УНИВЕРЗИТЕТ У БЕОГРАДУ
МАТЕМАТИЧКИ ФАКУЛТЕТ

Момир Глушица

Скривени Марковљеви модели и примена у биоинформатици

Магистарски рад

Ментор: проф. др Гордана Павловић-Лажетић

Београд
2013.

Предговор

Овај магистарски рад посвећен је неким математичким методама у решавању биоинформатичких проблема, и то скривеним Марковљевим моделима и њиховим применама у биоинформатици. Рад је организован у четири дела. У уводном делу дат је преглед и краћи опис тема обрађених у раду. У другом делу дат је увод у теорију скривених Марковљевих модела и приказани су основни алгоритми. Неке важније примене у биоинформатици описане су у трећем делу. У четвртом делу описана је сопствена имплементација алгоритама скривених Марковљевих модела и њихова примена на предвиђање и обележавање једне специфичне врсте секвенци у оквиру генома, тзв. CpG острва.

Задовољство ми је да се овом приликом захвалим свом ментору, проф. др Гордани Павловић-Лажетић, на изузетној помоћи. Својим идејама, саветима, сугестијама, разумевањем и подршком учинила је да овај рад буде бољи. Захваљујем се члановима комисије за преглед и оцену рада, проф. др Весни Јевремовић, проф. др Ненаду Митићу и др Милошу Бељанском на корисним саветима.

Београд, Момир Глушица

Садржај

1. Увод

1.1 Скривени Марковљеви модели

Скривени Марковљеви модели су статистички модели чије је математичке основе поставио Л.Е.Баум и његови сарадници. Скривени Марковљев модел је петорка (Σ,Q,π,A,E) где је Σ је алфабет – скуп симбола које модел емитује, Q = {1,2,3,…,k} је скуп стања, π је вектор почетних вероватноћа, A је матрица транзиционих вероватноћа, E је матрица емисионих вероватноћа.

Модел прелази из стања у стање и у сваком стању емитује симбол са неком вероватноћом. Тако се формирају две секвенце: секвенца стања и секвенца симбола. Модели се зову скривени јер је секвенца стања невидљива (скривена). Секвенца симбола је видљива (позната) као и параметри модела: транзиционе вероватноће преласка из стања у стање и вероватноће емитовања симбола. Модели се зову Марковљеви јер тренутно стање модела зависи од онолико претходних стања ког су реда модели.

Природно се намеће проблем декодирања, тј. како на основу познате секвенце симбола и модела (параметара) одредити секвенцу стања тако да вероватноћа емитовања дате секвенце симбола буде највећа. Важан је и проблем како одредити параметре модела да добијемо већу вероватноћу – проблем учења.

У овом раду се излажу алгоритми за решавање тих проблема: Витерби алгоритам, алгоритам унапред, алгоритам уназад, постериор декодирање, Баум-Велч алгоритам, Витерби учење.

1.2 Примене Скривених Марковљевих модела

Скривени Марковљеви модели (СММ) су нашли примену у препознавању говора, рукописа, покрета, записивању музичких нота, класификацији мелодија, финансијама, обради природних језика, препознавању образаца, медицини, биоинформатици, итд.

У биоинформатици, алгоритми засновани на СММ примењени су на анализу биолошких секвенци, на проналажење гена, на класификацију протеинских фамилија итд. У тим применама се често користе двоструки скривени Марковљеви модели (поравнање две секвенце) и профил скривени Марковљеви модели (поравнање више секвенци) који су описани у овом раду. У раду су приказане и основне идеје коришћења скривених Марковљевих модела за налажење гена и класификацију трансмембранских протеина.

1.3 CpG острва

CpG острва су делови ДНК секвенце где се нуклеотидна база гуанин (G) јавља иза нуклеотидне базе цитозин (C) (CG динуклеотид) чешће него у осталом делу секвенце. CpG острва (p је од фосфора чијим су атомом повезани суседни нуклеотиди једног ДНК ланца) се тако обележавају да би се

направила разлика у односу на водоничне везе између C и G нуклеотида као одговарајућих нуклеотида у комплементарним ланцима једне ДНК секвенце.

У оквиру геномске секвенце често се догађа да C нуклеотид (цитозин) метилацијом мутира у T нуклеотид (тимин), што има за последицу да се CpG динуклеотид генерално јавља знатно ређе него што би се могло очекивати. Код сисара је проценат метилације од 70% до 80%.

Тамо где се CpG метилација ређе дешава јављају се CpG острва. Она се могу наћи око промотера или на почетку гена па су зато важна у тражењу гена. Промотер је део ДНК секвенце (секвенца нуклеотида) који иницијализује транскрипцију гена (почетак синтезе протеина) а има и улогу у експресији гена. Најчешће се налази близу почетка гена. CpG острва су обично дугачка од неколико стотина до неколико хиљада нуклеотида.

Метилација цитозина око промотера може да "искључи" ген (да онемогући производњу протеина који је кодиран тим геном), што је уочено код бројних врста болести рака, па CpG острва могу да буду корисна у борби против ове болести.

У овом раду описана је једна примена скривених Марковљевих модела на предикцију и обележавање CpG острва. Развијен је и сопствени програм за реализацију овог задатка и показано је да даје веома добре резултате.

2.СКРИВЕНИ МАРКОВЉЕВИ МОДЕЛИ (CMM)

2.1. Увод

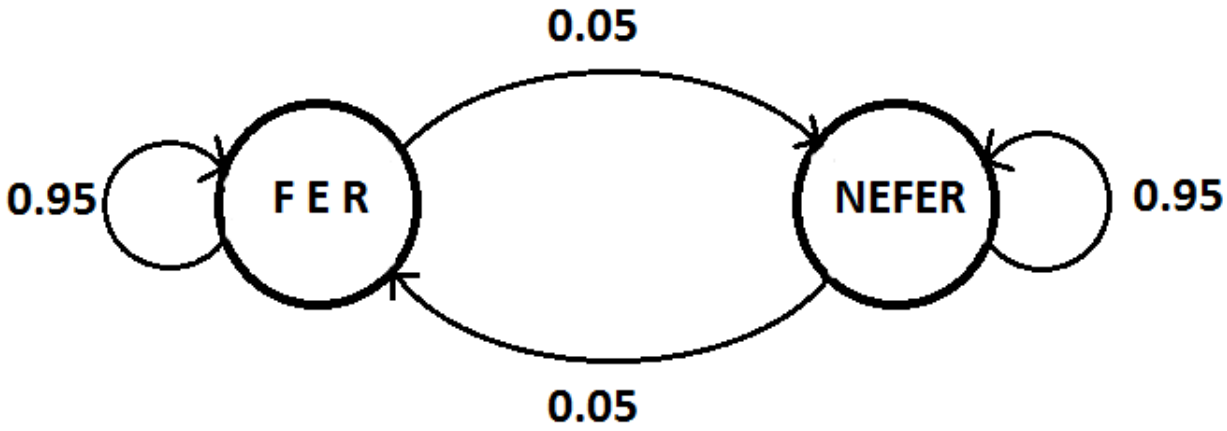
Скривени Марковљеви модели су статистички модели који су нашли примену у препознавању говора, рукописа, покрета, записивању музичких нота, класификацији мелодија, финансијама, обради природних језика, препознавању образаца, медицини, биоинформатици, итд.

У овом раду биће описани најважнији алгоритми, а касније и неке најважније примене у биоинформатици.

2.2. Уводни пример и основни проблеми

2.2.1. Уводни пример

Уводни пример (пример који може да мотивише) је пример непоштене коцкарнице, чији је модел представљен на слици 2.1. [1]



P(1|F) = 1/6	P(1|N) = 1/10
P(2|F) = 1/6	P(2|N) = 1/10
P(3|F) = 1/6	P(3|N) = 1/10
P(4|F) = 1/6	P(4|N) = 1/10
P(5|F) = 1/6	P(5|N) = 1/10
P(6|F) = 1/6	**P(6|N) = ½**

Слика 2.1. Модел непоштене коцкарнице

Пример се интерпретира на следећи начин. Коцкар баца коцкице играјући неку игру. Он користи две коцкице од којих је једна фер (даје сваки број са вероватноћом једне шестине) а друга нефер и даје шестицу са вероватноћом једне половине, док остали бројеви имају вероватноћу једне десетине. С времена на време, да не би био ухваћен у варању, мења коцкице (претпоставићемо да то ради случајно и да то не зависи од дужине коришћења неке коцкице).

Дијаграм на слици 2.1 представља непоштену коцкарницу. Два стања представљају две коцкице. Стрелице између стања (коцкица) носе тежину вероватноћа преласка из једног стања у друго (замена коцкица) и овде је 0.05 (не морају бити исте). Вероватноће да се не мења стање (коцкица) је 0.95. Те вероватноће ћемо звати транзиционе вероватноће између стања модела.
На слици 2.1 су приказане и вероватноће добијања (емитовања) бројева под условом да је коришћена фер или нефер коцкица (стање). Њих ћемо звати емисионе вероватноће.

На пример, P(6|N) = 1/2, значи да је вероватноћа да се добије шестица под условом да се користи нефер коцкица једнака једној половини.

Примећујемо да се појављују две интересантне секвенце. Једна је секвенца коцкица (стања) а друга њој одговарајућа секвенца бројева (симбола):

Ф Ф Ф Ф Ф Ф Ф Ф Ф Ф Ф Н Н Н Н Н Н Н Н Н Н Н

1 2 3 4 5 6 1 2 3 4 5 6 1 6 2 6 3 6 4 6 5 6 6 6

Ф је ознака за фер коцкицу, а Н за нефер коцкицу. Бројеви у другој секвенци се добијају (емитују) бацањем коцкице (из стања) које је изнад у првој секвенци. Наш коцкар (модел) из стања Ф добија (емитује) број 1, па број 2 итд. да би после дванаест бацања променио коцкицу и добијао (емитовао) 1, па 6, онда 2 и тако даље.

2..2.2. Основни проблеми скривених Марковљевих модела

Ови проблеми се уводе у литератури на разне начине. Овде ћемо их поставити уз помоћ горњег примера. Касније ћемо их детаљније проучити. Дакле, под горњим условима могу се уочити следећи проблеми, тј. поставити следећа питања.

