik állapotnak mekkora a valószínűsége. A differenciálegyenlet-rendszer általános megoldása

$$\mathbf{x}(t) = e^{\mathbf{Q}t}\mathbf{x}(0) = \sum_{n=0}^{\infty} \frac{(\mathbf{Q}t)^n}{n!}\mathbf{x}(0) \tag{5.2}$$

A megoldás helyességét itt csak reverzibilis modellekre bizonyítjuk.

Definíció Egy folytonos idejű Markov model reverzibilis, ha létezik egy π eloszlásfüggvény, hogy minden *i*-re és *j*-re teljesül az ún. részletes egyensúly (detailed balance):

$$\pi(i)q_{j,i} = \pi(j)q_{i,j} \tag{5.3}$$

Lemma 5.1. Reverzibilis Markov modellek rátamátrixa diagonalizálható.

Bizonyítás Jelölje $\mathbf{\Pi}^{1/2}$ azt a diagonális mátrixot, amely az *i*-edik pozíciójában $\sqrt{\pi(i)}$ áll. Ekkor

$$\mathbf{Q} = \mathbf{\Pi}^{1/2}\left(\mathbf{\Pi}^{-1/2}\mathbf{Q}\mathbf{\Pi}^{1/2}\right)\mathbf{\Pi}^{-1/2} \tag{5.4}$$

Ekkor $\mathbf{\Pi}^{-1/2}\mathbf{Q}\mathbf{\Pi}^{1/2}$ valós szimmetrikus mátrix, így diagonalizálható. De ekkor **Q** is, hiszen

$$\mathbf{Q} = \mathbf{\Pi}^{1/2}\left(\mathbf{W}\Lambda\mathbf{W}^{-1}\right)\mathbf{\Pi}^{-1/2} = \left(\mathbf{\Pi}^{1/2}\mathbf{W}\right)\Lambda\left(\mathbf{W}^{-1}\mathbf{\Pi}^{-1/2}\right) = \mathbf{V}\Lambda\mathbf{V}^{-1} \tag{5.5}$$

$\mathbf{V}=\mathbf{\Pi}^{-1/2}\mathbf{W}$ -re, ahol $\mathbf{\Lambda}$ diagonális mátrix, amely **Q** sajátértékeit tartalmazza.

Tétel 5.2. Az 5.1 egyenletben megadott differenciálegyenletrendszer megoldásait reverzibilis Markov modellekre az 5.2 képlet segítségével kaphatjuk meg.

Bizonyítás Behelyettesítve a diagonalizált formát a mátrix exponenciálisba, kapjuk, hogy

$$e^{\mathbf{Q}t} = \mathbf{V}e^{\Lambda t}\mathbf{V}^{-1} \tag{5.6}$$

De ekkor a megoldás

$$\mathbf{x}(t) = e^{\mathbf{Q}t}\mathbf{x}(0) = \mathbf{V}e^{\Lambda t}\mathbf{V}^{-1}\mathbf{x}(0) \tag{5.7}$$

alakba írható. Ez azt jelenti, hogy új változókat vezetünk be úgy, hogy az $\mathbf{x}(0)$ kezdeti értékeket a $\mathbf{V}^{-1}$ mátrix segítségével a **Q** mátrix sajátbázisába transzformáljuk. Az új változókra felírt differenciálegyenlet-rendszer szétesik független közönséges differenciálegyenletekké, amelyek

$$\frac{dv_i}{dt} = \lambda_i v_i \tag{5.8}$$

alakúak. Ezen differenciálegyenletek megoldásait a **V** mátrix segítségével transzformálhatuk vissza az eredeti változókba.

□

Reverzibilis Markov modellekre a differenciálegyenlet-rendszer megoldásait általában az (5.7) egyenlet segítségével szokás meghatározni, mivel

$$e^{\Lambda t} = \begin{pmatrix} e^{\lambda_1 t} & 0 \cdots & 0 \\ 0 & e^{\lambda_2 t} & \\ \vdots & \ddots & \\ 0 & & e^{\lambda_k t} \end{pmatrix} \tag{5.9}$$

ahol λ_1, λ_2, ... λ_k a **Q** mátrix sajátértékei.

Mivel a **Q** mátrix sorvektorainak az összege a **0** vektor, **Q** rangja nem lehet *k*, így van 0 sajátértéke. Pozitív sajátértéke nem lehet, mert ez ellentmondana a $\sum_{i=1}^{k} x_i(t) = 1$ feltételnek. Ha **Q**-nak csak egyetlen 0 sajátértéke van, akkor létezik egyetlen egyensúlyi állapot, amely globálisan stabilis, azaz bármely $\mathbf{x}(0) \geq \mathbf{0}$-ra, amely teljesíti a $\sum_{i=1}^{k} x_i(0) = 1$ feltételt, a folyamat ebbe az állapotba konvergál. Az *i*-edik állapot egyensúlyi valószínűségét π_i jelöli. Ha a folyamat reverzibilis, akkor az egyensúlyi eloszlás az a π eloszlás, amelyre a részletes egyensúly teljesül.

Általában nem szokták a **Q** mátrix minden egyes paraméterét maximum likelihood módszerrel megbecsülni. Helyette a rátamátrixot $\mathbf{Q}=\mathbf{Q}_0 s$ alakban írják fel, ahol $\mathbf{Q}_0$-ra teljesül, hogy

$$\sum_{i=1}^{k} \pi_i q_{j,i} = 1 \tag{5.10}$$

azaz egyensúlyi állapotban egységnyi idő alatt átlagosan egy mutáció történik Ezután *st*-re adnak Maximum Likelihood becslést. Általános jelenség, hogy az egyes folyamatok rátáját és az evolúciós időt nem lehet külön-külön becsülni, csupán csak a szorzatukat. Ha az evolúciós rátákat a felére csökkentjük, az eltelt időt pedig a kétszeresére növeljük, az immár kétszeres idő eltelte után az egyes állapotok valószínűsége ugyanakkora, mint az eredeti esetben. Úgy is lehetne szemléltetni ezt a jelenséget, hogy egy filmnek az utolsó képkockája nem változik meg attól, hogy felére lassítva játsszuk le, csupán kétszer annyi idő alatt érünk a film végére.

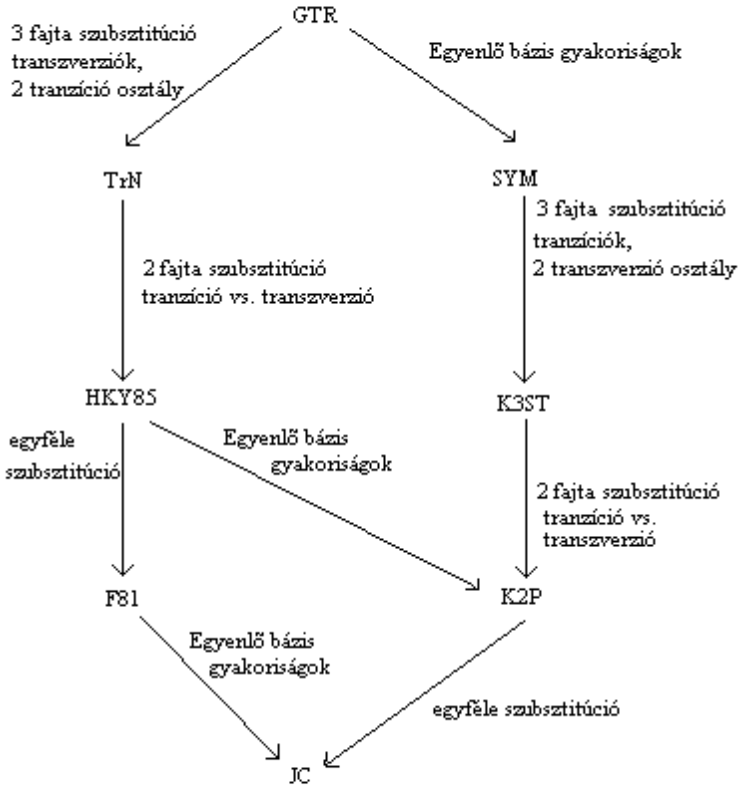
$\mathbf{Q}_0$ mátrix választására sokféle lehetőség van. Az általános **Q** mátrix DNS adatok esetében

$$\begin{pmatrix} x & \mu a \pi_A & \mu b \pi_A & \mu c \pi_A \\ \mu a \pi_C & y & \mu d \pi_C & \mu e \pi_C \\ \mu b \pi_G & \mu d \pi_G & z & \mu f \pi_G \\ \mu c \pi_T & \mu e \pi_T & \mu f \pi_T & q \end{pmatrix} \tag{5.11}$$

ahol *x*, *y*, *z* és *q* olyan értékeket vesz fel, hogy minden oszlop összege 0 legyen. Ekkor az egyensúlyi gyakoriságok valóban π_A, π_C, π_G és π_T. Például a mátrix első során ellenőrizve, $x = -\mu(a\pi_C + b\pi_G + c\pi_T)$ és valóban

$$\begin{aligned} &\left\langle \left(x, \mu a \pi_A, \mu b \pi_A, \mu c \pi_A, \right) \middle| \left(\pi_A, \pi_C, \pi_G, \pi_T, \right) \right\rangle = \\ &= -\mu(a\pi_C + b\pi_G + c\pi_T)\pi_A + \mu a \pi_A \pi_C + \mu b \pi_A \pi_G + \mu c \pi_A \pi_T = 0 \end{aligned} \tag{5.12}$$

Szintén lehet ellenőrizni, hogy a modell reverzibilis is, a transzponált helyeken álló értékek az egyensúlyi gyakoriságokkal szorozva valóban egyenlők.



5.1 ábra Nukleotid szubsztitúciós modellek osztályozása. Az általános reverzibilis modellből (GTR, General Time Reversible) egyre több megszorítással jutunk el a Jukes-Cantor modellig (JC). További rövidítések: TrN: Tamura & Nei, HKY85: Hasegawa-Kishino-Yano, 1985, F81: (Felsenstein, 1981), SYM: Zharkikh, 1994, K3ST: Kimura 3 paraméteres modellje, K2P: Kimura 2 paraméteres modellje.

Az általános modellre egyre több megkötést téve fokozatosan eljuthatunk a Jukes-Cantor modellhez, (**5.1** ábra). Ha egyenlő egyensúlyi gyakoriságokat tételezünk fel, a SYM modellt kapjuk. Erre tovább feltételezve, hogy a tranzíciók rátája, valamint az A↔T és C↔G, illetve az A↔C és G↔T transzverziók rátája megegyezik, Kimura három paraméteres modelljét kapjuk. Ha minden transzverzió rátáját azonosnak tekintjük, Kimura két paraméteres modelljéhez jutunk, végül a tranziciók rátáját a transzverziók rátájával egyenlővé téve eljutunk a Jukes-Cantor modellhez.

Az általános modellből a Tamura-Nei modellhez jutunk, ha feltesszük, hogy minden transzverzió rátája azonos, $a=c=d=f$. A Hasegawa-Kishino-Yano modellben az összes tranzíció rátája is azonos. Felsenstein 1981-es modelljében minden szubsztitúciós ráta azonos típusú, de az egyensúlyi gyakoriságok különbözőek.

Aminosav szekvenciák esetében a Jukes-Cantor modellhez analóg modell a Poisson modell, Felsenstein modelljének az analógja pedig a Hasegawa-Fujiwara modell. Bonyolultabb modellekben nem lehetséges a szubsztitúciók típusainak olyan elkülönítése, mint a nukleotidok esetében a tranzíciók és a transzverziók megkülönböztetése. Mégis, empirikus adatok azt mutatják, hogy fizikai-kémiailag

hasonló aminosavak szubsztitúciója sokkal gyakoribb, mint különböző aminosavak szubsztitúciója (pl.: egy hidrofób aminosav cseréje hidrofilre) (Dayhoff et al., 1978). A Dayhoff féle mátrixokat sok kutató felhasználja szubsztitúciós modellek készítésére. Kevéssé rokon fehérjék esetében a Dayhoff mátrixokat további empirikus adatok segítségével módosítják, emlythető például a Jones-Thorne-Thorton (JTT) mátrix.

A Dayhoff féle mátrixok meghatározása a következő approximáció segítségével történik. Olyan szekvenciákat evsznek, amelyekben a szubsztitúciók mennyisége csak 1% körül van. A tapasztalat szerint a beszúrások és törlések rátája mindig kisebb, mint a szubsztitúcióké, így ezekben a szekvenciákban a beszúrások-törlések helye egyértelműen meghatározható. Az idő mennyiségét 1 egységnyire állítják, ez az ún. 1 PAM egység (PAM = APM = Accepted Point Mutation). Az észlelt frekvenciákból kiindulva felírható egy **W** mátrix, ahol $\mathbf{w}_{i,j}$ mondja meg, hogy mekkora eséllyel látjuk a j indexű aminosavat 1 PAM idő elteltével, feltéve, hogy a t=0 időpontban az i indexű aminosavat láttuk. Ekkor

$$\mathbf{W} = e^{\mathbf{Q}t} = \sum_{i=0}^{\infty} \frac{(\mathbf{Q}t)^i}{i!} \approx \mathbf{I} + \mathbf{Q}t \tag{5.13}$$

és mivel t=1,

$$\mathbf{Q} \approx \mathbf{W} - \mathbf{I} \tag{5.14}$$

Mivel a Markov tulajdonság a gyakorlatban nem teljesül, hatékonyan számolni viszont csak a Markov modellekkel tudunk, egy ötlet a modellek megjavítására, hogy viszonylag távoli szekvenciák illesztéseit nézik. Ekkor azonban nem lehet az (5.13) képletben használt közelítést alkalmazni. Viszont ha a **W** mátrix diagonalizálható, és az összes sajátértéke pozitív, akkor lehet venni a logaritmusát, azaz azt a mátrixot, amelynek az exponenciálisa önmaga. Ezt a mátrixot lehet rátamátrixként használni, és a gyakorlat azt mutatja, hogy az ilyen módon meghatározott rátamátrixok jobban használhatóak például valószínűségi modelleken alapuló szekvenciaillesztésre, mint az (5.13) képletben használt becsléssel előállított rátamátrixok.

5.2. Felsenstein algoritmusa

A szubsztitúciós modell kiterjesztése kettőről több szekvenciára a következőképen tehető meg. A bemenő adat DNS szekvenciák többszörös illesztése. Csak azokat a pozíciókat tekintjük, ahol az illesztésben nincs beszúrás/törlés. Feltesszük, hogy az egyes pozíciók egymástól függetlenül evolválódtak, így egy evolúciós folyamat valószínűsége az egyes pontokon történt események valószínűségeinek a szorzata. Legyen adva egy gyökeres bináris fa, amely reprezentálja a szekvenciák leszármazási sorrendjét. Feltesszük azt is, hogy a fa egyes élein a karakterek szubsztitúciós folyamata független a többi élen történő folyamatoktól, és minden élre egy adott szubsztitúciós modell szerint megy. Általában feltesszük, hogy minden élen ugyanaz a szubsztitúciós modell van. A fa gyökerében a folyamat minden pozícióra az egyensúlyi állapotból indul.

Elég azt megmutatni, hogy hogyan kell egy pozícióra a karakterek észleléseinek a valószínűségeit kiszámolni, a teljes szekvenciák valószínűségeinek az észlelése az egyes pozíciókra vett észlelési valószínűségek szorzata. Egy pozícióra az

észlelés valószínűségét dinamikus programozással lehet megkapni, ezt hívják Felsenstein algoritmusának, az első publikálóról elnevezve. Jelölje $f(c,v)$ annak a valószínűségét, hogy a v pont alatti részfa levelein az adott karaktereket észleljük, feltéve, hogy a folyamat a c karakterből indult a v pontban. Az algoritmus inicializálása az a karaktert tartalmazó v levélre

$$f(c,v)=\begin{cases}1 & ha\ c=a\\ 0 & egyébként\end{cases} \tag{5.15}$$

A fő rekurzió egy v belső pontra, amelynek a két gyereke u_1 és u_2, valamint az ide futó éleken az eltelt idő rendre t_1 és t_2:

$$f(c,v)=\left(\sum_{c_1}P\left(c_1|c,t_1\right)f\left(c_1,u_1\right)\right)\times\left(\sum_{c_2}P\left(c_2|c,t_2\right)f\left(c_2,u_2\right)\right) \tag{5.16}$$

ahol $P\left(c_i|c,t_i\right)$ jelöli annak a valószínűségét, hogy a c karakterből indulva a Markov folyamat t_i idő elteltével c_i karakterben van. A termináció a teljes valószínűségre:

$$P=\sum_c f(c,root)\pi(c) \tag{5.17}$$

Ha a szubsztitúciót leíró modell reverzibilis, akkor teljesül az ún. emelő elv (Pulley Principle). Ez azt mondja ki, hogy ha a fa gyökeréből kiinduló két él közül az egyik hosszát lecsökkentjük, a másikat pedig megnöveljük, akkor a fa likelihood-ja nem változik meg. Valóban, bármely c_1-re és c_2-re, ha alkalmazzuk a reverzibilitást valamint a teljes valószínűség tételét

$$\begin{aligned}&\sum_c \pi(c)P\left(c_1|c,t_1\right)P\left(c_2|c,t_2\right)=\\ &=\sum_c \pi(c_1)P\left(c|c_1,t_1\right)P\left(c_2|c,t_2\right)=P\left(c_2|c_1,t_1+t_2\right)\end{aligned} \tag{5.18}$$

Az emelő elv folyománya, hogy egy adott fára az észlelési valószínűség független attól, hogy hol gyökereztetjük le, és egy legyökereztetett fán az észlelési valőszínűség megegyezik a gyökerezetlen fán vett észlelési valőszínűséggel, ahol a folyamatot bármelyik belső pontból indíthatjuk az egyensúlyi eloszlásból kiindulva.

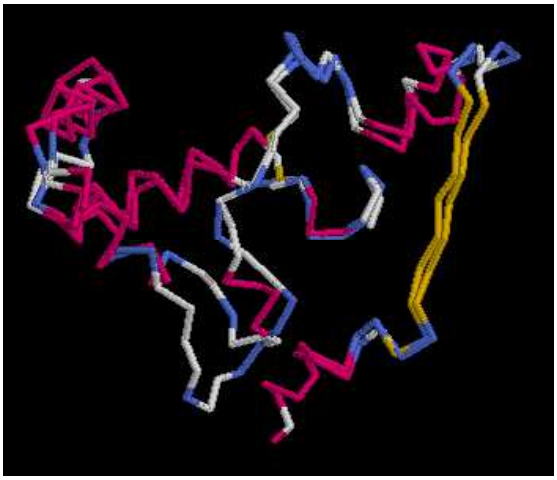
Egy adott topológia esetén a maximum likelihood élhosszakat egyszerű numerikus optimalizálás segítségével meg lehet határozni, de továbbra is fennáll az a probléma, hogy a lehetséges topológiák száma nagyon gyorsan nő a szekvenciák számával. Nevezetesen, a gyökerezetlen topológiák száma $(2n-5)!!$ Ez már tíz faj esetén is több mint 2 millió lehetőség. 2006-ban bebizonyították, hogy a Maximum Likelihood fa topológia és élhosszak megkeresése NP-nehéz probléma.

6. fejezet

Összehasonlító bioinformatika

6.1. Az összehasonlító bioinformatika alapelve

Az összehasonlító bioinformatika alapelve az, hogy *a struktúra mindig konzervatívabb, mint a szekvencia*. Ez alatt azt értjük, hogy a szekvenciák meg tudnak változni (a már tárgyalt szubsztitúciók, beszúrások és törlések révén) úgy, hogy a szekvenciális hasonlóságuk jelentősen csökken, miközben a struktúrális (és funkcionális) hasonlóságuk gyakorlatilag változatlan marad. Az alábbi példában két fehérje háromdimenziós térbeli szerkezetének egy részeletét mutatjuk be. A két fehérje közül az egyik egy élesztőgombában (*Candida clyndracea*), a másik a szarvasmarhában (*Bos taurus*) található meg. Bár a szekvenciális hasonlóság nagyon kicsi, a térbeli hasonlóság szemmel látható. Mindkét fehérje zsírsavak lebontásában vesz részt.



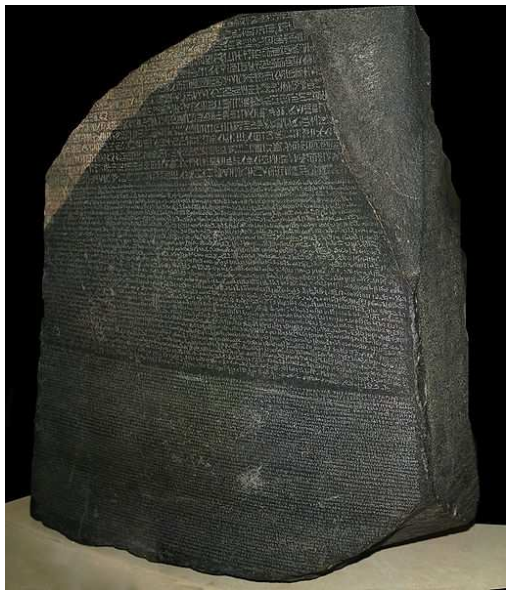
```
SGAM-VPSdpVdgtygneIYdlFVssAgÇ-gsas--dkla
SGVGlCpwAigq--dplfwAkrIAekvgÇpvd-dtskMAg

çLrsassdtLldATnnTP--GFlaysSLr-LSYLPRPdgk
çLkitdpraLtlaYkLplgste--ypKlhyLSFVPVidgd

NItddMYkLVrdgkyAsVpVIIGDQnDEGTvFGlsSlnVt
FIpddPvnlYa--nAAdvdYIAGTNdmDGHlFVGmdvpaI
```

6.1. ábra. Egy élesztőben (*Candida clyndracea*) és a szarvasmarhában (*Bos taurus*) található ún. alfa-beta hidroláz enzim egy részlete. Bal oldalt a fehérjék térbeli szerkezete látható, a két térszerkezet úgy van forgatva, hogy az aminosavak központi szénatomjai közötti távolságnégyzetek összege minimális legyen. Jobboldalt a szekvenciák ún. struktúrális illesztése látható. Az alacsony szekvenciális hasonlóság ellenére a szerkezeti hasonlóság nagy.

A struktúrális konzerváltság miatt lehetőség van arra, hogy a biológiai szekvenciák térszerkezetét *in silico* összehasonlító módszerekkel végezzük el. A számítógépes eljárásokra azért van szükség, mert az elsődleges térszerkezetek meghatározásának a tempójával a másodlagos és harmadlagos térszerkezetek meghatározása nem tud lépést tartani. Az UniProt adatbázis 2011 március 8-ai frissítésében 525997 fehérje szekvencia található, míg a PDB adatbázis 2011 március 15-ei frissítésében csak 71794 fehérje kísérletesen ellenőrzött háromdimenziós szerkezete található meg. Az oka annak, hogy az elsődleges térszerkezetekből (szekvenciákból) sokkal több van, mint másodlagos és harmadlagos térszerkezetekből az, hogy az elsődleges térszerkezet meghatározása nagyságrendekkel olcsóbb és gyorsabb, mint a

háromdimenziós szerkezet meghatározása. Ma már egy teljes emberi genom szekvenálás költsége 5000$ alatt van, és kevesebb, mint 10 nap alatt elvégezhető, míg egyetlen fehérje háromdimenziós szerkezetének a meghatározása még mindig több hónapos labormunkát igényel, és költségeiben összevethető egy teljes emberi genom szekvenálásával. (Az emberi genomban kb. 20-25 ezer gén van).

Az összehasonlító szerkezetpredikcióra két metódus létezik. Az egyik lehetőség az, hogy két olyan szekvenciát hasonlítunk össze, amelyekből az egyiknek a szerkezete ismert, és ezt próbáljuk a másik szekvenciára rávetíteni. Ezt a módszert valójában már a bioinformatika felfedezése előtt, a XIX. század elején is használták, amikor a rosetta kő segítségével megfejtették az egyiptomi hieroglifák jelentését. A rosettai kő (6.2. ábra) három különböző írásban tartalmazza ugyanazt a szöveget: egyiptomi démotikus írással, görög írással és hieroglifákkal. A hieroglifák megfejtői, Jean-François Champollion és Thomas Young a görög szöveget teljesen értették, és rájöttek, hogy az egyiptomi démotikus írással írt rész jelentése is ugyanaz, mint a görög nyelven írt részé. Ebből tételezték fel, hogy a hieroglifákkal írt rész jelentése is ugyanaz, mint a másik két írással írt szövegé. Ebből már viszonylag egyszerűen fejthették meg a hieroglifák jelentését, hiszen egy görög nyelvű fordítás a rendelkezésükre állt. A kövön egyébként V. Ptolemaiosz rendelete olvasható különböző adókról, templomokban felállítandó szobrokról, valamint arról, hogy ezt a rendeletet a fennt említett három nyelven kell kiadni. A három nyelven való kiadás értelme az, hogy az akkori Egyiptomban a különböző társadalmi rétegek más-más írást használtak, így a rendeleteket több írásban adták ki, hogy minden írástudó el tudja olvasni.



6.2. ábra A Rosettai kő

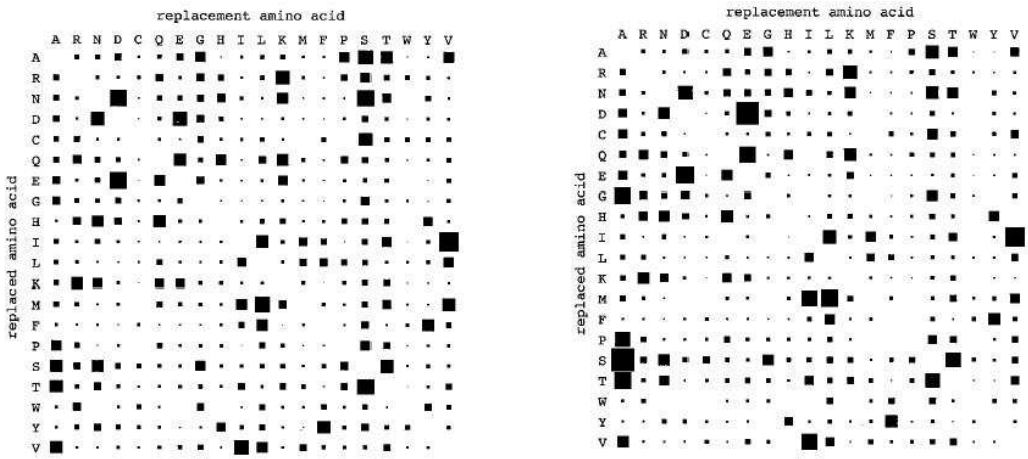
A másik metódus az összehasonlító szerkezetpredikcióra, amikor a biológiai szekvenciák szerkezetére úgy adunk meg predikciót, hogy egyiknek a szerkezete sem ismert, de feltesszük, hogy mindegyiké ugyanaz. A predikció során nem csak azt használjuk ki, hogy a különböző szerkezeti elemek karakter-összetétele más és más minden szekvenciára, hanem hogy a változások is jellemzőek az egyes szerkezeti elemekre. Az alábbiakban olyan szerkezetpredikciókra mutatok példát, amelyekben a szekvenciák szerkezetét sztochasztikus transzformációs nyelvtanok segítségével írjuk le, a szekvenciák változását pedig időfolytonos Markov modellekkel.

6.2. Fehérjék másodlagos térszerkezetének becslése

Ez első modellt, amelyben összekapcsolták a transzformációs nyelvtanokat az időfolytonos Markov modellekkel, Nick Goldman, Jeff Thorne és David Jones publikálta. Ez volt az első publikáció, amely felhívta a figyelmet arra, hogy evolúciós információkat felhasználva javítani lehet a struktúrára vonatkozó predikciókat.

A bemenő adatok többszörös szekvenciaillesztések, amelyben egy oszlopban levő aminosavak homológok. A cikkben bemutatott modellben ezt a szekvenciaillesztést egy rejtett Markov model bocsájtja ki. A Markov folyamat nagyon egyszerű, három emittáló állapota van, α, β, és hurok (loop), melyek a három

alapvető másodlagos térszerkezeti kategóriát reprezentálják, ill. a hurok csak annyit jelent, hogy az így jelölt régió nem tartozik bele az első két kategória egyikébe sem. Az egyes állapotok ugrási valószínűségét ismert térszerkezetű szekvenciákból határozták meg. Minden térszerkezetre rögzítettek egy időfolytonos Markov modellt, amely a karakterek szubsztitúciós folyamatát írja le, ld. 6.3. ábra. A szekvenciák egy evolúciós fa levelein vannak, és a rejtett Markov modellben az egyes emissziók valószínűsége a karaktereknek az evolúciós fa levelein való észlelési valószínűségeként van definiálva. Ezt a valószínűséget Felsenstein algoritmusával hatékonyan lehet kiszámolni.



6.3. ábra A Goldman-Thorne-Jones-féle rejtett Markov modell állapotaihoz tartozó időfolytonos Markov modellek rátamátrixai közül kettő, balra a hurkokhoz tartozó, jobbra az alfa-hélixekhez tartozó rátamátrix. A nagyobb négyzetek nagyobb rátát jelentenek.

	Analysis method							
Data set	HMM[a]		HMM (no tree)[b]		HMM (no tree) +2PSEFL[c]		HMM (no tree) + 9PSEFL[d]	
PSEFL only	**65.7**		**65.7**		**64.7**		**56.6**	
	72.5	*79.7*	72.5	*79.7*	82.4	*77.8*	61.8	*81.6*
	5.8	*95.7*	5.8	*95.7*	23.1	*86.8*	53.8	*75.5*
	81.3	*65.6*	81.3	*65.6*	67.1	*81.2*	54.2	*78.6*
close 3	**74.4**		**68.0**		**64.7**		**56.6**	
	82.4	*88.4*	80.4	*78.7*	81.4	*81.2*	58.8	*82.6*
	38.5	*94.2*	25.0	*92.6*	36.5	*83.3*	55.8	*74.7*
	81.3	*74.0*	74.2	*76.6*	63.2	*82.5*	55.5	*78.6*
med 5	**71.5**		**64.7**		**61.2**		**55.0**	
	73.5	*89.9*	66.7	*85.5*	63.7	*87.4*	53.9	*82.6*
	59.6	*89.5*	51.9	*81.7*	55.8	*75.1*	55.8	*73.2*
	74.2	*74.0*	67.7	*79.2*	61.3	*80.5*	55.5	*77.9*
all 7	**69.6**		**61.5**		**58.9**		**53.7**	
	66.7	*91.3*	59.8	*86.5*	58.8	*87.0*	52.9	*82.6*
	63.5	*84.0*	55.8	*77.8*	53.8	*75.1*	51.9	*72.8*
	73.5	*77.3*	64.5	*77.9*	60.6	*76.6*	54.8	*76.0*

6.4. ábra A predikciós jóság különböző tesztekben. Az egyes sorokban különböző adathalmazok vannak, rendre 1, közeli 3, közepesen távoli 5, illetve 7 szekvencia. Az egyes oszlopokban különböző elemzési metódusok vannak: GTJ-félr HMM, ugyanaz a HMM evolúciós fa nélkül, illetve az adatokhoz hozzáadva kétszer, illetve kilencszer az egyik szekvencia. Minden adat-elemzés párra 7 érték van megadva, középen fent az átlagos pontosság, a baloldali oszlopban a specificitás, a jobboldali oszlopban a szenzitivitás, az első sorban az alfa hélixekre, a második sorban a béta lemezekre, a harmadik sorban a hurkokra.

A modell biológiai magyarázata az, hogy a különböző másodlagos térszerkezetekben nem csak az egyes aminosavak előfordulási gyakorisága különbözik, hanem a preferált szubsztitúciók rátája is.

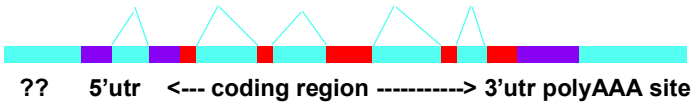
A modellben a rejtett Markov modell ugrási valószínűségei, a rátamátrixok, valamint az evolúciós fa topológiája rögzített, a fa élhosszai a forward algoritmussal kiszámolt teljes likelihood alapján numerikusan maximalizálva van. A maximum likelihood élhosszakra pedig a posterior decoding algoritmus számolja ki azt, hogy az egyes oszlopokban melyik struktúrális elem a legvalószínűbb.

Goldman és munkatársai számos vizsgálatot végeztek az adott modellen valódi biológiai szekvenciákon tesztelve, és az alábbi következtetésekre jutottak:

- Evolúciós információt használva jobb predikciót lehet készíteni, mint evolúciós információk nélkül: a 6.4. ábra első oszlopában szereplő átlagos pontosságok nagyobbak, mint a többi oszlopban.
- A predikció átlagos pontossága akkor volt a legnagyobb, amikor néhány közeli szekvencia alapján végezték a predikciót. Távoli fajok esetében a predikció jósága kissé romlott, de még így is jobb volt, mint az egyetlen szekvenciából végzett predikció. Azaz a túl távoli homológ szekvenciák több 'zajt' vittek be a vizsgálatba, mint információt.
- A β-lemezek predikciója a legnehezebb, viszont ezek a szerkezetek konzerválódnak a legjobban az evolúció során. Bár a becslés átlagos pontossága romlott, ahogy a vizsgálatba egyre több (és egyre távolabbi) szekvenciát vontak be, a β-lemezek predikciója javult.

6.3. Génkeresés páros rejtett Markov modellel

Minden faj genomjában vannak olyan szakaszok, amelyeknek a szerepe tisztázatlan, illetve nem tartalmaznak érdemi információt (szemét-DNS, junk-DNA). A maradék részben vannak a gének, amelyek a faj által termelt fehérjéket kódolják. A géneknek speciális szerkezetük van, ld. 6.5 ábra. Az ún. érett RNS-t (mature RNA) kódoló régiót az átíródás során kivágódó darabok, intronok szabdalják fel. Az érett RNS elején és végén át nem íródó szakaszok vannak, ezeket 5' és 3' UTR-nek (untranslated region) hívjuk. Az érett RNS többi része a tényleges kódoló régió.



6.5 ábra Egy Eukarióta gén szerkezete. Lila színnel jelölve az 5' és 3' UTR régió, pirossal a kódoló régió. A maradék DNS közül az intronok háromszögekkel vannak jelölve.

A kódoló régiókban három nukleinsav kódol egy aminosavat. A három nukleinsavból 64 variáció képezhető, ebből 3 ún. stop-kodon, itt áll meg az érett RNS átírása fehérjére. Mivel 61 kodon kódolja a 20 aminosavat, a kódolás redundáns. Általában 4 kodon kódolja ugyanazt az aminosavat, néhány aminosavnak 2, 6 vagy 1 kodonja van. Amikor 4 kodon kódolja ugyanazt az aminosavat, minden esetben a harmadik pozícióban különböznek. Az intronok nem feltétlenül két kodon közé szúródnak be, beékelődhetnek egy kodonba is. A 6.6 ábrán láthatjuk, hogy nem triviális feladat megmondani, hogy hol helyezkednek el a kódoló régiók egy genomban.

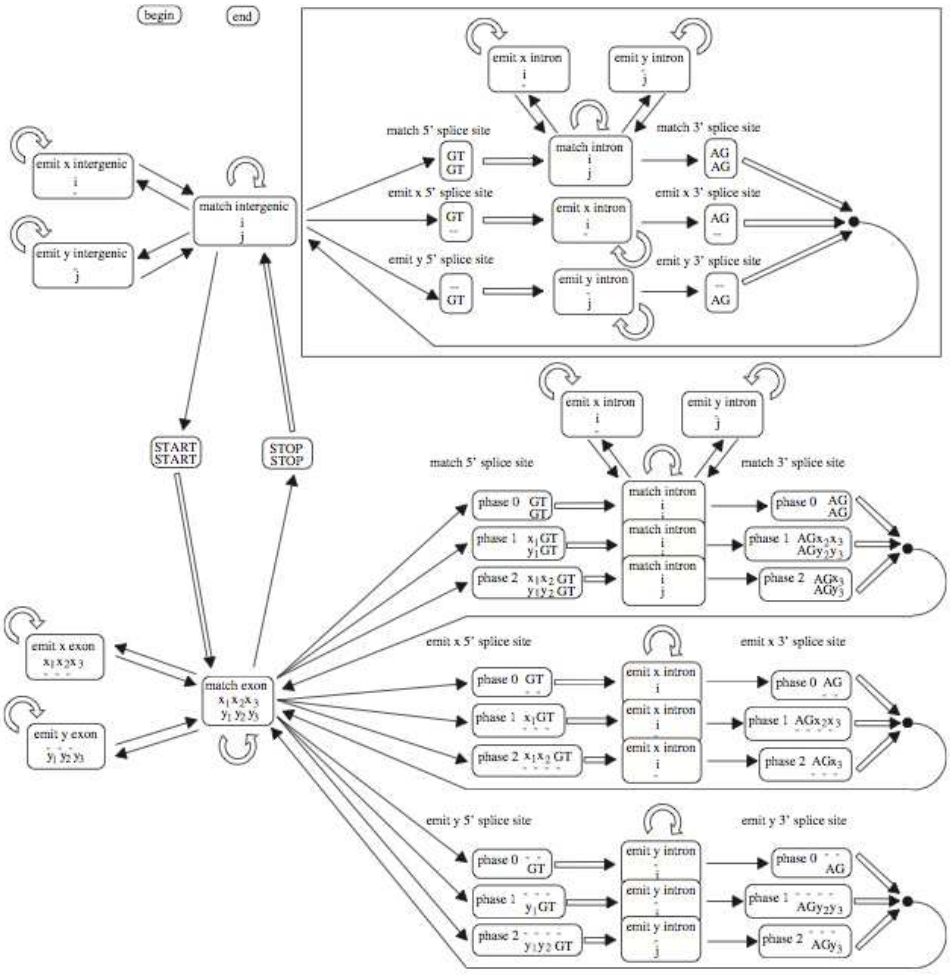
A páros rejtett Markov modellek olyan rejtett Markov modellek, amelyek két szekvenciát bocsájtanak ki. Bizonyos állapotok csak az egyik, más állapotok csak a másik szekvenciába bocsájtanak ki karaktereket, és vannak olyan állapotok is, amelyek mindkét szekvenciába bocsájtanak ki karaktert. A szemlélő csak a kibocsájtott szekvenciákat látja, az együttes kibocsájtási mintázatot (angolul: co-emission pattern) nem. Páros rejtett Markov modellekkel lehet megadni beszúrások és törlések egyszerű sztochasztikus modelljeit is.

6.6. ábra A génkeresési feladat az, hogy egy genomiális DNS szekvenciában megtaláljuk az exonokat és intronokat.

Irmtraud Meyer és Richard Durbin adott meg egy olyan páros rejtett Markov modellt, amelynek a segítségével lehet génszerkezetet prediktálni. A modelljük az Eukarióta gének szerkezetéből az alábbi tulajdonságokat vette figyelembe:

- A beszúrások-törlések az exonokban hármasával történnek az esetek döntő többségében. Minden más hosszúság ún. kereteltolódást (frame-shift) okoz.
- A szubsztitúciók a kodonok harmadik pozícióiban a leggyakoribbak, hiszen ezek általában nem változtatják meg a kódolt aminosavat. A második pozíció a legkonzervatívabb, az első pozíció is konzervatív, de egy picit kevésbé, mint a második
- Az intronokban több mutáció van, mint az exonokban. Az intronokban a beszúrások-törlések hosszai nem szükségképpen hárommal oszthatóak.
- Az intronok helyein a nukleinsavak speciálisak. Az intronok általában T-vel kezdődnek, és előtte egy G van, és egy A-val végződnek, amely után G van.

A fenti tulajdonságok mindegyikét könnyű egy páros rejtett Markov modellel leírni, lásd 6.7. ábra. A páros rejtett Markov modellnek lesznek olyan állapotai, amelyek a junk DNS régióban, 5', 3' UTR-ben, intronban, exonban match-eket modelleznek, illetve amelyek ugyanezen régiókban egyik vagy másik szekvenciába történő beszúrást modelleznek. Továbbá lehetséges, hogy egy intron csak az egyik vagy csak a másik szekvenciában legyen benne, mivel ilyen teljes intron beszúrásokat észleltek már több fajban. A Viterbi útvonal nem csak egy illesztését adja meg két homológ szekvenciának, hanem egyben mindkét szekvenicára szerkezetpredikció is lesz.

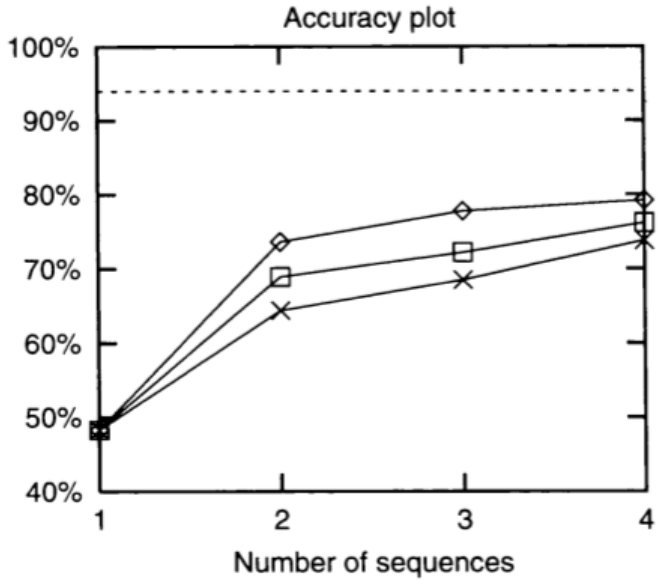


6.7. ábra A Meyer-Durbin féle páros rejtett Markov modell topológiája. A begin állapotból való ugrások valamint az end állapotba való ugrások nincsenek feltüntetve az ábrán az átláthatóság kedvéért.

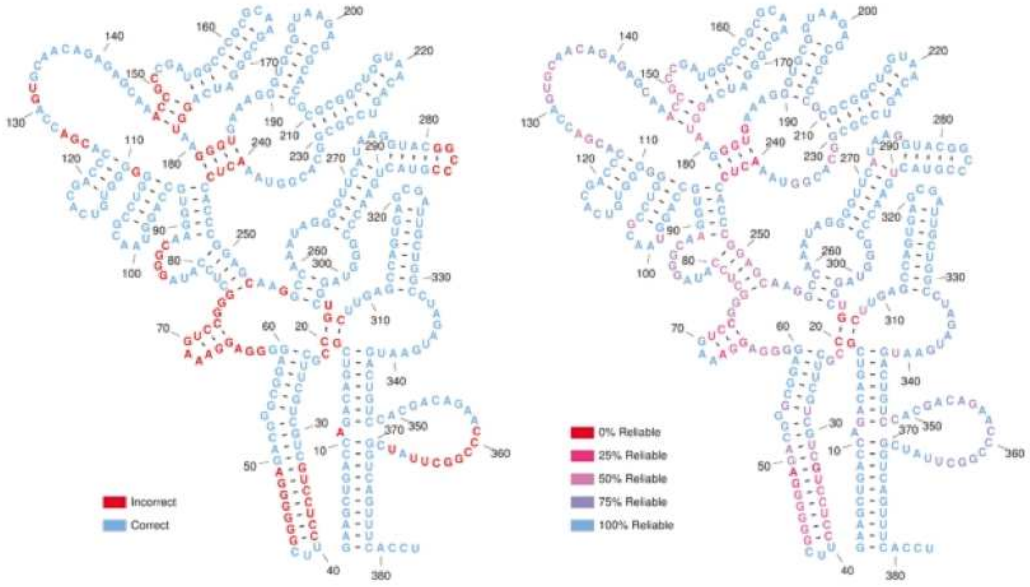
6.4. RNS másodlagos térszerkezet becslés a Knudsen-Hein nyelvtannal

Korábban már megmutattuk, hogy a Knudsen-Hein nyelvtan képes tetszőleges álcsomómentes másodlagos RNS térszerkezeteket generálni, amelyekben a párosodó bázispárok indexei közötti különbség legalább három. A Goldman-Thorne-Jones-féle HMM-hez hasonlóan a Knudsen-Hein nyelvtan is kiterjeszthető úgy, hogy a nyelvtan egy többszörös szekvenciaillesztést generáljon, amelyben a szekvenciaillesztés oszlopainak levezetési valószínűségei a nyelvtanban a megfelelő levezetési valószínűség és egy szubsztitúciós modellben való észlelési valószínűségek szorzata. Két szubsztitúciós modellt vezettek be a szerzők, az egyiket a nem bázispárosodó nukleinsavakra, a másikat pedig a párosodó nukleinsav-párokra. Ez utóbbit egy 16 elemű ABC feletti szubsztitúciós modellel adták meg, amelyben a modell 16 elemű ABC-je a négyelemű ABC-ből képzett rendezett párok.

Knudsen és Hein megmutatták, hogy az RNS térszerkezetek predikciója hatékonyabb filogenetikai információk figyelembevételével (6.8 ábra), valamint hogy a poszterior valószínűségek korrelálnak a predikció jóságával (6.9 ábra).



6.8. ábra RNS térszerkezetek predikciójának pontossága. Mindegyik esetben a térszerkezeteket a Knudsen-Hein nyelvtannal prediktálták. Rombusz: ML filogenetikai fa figyelembevételével, négyzet: filogenetikai fa nélkül, x: a Clustal-X nevű program által generált filogenetikai fát használva. 94%-nál a szaggatott vonal a predikció felső határát jelöli: az RNS tartalmazott egy álcsomót is, amelyet környezetfüggetlen nyelvtannal nem lehet prediktálni.