2.2.1.1. Проблем евалуације *(*енг. *The Evaluation Problem)*

Претпоставимо да смо направили модел који ради као коцкар. Могли бисмо се запитати колика је вероватноћа секвенце бројева од један до шест ако је генерисана таквим моделом. На пример, вероватноћа секвенце

3 5 4 1 2 6 5 3 4 3 5 6 2 5 3 6 6 3 6 6 6 3 5 6 1 4 6 3 4 6 6 4 3 6 1 5 4 3 6 1 3 5 2 4

могла би бити рецимо $1.8 * 10^{-25}$. (није тачно израчунато – само илустрација)

Уочљиво је да ће вероватноће дугачких секвенци бити веома мали бројеви и да би нам то могло правити проблеме при израчунавању. Најчешће се помажемо логаритамском трансформацијом или скалирањем *(*енг. *scaling).*

2.2.1.2. Проблем декодирања *(*енглески: *The Decoding Problem)*

Упоредо са постављањем овог проблема објаснићемо зашто се модели зову скривени. У горњем примеру ми не можемо да видимо да ли коцкар користи фер или нефер коцкицу. Скривена је секвенца стања (коцкица). Отуда назив *скривени* Марковљеви модели.

Могу да се поставе следећа питања:

Да ли на основу дате секвенце бројева (симбола) можемо утврдити да коцкар вара и када вара?

Да ли можемо да кажемо који је број добијен бацањем фер коцке, а који је добијен бацањем нефер коцке ?

Ово је проблем декодирања скривених Марковљевих модела где покушавамо да откријемо (највероватнију) *скривену* секвенцу коцкица (стања) која је произвела секвенцу бројева (1-6).

На пример, за следећу секвенцу, могли бисмо одредити којом коцком (стањем) је одговарајући број добијен.

3 5 4 1 2 6 5 3 4 3 5 6 2 5 3 6 6 3 6 6 6 3 5 6 1 4 6 3 4 6 6 4 3 6 1 5 4 3 6 1 3 5 2 4
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0
ФЕР НЕФЕР ФЕР

Нуле представљају фер коцку, а јединице нефер коцку.

Помоћу алгоритама које ћемо описати моћи ћемо да са великом вероватноћом одредимо да ли је и кад коцкар користио фер или нефер коцку.
Слично, у случају биолошких секвенци моћи ћемо одредити, на пример, где се вероватно налазе гени.

2.2.1.3. Проблем учења *(*енг. *The Learning Problem)*

Ако смо утврдили да коцкар вара, можемо ли сазнати када и како вара?
Колика је вероватноћа да се добије шестица ако се користи нефер коцка?
Колика је вероватноћа осталих бројева на нефер коцки?
Колико често и када коцкар мења коцке?
(Последње питање је у вези са транзиционим вероватноћама, а претходна три су у вези са емисионим вероватноћама – ово ће бити објашњено касније).

Одређивање ових вероватноћа зваћемо проблем учења скривених Марковљевих модела *(*енг. *Learning problem of HMM)*.

Ако се вратимо на пример из проблема декодирања, вероватноћа да добијемо шестицу ако користимо нефер коцку би могла бити 10/19. Ово је само процена која уопште не мора одговарати ономе како је секвенца добијена, али би поприлично добро описала шта се десило.

Касније ћемо описати математички апарат за решавање ових проблема и одговарајуће алгоритме.

2.3. Дефиниција скривених Марковљевих модела и генерисање секвенце

2.3.1. Дефиниција СММ

Скривен Марковљев модел је петорка (Σ,Q,π,A,E).

1) Σ је алфабет – скуп карактера које модел емитује.
   У горњем примеру Σ = {1,2,3,4,5,6}. У случају бацања новчића би био {писмо, глава}, а у случају ДНК секвенце би могао бити { A, T, G, C }.
2) Q = {1,2,3,…,k} је скуп стања
   У горњем примеру Q = { Фер коцка (стање), Нефер коцка (стање) }
3) π је вектор почетних вероватноћа
   У горњем примеру би могао бити (1/2,1/2) ако би коцкар са једнаком вероватноћом на почетку бирао коцку. Збир ових вероватноћа је 1.
4) A је матрица транзиционих вероватноћа
   Транзициону бероватноћу између два стања обележаваћемо са a_{ij} . То је веоватноћа преласка из стања i у стање j. Ове вероватноће не зависе од времена и константне су. Збир свих (одлазећих) вероватноћа из једног стања је 1. То значи да или остајемо у истом стању (стрелица од стања до самог себе) или прелазимо на неко друго стање у свакој тачки модела.
5) E је матрица емисионих вероватноћа.
   У редовима су обично стања а у колонама слова алфабета који се емитује. У горњем примеру матрица би била:

	1	2	3	4	5	6
F	1/6	1/6	1/6	1/6	1/6	1/6
N	1/10	1/10	1/10	1/10	1/10	1/2

   Збир вероватноћа емитовања симбола алфабета из једног стања једнак је 1.

Поред матричног записа чешће ћемо користити следеће обележавање:
$e_k(b) = P(x_i = b \mid \pi_i = k)$, односно:
вероватноћа да се из стања k емитује симбол b на позицији i.

Секвенцу симбола ћемо обележавати $x_1 x_2 ... x_i ... x_L$
Секвенцу стања ћемо обележавати $\pi_1 \pi_2 ... \pi_i ... \pi_L$
Тако бисмо могли писати у горњем примеру
$e_F(1) + e_F(2) + e_F(3) + e_F(4) + e_F(5) + e_F(6) = 1$
или $e_N(1) + e_N(2) + e_N(3) + e_N(4) + e_N(5) + e_N(6) = 1/10 + 1/10 + 1/10 + 1/10 + 1/10 + 1/2 = 1$

За транзициону вероватноћу (у раније спомињаној матрици) би било:
$a_{kl} = P (\pi_i = k | \pi_{i-1} = l)$
Индекс і се овде користи само да каже да су суседна стања.

2.3.2. Генерисање секвенце СММ

Ако је дата секвенца емитованих симбола $x_1 x_2 ... x_N$ и секвенца стања $\pi_1 \pi_2 ... \pi_N$, можемо израчунати вероватноћу да је та секвенца стања емитовала секвенцу симбола.
$P(x, \pi) = P(\pi_1)P(x_1| \pi_1) P(\pi_2| \pi_1) P(x_2| \pi_2) ..... P(\pi_N| \pi_{N-1}) P(x_N| \pi_N)$
(или према ознакама из дефиниције и уз примену комутативног закона)
$= a_{0\pi1} a_{\pi1\pi2} a_{\pi2\pi3} ..... a_{\pi N-1\pi N} e_{\pi1}(x_1) e_{\pi2}(x_2) ..... e_{\pi2}(x_2)$

Значење формуле је следеће:

- Почињемо из стања π_1 (са почетном вероватноћом)
- Из стања π_1 је емитован симбол x_1 (са емисионом вероватноћом)
- Прелазак из стања π_1 у стање π_2 (са транзиционом вероватноћом)
- Из стања π_2 је емитован симбол x_2 (са емисионом вероватноћом)
- ...
-
- Прелазак из стања π_{N-1} у стање π_N (са транзиционом вероватноћом)
- Из стања π_N је емитован симбол x_N (са емисионом вероватноћом)

При израчунавању вероватноће користили смо особину да је развијени модел - Марковљев модел првог реда.
$P(x_m| \pi_m,\pi_{m-1},\pi_{m-2},...,\pi_2,\pi_1) = P(x_m| \pi_m)$
Постоје наравно и Марковљеви модели вишег реда, али се мање користе јер углавном у пракси не дају боље резултате.

ПРИМЕР 1. рачунања вероватноће путање и секвенце:
x = 1, 2, 1, 5, 6, 2, 1, 5, 2, 4

Ако је π = фер,фер,фер,фер,фер,фер,фер,фер,фер,фер
$P(x, \pi) = (0.5)(1/6)^{10} (0.95)^9 = 0.00000000521158647211$ **~ $0.5 \cdot 10^{-9}$**

Ако је π= нефер,нефер,нефер,нефер,нефер,нефер,нефер,нефер,нефер,нефер

$P(x, \pi) = (0.5)(1/10)^9\ (0.5)(0.95)^9 = 0.00000000015756235243$ **~ $0.16 \cdot 10^{-9}$**

Можемо запазити да горња секвенца симбола x има већу вероватноћу ако је добијена секвенцом стања (путањом) која је састављена од фер стања.

ПРИМЕР 2.
x = 1, 6, 6, 5, 6, 2, 6, 6, 3, 6

Ако је π = фер,фер,фер,фер,фер,фер,фер,фер,фер,фер

$P(x, \pi) = (0.5)(1/6)^{10}\ (0.95)^9 = 0.00000000521158647211$ **~ $0.5 \cdot 10^{-9}$**

Ако је π= нефер,нефер,нефер,нефер,нефер,нефер,нефер,нефер,нефер,нефер

$P(x, \pi) = (0.5)(1/10)^4\ (0.5)^6(0.95)^9 = 0.00000049238235134735$ **~ $0.5 \cdot 10^{-7}$**

Можемо запазити да је вероватноћа добијања секвенце x помоћу путање од нефер стања око 100 пута већа.

ПРИМЕР 3.
x = 1, 2, 3, 4, 5, 6, 6, 6, 6, 6
Ако је π = фер,фер,фер,фер,фер,фер,фер,фер,фер,фер

$P(x, \pi) = (0.5)(1/6)^{10}\ (0.95)^9 = 0.00000000521158647211$ **~ $0.5 \cdot 10^{-9}$**

Ако је π= нефер,нефер,нефер,нефер,нефер,нефер,нефер,нефер,нефер,нефер

$P(x, \pi) = (0.5)(1/10)^5\ (0.5)^5(0.95)^9 = 0.00000009847647026947$ **~ $0.98 \cdot 10^{-8}$**

Ако је π = фер,фер,фер,фер,фер,нефер,нефер,нефер,нефер,нефер

$P(x, \pi) = (0.5)(1/6)^5\ 0.05\ (0.5)^5\ (0.95)^8 =$ **~ $0.67 \cdot 10^{-8}$**

Овај резултат је мало изненађујући. Ово би на први поглед требала бити највероватнија путања али није. Разлика није велика и узрок је мењање коцкице(0.05). Када бисмо на почетак секвенце додали један или више симбола који нису шестице и да су емитовани из фер стања, таква секвенца би постала највероватнија.