6.9. ábra Bal oldali ábra: *Klebsiella pneumoniae* baktérium RNase P RNS-ének a térszerkezete, kékkel azok a nukleinsavak vannak kiszínezve, amelyek térszerkezetét helyesen prediktálta a Knudsen-Hein nyelvtant használó Pfold program, míg pirossal a rosszul prediktált nukleinsavak. Jobb oldali ábra: a CYK predikcióhoz tartozó posterior decoding értékek minden nukleinsavra, színkóddal. Piros a legalacsonyabb érték, a kék a legmagasabb.

7. fejezet

Beszúrás-törlés modellek

7.1. A TKF91 modell

Az előző fejezetekben a szubsztitúciókat leíró folytonos idejű Markov modellekkel ismerkedtünk meg. A beszúrások és törlések leírására is meg lehet adni folytonos idejű Markov modelleket, az első publikált modell erre a folyamatra a Thorne-Kishino-Felsenstein modell volt 1991-ben, amelyet a szerzők tiszteletére TKF91 modellnek neveztek el.

Két szekvencia kapcsoltsági valószínűségét keressük, ez definíció szerint

$$P_t(A,B) = \sum_C P_\infty(C) P_t(A|C) P_t(B|C) \tag{7.1}$$

ahol $P_t(A|C)$ annak a valószínűsége, hogy a C szekvencia t idő alatt A-vá evolválódott., $P_\infty(C)$ pedig a C szekvencia egyensúlyi valószínűsége. A folyamatról feltesszük, hogy egyensúlyban van, így a C szekvencia egyensúlyi valószínűsége megegyezik a C szekvencia t idővel ezelőtti gyakoriságával.

Ha a folyamat reverzibilis, akkor, a Pulley Principle-ből adódóan

$$P_t(A,B) = P_\infty(A) P_{2t}(B|A) \tag{7.2}$$

azaz a kapcsoltsági valószínűség kiszámításához nem kell az összes lehetséges ősi szekvencián összegezni. Megállapodunk, hogy a továbbiakban azt az időt, ami alatt az A szekvencia B-vé evolválódott, t-vel fogjuk jelölni, bár tudjuk, hogy ez a két szekvencia szétválása óta eltelt idő kétszerese.

Az egyszerűség kedvéért a beszúrás és törlés folyamatát nem a karakterek segítségével, hanem ún. képzeletbeli linkek segítségével írjuk le. A szekvencia minden egyes karaktere asszociálva van egy halandó (mortal) linkkel. Ez a link mindig a karakter jobb oldalán található. Ezenkívül a szekvencia rendelkezik egy halhatatlan (immortal) linkkel is, amely a szekvencia bal végén található. Például, ha a halandó link jele $*$, a halhatatlané pedig o, akkor a GACTTAA szekvencia a következőképpen néz ki az adott modellben

o G * A * C * T * T * A * A *

Minden egyes link független a másik linktől, egy link születése vagy halálozása nem befolyásolja egy másik link születésének vagy halálozásának a valószínűségét. Mind a halandó, mind a hallhatatlan link képes halandó linkek szülésére. Az újszülött link mindig a szülője jobb oldalán helyezkedik el. Egy link születése össze van kapcsolva egy karakter születésével, amihez az újszülött link asszociálódik. Az új karakter mindig az újszülött link bal oldalán helyezkedik el. Az új karakter típusát a szubsztitúciós model egyensúlyi eloszlása szabja meg, azaz egy karaktert akkora valószínűséggel választunk, amekkora az egyensúlyi gyakorisága a szubsztitúciós

modellben. Az újszülött karakterek ilyen módon történő választása nem zavarja meg az egyensúlyt, azaz a születés utáni tetszőleges időpontban a született karakter az egyensúlyi eloszlásban lesz. A halandó linkek meghalhatnak, míg a halhatatlan nem. Ha egy link meghal, akkor a hozzá asszociált karakter is automatikusan törlődik. Hagyományosan a születési rátát λ-val, a halálozási rátát pedig μ-vel jelöljük.

Többszörös beszúrások és törlések nem lehetségesek ebben a modellben. Egy n hosszúságú szekvencia növekedési rátája $(n+1)\lambda$, mivel egy n hosszúságú szekvencia $n+1$ linkkel rendelkezik (beleszámítva a halhatatlant is). Hasonlóan, egy n hosszúságú szekvencia lecsökken $n-1$ hosszúságúra $n\mu$ rátával, mivel n halandó linket tartalmaz. A halhatatlan linkre két okból van szükség. Egyrészt ennek a segítségével modellezzük a szekvencia elején történő beszúrásokat, másrészről e nélkül a szekvencia hosszaknak nem lenne egyensúlyi eloszlásuk.

Képzeljünk el két szekvenciát, az A szekvencia legyen TGTC, a B szekvencia pedig GCACA. Számos lehetőség van arra, hogy az A szekvencia B-vé alakuljon. Egy lehetőséget az alábbi α illesztés reprezentál

```
- T G T - C -
G - C - A C A
```

azaz az első, ötödik és hetedik pozícióban egy-egy beszúrás történt, a másodikban és a negyedikben egy-egy törlés, a harmadikban pedig egy szubsztitúció. Jelölje az α illesztésben az egyes karakterek meglétének vagy meg nem létének az információját α'. Ha α'-t a linkek segítségével reprezentáljuk, akkor a következő illesztést kapjuk

```
○ -   *   *   * -   * -
○ *   -   *   - *   * *
```

Fontos megjegyezni egy lényeges különbséget a konvencionális illesztés és az itt bemutatott illesztés között. Az illesztésre az előbbi példa a következő volt

```
- T G T - C -
G - C - A C A
```

Ha a negyedik és az ötödik pozíciót felcseréljük, a következő illesztést kapjuk

```
- T G - T C -
G - C A - C A
```

Nem világos, hogy a hagyományos értelemben ez a két illesztés miben különbözik egymástól, de az itt bemutatott modell alapján a két illesztés világosan különbözik. Az első illesztésben a negyedik pozícióban található T-hez asszociált link szülte azt a linket, amely A-hoz van asszociálva. Magyarán A beszúrása mindenképpen megelőzte T törlését. A második illesztésben a negyedik pozícióban található A-hoz asszociált link a harmadik pozícióban található G-hez asszociált link leszármazottja. Ekképpen A beszúrása nem feltétlenül előzte meg T törlését.

A linkek csoportosíthatóak a fenti reprezentációban, azzal a céllal, hogy meghatározzuk $P(\alpha'|\theta)$-t, ahol θ a paraméterek halmaza. Egy tetszőleges tranzíció reprezentálható egy α illesztéssel. Az illesztés valószínűsége a tranzíció valószínűsége szorozva az ősi szekvencia valószínűségével. Az α illesztés

valószínűsége két komponensre bontható. Mivel α tartalmazza az összes információt α'-ről, ezért

$$P(\alpha \mid \theta)=P(\alpha, \alpha' \mid \theta)=P(\alpha \mid \alpha', \theta)\, P(\alpha' \mid \theta) \tag{7.3}$$

Általában, ha egy szekvencia n hosszúságú, $P(\alpha' \mid \theta)$ n+2 kifejezésnek a szorzata. Az első kifejezés az ősi szekvencia valószínűsége, a következő kifejezés azt írja le, hogy a halhatatlan linknek hány utódja van, a többi kifejezés pedig a halandó linkek sorsát írja le. Egy adott linkre ez a kifejezés függ attól, hogy a link életben maradt vagy meghalt, valamint a link utódjainak a számától. Mindkettő könnyen leolvasható az α' illesztésből. Egy link utódjainak a száma egy (ha a link túlélt) plusz a jobb oldala és a következő ősi link bal oldala között található leszármazottak száma.

Figyelemmel kísérve az egyes linkek sorsát, három fajta valószínűségi függvényt kell számításba venni: $p_n(t)$ fogja jelölni a valószínűségét annak, hogy egy halandó link t idő elteltével túlél, és saját magát is beleszámítva n utódja van a t-edik időpillanatban. $p_n'(t)$ jelöli a valószínűségét annak, hogy egy link meghal t idő alatt, de n utódot hagy maga után, és végül $p_n''(t)$ fogja jelölni annak a valószínűségé, hogy a halhatatlan linknek a t-edik időpillanatban n utódja van, önmagát is beleértve. A fenti illesztésre a valószínűségek

$$P(\alpha' \mid \theta) = \gamma_4\ p_2''(t)\ p_0'(t)\ p_1(t)\ p_1'(t)\ p_2(t) \tag{7.4}$$

$$P(\alpha \mid \alpha', \theta) = \pi_G\ \pi_T\ \pi_G f_{GC}(t)\ \pi_T\ \pi_A\ \pi_C f_{CC}(t) \pi_A \tag{7.5}$$

Definíció szerint $p_0''(t) = p_0(t) = 0$. A többi valószínűségi függvényt a következő differenciálegyenlet-rendszer írja le

$$\frac{dp_n(t)}{dt} = \lambda(n-1)p_{n-1}(t) - (\lambda+\mu)np_n(t) + \mu np_{n+1}(t) \qquad n > 0 \tag{7.6}$$

$$\frac{dp_n'(t)}{dt} = \lambda(n-1)p_{n-1}'(t) - (\lambda+\mu)np_n'(t) + \mu(n+1)p_{n+1}'(t) + \mu p_{n+1}(t) \qquad n > 0 \tag{7.7}$$

$$\frac{dp_0'(t)}{dt} = \mu p_1'(t) + \mu p_1(t) \tag{7.8}$$

$$\frac{dp_n''(t)}{dt} = \lambda(n-1)p_{n-1}''(t) - \left[\lambda n + \mu(n-1)\right]p_n''(t) + \mu n p_{n+1}''(t) \qquad n > 0 \tag{7.9}$$

a kezdeti feltételek a következők:

$$p_1(0) = p_1''(t) = 1 \tag{7.10}$$

$$p_n(0) = p_n''(t) = 0 \qquad n = 2, 3, \ldots \tag{7.11}$$

$$p_n'(0) = 0 \qquad n = 0, 1, \ldots \tag{7.12}$$

Vegyük észre, hogy a fenti dinamikát leíró differenciálegyenlet-rendszer végtelen dimenziós. Hogyan oldható meg egy ilyen differenciálegyenlet-rendszer? A TKF91 modellt átfogalmaztó a sorban állási rendszerek nyelvére, és megmutató, hogy abban a kontextusban a problémát gyakorlatilag megoldották.

A sorban állási rendszerek általános problémája a következő. Adott egy rendszer, amelyben egy vagy több kiszolgáló egység található. A rendszerbe vevők érkeznek, akiket ki kell szolgálni. Egy kiszolgáló egység csak egyetlen vevőt tud

egyidejűleg kiszolgálni. Ha minden kiszolgáló egység foglalt, akkor az újonnan érkezett vevőknek várakozniuk kell, ha van szabad kiszolgáló egység, akkor a vevő kiszolgálása a beérkezéssel egy időben elkezdődik Mind a kiszolgálás időtartamát, mind a tetszőleges két egymás után érkező vevő beérkezése közötti időtartamot egy folytonos valószínűségi változó írja le. A legegyszerűbb valószínűségi változók exponenciális eloszlásúak, azaz mind a kiszolgálási, mind a beérkezési folyamatot Poisson folyamatként képzeljük el. A rendszer állapotát a rendszerben tartózkodó vevők számával lehet leírni. A sorban állási rendszerek nyelvén a következő feladatok fogalmazhatóak meg

- Ismert rendszer, valószínűségi változók és a rendszer adott kezdeti állapota esetén megmondani t idő elteltével a rendszer lehetséges állapotainak a valószínűségét, azaz leírni a rendszer ún. tranziens viselkedését.
- A rendszer stacionárius állapotának a jellemzése, azaz a folyamat elindítása után végtelen idővel mekkorák a rendszer egyes állapotainak a valószínűségei.
- Egy új vevő nem csak a sor végére állhat, hanem egy ismerőse után is. Ilyen esetekben a tranziens viselkedés leírásakor nem csak a rendszer állapotaira lehet rákérdezni, hanem a rendszer ún. részállapotaira is, azaz a kezdeti állapotban a rendszerben tartózkodó vevők közül kinek hány ismerőse érkezett.

A kiszolgálási időt és az érkezési időt leíró valószínűségi változók függhetnek a rendszer állapotától. Például a rendszerben tartózkodó vevők számától függhet a következő vevő beérkezéséig eltelő időtartam. A hosszú sor elijeszthet a vevőket, de akár fel is bátoríthatja. Ez utóbbira példa az egykori szocialista országok vevőinek a magatartása. A szomszédom mesélte, hogy kiskorában, az ötvenes években napjában többször is felmászott a padlásra, ahonnan le lehetett látni a domb alján elterülő térre. Ha sor állt a bolt előtt, akkor ők is rohantak le a boltba, mert akkor biztosan lehetett kapni valamit, ami egyébként hiánycikk volt.

Ha a TKF modellt átfogalmazzuk a sorban állási rendszerek nyelvére, akkor a halandó linkek lesznek a vevők. Egy vevő kiszolgálása a sorban állási rendszerek nyelvén ugyanazt jelenti, mint egy link halála a TKF modellben. A TKF modellben bármely halandó link μ rátával halhat meg, függetlenül a szekvenciában található linkek számától. A sorban állási rendszerekben ez azt jelenti, hogy minden egyes vevő kiszolgálási idejét egy μ paraméterű exponenciális eloszlású valószínűségi változó írja le, és minden egyes újonnan beérkező vevő kiszolgálása azonnal elkezdődik, függetlenül a rendszerben tartózkodó vevők számától. Ez utóbbi azt jelenti, hogy a rendszer kimeríthetetlen kapacitású, azaz végtelen sok kiszolgálóegységgel van felszerelve. Egy link születése a sorban állási rendszerekben egy új vevő érkezését jelenti. Egy új link születési rátája egyenesen arányos a szekvenciában meglévő összes linkek számával. A sorban állási modellben tehát felbátorodó vevőkről van szó. Az új vevő beérkezéséig eltelő idő függ a rendszerben tartózkodó vevők számától, de továbbra is exponenciális eloszlású valószínűségi változó írja le, csupán az eloszlás paramétere függ a rendszerben tartózkodó vevők számától. A TKF modellben illeszteni kell a szekvenciákat, ezért a sorban állási rendszerben kíváncsiak vagyunk a részállapotok valószínűségeire is. A részállapotok folyamatát úgy lehet leírni, hogy minden vevőhöz λ paraméterű Poisson folyamattal csatlakozik egy ismerős. Ezenkívül van egy 'bolti csalogató', aki szintén λ paraméterű exponenciális várakozási idővel csalogatja be egymás után a vevőket. Ez a 'csalogató' a TKF modellben az immortális link.

Először az immortális link lehetséges részállapotainak a valószínűségeit számoljuk ki. Az egyszerűbb számolás kedvéért s sorban állási modellben csak a halandó linkek számát számoljuk, így $p''_n(t)$ indexelése el van csúsztatva eggyel. Így most $p''_n(t)$ a 'bolti csalogató' által behívott, t idő elteltével a rendszerben tartózkodó vevők számát jelenti. Az ezt leíró differenciálegyenlet-rendszer

$$\frac{dp''_n(t)}{dt} = \lambda n p''_{n-1}(t) - \left[\lambda(n+1) + \mu n\right] p''_n(t) + \mu(n+1) p''_{n+1}(t) \qquad n > 0 \qquad (7.13)$$

ahol $p''_{-1}(t) \equiv 0$, definíció szerint. A kezdeti érték feltétel:

$$p''_n(t) = \delta_{n,0} \qquad (7.14)$$

A sorban állási rendszerek elméletében az ilyen végtelen differenciálegyenlet-rendszerek analitikus megoldási technikája már régóta ismert. Be kell vezetni az ún. generátorfüggvényt

$$P(\xi,t) = \sum_{n=0}^{\infty} p''_n(t)\xi^n \qquad (7.15)$$

Végigszorozva (6.1.1)-et ξ^n-nel, és összegezve az összes n-re, P(ξ,t) egy parciális differenciálegyenletét kapjuk

$$\frac{\partial P(\xi,t)}{\partial t} = \left[\lambda\xi(\xi-1) + \mu(1-\xi)\right]\frac{\partial P(\xi,t)}{\partial \xi} + \lambda(\xi-1)P(\xi,t) \qquad (7.16)$$

A kapott parciális differenciálegyenlet elsőrendű és lineáris lesz. Lineáris azért, mert autonóm, lineáris differenciálegyenlet-rendszerből indultunk ki, és elsőrendű azért, mert elsőrendű volt a differenciálegyenlet-rendszer, és az n-edik differenciálegyenlet együtthatói n-nek elsőfokú függvényei voltak. A 7.16. egyenlet könnyen megoldható Lagrange (a karakterisztikák) módszerével. Ha $v(\xi,t,P)=C_1$ és $w(\xi,t,P)=C_2$ a

$$\frac{dt}{1} = \frac{d\xi}{(\xi-1)(\mu-\lambda\xi)} = \frac{dP}{\lambda(\xi-1)P} \qquad (7.17)$$

rendszernek két független megoldása, akkor (7.16) összes megoldása $\Phi\big(v(\xi,t,P)\big) = w(\xi,t,P)$ alakú, ahol Φ tetszőleges folytonosan differenciálható függvény.

A

$$\frac{dt}{1} = \frac{d\xi}{(\xi-1)(\mu-\lambda\xi)} \qquad (7.18)$$

differenciálegyenlet megoldásai

$$\frac{1-\xi}{\mu-\lambda\xi} e^{-t(\mu-\lambda)} = C_1 \qquad (7.19)$$

A

$$\frac{d\xi}{(\xi-1)(\mu-\lambda\xi)} = \frac{dP}{\lambda(\xi-1)P} \tag{7.20}$$

differenciálegyenlet megoldásai

$$P(\mu-\lambda\xi) = C_2 \tag{7.21}$$

Így a 7.16 egyenletben levő parciális differenciálegyenlet általános megoldása

$$P(\xi,t) = \frac{1}{\mu-\lambda\xi}\Phi\left(\frac{1-\xi}{\mu-\lambda\xi}e^{-t(\mu-\lambda)}\right) \tag{7.22}$$

A Φ függvényt a kezdeti érték feltétel szabja meg. A 7.14 egyenletből adódóan

$$P(\xi,0) = 1 \tag{7.23}$$

Ezért

$$\Phi(a) = \frac{\mu-\lambda}{1-\lambda a} \tag{7.24}$$

Azaz a 7.16 egyenletben levő parciális differenciálegyenletnek a 7.14 peremfeltétel melletti megoldása

$$P(\xi,t) = \frac{\mu-\lambda}{\mu-\lambda e^{-(\mu-\lambda)t} + \lambda\left(e^{-(\mu-\lambda)t}-1\right)\xi} \tag{7.25}$$

A $p''_n(t)$ valószínűségi függvények $P(\xi,t)$ Maclaurin-sorából (0 körüli Taylor-sorából) határozhatóak meg

$$\frac{\partial^n P(\xi,t)}{\partial\xi^n} = \frac{n!(\mu-\lambda)\lambda^n\left(1-e^{-(\mu-\lambda)t}\right)^n}{\left[\mu-\lambda e^{-(\mu-\lambda)t} + \lambda\left(e^{-(\mu-\lambda)t}-1\right)\xi\right]^{n+1}} \tag{7.26}$$

Az n-edik deriváltat elosztva $n!$-sal, és tekintve a ξ=0 pontban kapjuk, hogy

$$\frac{1}{n!}\frac{\partial^n P(\xi,t)}{\partial\xi^n}\bigg|_{\xi=0} = \frac{\mu-\lambda}{\mu-\lambda e^{-(\mu-\lambda)t}}\left[\lambda\frac{1-e^{-(\mu-\lambda)t}}{\mu-\lambda e^{-(\mu-\lambda)t}}\right]^n \tag{7.27}$$

Tehát annak a valószínűsége, hogy a halhatatlan linknek t idő múlva önnmagával együtt n leszármazottja lesz

$$p''_n(t) = [1-\lambda\beta(t)][\lambda\beta(t)]^{n-1} \tag{7.28}$$

ahol

$$\beta(t) = \frac{1-e^{(\lambda-\mu)t}}{\mu-\lambda e^{(\lambda-\mu)t}} \tag{7.29}$$

mert

$$1-\lambda\beta(t) = 1-\lambda\frac{1-e^{-(\mu-\lambda)t}}{\mu-\lambda e^{-(\mu-\lambda)t}} = \frac{\mu-\lambda}{\mu-\lambda e^{-(\mu-\lambda)t}} \tag{7.30}$$

A halandó linkek sorsát leíró függvényeket kicsit nehezebb kiszámolni. EmlékeztetőülKét generátorfüggvényt kell bevezetni, egyet $p_n(t)$-re, egyet pedig $p'_n(t)$-re:

$$P^{(1)}(\xi,t) = \sum_{n=1}^{\infty} p_n(t)\xi^n \tag{7.31}$$

$$P^{(2)}(\xi,t) = \sum_{n=1}^{\infty} p'_n(t)\xi^n \tag{7.32}$$

A 7.7-7.9 megfelelő egyenleteit ξ^n-nel szorozva, és az összes n-re összeadva egy parciális differenciálegyenlet-rendszert kapunk

$$\frac{\partial P^{(1)}(\xi,t)}{\partial t} = \frac{\partial P^{(1)}(\xi,t)}{\partial \xi}(\lambda\xi-\mu)(\xi-1) - P^{(1)}(\xi,t)\frac{\mu}{\xi} \tag{7.33}$$

$$\frac{\partial P^{(2)}(\xi,t)}{\partial t} = \frac{\partial P^{(2)}(\xi,t)}{\partial \xi}(\lambda\xi-\mu)(\xi-1) + P^{(1)}(\xi,t)\frac{\mu}{\xi} \tag{7.34}$$

Az első egyenlet csak $P^{(1)}(\xi,t)$-től függ. Ezt Lagrange módszerével megoldva a

$$\frac{dt}{1} = \frac{d\xi}{(\lambda\xi-\mu)(1-\xi)} = \frac{dP^{(1)}}{-P^{(1)}\frac{\mu}{\xi}} \tag{7.35}$$

rendszer két független megoldása

$$\frac{1-\xi}{\lambda\xi-\mu}e^{-(\mu-\lambda)t} = C_1 \tag{7.36}$$

$$\frac{P^{(1)}}{\xi}\left[\frac{(\lambda\xi-\mu)^{\lambda}}{(1-\xi)^{\mu}}\right]^{\frac{1}{\lambda-\mu}} = C_2 \tag{7.37}$$

A 7.33 egyenlet általános megoldása tehát

$$P^{(1)}(\xi,t)=\xi\left[\frac{(\lambda\xi-\mu)^{\lambda}}{(1-\xi)^{\mu}}\right]^{\frac{-1}{\lambda-\mu}}\Phi\left(\frac{1-\xi}{\lambda\xi-\mu}e^{-(\mu-\lambda)t}\right) \tag{7.38}$$

A Φ függvény a

$$P^{(1)}(\xi,0)=\xi \tag{7.39}$$

peremfeltételből számítható ki. Mivel a

$$f(x)=\frac{1-x}{\lambda x-\mu} \tag{7.40}$$

függvény inverze

$$f^{-1}(x)=\frac{1+\mu x}{1+\lambda x} \tag{7.41}$$

ezért

$$\Phi(a)=\left[\frac{\left(\lambda\frac{1+\mu a}{1+\lambda a}-\mu\right)^{\lambda}}{\left(1-\frac{1+\mu a}{1+\lambda a}\right)^{\mu}}\right]^{\frac{1}{\lambda-\mu}} \tag{7.42}$$

Ezt egyszerűsítve

$$\Phi(a)=\left[\frac{\left(\frac{\lambda-\mu}{1+\lambda a}\right)^{\lambda}}{\left(\frac{(\lambda-\mu)a}{1+\lambda a}\right)^{\mu}}\right]^{\frac{1}{\lambda-\mu}}=\left[\frac{(\lambda-\mu)^{\lambda-\mu}a^{-\mu}}{(1+\lambda a)^{\lambda-\mu}}\right]^{\frac{1}{\lambda-\mu}}=\frac{(\lambda-\mu)a^{-\frac{\mu}{\lambda-\mu}}}{1+\lambda a} \tag{7.43}$$

Így a 7.34 egyenletnek a 7.39 peremfeltétel melletti megoldása

$$P^{(1)}(\xi,t)=\xi\left[\frac{(\lambda\xi-\mu)^{\lambda}}{(1-\xi)^{\mu}}\right]^{\frac{-1}{\lambda-\mu}}\frac{(\lambda-\mu)\left(\frac{1-\xi}{\lambda\xi-\mu}e^{-(\mu-\lambda)t}\right)^{-\frac{\mu}{\lambda-\mu}}}{1+\lambda\frac{1-\xi}{\lambda\xi-\mu}e^{-(\mu-\lambda)t}} \tag{7.44}$$

Ezt egyszerűbb alakra hozva

$$P^{(1)}(\xi,t)=\frac{\xi\left(\lambda\xi-\mu\right)^{-1}e^{-\mu t}(\lambda-\mu)}{1+\lambda\dfrac{1-\xi}{\lambda\xi-\mu}e^{-(\mu-\lambda)t}}=\frac{\xi e^{-\mu t}(\lambda-\mu)}{\lambda e^{-(\mu-\lambda)t}-\mu+\lambda\xi\left(1-e^{-(\mu-\lambda)t}\right)}=$$
$$=\frac{\xi e^{-\mu t}(\mu-\lambda)}{\mu-\lambda e^{-(\mu-\lambda)t}+\lambda\left(e^{-(\mu-\lambda)t}-1\right)\xi} \tag{7.45}$$

Látható, hogy 7.45 csak egy $\xi e^{-\mu t}$ faktorban tér el 7.25-től. A ξ faktor a Maclaurin sort csúsztatja el, így

$$p_n(t)=e^{-\mu t}\left[1-\lambda\beta(t)\right]\left[\lambda\beta(t)\right]^{n-1} \tag{7.46}$$

Ezt az eredményt meg lehetett volna kapni generátorfüggvény nélkül is, csupán a 7.28 egyenletből elemi valószínűségszámítási ismeretek segítségével. Ugyanis amíg tudjuk, hogy a kezdő halandó link életben van, addig ez a rendszer és a halhatatlan link rendszere semmiben nem különbözik egymástól. Legyen E az az esemény, hogy a kezdő link életben van. Ennek a valószínűsége

$$P(E)=e^{-\mu t} \tag{7.47}$$

Ha F_n az az esemény, hogy a kezdő link túlélt, és önmagával együtt n utódja van, akkor

$$P(F_n)=P(F_n,E)=P(F_n\mid E)P(E)=\left[1-\lambda\beta(t)\right]\left[\lambda\beta(t)\right]^{n-1}e^{-\mu t} \tag{7.48}$$

A második parciális differenciálegyenlet megoldásához célszerűbb nem behelyettesíteni $P^{(1)}(\xi,t)$-t, hanem beevezetni a

$$Q(\xi,t)=P^{(1)}(\xi,t)+P^{(2)}(\xi,t) \tag{7.49}$$

függvényt. Ekkor a 7.33 és 7.34 egyenleteket összeadva kapjuk, hogy

$$\frac{\partial Q(\xi,t)}{\partial t}=\frac{\partial Q(\xi,t)}{\partial\xi}(\lambda\xi-\mu)(\xi-1) \tag{7.50}$$

Ennek a parciális differenciálegyenletnek az általános megoldása

$$Q(\xi,t)=\Phi\left(\frac{1-\xi}{\lambda\xi-\mu}e^{-(\mu-\lambda)t}\right) \tag{7.51}$$

a peremfeltétel

$$Q(\xi,0)=\xi \tag{7.52}$$

ekképpen

$$\Phi(a)=\frac{\mu a+1}{\lambda a+1} \tag{7.53}$$

És így

$$Q(\xi,t)=\frac{\mu\left(e^{-(\mu-\lambda)t}-1\right)+\xi\left(\lambda-\mu e^{-(\mu-\lambda)t}\right)}{\lambda e^{-(\mu-\lambda)t}-\mu+\lambda\xi(1-e^{-(\mu-\lambda)t})} \tag{7.54}$$

Jelölje $M(f(x))$ $f(x)$ Maclaurin sorát! $Q(\xi,t)$-t

$$\begin{aligned} Q(\xi,t) &= M\left(\frac{\mu\left(e^{-(\mu-\lambda)t}-1\right)}{\lambda e^{-(\mu-\lambda)t}-\mu+\lambda\xi(1-e^{-(\mu-\lambda)t})}\right)+ \\ &+\xi M\left(\frac{\left(\lambda-\mu e^{-(\mu-\lambda)t}\right)}{\lambda e^{-(\mu-\lambda)t}-\mu+\lambda\xi(1-e^{-(\mu-\lambda)t})}\right) \end{aligned} \tag{7.55}$$

alakban keressük.

$$\frac{\partial^n}{\partial\xi^n}\left(\frac{\mu\left(e^{-(\mu-\lambda)t}-1\right)}{\lambda e^{-(\mu-\lambda)t}-\mu+\lambda\xi(1-e^{-(\mu-\lambda)t})}\right)=\frac{n!\,\mu\left(e^{-(\mu-\lambda)t}-1\right)\lambda^n(e^{-(\mu-\lambda)t}-1)^n}{\left[\lambda e^{-(\mu-\lambda)t}-\mu+\lambda\xi(1-e^{-(\mu-\lambda)t})\right]^{n+1}} \tag{7.56}$$

és

$$\frac{\partial^n}{\partial\xi^n}\left(\frac{\left(\lambda-\mu e^{-(\mu-\lambda)t}\right)}{\lambda e^{-(\mu-\lambda)t}-\mu+\lambda\xi(1-e^{-(\mu-\lambda)t})}\right)=\frac{n!\left(\lambda-\mu e^{-(\mu-\lambda)t}\right)\lambda^n(e^{-(\mu-\lambda)t}-1)^n}{\left[\lambda e^{-(\mu-\lambda)t}-\mu+\lambda\xi(1-e^{-(\mu-\lambda)t})\right]^{n+1}} \tag{7.57}$$

Így $Q(\xi,t)$ Maclaurin sorának n-edik tagja

$$\left[\frac{\lambda\mu\left(e^{-(\mu-\lambda)t}-1\right)^2}{\left(\lambda e^{-(\mu-\lambda)t}-\mu\right)^2}+\frac{\lambda-\mu e^{-(\mu-\lambda)t}}{\lambda e^{-(\mu-\lambda)t}-\mu}\right]\left[\lambda\beta(t)\right]^{n-1}=[1-\mu\beta(t)][1-\lambda\beta(t)][\lambda\beta(t)]^{n-1} \tag{7.58}$$

Mivel

$$P^{(2)}(\xi,t)=Q(\xi,t)-P^{(1)}(\xi,t) \tag{7.59}$$

ezért

$$p_n'(t)=\left[1-\mu\beta(t)-e^{-\mu t}\right]\left[1-\lambda\beta(t)\right]\left[\lambda\beta(t)\right]^{n-1} \tag{7.60}$$

A kapcsoltsági valószínűség kiszámításához nem csak a tranzíciók valószínűségeit kell kiszámolni, hanem az ősi szekvenciák egyensúlyi valószínűségét is. A Thorne-Kishino-Felsenstein modellben egy adott n hosszúságú szekvencia valószínűsége az egyes karakterek valószínűségeinek a szorzata szorozva annak a valószínűségével, hogy a szekvencia éppen n hosszúságú. Könnyen beláthatóak az alábbi határértékek:

$$\lim_{t\to\infty}(p_n'(t)) = 0 \tag{7.61}$$

$$\lim_{t\to\infty}(p_n(t)) = 0 \tag{7.62}$$

$$\lim_{t\to\infty}(p_n''(t)) = \left(1-\frac{\lambda}{\mu}\right)\left(\frac{\lambda}{\mu}\right)^{n-1} \tag{7.63}$$

ezért a megadott születési-halálozási modell egyensúlyi eloszlása egy eggyel balra elcsúsztatott geometriai eloszlás (a szekvencia 0 hosszú is lehet), ha γ_n jelöli az n hosszúságú szekvencia egyensúlyi gyakoriságát, akkor

$$\gamma_n = \left(1-\frac{\lambda}{\mu}\right)\left(\frac{\lambda}{\mu}\right)^{n} \tag{7.64}$$

Ezek után meg tudunk adni egy dinamikus programozást a két szekvencia kapcsoltsági valószínűségére. Legyen adott két szekvencia, A és B, rendre n és m hosszúsággal. A kapcsoltsági valószínűséget a

$$P(A,B|\theta) = P_t(B|A,\theta)\gamma_n\prod_{x\in\Omega}\pi_x^{r_x} \tag{7.65}$$

képlettel számolható ki, ahol r_x az x karakterek száma az A szekvenciában. A $P(A,B|\theta)$ függvényt dinamikus programozási algoritmussal számítjuk ki. Ez három ok miatt lehetséges.

- Az egyes linkek egymástól függetlenül evolválódnak. Így $P(\alpha' \mid \theta)$ felbontható az egyes linkek sorsát leíró függvényekre, ahogy az a 7.4 képletben be lett mutatva.
- $n\geq 1$-től a 7.28, 7.46 és 7.60 egyenletekben szereplő képletek felírhatóak az első tag és $\lambda\beta(t)$ segítségével. Azaz

$$p_n(t) = p_1(t)\left[\lambda\beta(t)\right]^{n-1} \tag{7.66}$$

$$p_n'(t) = p_1'(t)\left[\lambda\beta(t)\right]^{n-1} \tag{7.67}$$

$$p_n''(t) = p_1''(t)\left[\lambda\beta(t)\right]^{n-1} \tag{7.68}$$

  Ez a felírás lehetőséget ad O(nm) idejű algoritmus készítésére.
- A szekvencia hosszak eloszlása geometriai, ezért minden egyes halandó link az A szekvenciában egy $\frac{\lambda}{\mu}$ faktorral járul hozzá a likelihoodhoz. Meg kell azonban jegyezni, hogy ez nem feltétlenül szükséges két szekvencia kapcsoltsági valószínűségének a kiszámításához.

Legyen $S(A_i,B_j)$ A_i és B_j összes lehetséges illesztésének a halmaza. Ezt a halmazt három diszjunkt részhalmazra bontjuk, az utolsó link sorsa alapján

- $S^0(A_i,B_j)$ azon illesztések halmaza, amelyben az A_i szekvencia utolsó linkjének nincs leszármazottja B_j-ben
- $S^1(A_i,B_j)$ azon illesztések halmaza, amelyben az A_i szekvencia utolsó linkjének pontosan egy leszármazottja van B_j-ben
- $S^2(A_i,B_j)$ azon illesztések halmaza, amelyben az A_i szekvencia utolsó linkjének legalább két leszármazottja van B_j-ben

Egy adott θ paraméterhalmaz mellett $\alpha(A_i,B_j)$ illesztés valószínűségét $p_\theta[\alpha(A_i,B_j)]$ jelöli. Az egyes S^i részhalmazok valószínűségeit az alábbiakban definiáljuk

$$P_\theta^k(A_i,B_j) = \sum_{\alpha(A_i,B_j)\in S^{ki}(A_i,B_j)} p_\theta\left[\alpha(A_i,B_j)\right] \qquad k=0,1,2 \tag{7.69}$$

Ezek után a dinamikus programozási algoritmus az alábbi szabályokat követi

$$\begin{aligned}
&a)\quad P_\theta^0(A_i,B_j) = \frac{\lambda}{\mu}\pi_{a_i} p_0'(t)\sum_{k=0}^{2} P_\theta^k(A_{i-1},B_j)\\
&b)\quad P_\theta^1(A_i,B_j) = \frac{\lambda}{\mu}\pi_{a_i}\left[f_{a_i,b_j}(t)p_1(t) + \pi_{b_j} p_1'(t)\right]\sum_{k=0}^{2} P_\theta^k(A_{i-1},B_{j-1})\\
&c)\quad P_\theta^0(A_i,B_j) = \lambda\beta(t)\pi_{b_j}\sum_{k=1}^{2} P_\theta^k(A_i,B_{j-1})
\end{aligned} \tag{7.70}$$

A kezdeti feltételek

$$\begin{aligned}
&a)\quad P_\theta^0(A_0,B_0) = P_\theta^2(A_0,B_0) = 0\\
&b)\quad P_\theta^1(A_0,B_0) = \gamma_0 p_1''(t)\\
&c)\quad P_\theta^0(A_i,B_0) = \gamma_i p_1''(t)\prod_{k=1}^{i}\left(\pi_{a_k} p_0'(t)\right) \qquad i\ge 1\\
&d)\quad P_\theta^1(A_i,B_0) = P_\theta^2(A_i,B_0) = 0, \qquad i\ge 1\\
&e)\quad P_\theta^2(A_0,B_j) = \gamma_0 p_{j+1}''(t)\prod_{k=1}^{j}\pi_{b_j}, \qquad j\ge 1\\
&f)\quad P_\theta^0(A_0,B_j) = P_\theta^1(A_0,B_j) = 0, \qquad j\ge 1
\end{aligned} \tag{7.71}$$

A 7.71 és 7.71 képletek helyessége könnyen ellenőrizhető, végiggondolva a lehetséges illesztéseket.

- (7.70a) A_{i-1} és B_j bármilyen illesztéséhez a_i-t hozzátéve egy olyan illesztést kapunk, amelyben A_i utolsó linkjének nincs leszármazottja. Eközben A-t eggyel megnöveltük
- (7.70b) A_{i-1} és B_{j-1} bármilyen illesztéséhez a_i-t és b_j-t hozzátéve egy olyan illesztést kapunk, amelyben A_i utolsó linkjének egy leszármazottja van. Ez vagy A utolsó, túlélt linkje, vagy ennek a valódi leszármazottja. Közben A-t szintén eggyel megnöveltük
- (7.70c) Ha egy illesztésben az utolsó linknek legalább két leszármazottja van, akkor az utolsó leszármazottat elhagyva egy olyan illesztést kapunk, amelyben az utolsó linknek legalább egy leszármazottja van. Az A szekvencia hossza nem változik meg, viszont ez a részlikelihood egy $\lambda\beta(t)$ szorzót kap a (4.4.11)-es képlet értelmében.
- (7.71a,b) Két üres szekvencia egyetlen módon illeszthető: egymás alá írjuk a két halhatatlan linket. Eszerint az utolsó linknek az illesztésben pontosan egy leszármazottja van.

- (7.71c,d) Egy nem üres szekvenciához egy üres szekvenciát egyféleképpen illeszthetünk. A halhatatlan link egyetlen leszármazottja az üres szekvencia halhatatlan linkje, minden halandó linknek — így az utolsónak is — nulla leszármazottja van.
- (7.71e,f) Egy üres szekvenciához egy nem üres szekvenciát egyféleképpen illeszthetünk, a leszármazott szekvencia összes linkje az üres szekvencia halhatatlan linkjének a leszármazottja. Így az utolsó linknek legalább két leszármazottja van.

Két szekvencia kapcsoltsági valószínűsége egyszerűen a

$$P(A,B|\theta) = \sum_{k=1}^{2} P_\theta^k(A,B) \tag{7.72}$$

képlettel adható meg.

A maximum likelihood paraméterek kiszámításához nem elegendő egyetlen paraméterre kiszámítani két szekvencia likelihoodját, meg kell keresni $P(A,B|\theta)$ függvény maximumát. A λ és μ paraméterek közül az egyik rögzíthető a

$$\lambda = \frac{\mu(n+m)}{n+m+2} \tag{7.73}$$

képlet segítségével. Ez a képlet közvetlenül adódik abból, hogy a minták számtani közepe az eloszlás várható értékének a maximum likelihood becslése.

7.2. A TKF92 modell

A Thorne-Kishino-Felsenstein modell (TKF91) nem enged meg többszörös beszúrásokat és törléseket. Az eredeti modell egy javított változatát egy évvel később publikálták, amely megenged többszörös beszúrásokat és törléseket (TKF92). Ez a modell ugyanazt a születési és halálozási folyamatot feltételezi, mint a TKF91 modell, de lehetőséget ad arra, hogy egy link ne egy, hanem tetszőleges számú karakterrel legyen asszociálva. Ekkor azonban a szekvencia széttörhetetlen fragmentumokból fog állni: ha egy link a születésekor több karakterrel asszociálódott, akkor nincs lehetőség arra, hogy ezek a karakterek külön-külön törlődjenek.

Bár a model biológus szemmel nézve nem realisztikus, mégis megmutatható, hogy ez a modell jobb, mint a TKF91 modell. Tekintsük például a következő három szekvencia illesztését:

```
A C G T C G T A T G
A C G T C - - - T G
A C G - - - - A T G
```

melyben a legfelső szekvencia az alatta levő két szekvencia őse. Ha hosszú beszúrásokat és törléseket megengedünk, akkor két törléssel keletkezhet a két szekvencia: az egyikből a `GTA` részstringet, a másikból a `TCGT` részstringet kell kivágni. Ha reverzibilis modellt szeretnénk felállítani, akkor megtehetjük, hogy az egyik modern szekvenciát tekintjük ősnek, a másikat a leszármazottnak, és az ősnek tekintett modern szekvencia és az ős közötti evolúciós utat megfordítjuk. Ezen az úton

a törlés megfordítása beszúrás lesz, így az ACGTCTG szekvenciából a GTA részstring beszúrásával, majd a TCGT részstring törlésével lehet eljutni az ACGATG szekvenciához.

Mivel a TKF91 modellben csak egyesével történhetnek a beszúrások és törlések, a TKF91 modell hét önálló besúrás-törlés eseménnyel tudja leírni ezt a tranzíciót. A TKF92 modellnek azonnban elég csak négy: a GT és A részstringek beszúrása után a TC és GT részstringek törlésével juthatunk el a másik szekvenciához.

A fragmentumok hossza r paraméterű geometriai eloszlást követ. Megmutatható, hogy a szekvenciák hosszának egyensúlyi eloszlása majdnem geometriai:

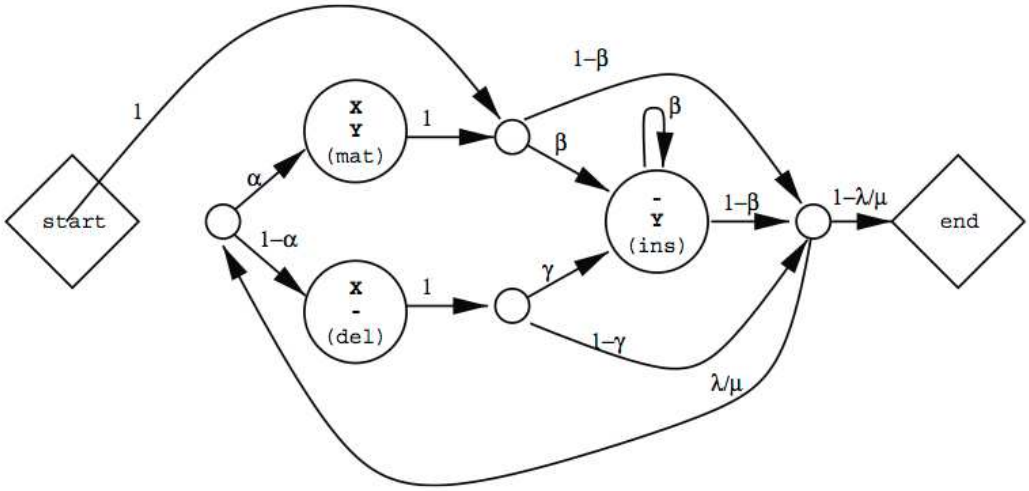
$$\gamma_0 = 1 - \frac{\lambda}{\mu} \tag{7.74}$$

$$\gamma_n = \left(1 - \frac{\lambda}{\mu}\right)\frac{\lambda}{\mu}(1-r)\left[\frac{\lambda}{\mu}(1-r) + r\right]^{n-1} \qquad n \geq 1 \tag{7.75}$$

Két szekvencia kapcsoltsági valószínűségének a kiszámolásához összegezni kell az összes lehetséges fragmentálódáson és az összes lehetséges tranzíción. A következő alfejezetben megmutatjuk, hogy a TKF modellek ekvivalensek páros rejtett Markov modellekkel, így a páros rejtett Markov modellek Forward algoritmusa alkalmazható a kapcsoltsági valószínűség kiszámolására.

7.3. A TKF modellek, mint páros rejtett Markov modellek

7.1. tétel A 7.1. ábrán látható, Σ ABC feletti szekvenciákat kibocsájtó páros rejtett Markov modell ekvivalens a TKF91 modellel, abban az értelemben, hogy létezik a páros rejtett Markov modell kibocsájtási utai és a Σ ABC feletti szekvenciák illesztései között egy valószínűségtartó bijekció. A páros rejtett Markov modellben a kibocsájtási valószínűségek egy karaktert kibocsájtó állapotok esetén a TKF91 modellben használt szubsztitúciós modell egyensúlyi eloszlását követik, az együttesen kibocsájtó állapot pedig $\pi(a)T(b|a,t)$ valószínűséggel bocsájtja ki az (a,b) párt, ahol π a szubsztitúciós modell egyensúlyi eloszlása, $T()$ pedig az átmeneti valószínűség a szubsztitúciós modellben. A szekvenciaillesztések valószínűségeit a TKF91 modellben kell számolni.