2.4. Алгоритми скривених Марковљевих модела

2.4.1. Проблем декодирања и Витерби алгоритам (енг. *Viterbi*)

За дату секвенцу емитованих симбола одредићемо секвенцу стања (путању) која максимизира вероватноћу

$P(x, \pi) = a_{0\pi 1}\, e_{\pi 1}(x_1)\; a_{\pi 1\pi 2}\, e_{\pi 2}(x_2) a_{\pi 2\pi 3} \ldots\ldots\; a_{\pi N-1 \pi N}\, e_{\pi N}(x_N)$.

Дакле, тражимо путању (секвенцу стања) π^* где је $\pi^* = \text{argmax}_\pi\, P(x, \pi)$.

Наивно израчунавање по свим путањама тражи експоненцијално време. Проблем ћемо решити ефикасније Витерби алгоритмом и динамичким програмирањем.

Нека имамо M стања и нека је дужина емитоване секвенце N. Дефинисаћемо динамички програмирану табелу (видети слику 2.2.) величине MxN тако да важи:

$$V_k(i) = \max_{\{\pi_1\pi_2\ldots\pi_{i-1}\}} P(x_1\, x_2 \ldots x_{i-1}\; ,\pi_1\; , \ldots ,\pi_{i-1}, x_i\; , \pi_i = k),$$

где је $V_k(i)$ вероватноћа највероватије путање (секвенце стања) која генерише (емитује) секвенцу симбола $x_1\, x_2 \ldots x_i$ и која завршава у стању $\pi_i = k$.

Претпоставимо даље да смо израчунали $V_k(i)$ за сва стања k и посматрајмо $V_m(i+1)$ где је m неко следеће стање у следећем тренутку (i+1).

$$V_m(i+1) = \max_{\{\pi_1\pi_2\ldots\pi_i\}} P(x_1\, x_2 \ldots x_i\; ,\pi_1\; , \ldots ,\pi_i, x_{i+1}\; , \pi_{i+1} = m)$$

$$= \max_{\{\pi_1\pi_2\ldots\pi_i\}}$$

$$\{ P(\, x_{i+1}\; , \pi_{i+1} = m \mid x_1\, x_2 \ldots x_i\; ,\pi_1\; , \ldots ,\pi_i) \cdot P(x_1\, x_2 \ldots x_i\; ,\pi_1\; , \ldots ,\pi_i) \}$$

(на основу формуле здружене вероватноће)

$$= \max_{\{\pi_1\pi_2\ldots\pi_i\}}$$

$$\{ P(\, x_{i+1}\; , \pi_{i+1} = m \mid \pi_i) \cdot P(x_1\, x_2 \ldots x_{i-1}\; ,\pi_1\; , \ldots ,\pi_{i-1}, x_i, \pi_i) \}$$

(марковљева особина – зависност само од претходног стања)

$$= \max_k$$

$$\{P(\, x_{i+1}\; , \pi_{i+1} = m \mid \pi_i=k) \max_{\{\pi 1\, \pi 2\, \ldots \pi i-1\}} P(x_1\, x_2 \ldots x_{i-1}, \pi_1, \ldots, \pi_{i-1}, x_i, \pi_i=k)\}$$

(према својствима проблема)

$$= \max_k \{\, P(\, x_{i+1} \mid \pi_{i+1} = m)\; P(\, \pi_{i+1} = m \mid \pi_i=k)\; V_k(i) \}$$

(према дефиницији)

$$= e_m(x_{i+1})\; \max_k\; a_{km}\; V_k(i)$$

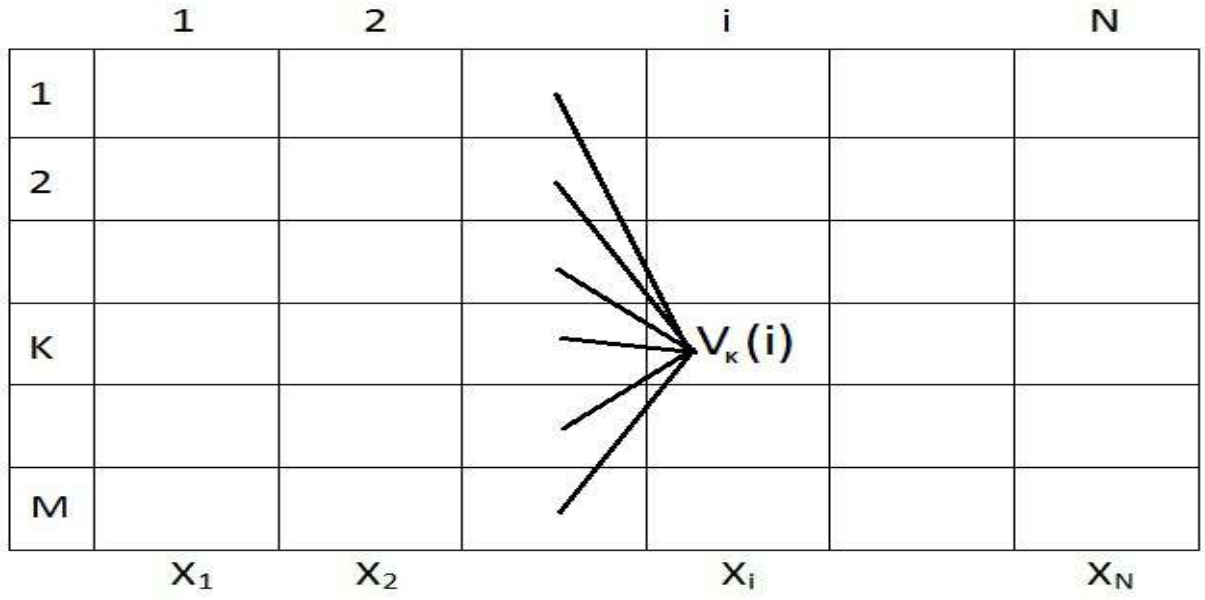
(промена ознака и $e_m(x_{i+1})$ не зависи од k)

Сада можемо да попунимо таблицу помоћу горње формуле.
Почињемо од имагинарног стања 0 и попуњавамо колону по колону.

$V_0(0)=1$ јер одатле крећемо, остали су 0 ($V_к(0)=0$ за $к \geq 1$).

$V_k(i) = e_k(x_i) \max_j a_{jk} V_j(i-1)$

На крају се одреди максимум последње колоне: $P(x, \pi^*) = \max_K V_K(N)$



Слика 2.2. Табела за рачунање по Витерби алгоритму

Одржаваћемо и матрицу показивача на претходно стање, који ће нам говорити из ког стања смо дошли у тренутно стање, да бисмо могли касније, враћајући се, да одредимо највероватнију (Витерби) путању.

Читање путање уназад почињемо од последње колоне где је највећа вредност.

Простор за динамичку матрицу је O(MN), а време извршавања је $O(M^2 N)$.

При импементацији се наилази на проблем да вероватноће са повећањем дужине секвенце постају веома мале. Једно веома често коришћено решење је логаритамска трансформација и горња формула постаје:

$V_m(i)= \log e_m(x_i) + \max_k (V_k(i-1) + \log a_{km})$

2.4.2. Алгоритам Унапред
(енг. *The Forward Algorithm*)

Алгоритам Унапред је алгоритам евалуације и даје вероватноћу да је секвенца x генерисана датим скривеним Марковљевим моделом. Та вероватноћа добија се сабирањем вероватноћа свих могућих различитих путања (секвенци стања) које генеришу секвенцу x.

$P(x) = \Sigma_\pi P(x,\pi) = \Sigma_\pi P(x|\pi) P(\pi)$

Овакав приступ није практичан јер број могућих путања расте експоненцијално са дужином секвенце. Број путања је број варијација са понављањем k^N где је k број стања а N дужина секвенце.

Проблем ће решити динамички програмирана табела као и у случају Витерби алгоритма, са вредностима:

$f_k(i) = P(x_1 x_2 \ldots x_i , \pi_i = k)$

где је $f_k(i)$ вероватноћа да се генерише првих i симбола секвенце која завршава у стању k. Ову вероватноћу ћемо звати вероватноћа унапред (forward possibility).

Треба да нађемо рекурентну формулу:

$f_k(i) = P(x_1 x_2 \ldots x_i , \pi_i = k)$ (дефиниција)

$= \Sigma_{\pi_1 \pi_2 \ldots \pi_{i-1}} P(x_1 x_2 \ldots x_{i-1} , \pi_1 , \ldots, \pi_{i-1}, \pi_i = k)\, e_k(x_i)$

$= \Sigma_m \Sigma_{\pi_1 \pi_2 \ldots \pi_{i-2}} P(x_1 x_2 \ldots x_{i-1} , \pi_1 , \ldots, \pi_{i-2}, \pi_{i-1} = m)\, a_{mk}\, e_k(x_i)$

$= \Sigma_m P(x_1 x_2 \ldots x_{i-1}, \pi_{i-1} = m)\, a_{mk}\, e_k(x_i)$

$= e_k(x_i) \Sigma_m f_m(i-1)\, a_{mk}$

Сада можемо да израчунамо $f_k(i)$ за свако k и за свако i .

Алгоритам:

$f_0(0)=1$, ($f_k(0)=0$ за $k \geq 1$).

$f_k(i) = e_k(x_i) \sum_j a_{jk} f_j(i-1)$

На крају се одреди збир последње колоне: $P(x, \pi^*) = \sum_K f_K(N)$

Можемо користити табелу као и код Витерби алгоритма. Крај алгоритма Унапред је кад се саберу вредности у последњој колони. То је P(x).

Ако упоредимо алгоритме Унапред и Витерби видимо да су они веома слични и да има само једна разлика. Тамо где Витерби рачуна максимум од претходних вероватноћа, Унапред рачуна збир.