7.1. ábra A TKF91 modellel ekvivalens páros rejtett Markov modell. Az ugrási valószínűségek rövidítései: $\alpha = e^{-\mu t}$, $\beta = \lambda \frac{1-e^{-(\mu-\lambda)t}}{\mu - \lambda e^{-(\mu-\lambda)t}}$, $\gamma = \frac{1-e^{-\mu t} - \mu \frac{1-e^{-(\mu-\lambda)t}}{\mu - \lambda e^{-(\mu-\lambda)t}}}{1-e^{-\mu t}}$. A kibocsájtási valószínűségek egy szubsztitúciós modell egyensúlyi és kapcsoltsági valószínűségeivel egyeznek meg, részletesen lásd a szöveget. Az ürek körök nem emittáló ún. csendes állapotok (silent state, mute state).

Bizonyítás A rejtett Markov modell három állapota a három lehetséges szekvenciaillesztési oszlopnak felel meg: összeillesztés, beszúrás, törlés. Könnyen ellenőrizhető, hogy ezek bármelyikével kezdődhet egy kibocsájtási út, bármelyikével befejeződhet, és bármelyik állapotból bármelyikbe lehet ugrani, és lehetséges üres szekvenciaillesztést is készíteni. Így létezik egy természetes bijekció a kibocsájtási utak és szekvenciaillesztések között. Az alábbiakban ennek a bijekciónak a valószínűségtartását bizonyítjuk be.

A kapcsoltsági valószínűség az ősi szekvencia egyensúlyi állapotban vett valószínűsége szorozva az átmenet valószínűségével. Az átmenet valószínűsége a 7.1. fejezetben ismertetett mintázatok valószínűségeinek a szorzata. Először azt bizonyítjuk, hogy minden illesztésre az ősi szekvencia valószínűségét tartalmazza a kibocsájtási út valószínűsége. Valóban, az END állapotba való ugráskor van egy $1-\frac{\lambda}{\mu}$ ugrási valószínűség, és minden ősi karakter kibocsájtása előtt van egy $\frac{\lambda}{\mu}$ ugrási valószínűség. Az ősi szekvencia karaktereinek az egyensúlyi állapotban vett valószínűségeit az összeillesztés és törlés állapotok kibocsájtási valószínűségei tartalmazzák.

Ezután belátjuk, hogy a maradék ugrási valószínűségek éppen a megfelelő szekvenciaillesztési mintázatoknak a TKF91 modellben vett valószínűségeit adják meg. A halhatatlan link leszármazottjaira valóban $(1-\lambda\beta(t))(\lambda\beta(t))^{n-1}$ valószínűség jut: mire a START állapotból eljutunk az END előtti csendes állapotig, pontosan egyszer volt egy $(1-\lambda\beta(t))$ ugrási valószínűség, és annyiszor $\lambda\beta(t)$ ugrási valószínűség, ahány karaktert szúrtunk be a szekvenciaillesztés elejére.

Minden túlélő halandó link kap egy $e^{-\mu t}$ szorzót, majd abba a csendes állapotba jutunk, amelyből az előbb már megmutattuk, hogy $(1-\lambda\beta(t))(\lambda\beta(t))^{n-1}$ ugrási valószínűség generálódik, mire onnan az END előtti csendes állapotig eljutunk.

A legtrükkösebb a meghalt halandó linkek mintázatai valószínűségeinek a generálása. Először egy $1-e^{-\mu t}$ ugrási valószínűség jön be a kibocsájtási út valószínűségébe, de ez kiesik a következő ugrási valószínűségeknél. Valóban, ha a link kihalt, $\mu\beta(t)$ lesz az ugrási valószínűségek szorzata, ha pedig vannak leszármazottjai, akkor lesz egy $1-e^{-\mu t}-\mu\beta(t)$ szorzó az első két ugrási valószínűség szorzatából, minden további leszármazottra egy $\lambda\beta(t)$ ugrási valószínűség, végül az END előtti csendes állapotba jutva egy $(1-\lambda\beta(t))$ ugrási valószínűség. Ezek pont ki is adják a kérdéses mintázat valószínűségét.

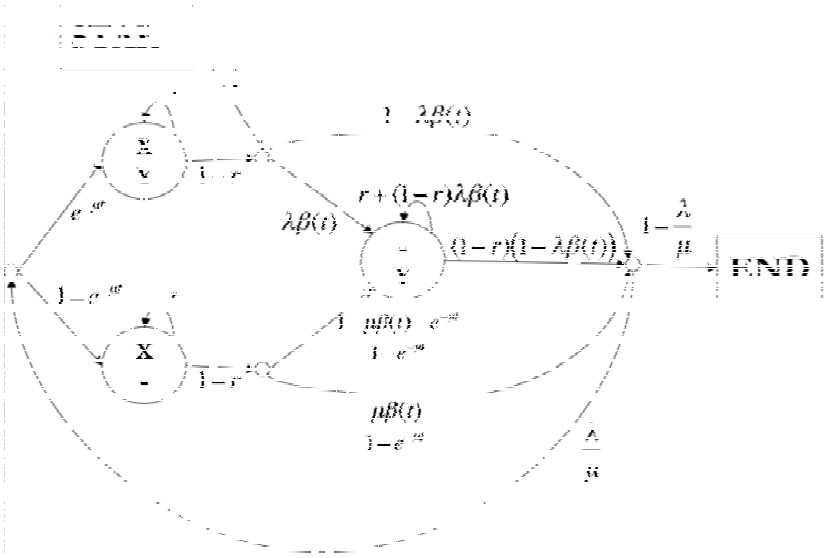
□

7.2. tétel A A 7.2. ábrán látható, Σ ABC feletti szekvenciákat kibocsájtó páros rejtett Markov modell ekvivalens a TKF92 modellel, abban az értelemben, hogy létezik a páros rejtett Markov modell kibocsájtási utai és a Σ ABC feletti szekvenciák illesztései között egy valószínűségtartó bijekció. A páros rejtett Markov modellben a kibocsájtási valószínűségek egy karaktert kibocsájtó állapotok esetén a TKF92 modellben használt szubsztitúciós modell egyensúlyi eloszlását követik, az együttesen kibocsájtó állapot pedig $\pi(a)T(b|a,t)$ valószínűséggel bocsájtja ki az (a,b) párt, ahol π a szubsztitúciós modell egyensúlyi eloszlása, $T()$ pedig az átmeneti valószínűség a szubsztitúciós modellben. A szekvenciaillesztések valószínűségeit a TKF92 modellben kell számolni, úgy, hogy összegzünk az ősi szekvencia összes lehetséges fragmentálódásán.

Bizonyítás A rejtett Markov modell három állapota a három lehetséges szekvenciaillesztési oszlopnak felel meg: összeillesztés, beszúrás, törlés. Könnyen ellenőrizhető, hogy ezek bármelyikével kezdődhet egy kibocsájtási út, bármelyikével befejeződhet, és bármelyik állapotból bármelyikbe lehet ugrani, és lehetséges üres szekvenciaillesztést is készíteni. Így létezik egy természetes bijekció a kibocsájtási utak és szekvenciaillesztések között. Az alábbiakban ennek a bijekciónak a valószínűségtartását bizonyítjuk be.

A páros rejtett Markov modell csak annziban különbözik a TKF91 modellel ekvivalens páros rejtett Markov modelltől, hogy mind az összeillesztés, mind a törlés állapot közvetlenül is tud önmagába ugrani r valószínűséggel, és 1-r valószínűséggel lép tovább, valamint a beszúrás állapot önmagába ugrásának a valószínűsége is más. Elég annyit belátni, hogy ezek a változtatások pontosan a fragmentáltságot írják le. Valóban, mind az összeillesztés, mind a törlés állapotnál az r és 1-r ugrási valószínűségek a megfelelő geometriai eloszlásban vett valószínűséget adják ki. A beszúrás állapotban a következő karakter r valószínűséggel tartozik ugyanahozz a fragmenthez, 1-r valószínűséggel pedig egy új fragmentet kezdünk. Az első fragmenthez tartozó 1-r szorzót az állapot elhagyásakor kapjuk meg.

□



7.2. ábra A TKF92 modellel ekvivalens páros rejtettMarkov modell. Az ugrási valószínűségekben szokásosan $\beta(t) = \frac{1-e^{-(\mu-\lambda)t}}{\mu-\lambda e^{-(\mu-\lambda)t}}$. A kibocsájtási valószínűségek egy szubsztitúciós modell egyensúlyi és kapcsoltsági valószínűségeivel egyeznek meg, részletesen lásd a szöveget. Az ürek körök nem emittáló ún. csendes állapotok (silent state, mute state).

Mivel mind a TKF91, mind a TKF92 modell ekvivalens egy-egy páros rejtett Markov modellel, a kapcsoltsági valószínűségek kiszámolhatóak a Forward algoritmussal is, a legvalószínűbb illeszés pedig a Viterbi algoritmussal. A TKF91 modell esetében a Forward algoritmusnál egy picit hatékonyabb algoritmus is létezik, ezt mutatjuk be a következő alfejezetben.

7.4. Az 1-állapot rekurzió

Az 1-állapot rekurzió (one-state recursion) elnevezés arra utal, hogy a TKF91 modellben a kapcsoltsági valószínűség kiszámolható úgy is, hogy nem osztjuk a lehetséges illesztéseket 3 halmazra, illetve a páros rejtett Markov modell 3 állapotára nem kell külön-külön számolni. Ezt precízen az alábbi tétel mondja ki.

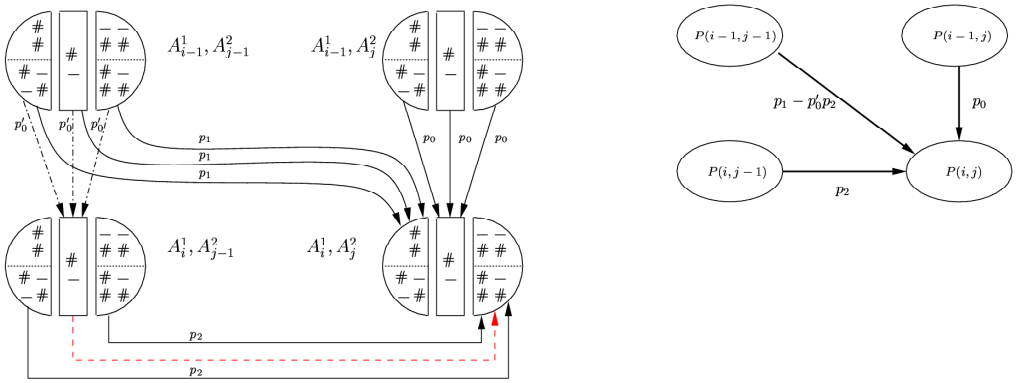
7.3. tétel Két szekvenciának a kapcsoltsági valószínűségét az alábbi dinamikus programozási rekurzióval is ki lehet számolni:

$$P(0,0) = \left(1-\frac{\lambda}{\mu}\right)\left(1-\lambda\beta(t)\right) \tag{7.76}$$

$$\begin{aligned} P(i,j) &= P(i-1,j)\lambda\beta(t)\pi(a_i) + P(i,j-1)\lambda\beta(t)\pi(b_j) + \\ &+P(i-1,j-1)\pi(a_i)\frac{\lambda}{\mu}\left[e^{-\mu t}(1-\lambda\beta(t))T(b_j \mid a_i,t) - \lambda\beta(t)\mu\beta(t)\pi(b_j)\right] \end{aligned} \tag{7.77}$$

ahol $P(i,j)$ az i és j hosszú prefixek kapcsoltsági valószínűségét adja meg.

Bizonyítás Tegyük fel, hogy az (i,j) indexpárra számolunk, lásd még 7.3 ábra is. Vegyük észre, hogy a 7.70 képletben megadott rekurzióban nem kellett az illesztések részhalmazait szétválasztani, amikor mindkét index eggyel változott, illetve amikor az első index változott eggyel (7.70a és 7.70b képletek). Egyetlen esetben van szükség az illesztések megkülönböztetésére, amikor a második index nő eggyel. Ekkor azokat az illesztéseket ki kell hagyni a rekurzióból, amelyekben az utolsó linknek nincs leszármazottja. De ezek pont úgy keletkeztek, hogy az $(i\text{-}1,j\text{-}1)$ indexű illesztésekben növeltük az első indexet. Így számolhatunk úgy is, hogy az összes illeszés halmazának a valószínűségére szorzunk rá $\lambda\beta(t)\pi(b_j)$-vel, majd az így elkövetett hibát javítjuk ki az $(i\text{-}1, j\text{-}1)$ indexű illesztések valószínűségeiből kiindulva.



7.3. ábra Az 1-állapot rekurzió bizonyításának a grafikus szemléltetése.

8. fejezet

RNS térszerkezetpredikció energiamodellekben

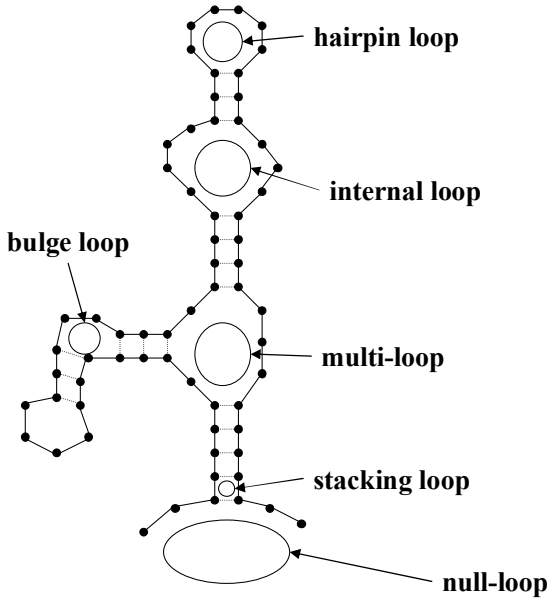
8.1. A Zucker-Tinoco energiamodell

A Zucker-Tinoco energiamodell az álcsomómentes RNS térszerkezetek szabadenergiáira ad meg egy értéket. Ebben a modellben minden térszerkezetet szétdarabolható körökre, és a térszerkezet szabadenergiája az egyes körökhöz rendelt szabadenergiák összege. Egy S térszerkezet szabadenergiáját $\Delta G(S)$ fogja jelölni, a Δ jelölés mutatja, hogy itt egy szabadenergia-különbségről van szó, $\Delta G(S)$ az S térszerkezet szabadenergiája és azon térszerkezet szabadenergiája közötti különbség, amelyben egyetlen bázispárosodás sincsen.

A körökre való bontáshoz először bevezetjük a bázispárok parciális rendezését.

Definíció $(i,j) < (i',j')$ ha $i > i'$ és $j' > j$. Azaz, egy (i,j) bázispár kisebb egy (i',j') bázispárnál, ha az (i,j) intervallumot magába foglalja az (i',j') intervallum.

Könnyű belátni, hogy egy álcsomómentes RNS térszerkezet bázispárjai részbenrendezett halmazt alkotnak. Ezen részbenrendezés segítségével definiálhatjuk a köröket, lásd még 8.1 ábra.



8.1. ábra Álcsomómentes RNS térszerkezet körökre való bontása a Zucker-Tinoco energiamodellben.

Definíció Egy álcsomómentes térszerkezet *null-loop*ja a maximális bázispárokból és azon pozíciókból áll, amelyek nincsenek benne egyetlen bázispár intervallumában.

Definíció Egy álcsomómentes térszerkezetben minden $(i, i+1, j-1, j)$ négyes *stacking-loop*ot alkot, amelyben mind (i,j), mind $(i+1,j-1)$ bázispár.

Definíció Egy álcsomómentes térszerkezetben minden (i, i+1, k, k+1, ... j-1, j) pozíciók, valamint minden (i, i+1, ... k'-1, k', j-1, j) pozíciók *bulge-loop*ot alkotnak, amelyre mind (i,j), mind (i+1,k) vagy (k', j-1) bázispárt alkot, valamint k és j, illetve i és k' közötti pozíciók egyike sem szerepel bázispárban.

Definíció Egy álcsomómentes térszerkezetben minden (i, i+1, ... k-1, k, l, l+1, ... j-1, j) pozíciók egy *internal-loop*ot alkotnak, amelyre mind (i,j), mind (k,l) bázispárt alkot, valamint mind az i és k, mind az l és j közötti pozíciók egyike sem vesz részt bázispárban.

Definíció Egy álcsomómentes térszerkezetben minden (i,j) bázispárra az i-től j-ig terjedő szakaszban azok a pozíciók, amelyek nem szerepelnek egyetlen, (i,j)-nél kisebb bázispárosodás intervallumában, valamint az (i,j) által fedett bázispárok (i,j)-vel együtt egy *multi-loop*-ot alkotnak, amennyiben az (i,j) által fedett bázispárok száma legalább 2.

Definíció Egy álcsomómentes térszerkezetben minden (i,j) bázispárra az i-től j-ig terjedő szakasz az (i,j) bázispárral egy *hairpin-loop*ot alkot, ha a kérdéses szakaszban egyik pozíció sem vesz részt bázispárban.

A loop-ok közül a null-loop és a multi-loop szabadenergiája fontos algoritmikai szempontból. A Zucker-Tinoco energiamodellnek egy egyszerűbb, itt tárgyalt változatában a null-loop szabadenergiája 0. A bonyolultabb modell feltételez egy ún. *dangling* energiát, amely a maximális bázispárban résztvevő nukleinsavak és a szomszédos nukleinsavak közötti kölcsönhatásból származó energia. A dangling energiák figyelembevétele az alább bemutatott dinamikus programozási algoritmust komplikáltabbá teszi, de a futási idő nagyságrendjén nem változtat.

Egy L multi-loop-ra számolt szabadenergia tag lineáris közelítése

$$\Delta G(L) = a + b \times l_s(L) + c \times l_d(L) + \Delta G_{dangling} \qquad (8.1)$$

ahol a, b, c konstansok, $l_s(L)$ a multi-loopban szereplő nem párosodó nukleinsavak száma, $l_d(L)$ a multi-loopban szereplő párosodó nukleinsav-párok száma. Mint a null-loop esetén, a dangling energiától itt is el fogunk tekinteni. A multi-loop energiáját jobban írná le a

$$\Delta G(L) = a + 6b + 1.75 \times \mathrm{RT} \ln(l_s(L)/6) + c \times l_d(L) + \Delta G_{dangling} \qquad (8.2)$$

képlet, ez utóbbi azonban az alább bemutatott dinamikus programozás futási idejét exponenciálisra növelné meg.

A többi kör energiája csak a benne szereplő bázispároktól függ, a továbbiakban a bázispárokban szerplő nukleinsavak indexeivel fogunk rájuk hivatkozni.

8.2. A Zucker-Sankoff algoritmus

A Zucker-Sankoff algoritmus egy RNS szekvenciára megadja a Zucker-Tinoco energiamodellben minimális szabadenergiájú térszerkezetet. Az algoritmusnak azt a változatát mutatjuk itt be, amelyik nem veszi figyelembe a dangling energiákat,

így a null-loop energiája 0 lesz, egy multi-loop energiáját a 8.1. képlet írja le (elhanyagolva belőle a dangling energiákat), az összes többi loop energiáját pedig táblázatból lehet kiolvasni. A Zucker-Sankoff algoritmus azon változatát mutatjuk itt be, amelyik minden térszerkezetet pontosan egyszer vesz figyelembe, ennek a következő alfejezetben lesz jelentősége.

A dinamikus programozás több táblázatot tölt ki, ezek közül a legmagasabb szinten levő (amelyik táblázatra csak önmaga hivatkozik, más táblázatok nem) az F^5 táblázat. $F^5(j)$ megadja az első pozíciótól a j-edikig terjedő résstring minimális szabadenergiájú térszerkezetének a szabadenergiáját. Az 5-ös felső index arra utal, hogy az RNS szekvenciát az 5' irányból kezdjük olvasni. A dinamikus programozási rekurzió:

$$F^5(j) = \min\left\{ F^5(j-1), \min_{2 \le l \le j-m-1} \left\{ F^5(l-1) + C(l,j) \right\} \right\} \tag{8.3}$$

ahol $C(l,j)$ megadja az l-től j-ig terjedő részstring azon térszerkezetei közül a minimális szabadenergiájúnak a szabadenergiáját, amelyben (l,j) bázispárt alkot. A képlet helyessége onnan látszik, hogy a null-loop szabadenergiája nulla, és az utolsó nukelinsav vagy nem bázispárosodik, vagy valamelyik l pozícióban levő nukelinsavval párosodik. Ha (l,j) bázispárt alkot, akkor nincs olyan loop, amelyik tartalmazna pozíciókat az 1,... l-1 tratományból és az l, ... j tartományból is.

Egy bázispár vagy egy hairpin-loopot, vagy egy stacking loopot, vagy egy bulge-loopot, vagy egy internal loopot vagy egy multi-loopot zár le. Így $C(i,j)$-re a dinamikus programozási rekurzió:

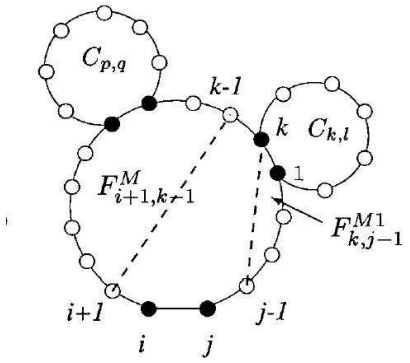
$$\begin{aligned} C(i,j) = \min\{ & H(i,j), \\ & C(i+1,j-1) + \mathit{Stacking}(i,i+1,j-1,j), \\ & \min_{\substack{i+1 \le p \le j-2 \\ p+1 \le q \le j-1 \\ p=i+1 \Rightarrow q \ne j-1}} \{ C(p,q) + L(i,p,q,j) \}, \\ & \min_{i+3 \le k \le j-2} \left\{ F^M(i+1,k-1) + F^{M1}(k,j-1) + a + c \right\} \end{aligned} \tag{8.4}$$

ahol $H(i,j)$ jelöli az (i,j) bázispár által lezárt hairpin loop szabadenergiáját, $\mathit{Stacking}(i,i+1,j-1,j)$ az adott bázispárok által alkotott stacking loop szabadenergiáját, $L(i,p,q,j)$ az (i,j) és (p,q) bázispárok által határolt bulge-loop vagy internal-loop szabadenergiáját, F^M és F^{M1} jelentését pedig lásd alább.

$F^M(i,j)$ értéke definíció szerint az i-edik pozíciótól a j-edik pozícióig terjedő részstring minimális szabadenergiájú térszerkezetének a szabadenergiája, amennyiben ez a térszerkezet legalább egy bázispárosodást tartlamaz, egyébként plusz végtelen, hozzáadva ehhez a null-loopban szereplő nem-párosodó nukleinsavak száma szorozva a 8.1 képletben szereplő b, továbbá hozzáadva a null-loopban szereplő bázispárok száma szorozva a 8.1. képletben szerplő c. $F^{M1}(i,j)$ értéke definíció szerint az i-edik pozíciótól a j-edik pozícióig terjedő részstring azon térszerkezetei közül a minimális szabadenergiájúnak a szabadenergiája, amelyben az i-edik pozícióban szereplő nukleinsav bázispárosodik, és a null-loopban ezen kívül nincs másik bázispár, hozzáadva ehhez a null-loopban szereplő nem-párosodó nukleinsavak száma szorozva a 8.1 képletben szereplő b, továbbá hozzáadva a 8.1. képletben szerplő c.