2.4.3. Алгоритам Уназад
(енг. *The Backward algorithm*)

Како да ухватимо коцкара да вара у примеру са непоштеном коцкарницом?
Како да израчунамо да је у i – том бацању коцке користио нефер коцку?
Како да израчунамо вероватноћу да је i - ти симбол секвенце емитован из неког стања k ?

Ова питања могу да нас мотивишу да рачунамо вероватноћу:

$P(\pi_i = k \mid x)$

Ако је израчунамо за свако к, добијамо расподелу вероватноћа емитовања симбола на i–тој позицији.
Дакле,

$P(\pi_i = k \mid x) = P(x_1 x_2 \ldots x_i , \pi_i = k , x_{i+1} x_{i+2} \ldots x_N)$

$= P(x_1 x_2 \ldots x_i , \pi_i = k) P(x_{i+1} x_{i+2} \ldots x_N \mid x_1 x_2 \ldots x_i , \pi_i = k)$

$= P(x_1 x_2 \ldots x_i , \pi_i = k) P(x_{i+1} x_{i+2} \ldots x_N \mid \pi_i = k)$

Први чинилац овог производа је унапред вероватноћа а други ћемо дефинисати као уназад вероватноћу*(backward possibility)*.

$b_k(i) = P(x_{i+1} x_{i+2} \ldots x_N \mid \pi_i = k)$

$= \Sigma_{\pi i+1 \, \pi i+2 \, \ldots \pi N} P(x_{i+1}, x_{i+2}, \ldots x_N, \pi_{i+1} , \ldots, \pi_N \mid \pi_i = k)$

$= \Sigma_m \Sigma_{\pi i+1 \, \pi i+2 \, \ldots \pi N} P(x_{i+1}, x_{i+2}, \ldots x_N, \pi_{i+1}=m, \pi_{i+2}, \ldots, \pi_N \mid \pi_i = k)$

$= \Sigma_m e_m(x_{i+1}) a_{km}$

$\Sigma_{\pi i+1 \, \pi i+2 \, \ldots \pi N} P(x_{i+1}, x_{i+2}, \ldots x_N, \pi_{i+1}=m, \pi_{i+2}, \ldots, \pi_N \mid \pi_i = k)$

$= \Sigma_m e_m(x_{i+1}) \, a_{km} \, b_m(i+1)$

Добили смо рекурентну формулу којом можемо израчунати вероватноће уназад.

Алгоритам Уназад:

Почетак:

За свако k нека је $b_k(N) = 1$

Попуњавање матрице уназад:

$b_k(i) = \Sigma_m \, e_m(x_i+1) \, a_{km} \, b_m(i+1)$

Крај:

$P(x) = \Sigma_m \, a_{0m} \, e_m(x_1) \, b_m \, (1)$

Алгоритам Уназад је алгоритам евалуације и, као алгоритам Унапред, рачуна вероватноћу да је секвенца x генерисана датим скривеним Марковљевим моделом.

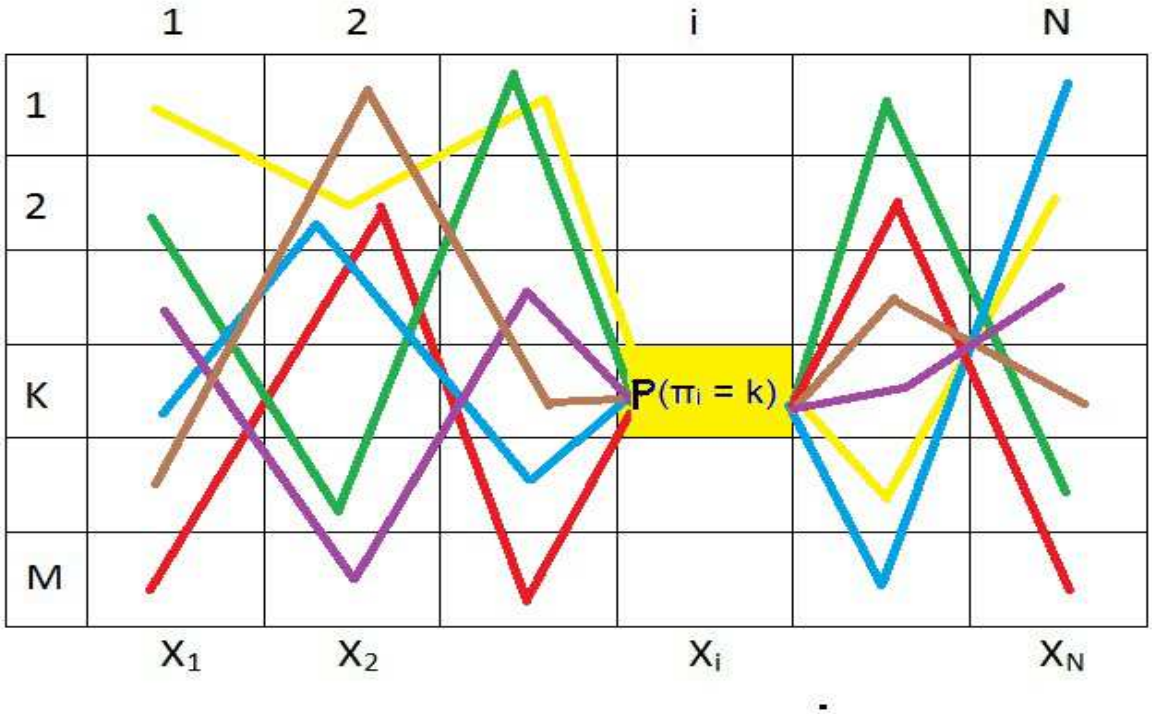
Као код Витерби алгоритма простор за динамичку матрицу је O(MN), а време извршавања је $O(M^2 N)$.

И овде имамо проблем израчунавања који се решава скалирањем: после неколико позиција секвенце симбола множимо добијену вредност константом да бисмо остали у опсегу покретног зареза. Чуваћемо запис о овим константама да бисмо на крају могли израчунати вероватноћу.

2.4.4. Постериор декодирање
(енг. Posterior Decoding)

Ово може бити алтернатива Витерби декодирању. Знајући вероватноће унапред и уназад, можемо израчунати

$$P(\pi_i = k \mid x) = P(\pi_i = k \mid x)/P(x)$$
$$= P(x_1 x_2 \ldots x_i \ , \pi_i = k \ , x_{i+1} x_{i+2} \ldots x_N \ )/P(x)$$
$$= P(x_1 x_2 \ldots x_i \ , \pi_i = k) \, P(x_{i+1} x_{i+2} \ldots x_N \mid \pi_i = k \,)/P(x)$$
$$= f(i)_k \, b(i)_k /P(x)$$



Слика 2.3. Постериор декодирање

На свакој позицији i у секвенци симбола можемо одредити највероватније стање које га је емитовало:

$$\pi^*_i = \text{argmax}_k \, P(\pi_i = k \mid x),$$

те од њих можемо направити секвенцу. Она се обично назива постериор путања (енг. *posterior decoding parse):* $\pi^* = \pi^*_1, \ldots, \pi^*_N$
Ова секвенца може бити алтернатива секвенци добијеној Витерби алгоритмом.

Пре него што упоредимо ове две секвенце, сетимо се питања од раније - да ли можемо ухватити коцкара да вара. На основу постериор вероватноће могли бисмо са приличном сигурношћу тврдити да ли користи нефер коцку или не у неком тренутку (бацању коцке).

Мада постериор декодирање може често бити корисније (посебно ако се траже информације на локалном нивоу) него оно које може да произведе Витерби алгоритам, дешава се да даје немогућу путању (ако неке транзиције нису могуће – вероватноћа им је нула). Зато треба бити пажљив са његовом употребом.

Да ли је боље користити Витерби или постериор декодирање? Одговор делом зависи од тога колико је вероватна оптимална Витерби путања, односно колико се оптимална путања разликује од осталих путања. Ако се оптимална путања само мало разликује од неких других путања ,тада ћемо вероватно од постериор декодирања добити корисније информације.

На пример, биолог је идентификовао промотер регион, али жели пре скупог експеримента да процени вероватноћу да је то промотер. Примениће постериор декодирање и узети у обзир све путање. Има ситуација где је Витерби алгоритам прикладнији.

Исто тако, сама апликација игра веома важну улогу у одлуци који ће се алгоритам применити.

2.4.5. Учење параметара (енг. *Learning – parameter (re)estimation)*

При употреби скривених Марковљевих модела најтеже је направити сам модел и подесити све параметре. Први део посла је дизајн модела – која стања постоје и како су повезана. Тај део проблема нема сасвим разрађену теорију и сматра се донекле уметношћу. Други део је процена параметара – одређивање транзиционих и емисионих вероватноћа. Овде постоји доста добро разрађена теорија и сада ћемо дати њен преглед.

Постоје два случаја - ако знамо “скривену“ секвенцу стања и ако је не знамо.

2.4.5.1. Вођено учење (енг. *Supervised Learning*)

Ово је лакши случај и чест је у пракси, на пример, када имамо ДНК секвенце које су већ обележене на основу експерименталних података.
У случају непоштене коцкарнице, коцкар би нам омогућио да га гледамо кад мења фер и нефер коцку.

Дата је секвенца x и одговарајућа путања (секвенца стања). Тражимо транзиционе и емисионе вероватноће које ће максимизирати P(x,π).
Нека је A_{km} број транзиција из стања k у стање m у путањи.
Нека је $E_k(b)$ број емитовања симбола b из стања k у путањи.

Метода максималне веродостојности ових параметара даје

$$a_{km} = A_{km} / \Sigma_m A_{km}$$
$$e_k(b) = E_k(b) / \Sigma_d E_k(d)$$

што потпуно одговара интуицији да су они просечна фреквенција појављивања транзиција и емисија у датим секвенцама.

a_{km} = број стања m после стања k / број стања k
$e_k(b)$ = број емитовања симбола b из стања k/ број стања k

За почетне вероватноће се могу узети тоталне фреквенције стања у путањи.

Овај метод може довести до проблема преподешавања (енг. *overfitting*). Проблем је нарочито изражен ако имамо мало података. Бројање неких транзиција и емисија може да буде 0, а да вероватноћа не би требала бити 0 (мала је али није 0). То може да произведе погрешан скривени Марковљев модел који би такве транзиције и емисије сматрао немогућим. Проблем се решава псеудобројањем. На пример, може се користити Лапласова формула, где би се избројало да таквих транзиција и емисија има по једна уместо 0 (вероватноћа би им била 1/нешто). Други приступ је да се узму вероватноће које подразумевају неко претходно знање о проблему (за случај 0).
A_{km} = (број транзиција из стања k у стање m у путањи) + R_{km}
$E_k(b)$ = (број емитовања симбола b из стања ky путањи) + $R_k(b)$

2.4.5.2. Невођено учење (енг. *Unsupervised Learning*)

Овде не знамо секвенцу стања и не постоји формула као у претходном случају. Зато ћемо морати итеративно да рачунамо параметре и биће потребно користити неки од метода оптимизације. Сви методи за оптимизацију непрекидних функција могу се применити а највише је у употреби Баум-Велч

(енг. Baum-Welch) алгоритам. Користи се и алгоритам Витерби обучавања (енг. Viterbi training).
Основна идеја је да после почетне процене параметара модела (енг. best guess) моделом израчунамо путању и опет на основу ње проценимо параметре и опет рачунамо путању и то све понављамо док не задовољимо неки критеријум.

Баум-Велч (енг. Baum-Welch) алгоритам максимизовања очекивања

Тражићемо параметре система тако да добијемо максимум од

$$P(x) = \Sigma_{\pi} P(x,\pi).$$

Да бисмо израчунали A_{km} , уместо да бројимо колико пута ће стање m да буде после стања k, ми ћемо сабирати за све позиције i, вероватноћу да је стање k на позицији i, а да је стање m на позицији i+1.
Ово је веома слично рачунању код постериор декодирања, са разликом да овде специфицирамо две позиције уместо једне. И овде ћемо користити унапред и уназад вероватноће али ћемо морати додати између њих израз за транзицију између позиција i и i+1:

$$A_{km}=\Sigma_i P(\pi_i = k, \pi_{i+1}=m \mid x,\theta)=(\Sigma_i f_k(i)\, a_{km}\, e_m(x_{i+1})\, b_m(i+1))/P(x|\,\theta)$$

(θ је скуп свих параметара)
Слично ћемо рачунати и $e_k(b)$ користећи уназад и унапред вероватноће.

$$E_k(b) = (\Sigma_{x_i=b} f_k(i)\, b_k(i)\,)/\, P(x|\theta)$$

Баум-Велч алгоритам:
Почетак:
Изабрати параметре (најбоља процена).
Итерација:
Израчунати унапред и уназад вероватноће и затим A_{km} и $E_k(b)$.
Израчунати нове вредности за параметре

$$a_{km} = A_{km} / \Sigma_d A_{kd}$$

$$e_k(b) = E_k(b) / \Sigma_d E_k(d)$$

Израчунати нову вероватноћу модела $P(x|\theta_{new})$, и ставити $\theta = \theta_{new}$.
Нова вероватноћа модела је гарантовано већа [3].
Крај:
Ако се вероватноћа не мења преко задатог прага.

Време извршавања је $O(K^2N) \cdot$ број итерација.

Баум-Велч гарантује конвергенцију али ка локалном максимуму. Да би се то побољшало, може се цела процедура поновити неколико пута са различитим почетним вредностима за непознате параметре и изабрати најбољи резултат.

Према томе, алгоритам је “осетљив“ на велики број параметара (велики број локалних максимума) и почетне параметре.

2.4.5.3. Витерби учење (енг. *Viterbi training*)

Витерби алгоритам даје у пракси нешто слабије резултате од Баум-Велч алгоритма али има лакшу имплементацију и често се користи. Сматра се да, ако је добар за декодирање и поравнавање, добар је и за учење.

Почетак:
Изабрати параметре (најбоља процена)
Итерација:
Помоћу Витерби алгоритма наћи путању π^*.
Израчунати A_{km} , $E_k(b)$ помоћу путање π^* узимајући у обзир псеудобројање (pseudocounts) .
Израчунати нове параметре a_{km}, $E_k(b)$.
Крај:
Када нема промена у путањи.

Алгоритам конвергира јер је одређивање путање дискретан процес и завршава се кад се путања више не мења. Тада се ни параметри неће више мењати. Овај алгоритам не максимизира $P(x|\theta)$ као Баум-Велч, него $P(x, \pi^*|\theta)$.

2.4.6. Избор топологије модела

До сада смо сматрали да су све транзиције између свих стања могуће. Нормално би било да са таквим моделом почнемо и пустимо да модел сам одреди које ће транзиције да буду могуће и са којом вероватноћом.

Међутим, у пракси то скоро никад не фунционише. Чак и обиље података за учење води ка веома лошем моделу. Проблем није недовољно података за учење него локални максимуми. Што су мања ограничења модела, то је проблем локалних максимума већи.

Било је покушаја да се додавањем или одузимањем могућих транзиција развију општи методи конструкције модела, али у пракси су успешни модели конструисани пажљивим проучавањем природе проблема и на основу тога сазнања одређене су могуће транзиције између стања.

Треба да изаберемо модел тако да интерпретира наше знање о проблему.

Да бисмо онемогућили транзицију из стања k у стање m довољно је да ставимо $a_{km} = 0$. Ако користимо Баум-Велч или Витерби алгоритме a_{km} ће остати 0 после итерација, јер ће очекивани број транзиција бити 0. Дакле, математика се неће променити ако све транзиције нису могуће.

2.5. Пример

Посматрајмо модел са два стања "Непоштена коцкарница" из одељка 1.2. Уводни пример и основни проблеми. Нека је почетни вектор (0.5 0.5), и нека су транзициона и емисиона матрица, редом:

T: 0.9 0.1
0.1 0.9

E: 0.166 0.166 0.166 0.166 0.166 0.166
0.100 0.100 0.100 0.100 0.100 0.500

Нека је секвенца симбола 1234566666. Фер стање ћемо означавати са 0, а нефер са 1.
Приметимо да су у споменутом одељку транзиционе вероватноће биле 0.95 и 0.05.

Применом Витерби алгоритма добићемо следеће:

Симбол	Фер стање	Нефер стање	Траг уназад	
			Фер	Нефер
1	8.300000000E-002	5.000000000E-002	0	0
2	1.240020000E-002	4.500000000E-003	0	1
3	1.852589880E-003	4.050000000E-004	0	1
4	2.767769281E-004	3.645000000E-005	0	1
5	4.135047305E-005	3.280500000E-006	0	1
6	6.177760674E-006	2.067523653E-006	0	0
6	9.229574447E-007	9.303856437E-007	0	1
6	1.378898422E-007	4.186735397E-007	0	1
6	2.060074243E-008	1.884030929E-007	0	1
6	3.127491341E-009	8.478139178E-008	1	1

0.083 = 0.5 * 0.166 (почетна вероватноћа * вероватноћа емитовања симбола 1 из Фер стања)
0.05 = 0.5 * 0.1 (почетна вероватноћа * вероватноћа емитовања симбола 1 из нефер стања)

Израчунаћемо другу вредност у колони Нефер стање да илуструјемо како даље ради алгоритам.
Бирамо максимум од две вредности: 0.083 * 0.1 = 0.0083 и 0.05 * 0.9 = 0.045.
Већа је вредност ако остајемо у Нефер стању и у траг матрицу уписујемо 1 (одакле долазимо).
0.045 множимо са вероватноћом да из Нефер стања емитујемо симбол 2
0.045 * 0.1 = 0.0045 и то уписујемо у Нефер стање Витерби матрице.
Слично рачунамо остале вредности.

Интерпретација матрице трага уназад је следећа:
Секвенцу стања градимо уназад.
У последњем реду видимо да је највећа вероватноћа у Нефер стању и износи $8.478 * 10^{-8}$.
То значи да је последње стање у траженој секвенци стања Нефер. Из њега је емитован последњи симбол.
Сада почињемо да користимо матрицу траг уназад.

У последњој врсти матрице траг уназад, у колони Нефер стање читамо да смо дошли из Нефер стања. Слично радимо и са претходним врстама док не добијемо секвенцу Нефер, Нефер, Нефер, Нефер, Нефер.
Следеће читање је 0 (шеста врста, нефер колона), а то значи да смо дошли из Фер стања. Онда гледамо у пету врсту где је колона Фер стање и ту је 0.
Тако идемо до друге врсте (прву врсту не користимо), те је коначно секвенца стања од почетка:
Фер, Фер, Фер, Фер, Фер, Нефер, Нефер, Нефер, Нефер, Нефер. Или:

Секв. симбола	1	2	3	4	5	6	6	6	6	6
Секв. стања	0	0	0	0	0	1	1	1	1	1

Подсетимо се да је вероватноћа да је ова секвенца генерисана нашим моделом из секвенце стања 1234566666 приближно $8.478139178 * 10^{-8}$.

Матрица Унапред се рачуна као и Витерби само се сабира уместо тражења максимума. Уназад се рачунање разликује али није компликовано. Дајемо само резултате:

Матрица Унапред:

Фер стање	Нефер стање
8.300000000E-002	5.000000000E-002
1.323020000E-002	5.330000000E-003
2.065069880E-003	6.120020000E-004
3.186806733E-004	7.573087880E-005
4.886802517E-005	1.000258582E-005
7.466925886E-006	6.944564880E-006
1.230838504E-006	3.498400490E-006
2.419607207E-007	1.635822146E-006
6.330357929E-008	7.482180017E-007
2.187797357E-008	3.398632797E-007

Матрица Уназад:

Фер стање	Нефер стање
3.180744623E-006	1.954788992E-006
2.008427154E-005	1.801544538E-005
1.225474386E-004	1.775684211E-004
6.968061195E-004	1.844460439E-003
3.333429695E-003	1.987917230E-002
7.621758256E-003	4.389478024E-002
1.860018378E-002	9.685781598E-002
5.312036000E-002	2.132800400E-001
1.994000000E-001	4.666000000E-001
1.000000000E+000	1.000000000E+000

Вероватноћу да наш модел генерише секвенцу 1234566666 можемо звати унапред збир или уназад збир и она износи 3.617412533E-007. Добија се ако се саберу елементи последњег реда матрице Унапред или ако се на прву врсту матрице Уназад примени формула (крај Уназад алгоритма) из одељка 1.4.3. :
$0.5 \cdot 0.166 \cdot 3.180744623E\text{-}006 + 0.5 \cdot 0.1 \cdot 1.954788992E\text{-}006$.