Amennyiben az adott részstringnek nincs olyan térszerkezete, amelyben az *i*-edik pozícióban szereplő nukleinsav párosodna, $F^{M1}(i,j)$ definíció szerint végtelen.

A 8.4 képlet utolsó sorában minden $F^{M}(i+1,k-1)+F^{M1}(k,j-1)+a+c$ összeg egy multi-loop szabadenergiája lesz, amint azt a 8.2. ábráról leolvasható. Továbbá a minimalizálásban minden *k*-ra más-más multi-loop szabadenergiája szerepel. Valóban, a multi-loop, amely legalább kettő olyan bázispárt tartalmaz, amelyet a záró (*i,j*) bázispár fed, szétbontható két részre. Az egyik rész legalább egy bázispárt tartlamaz, a másik pedig pontosan egyet. A kiválasztott bázispár a kör jobboldalán van, a multi-loop felvágása pedig ennek a bázispárnak a bal bázisa, ez a kettévágás egyértelmű.



8.2. ábra Magyarázat a multi-loop felbontáshoz. Részletesen lásd a szöveget.

$F^{M}(i,j)$ és $F^{M1}(i,j)$-re a rekurzió

$$F^{M1}(i,j)=\min_{i+1\le l\le j}\left\{C(i,l)+(j-l)b+c\right\} \tag{8.5}$$

$$F^{M}(i,j)=\min\left\{\min_{i+2\le k\le j-1}\left\{F^{M}(i,k-1)+F^{M1}(k,j)\right\},\ \min_{i\le k\le j-1}\left\{F^{M1}(k,j)+(k-i)b\right\}\right\} \tag{8.6}$$

A Zucker-Sankoff algoritmus a fenti rekurziókat számolja. Mint általában a dinamikus programozásban, a kitöltési fázisban a minimális szabadenergiájú térszerkezet szabadenergiáját kapjuk meg, magát a térszerkezetet a visszakeresési fázisban kapjuk meg.

Az algoritmusban a leglassab rész az internal loop és bulge loopok számolása. Ezt kiemelve a 8.4 képletből, jelöljük a minimális szabadenergiájú, internal looppal vagy bulge looppal lezáruló, *i*-től *j*-ig terjedő részstring térszerkezetének a szabadenergiáját *VBI*(*i,j*):

$$VBI(i,j)=\min_{\substack{i+1\le p\le j-2\\ p+1\le q\le j-1\\ p=i+1\Rightarrow q\ne j-1}}\left\{C(p,q)+L(i,p,q,j)\right\} \tag{8.7}$$

Mivel $O(n^2)$ (*i,j*) párra kell végignézni $O(n^2)$ esetet, a futási idő $O(n^4)$ lesz. Amennyiben *VBI* számolását le tudnánk csökkenteni $O(n^3)$-re, a teljes Zucker-Sankoff algoritmus futási ideje $O(n^3)$-re csökkenne le. A gyakorlatban használt energiafügvényekkel ez meg is tehető. Az internal loopokra használt energiafüggvény

$$L(i,p,q.j) = size(p-i+j-q-2) + \\ asymmetry(p-i-1, j-q-1) + stacking(i\cdot j) + stacking(p\cdot q) \tag{8.8}$$

továbbá

$$asymmetry(k+1,l+1) = asymmetry(k,l) + f(k+1) \tag{8.9}$$

Azaz, az aszimmetriából származó szabadenergia változása csak a loop méretétől függ, miközben a loop két része közötti különbség változatlan marad. Jelölje VBI'(*i,j,l*) az *i*-vel és *j*-vel lezárt, *l* hosszú belső loopot tartalmazó térszerkezetek közül az optimális energiájút. Ekkor:

$$\begin{aligned} VBI'(i,j,l) = \min\{ & VBI'(i+1,j-1,l-2) + size(l) - size(l-2) \\ & + f(l-2) + stacking(i\cdot j) - stacking(i+1\cdot j-1), \\ & C(i+1,j-l-1) + L(i,i+1,j-l-1,j), \\ & C(i+l+1,j-1) + L(i,i+l+1,j-1,j)\} \end{aligned} \tag{8.10}$$

Ez az algoritmus már köbös időben fut, és a *VBI* értékek kiolvasása egyszerűen az *l* indexen való végigjárással állapítható meg. Azonban a memóriaigény így felmegy négyzetesről köbösre. a memóriaigény visszaszorítható négyzetesre, úgy, hogy közben megőrizzük a köbös futási időt, ha figyelembe vesszük a következőt: Vegyük észre, hogy minden *VBI*'(*i,j,l*)-et csak egyszer nézünk meg. Ezért ezeket nem tároljuk, hanem előreküldjük, amikor éppen kéznél vannak. Mire a dinamikus programozási táblázatban *i*-hez és *j*-hez érünk, *VBI*(*i,j*)-re majdnem az összes lehetőséget megnéztük, kivéve a rövid ciklusokat. Az algoritmust a 8.3. ábrán mutatjuk be.

Algorithm 1. Computation of *VBI*(*i*, *j*).
```
/* VBI(i,j) has already been calculated for all loops of
size greater than 2; examine loops smaller than 2 */
for a = 1 to 2 do
    Let E be the energy of optimal loop of size a closed
    by i · j
    if E < VBI(i, j) then
        VBI(i, j) = E
    for b = 1 to min(i – 1, n – j) do

        /* Examine loops of size a + 2b with closing pair
        i – b · j + b.
        Invariant: E = VBI'(i – b + 1, j + b – 1,a +
        2(b – 1)) */
        E = E – size(a + 2b – 2) + f (a + 2b – 2)
            + size(a + 2b) + stacking(i – 1 · j + 1)
            – stacking(i · j)
        if a bulge closed by i – b · j + b of size a + 2b has
        energy lower than E then
            Let E be the energy of this bulge
        if E < VBI(i – b, j + b) then
            VBI(i – b, j + b) = E
    end for
end for
```

8.3. ábra Köbös futási idejű, nényzetes memóriaigényű algoritmus az internal loopok és bulge loopok energiáinak számolásához.

8.3. Az RNS térszerkezetek Boltzmann eloszlása. Az algebrai dinamikus programozás

Egy *S* struktúra előfordulási valószínűsége a Boltzmann eloszlásban:

$$P(S) \propto e^{\frac{-\Delta G(S)}{RT}} \tag{8.11}$$

Az arányossági tényező reciproka a partíciófüggvény, *Z*:

$$Z = \sum_{S} e^{\frac{-\Delta G(S)}{RT}} \tag{8.12}$$

Ha ki tudnánk számolni a partíciófüggvényt, akkor a minimális szabadenergiájú térszerkezet valószínűségét a Boltzmann eloszlásban pontosan meg tudnánk mondani. Ezt természetesen meg tudjuk tenni, mivel a Zucker-Sankoff algoritmusban minden térszerkezetet pontosan egyszer vettünk figyelembe. Pusztán annyit kell tenni, hogy minimalizálás helyett összeadni kell, összeadás helyett szorozni, a kezdeti energiákat pedig a megfelelő exponensre kell emelni.

Felmerülhet a kérdés, hogy szükséges-e újra implementálni a dinamikus programozási rekurziót. A Zucker-Sankoff féle dinamikus programozási rekurzió az egyik legbonyolultabb rekurzió, főleg, ha a fent említett köbös idejű gyorsítást is beépítjük az algoritmusba, továbbá a dangling energiákkal is törődünk, amivel itt most nem foglalkoztunk. Ha a dangling energiákat figyelembe vesszük, akkor további esetszétválasztást kell végezni, mert minden nem-párosodó nukleinsav csak egy bázispárral léphet kémiai kapcsolatba, amelyből a dangling energiák származnak. Ezért nyomon kell követni, hogy egy nem-párosodó nukleinsavat figyelembe vettünk-e már egy dangling energiában vagy nem. A dinamikus programozásban az egyes esetek közötti kapcsolat ugyanaz, csak a számolás más. Ha szét tudnánk választani az esetek közötti kapcsolatokat a konkrét számolástól, akkor a bonyolult részt csak egyszer kellene leprogramozni, és csak a számolásokra kellene külön függvényeket írni. Ezt a fajta programozási eljárást hívják *algebrai dinamikus programozás*nak.

Az algebrai dinamikus programozás legkönnyebben objektumorientált programozási nyelven valósítható meg. Ekkor a fenti dinamikus programozásban a dinamikus programozási táblázat egyes elemei nem egyszerű változók, hanem objektumok lesznek, a rekurzióban szereplő műveletek (összeadás, minimalizálás) pedig függvények, amelyeknek az argumentumai az adott objektumok, az értékei pedig a létrehozandó új objektumok. Az objektumoknak vannak privát változói, amelyek megfelelnek az egyes dinamikus programozási algoritmusok (energiaminimumot számoló, partíciófüggvényt számoló algoritmus) változóinak. A függvényekbe azt kell leírni, hogy az egyes változókkal mi történjen. Az a függvény, amely a Zucker-Sankoff algoritmusban az összeadás volt, a Zucker-Sankoff féle algoritmushoz használt változókat összeadja, a partíciófüggvényhez tartozó változókat összeszorozza, és a függvény visszatérő értékéhez tartozó objektum

megfelelő változóinak ezeket az értékeket adja. Az a függvény, amelyik a Zucker-Sankoff algoritmusban a minimalizálás volt, a Zucker-Sankoff féle algoritmushoz tartozó változóknak a minimumát veszi, a partíciófüggvényhez tartozó változókat összeadja, és a függvény visszatérő értékéhez tartozó objektum megfelelő változóinak ezeket az értékeket adja.

További kérdésként felmerülhet, hogy tudunk-e még többet kihozni ebből az ötletből? Ki tudjuk-e számolni a Boltzmann eloszlás várható értékét, varianciáját, és általánosan a *k*-adik momentumát? Erre a kérdésre is igen a válasz. A dinamikus programozásban nem közvetlenül a várható értéket, illetve varianciát kell számolni, hanem a

$$\sum_S \Delta G(S) e^{\frac{-\Delta G(S)}{RT}} \tag{8.13}$$

és

$$\sum_S \Delta G^2(S) e^{\frac{-\Delta G(S)}{RT}} \tag{8.14}$$

mennyiségeket, hiszen ezekből a várható érték és variancia már könnyen számolható:

$$E_B[\Delta G] := \frac{\sum_S \Delta G(S) e^{\frac{-\Delta G(S)}{RT}}}{Z} \tag{8.15}$$

$$V_B[\Delta G] := \frac{\sum_S \Delta G^2(S) e^{\frac{-\Delta G(S)}{RT}}}{Z} - E_B^2[\Delta G] \tag{8.16}$$

Ezek a mennyiségek viszont könnyen számolhatóak, hiszen ha az *S* térszerkezetek felbonthatóak S_1 és S_2 térszerkezetek úniójára, akkor

$$\begin{aligned}
&\sum_S \Delta G(S) e^{\frac{-\Delta G(S)}{RT}} = \sum_{S_1}\sum_{S_2} \Delta G(S_1 \cup S_2) e^{\frac{-\Delta G(S_1 \cup S_2)}{RT}} = \\
&= \sum_{S_1}\sum_{S_2} \left[\Delta G(S_1) + \Delta G(S_2)\right] e^{\frac{-\Delta G(S_1)}{RT}} e^{\frac{-\Delta G(S_2)}{RT}} = \\
&= \sum_{S_1} e^{\frac{-\Delta G(S_1)}{RT}} \sum_{S_2} \Delta G(S_2) e^{\frac{-\Delta G(S_2)}{RT}} + \sum_{S_2} e^{\frac{-\Delta G(S_2)}{RT}} \sum_{S_1} \Delta G(S_1) e^{\frac{-\Delta G(S_1)}{RT}}
\end{aligned} \tag{8.17}$$

Azaz, ha Z_i jelöli a partíciófüggvény számolásához tartozó privát változót, X_i pedig a $\sum_S \Delta G(S) e^{\frac{-\Delta G(S)}{RT}}$ mennyiség számolásához tartozó privát változót, akkor a számolandó mennyiség akkor, amikor az argumentumok az *i* és *j* indexű objektumok:

$$Z_i X_j + Z_j X_i \tag{8.18}$$

Hasonlóan megmutatható, hogy ha Y_i jelöli a $\sum_S \Delta G^2(S) e^{\frac{-\Delta G(S)}{RT}}$ mennyiség számolásához tartozó privát változót, akkor a számolandó mennyiség akkor, amikor az argumentumok az *i* és *j* indexű objektumok:

$$Z_i Y_j + 2X_i X_j + Z_j Y_i \tag{8.19}$$

Hasonló képleteket lehet megadni a *k*-adik momentum kiszámolásához, amelyhez először a $\sum_S \Delta G^k(S) e^{\frac{-\Delta G(S)}{RT}}$ mennyiséget határozzuk meg minden $i<k$ $\sum_S \Delta G^i(S) e^{\frac{-\Delta G(S)}{RT}}$ segítségével, majd ezekből határozhatjuk meg a *k*-adik momentumot.

9. fejezet

Sztochasztikus becslések elméleti számítástudományi alapjai

Az előző fejezetekben megismerkedtünk azokkal a bioinformatikai modellekkel, amelyekben a fontos kérdések a bemenő adat polinom függvénye idejében megválaszolhatóak. Láthattuk, hogy a hatékony számoláshoz az kell, hogy a modellek nagyon egyszerűek legyenek, és még a legegyszerűbb modellek esetén is komoly algoritmikai eszközöket kellett használni a hatékony számoláshoz. A bonyolultabb modellek biológus szemmel nézve realisztikusabbak, ugyanakkor a modell bonyolultságával a számolásigény is gyorsan növekszik. Nagyon hamar el lehet jutni arra szintre, ahol a hatékony számolás nem ismert, és elfogadott bonyolultságelméleti sejtéseket igaznak feltételezve, nem is lehetséges.

Felmerül a kérdés, hogy ha az egzakt számolás komputációs költsége nagy, lehetséges-e közelítő számolást gyorsan elvégezni. Továbbá, ez a közelítő számolás lehet sztochasztikus is, nem csak determinisztikus. Több okot is fel lehet sorolni, hogy miért elégedhetünk meg hatékony sztochasztikus becslésekkel:

- A modell nem tökéletes képe a valóságnak, csak modellezi azt. Felesleges a modellben pontosabban számolni, mint amennyire pontosan modellezi a modell a valóságot.
- Még ha a modell tökéletes képe is lenne a valóságnak, akkor is csak egy mintát veszünk a valóságból. A mintavételezés során is elkövetünk mintevételi hibát, ráadásul ez a hiba is sztochasztikus, nagy valószínűséggel kicsi a hiba, és kis valószínűséggel nagy. Ha a számolás során is elkövetünk egy sztochasztikus hibát, ami így viselkedik (nagy valószínűséggel kicsi a hiba és kis valószínűséggel nagy), és ennek a hibának a mértéke lényegesen kisebb a mintavételezési hibánál, akkor a teljes hibának csak elenyésző részét fogja kitenni a számolásból eredő hiba.

A sztochasztikus becsléseket is mintavételezéssel készítjük el, és mint az alábbiakban látni fogjuk a mintavételezés és becslés bonyolultságelméleti szemmel nézve nagyon gyakran ekvivalens problémák. A következő alfejezetben ennek az egzakt definícióit adjuk meg.

9.1. Fontosabb bonyolultságelméleti osztályok

Definíció Egy döntési probléma NP-beli, ha létezik olyan nem-determinisztikus Turing gép, amely minden problémabeli feladatot meg tud oldani a feladat polinom függvényének idejében.

Ezzel ekvivalens definíció, hogy egy probléma NP-beli, ha létezik olyan tanu, amely polinom időben ellenőrizhető. Pl. az a döntési kérdés, hogy létezik-e Hamilton út egy gráfban, NP-beli, hiszen minden Hamilton útról a gráf méretének polinom idejében eldönthető, hogy tényleg Hamilton út.

Definíció Egy döntési probléma P-beli, ha létezik olyan determinisztikus Turing gép, amely minden problémabeli feladatot meg tud oldani a feladat polinom függvényének idejében.

Például annak eldöntése, hogy létezik-e teljes párosítás egy gráfban, P-beli probléma. Nagyon sokszor olyan egész értékű optimalizálási feladatokra is azt mondjuk, hogy P-beliek, amelyekre a lehetséges értékek intervallumának nagysága csak $O(c^{\text{poly}(n)})$, ahol $c > 1$, és n a bemenő adat mennyisége. Ez alatt azt értjük, hogy minden k-ra P-beli probléma annak eldöntése, hogy létezik-e olyan megoldás, amelynek az értéke legalább/legfeljebb k. Könnyen belátható, hogy ha ezt minden k-ra meg tudjuk oldani polinom időben, akkor az optimalizálási problémát is meg tudjuk oldani hatékonyan bináris kereséssel.

Mivel minden determinisztikus Turing gép egyben nem-determinisztikus Turing gép is, ezért minden P-beli probléma egyben NP-beli is. Azt viszont nem tudjuk, hogy minden NP-beli probléma P-beli probléma-e. Az NP-beli problémáknak van egy részhalmaza, amely különösen nehéznek tűnik, és annyit tudunk, hogy ha ezek akármelyike P-beli, akkor P = NP, ha egyike sem P-beli, akkor viszont értelemszerűen P != NP. Ezeket a nehéz NP beli problémákat definiáljuk alább.

Definíció Egy A probléma polinomiálisan visszavezethető B problémára, ha minden A-beli feladat polinom időben megoldható B-beli feladatok megoldásainak segítségével. A számolási időbe bele kell számolni a B-beli feladatok megkonstruálásának idejét, de nem kell beleszámolni a B-beli feladatok megoldásával töltött időt.

Definíció Egy A probléma NP-nehéz, ha minden NP-beli probléma polinomiálisan visszavezethető A-ra. Egy probléma NP-teljes, ha NP-beli és NP-nehéz.

Definíció A SAT az a döntési probléma, hogy egy konjuktív normál formába felírt logikai kifejezés kielégíthető-e.

Tétel 9.1. A SAT NP-teljes probléma.

Bizonyítás (vázlat) Világos, hogy a SAT NP-ben van, hiszen ha a logikai változók bizonyos értékei kielégítenek egy konjuktív normál formát, akkor az ellenőrizhető polinom időben. Továbbá a SAT NP-nehéz probléma is. Ennek bizonyítása azon alapszik, hogy minden NP-beli problémához konstruálható egy nem-determinisztikus Turing gép, amely polinom időben megoldja azt. A nem-determinisztikus Turing gép működése minden adott feladatra felírható logikai formulákkal, ez a formula átírható konjuktív normál formába, és megmutatható, hogy a formula kielégíthető akkor és csak akkor, ha a döntési feladatra a válasz „igen”.

A bizonyítás következménye a következő erősebb tétel, amire a későbbiekben hivatkozni fogunk:

Tétel 9.2. Minden NP-beli döntési feladathoz konstruálható a feladat polinom idejében egy olyan konjuktív normál forma, amely kielégíthető akkor és csak akkor, ha a döntési feladatra a válasz „igen”.

Definíció Egy döntési probléma RP-beli (random polinom), ha létezik olyan polinom időben futó random algoritmus, amelyre ha a helyes válasz „nem”, akkor az algoritmus is 1 valószínűséggel „nem”-et fog mondani, míg ha a válasz „igen”, akkor a helyes válasz több, mint ½ valószínűséggel „igen”.

Sokáig a Miller-Rabin teszt volt ismert, mint RP-beli algoritmus egy természetes szám összetettségének a vizsgálatára. Ha egy szám nem összetett, akkor Miller-Rabin teszt minden esetben azt mondja, hogy a szám nem összetett, míg ha a szám összetett akkor az esetek legalább ¾ részében a Miller-Rabin teszt is ezt mondja. Pár éve három indiai matematikus megmutatta, hogy annak eldöntése, hogy egy természetes szám összetett, P-beli döntési probléma. A három döntési osztályról (P, RP, NP) a következő tételt ismert:

Tétel 9.3 $P \subseteq RP \subseteq NP$.

Nem tudjuk, hogy ebben a láncban bárhol is valódi tartalmazás van-e. Egy általánosan elfogadott sejtés a következő, amelyet mi is mindig feltételezünk a kurzus során:

Sejtés 9.4 P = RP != NP.

Azaz, a sztochasztika nem segít a döntési problémákban, ha egy probléma RP-ben van, akkor P-beli is. Ugyanakkor vannak nehéz döntési problémák, az NP-teljes problémákra nem létezik polinom futási idejű algoritmus. Mégegyszer hangsúlyozzuk, hogy mind az P = RP, mind a P != NP állítás csak sejtés, nincs bizonyítva.

Definíció Egy döntési probléma BPP-beli (Bounded Probabilistic Polynomial), ha létezik olyan random algoritmus, amelyik legalább 2/3 valószínűséggel „igen”-t mond, ha a tényleges válasz „igen” és legalább 2/3 valószínűséggel „nem”-et mond, ha tényleges válasz „nem”.

Mint látható a BPP osztály gyengébb, mint az RP, így nyilván

$P \subseteq RP \subseteq BPP \subseteq NP$

Papadimitriou tétele szerint a 9.4 sejtésnek mondana ellen, ha a BPP összemetszene az NP-teljes osztállyal, nevezetesen:

Tétel 9.5. NP-teljes $\cap$ BPP $\neq \emptyset \Rightarrow$ RP = NP.

Bizonyítás A tételt három lépésben bizonyítjuk. Először azt mutatjuk meg, hogy (1) ha NP-teljes $\cap$ BPP $\neq \emptyset$, akkor BPP = NP. Ekkor létezik a SAT-ra is BPP algoritmust, és (2) megmutatjuk, hogy ekkor SAT benne van RP-ben is. Végül azt látjuk be, hogy (3) ha SAT RP-ben van, akkor NP = RP.