Матрица постериор декодирања:
7.298083956E-001 2.701916044E-001
7.345552297E-001 2.654447703E-001
6.995857456E-001 3.004142544E-001
6.138604355E-001 3.861395645E-001
4.503166968E-001 5.496833032E-001
1.573254460E-001 8.426745540E-001
6.328783953E-002 9.367121605E-001
3.553103350E-002 9.644689665E-001
3.489437159E-002 9.651056284E-001
6.047962010E-002 9.395203799E-001

У прва четири реда вероватноћа Фер стања (прва колона) је већа а после је већа вероватноћа Нефер стања. Секвенца стања је 0 0 0 0 1 1 1 1 1 1.

Вратимо се на пример 3. из одељка 2.3.2. Транзиционе вероватноће су 0.95 и 0.05.
Витерби матрица:
8.300000000E-002 5.000000000E-002 0 0
1.308910000E-002 4.750000000E-003 0 1
2.064151070E-003 4.512500000E-004 0 1
3.255166237E-004 4.286875000E-005 0 1
5.133397156E-005 4.072531250E-006 0 1
8.095367316E-006 1.934452344E-006 0 1
1.276639426E-006 9.188648633E-007 0 1
2.013260374E-007 4.364608101E-007 0 1
3.174911610E-008 2.073188848E-007 0 1
5.006835609E-009 9.847647027E-008 0 1
Витерби алгоритам даје :
1 2 3 4 5 6 6 6 6 6
1 1 1 1 1 1 1 1 1 1
Вероватноћа је $9.847647027*10^{-8}$, а тако је и у ранијем примеру.

Ако додамо симбол 5 на почетак добијамо
5 1 2 3 4 5 6 6 6 6 6
0 0 0 0 0 0 1 1 1 1 1
Витерби матрица:
8.300000000E-002 5.000000000E-002 0 0
1.308910000E-002 4.750000000E-003 0 1
2.064151070E-003 4.512500000E-004 0 1
3.255166237E-004 4.286875000E-005 0 1
5.133397156E-005 4.072531250E-006 0 1
8.095367316E-006 3.868904688E-007 0 1
1.276639426E-006 2.023841829E-007 0 0
2.013260374E-007 9.613248687E-008 0 1
3.174911610E-008 4.566293126E-008 0 1
5.006835609E-009 2.168989235E-008 0 1
7.895779756E-010 1.030269887E-008 0 1

3. НЕКЕ ВАЖНИЈЕ ПРИМЕНЕ СКРИВЕНИХ МАРКОВЉЕВИХ МОДЕЛА У БИОИНФОРМАТИЦИ

У биоинформатици, алгоритми засновани на СММ примењени су на анализу биолошких секвенци, на проналажење гена, на класификацију протеинских фамилија итд. У том смислу се може поставити, на пример, питање: да ли одређена секвенца припада одређеној фамилији секвенци, или, шта се може рећи о структури секвенце ако претпоставимо да она припада одређеној фамилији. У случају протеинских секвенција питање би се могло односити на идентификацију алфа хеликса и бета трака. Скривени Марковљеви модели су посебно погодни за ту врсту проблема и овде ћемо описати неке њихове важније примене.

3.1. Двоструки скривени Марковљеви модели *(енг. pair HMM)*

Модели за поравнање две секвенце који једноставно представљају празнине, где је скор празнина пропорционалан дужини, није идеалан за биолошке секвенце. Такви модели кажњавају додатне празнине исто као прву. У биолошким секвенцама празнине се обично не јављају појединачно и свака представља посебан мутациони догађај.

Ако нам је дата општа функција казне празнине m(g), за рачунање скора поравнања префикса секвенци s1 и s2 дужине i односно j, редом, можемо користити верзије динамичког програмирања, па ће уз подешавања рекурентна веза бити:

$$F(i,j) = \max \{ F(i-1, j-1) + s(x_i, y_j) ;$$
$$F(k, j) + m(i - k), \quad (k = 0, 1, 2, \ldots i-1) ;$$
$$F(i, k) + m(j - k), \quad (k = 0, 1, 2, \ldots, j - 1) ; \quad \}$$

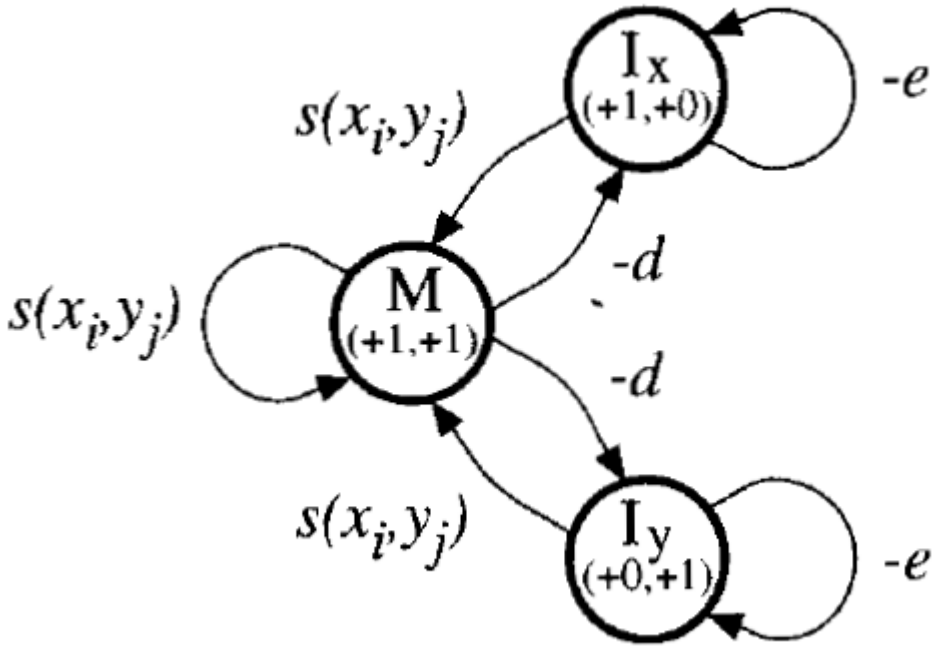
Међутим, ово израчунавање сада захтева $O(n^3)$ операција за поравнање две секвенце дужине n , уместо $O(n^2)$ у верзији линеарног рачунања казне за празнину. То је неприхватљиво повећање времена израчунавања. Стандардна алтернатива је афина казнена функција: $m(g) = - d - (g - 1) * e$. Време је $O(n^2)$ али ћемо морати рачунати три вредности уместо једне: ако се упарују оба елемента секвенци (M), елемент и празнина (I_x), празнина и елемент друге секвенце (I_y).

Рекурентне везе ће бити:

$$M(i, j) = \max \{ M(i - 1, j - 1) + s(x_i, y_j) ;$$
$$I_x(i - 1, j - 1) + s(x_i, y_j) ;$$
$$I_y(i - 1, j - 1) + s(x_i, y_j) ; \quad \}$$
$$I_x(i, j) = \max \{ M(i - 1, j) - d ;$$
$$I_x(i - 1, j) - e ; \quad \}$$
$$I_y(i, j) = \max \{ M(i, j - 1) - d ;$$
$$I_x(i, j - 1) - e ; \quad \}$$

Поравнање се стандардно налази пратећи уназад путању *(енг. traceback procedure).*

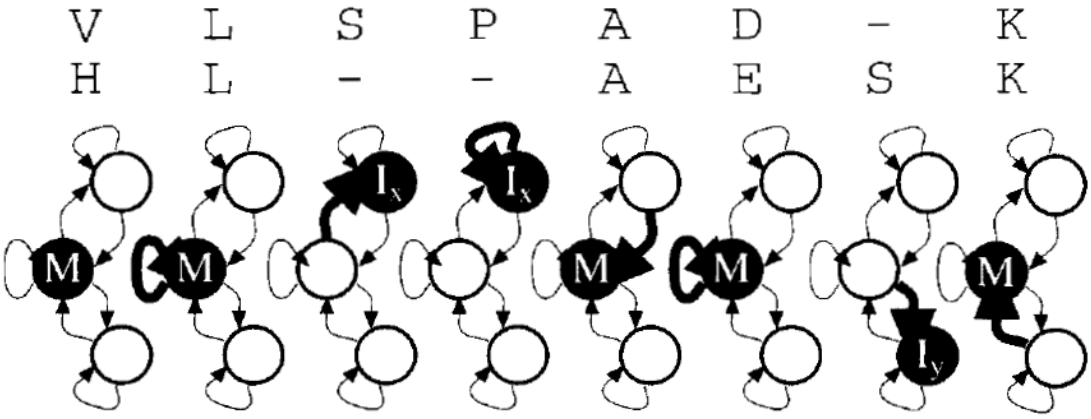
Овај систем се може елегантно приказати као на слици 3.1. Може се направити 1 – 1 пресликавање између поравнања и низа генерисаног моделом са слике 3.1. Низ генеришу стања аутомата M, I_x , I_y . Збир "+ 1" израза за x је једнак дужини секвенце x , док је збир "+ 1" израза за y једнак дужини секвенце y.