(1) Ehhez azt kell belátni, hogy az NP-teljességet megadó polinom reducibilitás polinom reducibilitást ad BPP-re is. Ehhez először azt kell észrevenni, hogy ha egy BPP algoritmust sokszor megismételünk, és a többségi választ fogadjuk el, akkor exponenciálisan csökken a valószínűsége annak, hogy rossz választ adunk. Valóban,

ha $2k$-szor ismételjük meg az algoritmust, akkor annak a valószínűsége, hogy rossz választ adunk

$$P(\text{rossz válasz}) = \sum_{i=k}^{2k} \binom{2k}{i} \left(\frac{1}{3}\right)^i \left(\frac{2}{3}\right)^{2k-i} \le (k+1) \binom{2k}{k} \left(\frac{1}{3}\right)^k \left(\frac{2}{3}\right)^k \tag{9.1}$$

Felhasználva a Stirling formulát, kapjuk, hogy alkalmas c konstansokra

$$P(\text{rossz válasz}) = (k+1)c_1 \frac{4^k}{\sqrt{k}} \left(\frac{1}{3}\right)^k \left(\frac{2}{3}\right)^k \le (k+1)c_2 \left(\frac{8}{9}\right)^k \le c\left(\frac{9}{10}\right)^k \tag{9.2}$$

Ha a polinom reducibilitás során m feladatot kell megoldani, mindegyiket annyiszor oldjuk meg BPP-vel, hogy a hiba valószínűsége kisebb legyen $1/3m$-nél. Ekkor annak a valószínűsége, hogy egyszer sem hibázunk az eljárás során legalább

$$\left(1 - \frac{1}{3m}\right)^m \ge \frac{2}{3} \tag{9.3}$$

De ahhoz, hogy a hiba valószínűsége kisebb legyen $1/3m$-nél, elég $O(\log(m))$-szer megismételni a feladatot, hiszen annak a valószínűsége, hogy a többségi döntés hibás választ ad, exponenciálisan csökken az ismétlések számával. Ugyanakkor m polinom függvénye kell, hogy legyen a bemenő feladat méretének, a polinom reducibilitás miatt.

(2) Ha BPP = NP, akkor a SAT-ra is van BPP algoritmus. De ha van a SAT-ra BPP algoritmus, akkor megtehetjük a következőt. Adott konjuktív normál formára a normál formából kitörlünk minden olyan klózt, amelyben x_1 logikai változó negálás nélkül szerepel, és kitöröljük x_1-et minden olyan klózból, amelyben negálva szerepel. Ezzel egy másik konjuktív normál formát kapunk, amelyik ha kielégíthető, akkor az eredeti normál forma is kielégíthető úgy, hogy x_1-nek igaz értéket adunk. Ha viszont nem elégíthető ki, akkor azokat a klózokat töröljük ki, amelykben x_1 negálva szerepel és kihúzzuk x_1-et azokból a klózokból, amelyben negálás nélkül szerepel. Ha ez a konjuktív normál forma kielégíthető, akkor az eredeti is, úgy, hogy x_1-nek hamis értéket adunk. Ilyen módon rekurzívan megadhatjuk egy megoldását a normál formának, amennyiben az kielégíthető. Ha az egyes iterációkat BPP algoritmussal tudjuk megoldani, akkor ezeket annyiszor kell ismételni, és a többségi választ elfogadni, hogy annak a valószínűsége, hogy egyszer sem hibázunk a rekurzió során legalább ½ legyen. A megkonstruált megoldást determinisztikusan leellenőrizzük, és akkor válaszolunk „igen”-nel a feladatra, ha a kapott eredmény tényleg megoldás. Így ha a konjuktív normál forma nem kielégíthető, 1 valószínűséggel „nem” választ fogunk adni, ha kielégíthető, akkor pedig legalább ½ valószínűséggel „igen”-t fogunk mondani. A teljes futási idő csak polinom mérete lesz a feladat méretének, hiszen a BPP-k ismétléseinek a száma csak $O(\log(m))$, ahol m a logikai változók száma, és a rekurzió során kapott konjuktív normál formák rövidebbek, mint az eredeti feladat.

(3) Viszont ha SAT RP-ben van, akkor RP = NP, a 9.2 tételből adódóan.

□

Az eldöntési problémák legfontosabb osztályainak ismertetése után most áttérünk a leszámolási problémákra.

Definíció A #P-beli leszámolási problémák az NP-beli döntési problémák polinom időben ellenőrizhető tanui számára kérdeznek rá.

Például #P-beli leszámolási probléma az, hogy hány teljes párosítás létezik. A #P-beli problémák esetében is vannak könnyű és nehéznek tűnő problémák.

Definíció Egy #P-beli probléma FP-ben van (függvény polinom), ha létezik olyan algoritmus, amely polinom időben kiszámolja a tanuk számát.

Definíció Egy probléma #P-nehéz, ha minden #P-beli probléma polinom reducibilis rá. Egy probléma #P-teljes, ha #P-beli és #P-nehéz.

Tétel 9.6. Létezik #P-teljes probléma.

Például a fent említett teljes párosítások száma #P-teljes probléma. A #P-teljes problémák ugyanazt a szerepet játszák, mint az NP-teljes problémák a döntési problémáknál: ha létezne polinom idejű algoritmus egyetlen #P-teljes problémára, akkor #P benne lenne FP-ben. Az alábbi tétel azt mondja, hogy ez ellentmondana a 9.4 sejtésünknek.

Tétel 9.7. $\#P \subseteq FP \Rightarrow P = NP$.

Bizonyítás Ha minden #P-beli problémát meg tudnánk oldani polinom időben, akkor bármely NP-teljes problémára ki tudnánk számolni a tanuk számát polinom időben. De ekkor az adott NP-teljes problémát is meg tudnánk oldani polinom időben, hiszen ha a tanuk száma 0, akkor a válasz a döntési problémára „nem”, egyébként a válasz az, hogy „igen”. Egy polinom futási idejű algoritmus egy NP-teljes problémára már indikálja azt, hogy P = NP.

□

Definíció Egy #P-beli leszámolási probléma FPRAS-ban van (Fully Polynomial Randomized Approximation Scheme), ha létezik rá olyan sztochasztikus algoritmus, amely minden problémabeli feladatra és ε, $\delta > 0$-ra megbecsüli a tanuk számát, úgy, hogy teljesüljön a

$$P\left(\frac{f}{1+\varepsilon} \le \hat{f} \le f(1+\varepsilon)\right) \ge 1-\delta \qquad (9.4)$$

egyenlőtlenség, ahol f a tanuk tényleges száma, $\hat{f}$ a becslés, és az algoritmus futási ideje polinom függvénye mind a feladat méretének, mind $1/\varepsilon$-nak, mind $-\log(\delta)$-nak.

Vannak olyan leszámolási problémák, amelyre nem remélhetünk FPRAS algoritmust, feltételezve a 9.4. sejtést. A következő tétel ezt mondja ki:

Tétel 9.8. Ha bármely NP-teljes döntési probléma tanui számára létezik FPRAS algoritmus, akkor RP = NP.

Bizonyítás Tegyük fel, hogy létezik ilyen A NP-teljes probléma és FPRAS a tanui számára. Ekkor állítsuk ε-t ½-re, δ pedig legyen 1/3. Ekkor az FPRAS egy BPP algoritmus lesz a döntési problémára, hiszen ha a válasz „nem”, akkor a tanuk száma 0, amit az FPRAS is 0-nak fog becsülni legalább $1 - \delta$ valószínűséggel, viszont ha a tanuk száma legalább 1, akkor az FPRAS legfeljebb 1/3 valószínűséggel fog 2/3-nál kisebb értéket adni. Így az az algoritmus, amely az FPRAS 0 kimenetére „nem”-et mond, minden más kimenetre pedig „igen”-t, BPP-beli algorimus lesz. De ekkor NP = RP, Papadimitrou tétele (9.5 tétel) miatt.

□

Ugyanakkor igaz a következő tétel is:

Tétel 9.9. #P-teljes $\cap$ FPRAS $\neq \emptyset$.

Például a fent említett teljes párosítások száma egyrészt #P-teljes, másrészt benne van FPRAS-ban is, így benne van a kettő metszetében is. A tétel azonban nem mondja azt, hogy #P egyenlő lenne FPRAS-szal, ugyanis a #P-teljességhez használt polinomiális visszavezethetőség nem őrzi meg az FPRAS algoritmust. Valóban, az FPRAS-ban nagy valószínűséggel kis relatív hibával kell megmondani a tanuk számát. A relatív hiba azonban nem őrződik meg a négy alapművelet közül a kivonásnál, így ha a polinomiális visszavezethetőség a pontos számolásnál tartalmaz kivonást, akkor nem használhatjuk ezt a redukciót az FPRAS során. Enumeratív kombinatorikában az egyik leggyakoribb technika a logikai szita, amely tartalmaz kivonásokat is.

A #P-beli problémák egy részhalmazára a tanuk becslése és a tanukból való közel egyenletes mintavételezés gyakorlatilag ugyanaz a feladat, ezt mutatjuk be alább.

Definíció Egy #P-beli probléma önhasonló leszámolási probléma, ha mind a feladatokat, mind a tanukat le lehet írni egy Σ ABC feletti véges hosszú stringek segítségével, amelyre a következők teljesülnek:

- Bevezetünk egy R relációt Σ^*-on. xRy akkor és csak akkor, ha x feladatnak y tanuja. Feltesszük, hogy $|y| = O(\text{poly}(|x|))$.
- Minden x feladat-ra megadható egy Σ_1 ABC, úgy, hogy $|\Sigma_1| = O(\text{poly}(|x|))$, $\Sigma_1 \subseteq \Sigma^*$, és x minden y tanuja egyben eleme Σ_1^*-nak is.
- Minden x-re és $w \in \Sigma_1^*$-ra létezik egy x', $|x'| \leq O(\text{poly}(|x|))$, úgy, hogy $x'Rz$ akkor és csak akkor, ha $xRwz$, ahol wz a w és z stringek konkatenációját jelöli.

Azaz egy probléma akkor önhasonló, ha minden x feladatának minden w parciális megoldásához létezik egy x' feladat, amely leírása nem hosszabb lényegesen x leírásánál, és a w parciális megoldások befejezései éppen x' megoldásai.

Bár a definíció eléggé absztrakt, látni fogjuk, hogy az önhasonló leszámolási problémák egészen természetesek. Önhasonló leszámolási probléma például egy gráf teljes párosításainak a száma, egy részben rendezett halmaz teljes rendezéseinek a száma, stb.

Definíció Egy #P-beli probléma FPAUS-ban van (Fully Polynomial Almost Uniform Sampler), ha minden feladatára és $\delta > 0$-ra létezik olyan algoritmus, amely egy random, polinom időben ellenőrizhető tanut ad meg a π eloszlásból, melyre teljesül, hogy

$$d_{TV}(\pi, U) \leq \delta \tag{9.5}$$

ahol d_{TV} jelöli a variációs távolságot, U a polinom időben ellenőrizhető tanuk egyenletes eloszlását, és az algoritmus futási ideje polinom függvénye mind feladat méretének, mind -log(δ)-nak.

Önhasonló problémákra a random mintavételezés és a tanuk számának becslése egyforma nehéz problémák, ahogy ezt a következő tétel is kimondja.

Tétel 9.10 (Jerrum, Vailant, Vazirani) Bármely #X önhasonló leszámolási problémára létezik #X-re FPAUS algoritmus akkor és csak akkor, ha létezik #X-re FPRAS algoritmus.

A tétel bizonyítása konstruktív, azaz egy FPAUS algoritmusból meg lehet konstruálni az FPRAS algoritmust, és egy FPRAS algoritmusból meg lehet konstruálni egy FPAUS-t.

Az önhasonló leszámolási problémák azért is érdekesek, mert minden önhasonló probléma vagy nagyon jól approximálható, vagy csak nagyon rosszul, mint azt a következő tétel kimondja.

Tétel 9.11. (Sinclair és Jerrum) Ha egy önhasonló leszámolási problémára létezik polinom faktorú determinisztikus approximáció, akkor létezik rá FPRAS is.

Önhasonló leszámolási problémákra általában FPAUS-t adunk meg, ami bizonyítja azt, hogy létezik rá FPRAS is. Az FPAUS legtöbbször Markov láncokkal van megadva, ha egy Markov lánc a tanuk terén bolyong, és gyorsan konvergál az egyenletes eloszláshoz, valamint egy lépés a Markov láncban kevés számolást igényel, akkor könnyen konstruálható belőle FPAUS: adott lépésszám utáni állapota a Markov láncnak olyan eloszlásból fog származni, amelyre teljesül a 9.5. egyenlőtlenség. A Markov láncok konvergenciasebességének az elméletét a következő alfejezetben ismertetjük.

9.2. Markov láncok konvergenciasebessége

Az alábbiakban minden Markov lánc véges halmazon bolyong, diszkrét időben, ezt nem tüntetjük fel a továbbiakban.

Definíció Egy Markov lánc gráfja egy irányított gráf, amelynek pontjai a Markov lánc lehetséges állapotai, és egy u csúcsból akkor és csak akkor megy v-be él, ha az u-ból v-be való átmenet valószínűsége nem 0.

Definíció Egy Markov lánc irreducibilis, ha a gráfjában bármely két pont között megy irányított út.

Definíció Egy Markov lánc aperiodikus, ha a gráfjában az irányított körök hosszainak legnagyobb közös osztója 1.

Tétel 9.12. Ha egy Markov lánc irreducibilis, aperiodikus és reverzibilis, akkor a részletes egyensúlyban szereplő eloszlás globálisan stabilis lesz.

Észrevétel 9.13. Egy irreducibilis, aperiodikus, reverzibilis Markov lánc minden sajátértéke valós, és abszolút értékben nem nagyobb 1-nél. A legnagyobb sajátérték 1, egyszeres algebrai és geometriai multiplicitással, az összes többi ennél abszolútértékben kisebb.

Bizonyítás A lánc átmeneti gráfja az 5. fejezetben ismertetett technikával szimmetrikus valós mátrixszá alakítható, így minden sajátértéke valós. Az észrevétel többi része a Perron-Frobenius tételből adódik.

Az alábbiakban arra adunk tételeket, hogy milyen gyors a konvergencia a stacionárius eloszláshoz.

Definíció Egy *r* állapotú Markov lánc második legnagyobb sajátérték modulusa (Second Largest Eigenvalue Modulus, SLEM)

$$\rho = \max\left\{\lambda_2, |\lambda_r|\right\} \tag{9.6}$$

Definíció Egy Markov lánc relaxációs ideje

$$\tau_i(\varepsilon) := \min\left\{n_0 \in \mathrm{N} : d_{TV}\left(P^n \delta_i, \pi\right) \le \varepsilon \ \forall n \ge n_0\right\} \tag{9.7}$$

Tétel 9.14

$$\tau_i(\varepsilon) \le \frac{1}{1-\rho}\left(\ln\left(\frac{1}{\pi(i)}\right) + \ln\left(\frac{1}{\varepsilon}\right)\right) \tag{9.8}$$

$$\max_{i \in I}\left\{\tau_i(\varepsilon)\right\} \ge \frac{\rho}{2(1-\rho)}\ln\left(\frac{1}{2\varepsilon}\right) \tag{9.9}$$

Az alábbiakban Markov láncon nem egyetlen Markov láncot fogunk érteni, hanem Markov láncok egy osztályát, amely egy adott probléma minden feladatához tartalmaz egy Markov láncot. A konvergenciasebességet a feladat méretének függvényében szeretnénk mérni, és az alábbi tételekből az derül ki, hogy a kulcskérdés az, hogy a második legnagyobb sajátérték a feladat méretének milyen függvénye sebességgel konvergál az 1-hez.

Ha egy NP-beli probléma minden *x* feladatára egy Markov lánc a polinom időben ellenőrizhető tanuk egyenletes eloszlásához konvergál, akkor a 9.8. képletben szereplő $\ln\left(\frac{1}{\pi(i)}\right)$ kifejezés $O(\mathrm{poly}(|x|))$ lesz. Így a relaxációs idő polinom függvénye lesz mind a probléma méretének, mind a kívánt variációs távolság logaritmusa -1-szeresének, amennyiben $\frac{1}{1-\rho}$ polinomiális függvénye a probléma méretének. Most megmutatjuk, hogy a SLEM helyett elég a második legnagyobb sajátértéket tekinteni.

Definíció Egy **T** átmeneti mátrixú Markov lánc lusta változata (lazy Markov chain) a $\frac{\mathbf{T}+\mathbf{I}}{2}$ átmeneti mátrixú Markov lánc, ahol **I** az identikus mátrix.

Egy ilyen Markov láncot könnyen előűllíthatunk, ha minden lépésben ½ valószínűséggel helybenmaradunk, ½ valószínűséggel pedig úgy lépünk, ahogy az eredeti Markov láncban léptünk. A lusta Markov láncok gyakorlati hasznát az alábbi észerevétel adja meg.

Észrevétel 9.15. Egy irreducibilis, reverzibilis Markov lánc lusta változata irreducibilis, aperiodikus és reverzibilis lesz, ugyanazzal a stacionárius eloszlással, ugyanazokkal a sajátvektorokkal, és a sajátértékekre teljesül a

$$\lambda'_i = \frac{\lambda_i + 1}{2} \tag{9.10}$$

egyenlőség, ahol λ'_i a lusta Markov lánc *i*-edik sajátértéke, λ_i pedig az eredeti Markov lánc *i*-edik sajátértéke. Speciálisan, a lusta Markov lánc minden sajátértéke nemnegatív valós lesz, és

$$\frac{1}{1-\rho'} = \frac{2}{1-\lambda_2} \tag{9.11}$$

ahol ρ' a lusta Markov lánc SLEM-je, λ_2 pedig az eredeti Markov lánc második legnagyobb sajátértéke.

Bizonyítás Minden pozitív átmeneti valószínűség megmarad pozitívak, így a lusta Markov lánc is irreducibilis lesz, ha az eredeti is az volt. Mivel lesznek 1 hosszú hurkok a lusta Markov láncban, a lusta Markov lánc aperiodikus is lesz. Minden $i \neq j$-re mind az *i*-ből *j*-be, mind a *j*-ből *i*-be történő átmeneti valószínűségek a felükre csökkennek, így a reverzibilitás is megmarad. Minden sajátérték-sajátvektor párra fennáll a

$$\frac{\mathbf{T}+\mathbf{I}}{2}\mathbf{v}_i = \frac{\lambda_i + 1}{2}\mathbf{v}_i \tag{9.12}$$

egyenlőség is. Mivel az eredeti Markov lánc minden sajátértéke valós volt és abszolút értékben 1 vagy kisebb, a lusta Markov lánc minden sajátértéke nem-negatív lesz. Továbbá a lusta Markov láncnak a SLEM-je a második legnagyobb sajátértéke lesz, amely kétszer olyan közel lesz az 1-hez, mint az eredeti Markov lánc második legnagyobb sajátértéke.

□

A második legnagyobb sajátértéknek az 1-től való távolságát *spektrális rés*-nek hívják.

Tétel 9.16. Ha egy P-beli döntési problémához lehet egy olyan irreducibilis, reverzibilis Markov láncot készíteni, amely a polinom időben ellenőrizhető tanuk egyenletes eloszlásához konvergál, a spektrális résének a reciproka polinomiálisan növekszik a feladat méretével, egy lépés a Markov láncban a feladat méretének

polinom idejében megtehető, és a feladathoz egy tanút meg lehet polinom időben konstruálni, akkor a P-beli problémához tartozó #P-beli feladatra létezik FPAUS.

Bizonyítás A Markov lánc lusta változata aperiodikus lesz, a stacionárius állapota globálisan stabilis, és teljesül rá a 9.11 képlet. A relaxációs idő teljesen polinomiális lesz, azaz polinomiális mind a feladat méretében, mind a kívánt variációs távolság logaritmusa -1-szeresében. Így a következőképpen definiált mintavételezés FPAUS lesz:

- Vegyünk egy tetszőleges tanut. Ez polinom időben megadható.
- Ebből az állapotból kiindulva a lusta Markov láncban lépjünk

$$\frac{1}{1-\rho}\left(\ln\left(\frac{1}{\pi(i)}\right)+\ln\left(\frac{1}{\varepsilon}\right)\right) \tag{9.13}$$

  lépést. Ez poly($|x|$,-log(ε)) időben megtehető
- A random tanu legyen a lusta Markov lánc állapota az adott lépésszám után

□

Persze sokszor csak polinom felső becslésünk van a tanuk számának a logaritmusára, de ilyen felső becslések nagyon természetesek. Pl. egy k változós 2SAT feladat tanui a számának felső becslése 2^k, egy n pontú gráf teljes párosításai számának felső becslése a K_n teljes gráf teljes párosításainak a száma, stb. Az alábbiakban a spektrális rés reciprokára adunk meg jól használható becsléseket.

Definíció Egy Markov lánc állapotterében egy S részhalmaz ergodikus folyama

$$F(S) := \sum_{i\in S, j\in\bar{S}} \pi(i)T(j|i) \tag{9.14}$$

Definíció Egy Markov lánc konduktanciája

$$\Phi := \min\left\{\frac{F(S)}{\pi(S)}\middle| S\subset I, 0<\pi(S)\le\frac{1}{2}\right\} \tag{9.15}$$

ahol I az állapottér, $\pi(S)$ pedig az S részhalmaz valószínűsége a stacionárius állapotban. Azaz a konduktancia a feltételes kifolyási sebesség (mi a valószínűsége, hogy a következő lépésben elhagyjuk az S halmazt, feltéve, hogy S-ben vagyunk stacionárius állapotban) minimuma az összes olyan részhalmazon, amelynek a valószínűsége legfeljebb ½.

Tétel 9.17. A második legnagyobb sajátértékre teljesül a

$$1-2\Phi\le\lambda_2\le 1-\frac{\Phi^2}{2} \tag{9.16}$$

egyenlőtlenség.

A második legnagyobb sajátértékre hatékony felső becslést lehet adni a Poincarré koefficiens segítségével is. Számos változata van a tételnek, ebből kettőt mutatunk be:

Tétel 9.18. Legyen adva egy Γ útvonalrendszer, amely a Markov lánc gráfjában minden (i,j) párra tartalmaz pontosan egy irányított utat i-ből j-be. Egy út Q-mértékét definiáljuk a kapacitások reciprokainak összegeként, azaz

$$\left|\gamma_{ij}\right|_Q := \sum_{e\in\gamma_{ij}} \frac{1}{Q(e)} \tag{9.17}$$

ahol az $e = (x,y)$ él kapacitása

$$Q(e) := \pi(x)T\left(y|x\right) \tag{9.18}$$

Ekkor a Γ útvonalrendszer Poincarré koefficiense

$$\kappa := \kappa(\Gamma) := \max_e \sum_{\gamma_{ij}|e\in\gamma_{ij}} \left|\gamma_{ij}\right|_Q \pi(i)\pi(j) \tag{9.19}$$

melyre teljesül a

$$\lambda_2 \le 1 - \frac{1}{\kappa} \tag{9.20}$$

egyenlőtlenség.

Tétel 9.19. Legyen adva egy Γ sztochasztikus útvonalrendszer, amely a Markov lánc gráfjában minden (i,j) párra tartalmaz irányított utak egy eloszlását i-ből j-be. Az i-ből j-be menő utak eloszlását jelölje Γ_{ij}, egy partikuláris, i-ből j-be menő γ út valószínűségét $p_{ij}(\gamma)$ jelöli, az út hosszát (benne levő élek számát) $|\gamma|$. Ekkor definiáljuk a sztochasztikus Poincarré koefficienst:

$$\kappa := \kappa(\Gamma) := \max_e \sum_{(x,y)} \sum_{\gamma\in\Gamma_{xy}|e\in\gamma} \pi(x)\pi(y)p_{xy}(\gamma)\frac{|\gamma|}{Q(e)} \tag{9.21}$$

melyre teljesül a

$$\lambda_2 \le 1 - \frac{1}{\kappa} \tag{9.22}$$

egyenlőtlenség.