Слика 3.1. Коначни аутомат за поравнање секвенци

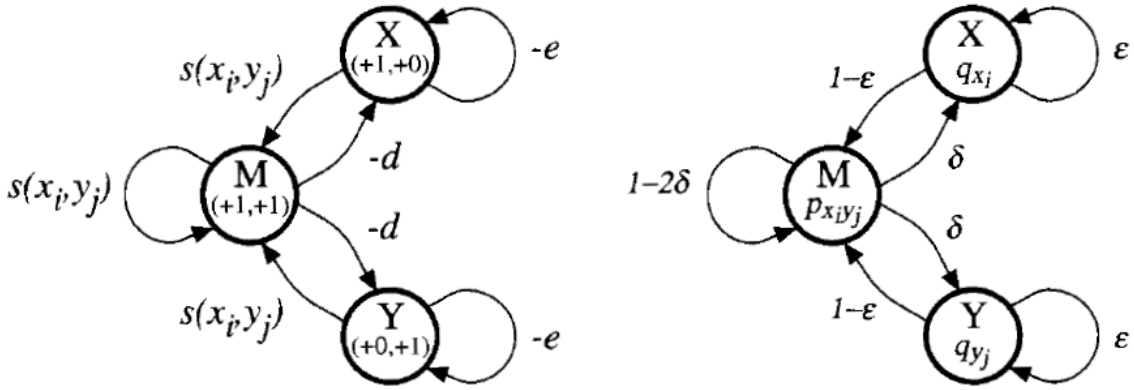
Свака путања кроз овај модел одговара једном поравнању. Ако израчунамо скор овим моделом добићемо исти резултат као код глобалног поравнања.

Пример (вежбање 10.2 из [1]). Израчунај скор поравнања са слике 3.2 ако је d = 12, а e = 2.
S = s(V,H) + s(L,L) – d – e +s(A,A) + s(D,E) – d + s(K,K)
S = – 3 + 4 – 12 – 2 + 4 + 2 – 12 + 5 = – 12
Коришћена је уобичајена матрица сличности аминокиселина.



Слика 3.2 Пример генерисања поравнања коначним аутоматом.

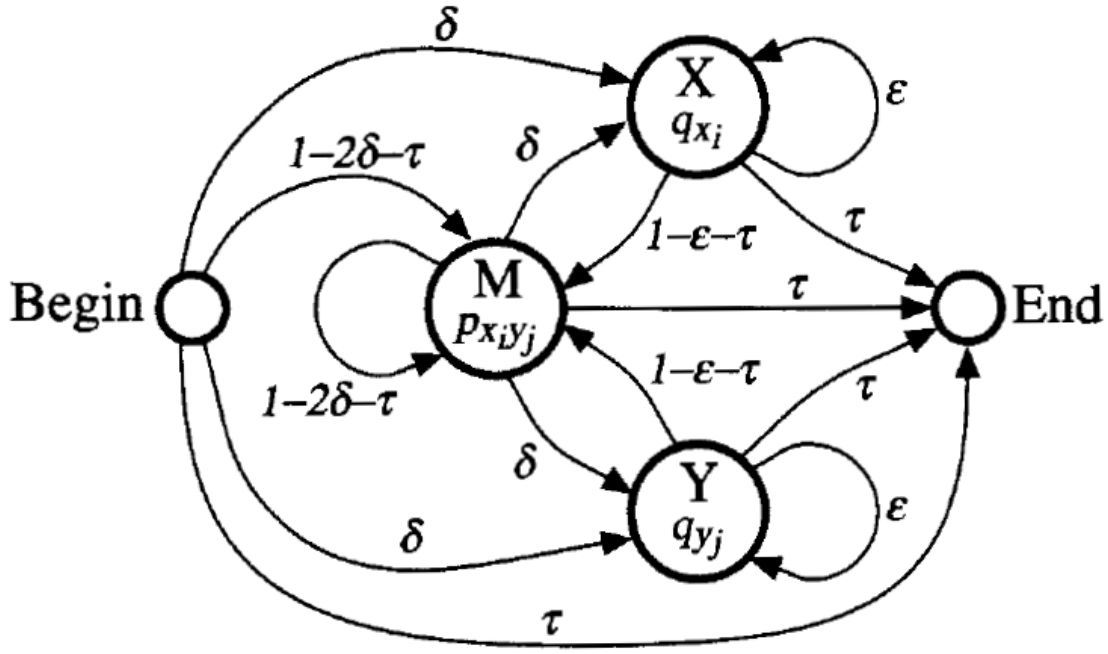
Следећи модел (Слика 3.3 лево) даје нам идеју како да направимо скривени Марковљев модел за поравнање две секвенце (Слика 3.3 десно).



Слика 3.3. Коначни аутомат за поравнање две секвенце (лево) и одговарајући статистички модел (десно).

На слици 3.2. десно, у круговима су емисионе вероватноће, транзиционе вероватноће су поред лукова.

Да бисмо проширили скуп могућих генерисаних секвенци до нивоа наших потреба, додаћемо још два стања: почетно и завршно. Неопходне су и нове транзиционе вероватноће. Тако долазимо до модела на слици 3.4.



Слика 3.4. Потпуни статистички модел са слике 3.3 десно

За разлику од стандардног скривеног Марковљевог модела овај модел ће уместо једне секвенце емитовати две поравнате секвенце. То наравно захтева и нови назив: двоструки скривени Марковљев модел (ДСММ). Сви алгоритми који се користе код стандардног СММ могу се применити и код ДСММ.

За ДСММ, Витерби алгоритам изгледаће овако:

Почетак:
$V^M(0, 0) = 1$;
$V^*(i, 0) = 0$, $V^*(0, j) = 0$, за све остале (i,j)

Итерација: (i = 1, 2, …, n, j = 0, 1, … , m)
$V^M(i, j) = p_{xiyj} * \max \{ (1 – 2*\delta – t) * V^M(i – 1 , j – 1) ;$
$(1 – \varepsilon – t) * V^X(i – 1 , j – 1) ;$
$(1 – \varepsilon – t) * V^Y(i – 1 , j – 1) ; \}$

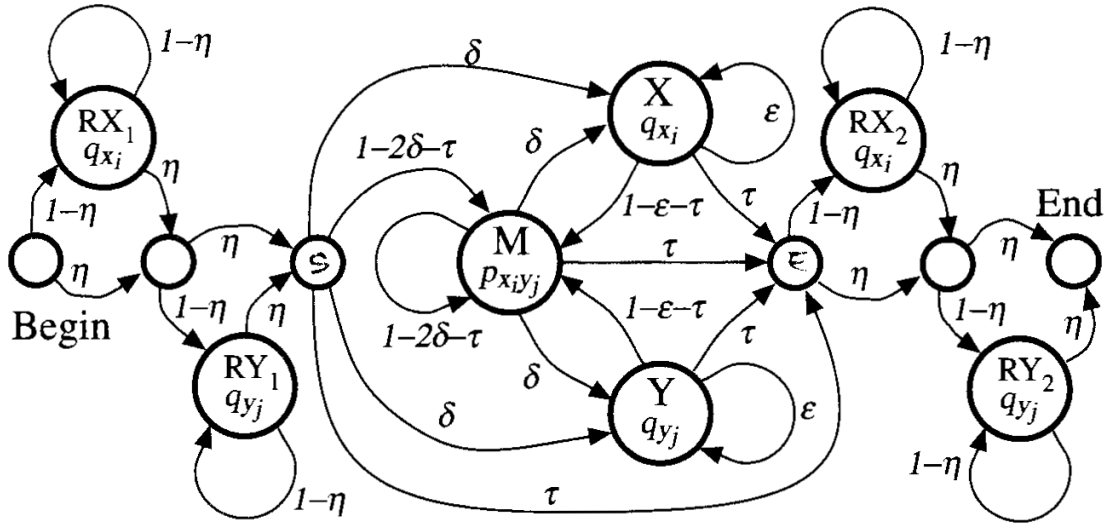
$V^X(i, j) = q_{xi} * \max \{ 2 * \delta * V^M(i – 1 , j) ;$
$\varepsilon * V^X(i – 1 , j) ; \}$

$V^X(i, j) = q_{yi} * \max \{ 2 * \delta * V^M(i , j – 1) ;$
$\varepsilon * V^Y(i , j – 1) ; \}$

Крај:
$Viterbi = t * \max(V^M(n, m), V^X(n, m), V^Y(n, m))$

Опширније и детаљније о трансформацији у логаритамски простор и о другим раније споменутим алгоритмима може се видети на пример у књизи [1].

Могуће је изградити ДСММ и за локално поравнање пара секвенци (Слика 3.5.).



Слика 3.5. ДСММ за локално поравнање пара секвенци

Иако делује доста компликованије него модел за глобално поравнање, пажљивијом анализом Слике 3.5. може се видети да је само додат мини модел на почетку и на крају. Мини модел је у ствари случајни *(енг. random)* модел који на почетку и крају поравнања секвенци може да генерише непоравнате секвенце произвољне дужине.

3.2.Скривени Марковљеви Модели за конструкцију профила (ПСММ) (енг: Profile Hidden Markov Models)

3.2.1. Увод

Функционалне биолошке секвенце обично иду у фамилијама и многи методи за њихову анализу су засновани на одређивању односа између секвенце и фамилије. Одређивање да ли секвенца припада фамилији и поравнање са осталим члановима често омогућава сазнање о њеној функцији. Упоређивање по две секвенце може се користити у решавању овог задатка, али тим поступком могу да се превиде секвенце које су у даљој релацији са фамилијом.

Чак и када знамо да секвенца припада фамилији, њено поравнање се може поправити ако узмемо у обзир делове који су добро очувани.

Конструисаћемо скривени Марковљев модел који ће бити заснован на концензус секвенци (сатављеној од симбола који су заступљени, на пример, више од пола на некој позицији). Зваћемо га скривени Марковљев модел за конструкцију профила (ПСММ).

Претпоставићемо да је дато вишесеквенцијално поравнање, на основу кога ћемо изградити модел и користити га да нађемо нове чланове фамилије те да их поравнамо са осталим члановима.

Ако погледамо неко вишесеквенцијално пораванање, приметићемо да се празнине групишу а да преостали симболи формирају блокове. Модел ћемо почети да градимо на тим блоковима без празнина.

Једноставним рачунањем се долази до матрице где редови представљају симболе, а колоне позицију у вишесеквенцијалном поравнању. Сваки елемент матрице је број који представља релативну фреквенцију појављивања симбола (у чијем је реду елемент) на позицији која одговара његовој колони. Ево једноставног примера.