10. fejezet

Sztochasztikus mintavételezések

10.1. Az elutasítás (rejection) metódus

Adott egy π eloszlás, amelyre bármely x pontban $\pi(x)$ kiszámolható egy ismeretlen normalizációs konstans erejéig, azaz egy

$$f(x) \propto \pi(x) \tag{10.1}$$

függvény kiszámolható minden pontban. Továbbá van egy p eloszlás, amelyből tudunk mintavételezni, egy g függvény, amely arányos p-vel és kiszámolható minden x pontban, és ismert egy c ún. boritékoló konstans (enveloping constant) úgy, hogy minden x-re

$$f(x) \le cg(x) \tag{10.2}$$

Az elutasítás metódus a következő lépésekből áll:

- Veszünk egy random x-et p-ből.
- Vesszünk egy random u-t az egyenletes eloszlásból a [0,1] intervallumon. Ha

$$ucg(x) \le f(x) \tag{10.3}$$

akkor elfogadjuk x-et, egyébként elutasítjuk, és újrakezdjük a procedúrát.

Tétel 10.1. Az elutasítás metódussal generált random számok a π eloszlást követik.

Bizonyítás Jelölje A azt az eseményt, hogy elfogadjuk az első pontban generált értéket. A Bayes tételből adódóan

$$P(x|A) = \frac{P(A|x)P(x)}{P(A)} \tag{10.4}$$

A jobb oldalon szereplő valószínűségeket könnyen kiszámolhatjuk:

$$P(A|x) = \frac{f(x)}{cg(x)} \tag{10.5}$$

$$P(x) = p(x) \tag{10.6}$$

$$\begin{aligned} P(A) &= \int_{x'} P(A|x')P(x')dx' = \int_{x'} \frac{f(x')}{cg(x')} p(x')dx' = \\ &= \int_{x'} \frac{z_1 \pi(x')}{cz_2 p(x')} p(x')dx' = \frac{z_1}{cz_2} \end{aligned} \tag{10.7}$$

ahol z_1 és z_2 rendre a π-hez és p-hez tartozó ismeretlen normalizációs konstansok. Visszaírva ezeket a Bayes tételbe:

$$P(x|A) = \frac{\frac{f(x)}{cg(x)}p(x)}{\frac{z_1}{cz_2}} = \frac{\frac{z_1\pi(x)}{cz_2p(x)}p(x)}{\frac{z_1}{cz_2}} = \pi(x) \tag{10.8}$$

□

10.2. Fontossági mintavételezés (Importance Sampling)

Egy X halmazon értelmezett f függvény π eloszlás melletti várható értékét szeretnénk meghatározni, azaz a

$$E_\pi[f(x)] := \int_x f(x)\pi(x)dx \tag{10.9}$$

értékre vagyunk kíváncsiak. A π eloszlás helyett csak a p eloszlásból tudunk mintavételezni, és a mintákat súlyozzuk a

$$w(x) := \frac{\pi(x)}{p(x)} \tag{10.10}$$

ún. fontossági súlyokkal, így az f függvény helyett a

$$g(x) := w(x)f(x) = \frac{\pi(x)}{p(x)}f(x) \tag{10.11}$$

függvényt tekintjük.

Tétel 10.2. A fontossági mintavételezésből számolt súlyozott átlag torzítatlan becslése a keresett várható értéknek, azaz

$$E_\pi[f(x)] = E_p[g(x)] \tag{10.12}$$

Bizonyítás A kérdéses várható érték definíciójából rögtön adódik, hogy

$$E_p[g(x)] := \int_x g(x)p(x)dx = \int_x f(x)\frac{\pi(x)}{p(x)}p(x)dx = E_\pi[f(x)] \tag{10.13}$$

10.3. Markov lánc Monte Carlo (Markov chain Monte Carlo)

Definíció Legyen adott egy irreducibilis, aperiodikus Markov lánc egy X halmazon $T(\cdot|\cdot)$ átmeneti valószínűségekkel, amelyre teljesül, hogy

$$\forall x,y \quad T(y|x) \neq 0 \Rightarrow T(x|y) \neq 0 \tag{10.14}$$

valamint minden x és y párra a $T(y|x)$ egy ismeretlen normalizációs konstans erejéig számolható. Legyen adott egy π eloszlás, amely sehol nem 0 X-en, és X bármely pontjára egy ismeretlen normalizációs konstans erejéig számolható. Ekkor a Metropolis-Hastings algoritmus a következő két lépésből áll:

- Ha az aktuális állapot x_t, vegyünk egy véletlen y-t a $T(\cdot|x_t)$ feltételes eloszlásból.
- Vegyünk egy véletlen u-t az egyenletes eloszlásból a [0,1] intervallumon. Legyen $x_{t+1} = y$, ha

$$u \leq \frac{\pi(y)T(x|y)}{\pi(x)T(y|x)} \tag{10.15}$$

  És x_t egyébként.

Tétel 10.3. A Metropolis-Hastings algoritmus által definiált Markov láncnak π globálisan stabilis stacionárius eloszlása.

Bizonyítás Megmutatjuk, hogy a Markov lánc aperiodikus, irreducibilis és reverzibilis a π eloszlásra. Az aperiodikusság és irreducibilitás közvetlenül adódik abból, hogy az eredeti Markov lánc is aperiodikus és irreducibilis volt, és mindkét tulajdonság megőrződik a 10.14 egyenletben megfogalmazott feltétel miatt, valamint amiatt, hogy π sehol nem 0 X-en. Jelölje T' a Metropolis-Hastings algoritmus által definiált Markov lánc átmeneti mátrixát. Ekkor

$$\begin{aligned} T'(y|x)\pi(x) &= T(y|x)\min\left\{1, \frac{\pi(y)T(x|y)}{\pi(x)T(y|x)}\right\}\pi(x) = \\ &= \min\left\{T(y|x)\pi(x), T(x|y)\pi(y)\right\} = \\ &= T(x|y)\min\left\{1, \frac{\pi(x)T(y|x)}{\pi(y)T(x|y)}\right\}\pi(y) = T'(x|y)\pi(y) \end{aligned} \tag{10.16}$$

□

Megjegyzések: Az eredeti Markov lánc nem szükséges, hogy aperiodikus legyen, amennyiben van olyan x és y pár, amelyre az elutasítás valószínűsége nem 0. A nem 0 valószínűségű elutasítás azt eredményezi, hogy nem 0 valószínűséggel marad a Markov lánc az aktuális állapotban. Ez egy 1 hosszúságú hurok, amitől a Metropolis-Hastings algoritmus által definiált Markov lánc aperiodikussá válik. Amennyiben az aperiodikusság kérdéses a Metropolis-Hastings algoritmus által definiált Markov láncban, a Markov lánc lusta verziója már mindenképpen aperiodikus lesz.

Hasonlóan, nem szükséges a 10.14 feltétel sem, valamint az sem, hogy a π eloszlás ne legyen 0 sehol X-en, amennyiben garantálni tudjuk, hogy a Metropolis-Hastings algoritmus által definiált Markov lánc irreducibilis marad.

10.4. Változatok MCMC-re

a) Metropolis algoritmus

Ha a kiindulási Markov láncban minden x és y párra $T(y|x) = T(x|y)$, akkor a 10.5 képletben szereplő ún. Metropolis-Hastings hányados leegyszerűsödik:

$$u \leq \frac{\pi(y)}{\pi(x)} \tag{10.17}$$

b) Gibbs-féle mintavételezés (Gibbs sampling)

A Gibbs-féle mintavételezés akkor alkalmazható, ha az X állapottér minden eleme felírható egy n dimenziós vektorként, és minden $\mathbf{x}$ vektorra és i koordinátára lehetséges a

$$\pi\left(\left|\mathbf{x}_{[-i]}\right.\right) \tag{10.18}$$

eloszlásból mintavételezni, ahol $\mathbf{x}_{[-i]} = x_1, x_2, \ldots x_{i-1}, x_{i+1}, \ldots x_n$. Azaz minden pontra és i koordinátára lehetséges az adott i kooridnáta értékét mintavételezni, úgy, hogy a kérdéses értékek a többi koordinátával együtt a kívánt π eloszlást kövessék, feltéve, hogy az i-edik koordináta kivételével az összes koordinátaérték rögzített. Ekkor a Gibbs-féle mintavételezés minden lépésben kiválaszt egy véletlen koordinátát tetszés szerinti, de az aktuális ponttól független eloszlásból, és a kiválaszott koordinátára egy új értéket választ, a $\pi\left(\left|\mathbf{x}_{[-i]}\right.\right)$ eloszlást követve.

Tétel 10.4. Ha a Gibbs-féle mintavételezés irreducibilis Markov lánc, akkor π globálisan stabilis stacionárius eloszlása.

Bizonyítás Megmutatjuk, hogy a Markov lánc aperiodikus, és a Gibbs-féle mintavételezés egy speciális esete a Metropolis-Hastings algoritmusnak, amelyben minden Metropolis-Hastings hányados 1-gyel egyenlő.

Az aperiodikusság közvetlenül adódik abból, hogy a Markov lánc gráfjában minden pontra létezik egy hosszú hurok. A 10.14 feltétel is teljesül, hiszen bármi is a mintavételezett új koordinátaérték, az új vektor esetén is nem nulla valószínűséggel választhatjuk ugyanazt a koordinátát, és nem nulla valószínűséggel választhatjuk az eredeti koordinátaértéket. Azt kell még belátni, hogy a Metropilis-Hastings hányados mindig 1. Legyen az aktuális pont az $\mathbf{x}^a$ vektor, legyen i a kiválasztott koordináta, jelölje p azt az eloszlást, amely alapján a koordinátákat választjuk, és az új koordinátaértékű vektor legyen $\mathbf{x}^b$. Ekkor a Metropils-Hastings hányados

$$\frac{\pi(\mathbf{x}^b)T\left(\mathbf{x}^a\middle|\mathbf{x}^b\right)}{\pi(\mathbf{x}^a)T\left(\mathbf{x}^b\middle|\mathbf{x}^a\right)}=\frac{\pi(x_i^b\middle|\mathbf{x}_{[-i]}^b)\pi\left(\mathbf{x}_{[-i]}^b\right)p(i)\pi(x_i^a\middle|\mathbf{x}_{[-i]}^b)}{\pi(x_i^a\middle|\mathbf{x}_{[-i]}^a)\pi\left(\mathbf{x}_{[-i]}^a\right)p(i)\pi(x_i^b\middle|\mathbf{x}_{[-i]}^a)}=$$
$$=\frac{\pi(x_i^b\middle|\mathbf{x}_{[-i]})\pi\left(\mathbf{x}_{[-i]}\right)p(i)\pi(x_i^a\middle|\mathbf{x}_{[-i]})}{\pi(x_i^a\middle|\mathbf{x}_{[-i]})\pi\left(\mathbf{x}_{[-i]}\right)p(i)\pi(x_i^b\middle|\mathbf{x}_{[-i]})}=1 \qquad (10.19)$$

A második egyenlőség abból adódik, hogy $\mathbf{x}_{[-i]}^a=\mathbf{x}_{[-i]}^b$, és ennek a közös jelölésére vezettük be $\mathbf{x}_{[-i]}$-t.

□

c) Párhuzamos melegítés (Parallel Tempering)

A párhuzamos melegítés során egy adott állapottér önmagával vett *k*-szoros Descartes szorzatát vesszük, és ez lesz az új állapottér. Az új állapottér eloszlása koordinátánkémt definiáljuk, az egyes koordináták eloszlásai függetlenek egymástól, és minden koordináta egy Boltzmann eloszlás lesz. Az eredetileg elérni kívánt π eloszlásra úgy tekintünk, mint egy Boltzmann eloszlásra T=1 hőmérsékleten, így minden eredeti x állapotra definiálni tudjuk az állapot szabadenergiáját:

$$\Delta G(x)=-\ln(\pi(x)) \qquad (10.20)$$

hiszen

$$\pi(x)\propto e^{-\frac{\Delta G(x)}{T}} \qquad (10.21)$$

Mivel itt hipotetikus energiákról és hőmérsékletekről van szó, eltekintünk a Kelvin és Jule/mol mértékegységek közötti átváltást megadó $R\approx 8.314\frac{J}{K\times mol}$ ún. egyetemes gázállandótól. Mivel 10.21-ben csak egy arányosság van, ezért elég, ha $\pi(x)$-et csak egy ismeretlen normalizációs konstans erejéig tudjuk kiszámolni. Ekkor a szabadenergiában megjelenik egy ismeretlen additív konstans, de a Boltzmann eloszlás invariáns a szabadenergiák konstans eltolására. Az első koordinátán a hőmérsékletet T=1-re állítjuk, a többi koordinátán pedig magasabb hőmérsékleteket veszünk.

Minden koordinátára egy Markov láncot definiálunk, amelyek egy Metropolis-Hastings algoritmus szerint konvergálnak a kívánt eloszláshoz. A szorzatteren vett Markov láncot úgy definiáljuk, hogy adott p valószínűséggel egy koordináta Markov láncán lépünk, 1-p valószínűséggel pedig két lánc állapotát cseréljük fel. Ha egy koordináta Markov láncán lépünk, akkor valami q eloszlásból kisorsoljuk az egyik koordinátát, és azon vesszük a Metropolis-Hastings lépést. Ha cserét választunk, akkor valami q' eloszlásból kiválasztunk két szomszédos koordinátát, ha a kiválaszott két koordináta i és i+1, akkor

$$\min\left\{1,\frac{\pi_i(x_{i+1})\pi_{i+1}(x_i)}{\pi_i(x_i)\pi_{i+1}(x_{i+1})}\right\} \qquad (10.22)$$

valószínűséggel fogadjuk el a két koordináta felcsrélését, ahol x_i és x_{i+1} a két koordináta állapota a csere előtt, π_i és π_{i+1} pedig a két koordináta Boltzmann eloszlása.

Tétel 10.5. Ha az egyes kordinátákon vett Metropolis-Hastings algoritmusok által definiált Markov láncok aperiodikusak és irreducibilisek, akkor a párhuzamos melegítéssel definiált Markov láncnak a koordinátánként vett Boltzmann eloszlások szorzata globálisan stabilis stacionárius eloszlása.

Bizonyítás A párhuzamos melegítés aperiodikussága és irreducibilitása adódik a koordinátákon vett Metropolis-Hastings Markov láncok aperiodikusságából és irreducibilitásából. Be kell látnunk, hogy a párhuzamos melegítés Markov lánca reverzibilis a Boltzmann eloszlások szorzatára. Ezt úgy látjuk be, hogy minden elfogadási hányados pontosan a megfelelő Metropolis-Hastings hányados. Ha egy koordinátán veszünk egy Metropolis-Hastings lépést, akkor annak a Metropolis-Hastings hányadosa megegyezik a teljes térre vett Metropolis-Hastings hányadossal: mind a koordináta választás valószínűsége (p), mind az adott koordináta kiválasztásának a valószínűsége (q eloszlás) kiesik a hányadosból, ugyanígy kiesik a többi koordináta megfelelő Boltzmann eloszlásban vett valószínűsége. Ha koordinátákat cserélünk fel, akkor a teljes térre vett Metropolis-Hastings hányadosból kiesik a csere választásának a valószínűsége (1-p), valamint az adott koordinátapár kiválasztásának a valószínűsége (q' eloszlás). Hasonlóan kiesik a két koordinátán kívül az összes többi koordináta megfelelő Boltzmann eloszlásban vett valószínűsége, és ami megmarad, az pontossan a 10.22 képletben szereplő hányados.

□

A párhuzamos melegítés akkor segítheti elő a konvergenciát, ha az eredeti állapottér kívánt π eloszlásának több lokális maximuma van, és a Markov lánc nem tud gyorsan keveredni a lokális maximumokat elválasztó kis valószínűségű állapotok miatt. Az alábbiakban erre mutatunk egy játékpéldát, és bizonyítjuk is a párhuzamos melegítés gyors konvergenciáját.

Tétel 10.6. Legyen a π_n eloszlás a [0, 2n+3] intervallum egészein értelmezve következőképpen

$$\pi_n(x) = \left(\frac{1}{2}\right)^{n+2.5-0.5|n-x|-0.5|n+3-x|} \tag{10.23}$$

Legyen M_n^1 Markov lánc a [1, 2n+2] intervallum vett random sétára rátett Metropolis-Hastings algoritmussal definiált Markov lánc amely a 10.23 képletben definiált eloszláshoz konvergál, M_n^2 Markov lánc pedig egy párhuzamos melegítéssel definiált Markov lánc, amely Boltzmann eloszlásai a 10.23 képletben megadott eloszlásból vannak származtatva, két koordinátája van, az első koordinátán a hőmérséklet T=1, a második koordinátán a hőmérséklet T=∞, p = ½, q pedig az egyenletes eloszlás, a koordinátákon vett lépések a random sétára rátett Metropolis-Hastings algritmus által definiált Markov láncok. Ekkor az M_n^1 Markov lánc spektrális résének a reciproka $\Omega(2^n)$, az M_n^2 Markov lánc spektrális résének a reciproka $O(n^2)$.

Bizonyítás Az M_n^1 Markov láncra tekintsük az [1, n+1] halmazt. Ennek ergodikus folyama

$$\pi(n+1)T(n+2|n+1) = \left(\frac{1}{2}\right)^{n+1} \times \frac{1}{2} = \left(\frac{1}{2}\right)^{n+2} \tag{10.24}$$

Mivel az [1, n+1] halmaz valószínűsége pontosan ½ a stacionárius eloszlásban, a Markov lánc konduktanciája így legfeljebb $\left(\frac{1}{2}\right)^{n+1}$. De ekkor a spektrális rés reciproka legalább 2^n, a Cheeger egyenlőtlenségből (9.16 tétel) adódóan.

Az M_n^2 Markov láncra készítünk egy multicommodity flow-t a következőképpen. Az (x_1, y_1) pontból úgy megyünk az (x_2, y_2) pontba, hogy a második koordinátán y_1-ből elmegyünk x_2-be, felcseréljük a két koordinátát, majd a második koordinátán x_1-ből elmegyünk y_2-be. Erre az útvonalrendszerre alkalmazzuk a 9.18 tételt. A 9.19 képletben szereplő maximumot felülről becsüljük. Egy (a,b) pont koordinátacseréjének a kapacitása pont

$$\frac{1}{2}\min\{\pi((a,b)),\pi((b,a))\} = \frac{1}{4n+4}\min\{\pi_1(a),\pi_1(b)\} \tag{10.25}$$

Minden út hossza legfeljebb 4n+3. Az (a,b) pont koordinátacseréjét minden ((a,y_1), (b,y_2)) pontpárra alkalmazzuk így a 9.19-ben szereplő szummázás felső becslése koordinátacserékre:

$$\begin{aligned}
&\sum_{(x,y)} \sum_{\gamma\in\Gamma_{xy}|e\in\gamma} \pi(x)\pi(y)p_{xy}(\gamma)\frac{|\gamma|}{Q(e)} \le \\
&\le (2n+2)^2 \frac{\pi_1(a)}{2n+2}\frac{\pi_1(b)}{2n+2}\frac{4n+3}{\frac{1}{4n+4}\min\{\pi_1(a),\pi_1(b)\}} = \\
&= \max\{\pi_1(a),\pi_1(b)\}(4n+3)(4n+4) \le (4n+3)(4n+4)
\end{aligned} \tag{10.26}$$

A második koordinátában (a,b)-ből (a,b+1)-be vagy (a,b-1)-be lépés kapacitása $\frac{1}{8}\frac{\pi_1(a)}{2n+2}$. Ilyen lépést vagy akkor használunk, amikor y_1-ből elmegyünk x_2-be vagy amikor x_1-ből elmegyünk y_2-be. Mindkét esetben y_1 és y_2 tetszőleges, az első esetben $a = x_1$, a második esetben $a = x_2$. Jelölje c x_2-t, ha $a = x_1$, és x_1-et, ha $a = x_2$. Ekkor a a 9.19-ben szereplő szummázás felső becslése a második koordinátában tett lépésekre

$$\begin{aligned}
&\sum_{(x,y)} \sum_{\gamma\in\Gamma_{xy}|e\in\gamma} \pi(x)\pi(y)p_{xy}(\gamma)\frac{|\gamma|}{Q(e)} \le \\
&\le (2n+2)^2 \sum_c \frac{\pi_1(a)}{2n+2}\frac{\pi_1(c)}{2n+2}\frac{4n+3}{\frac{1}{8}\frac{\pi_1(a)}{2n+2}} = \\
&= 8(4n+3)(2n+2)
\end{aligned} \tag{10.27}$$

Így a Poincarré koefficiens felső becslése $8(4n+3)(2n+2)$, ami a 9.18 tétel értelmében felső becslése a spektrális rés reciprokának is.

d)Metropolizált fontossági mintavételezés (Metropolized Importance Sampling)

Minden fontossági mintavételezést fel tudunk használni egy Markov lánc Monte Carlohoz is, ha a fontossági mintavételezésre úgy tekintünk, mint egy nulladrendű Markov láncra (azaz, amelyben a következő állapot független az aktuális állapottól). Ha π a céleloszlás, p a fontossági eloszlás, akkor a Metropolis-Hastings hányados a fontossági súlyok hányadosa

$$\frac{\pi(y)p(x)}{\pi(x)p(y)} = \frac{w(y)}{w(x)} \tag{10.28}$$

Tétel 10.7. (Liu) A Metropolizált fontossági mintavételezés Markov láncának második legnagyobb sajátértéke

$$\lambda_2 = 1 - \frac{1}{\max_i \{w(x_i)\}} \tag{10.29}$$

azaz, a spektrális rés reciproka a legnagyobb fontossági súly.

A tétel bizonyításától itt eltekintünk, habár nem túl bonyolult lineáris algebrai eszközökkel bizonyítható. A tétel azt mondja, hogy a fontossági mintavételezésben használt eloszlás farkának nem szabad lényegesen gyorsabban csökkennie, mint a céleloszlás farkának. Ha egy kis valószínűségű állapotot valószínűsége a fontossági eloszlásban még kisebb, mint a céleloszlásban, akkor a konvergencia nagyon lassú lesz. Magas dimenziós térben az eloszlás farkán a nagyságrendet különösen nehéz megbecsülni, ezért magas dimenziós tereken ritkán használnak fontossági mintavételezést.

11. fejezet

Dupla vágás és kötés utak sztochasztikus becslése

Angol nyelvű kézirat letölthető a http://www.renyi.hu/~miklosi/dcj.pdf url-ről.