Ако је поравнање

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

Матрица је

A	1	0	0.75	0	0
T	0	1	0	0.25	0.75
C	0	0	0.25	0	0.25
G	0	0	0	0.75	0

Оваква матрица се користи као први корак у изградњи ПСММ. Свака колона ће бити представљена једним стањем у ПСММ. Транзиционе вероватноће ће бити 1 (за сада – после ће бити другачије кад додајемо нова стања) између стања а емисионе ће одговарати елементима матрице (Слика 3.6).

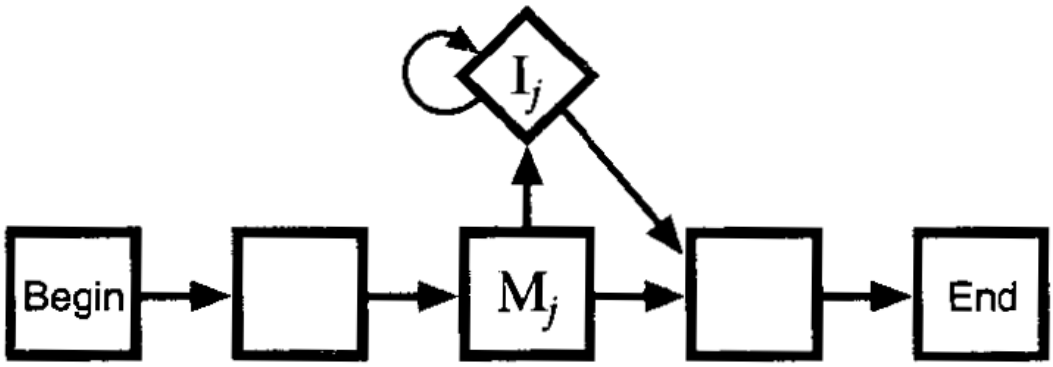
Слика 3.6. Први корак конструкције ПСММ.

На почетку смо додали почетно стање (енг.: *begin*), а на крају крајње стање (енг.: *end*). Овакав тривијалан СММ моделира одговарајући блок (матрицу) вишеструког поравнања. Стања између почетног и крајњег ћемо звати стања поклапања (енг: *match*) и обележавати са M_j.

Следећи корак је да моделујемо стања уметања (енг: *insertions*). Увешћемо нова стања, звати их стање уметања (енг: *insertion*) и обележавати их са I_j (слика 3.7).

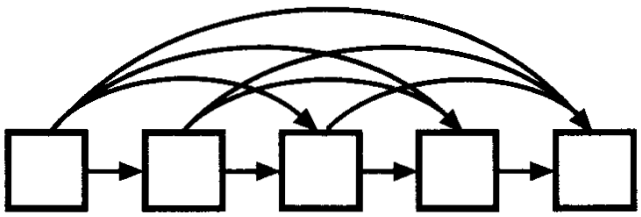
Ако смо имали рецимо k стања поклапања , овде ћемо имати k+1 стање уметања. После сваког поклапање стања биће једно уметање стање и после почетног стања биће једно уметање стање (слика 3.10).

Потребно је још одредити емисионе вероватноће као и транзиционе вероватноће за транзицију из поклапање стања M_k у I_k уметање стања, повратна транзиција I_k на само себе, те транзицију од стања I_k у стање M_{k+1}. Графички приказ једног уметање стања представљен је на Слици 3.7. (приказано као ромб).



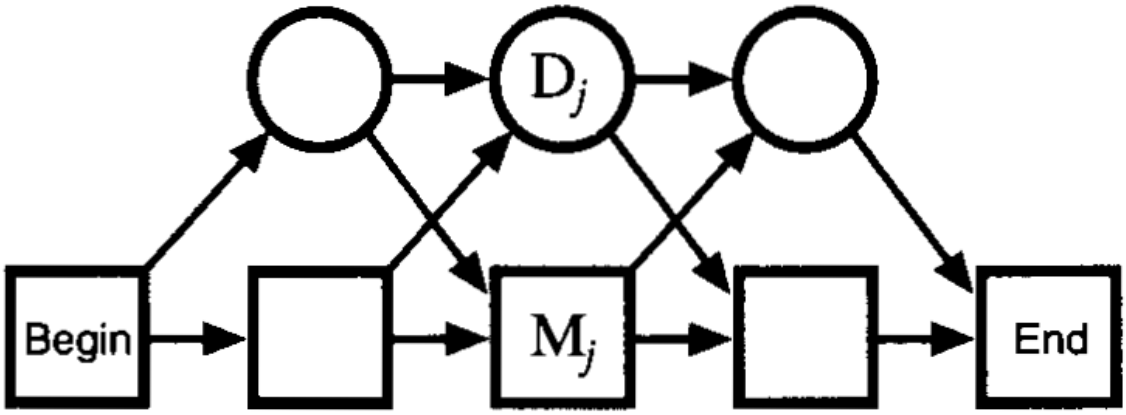
Слика 3.7. Уметање стање

Увешћемо и стања која ћемо звати стања брисања (енг. *deletion*). Делови поравнања који не одговарају стањима која емитују симболе, могли би се моделовати транзицијама које би биле скокови унапред као на Слици 3.8.:



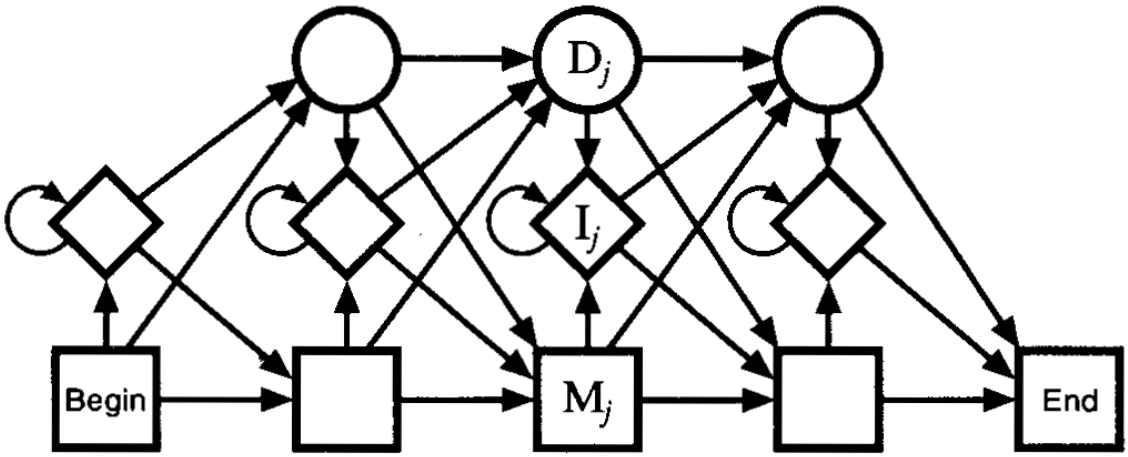
Слика 3.8. Моделовање поравнања транзицијама унапред.

Међутим, то би захтевало много транзиција, па ћемо уместо тога увести тиха (енг. *silent)* стања (која не емитују симболе) као на Слици 3.9.)



Слика 3.9. Тиха стања

Тиха стања су обележена круговима и могуће је користити више њих у низу да би се стигло из неког поклапање стања у неко касније поклапање стање. Коначно добијамо следећи модел (Слика 3.10.):



Слика 3.10. Скривени Марковљев модел за конструкцију профила

Слика 3.10. приказује које ће транзиције бити могуће у моделу који ћемо звати скривени Марковљев модел за конструкцију профила (ПСММ).

Пример за конструкцију ПСММ је у одељку 3.2.4.

3.2.2. Основна параметризација ПСММ (енг. Basic profile HMM parameterization)

ПСММ може свим могућим секвенцама симбола датог алфабета доделити вероватноћу. Тако дефинише расподелу вероватноћа у целом простору секвенци. Наш циљ је да одредимо параметре модела тако да ова расподела достиже највећу вредност око чланова фамилије. На ову расподелу утичу транзиционе и емисионе вероватноће као и дужина модела. Дужина модела зависи од броја поклапања и уметања стања. Једна проста метода је хеуристика која веома добро ради у пракси и каже да ако у колони вишеструког поравнања има више од пола празнина она треба да буде моделована уметање стањем. Остале колоне ће наравно бити моделоване поклапање стањима.

Како да проценимо параметре модела?

Пошто нам је дато вишеструко поравнање, можемо да пребројимо појављивања одговарајућих транзиција и емисија слично већ виђеној идеји код учења.

Нека је A_{km} број транзиција из стања k у стање m у путањама.

Нека је $E_k(b)$ број емитовања симбола b из стања k у путањама.

$a_{km} = A_{km} / \Sigma_m A_{km}$

$e_k(b) = E_k(b) / \Sigma_d E_k(d)$

или

a_{km} = број колико је пута стање m после стања k / колико пута стање k

$e_k(b)$ = колико пута се из стања k емитвао симбол b / колико пута стање k

Индекси означавају стања, a_{km}, $e_k(b)$ вероватноће, а A_{km}, E_k одговарајуће фреквенције.

Сетимо се и проблема када имамо мало података (само неколико секвенци у вишеструком поравнању. Пребројавање неких транзиција и емисија може да да 0, а да вероватноћа не би требала бити 0, што се решава псеудобројањем где би се избројало да таквих транзиција и емисија има по једна уместо 0. То је Лапласово правило (енг. *Laplace's rule*).

Поред примене методе максималне веродостојности која даје горње формуле и псеудобројања, постоје и други (бољи) приступи процени параметара.

3.2.3. Формуле за ПСММ

ПСММ користимо за одређивање да ли нека секвенца припада фамилији. То ћемо да урадимо упоређујући саквенцу са моделом. Прво ћемо да утврдимо припадност фамилији глобално, а после локално.

Користећи СММ можемо да бирамо да ли ћемо да користимо Витерби алгоритам (највероватнија секвенца стања π^*) за $P(x, \pi^* \mid M)$ или да рачунамо пуну вероватноћу секвенце за $P(x \mid M)$ користећи алгоритме унапред и уназад за сумирање по свим путањама.

Резултат који ћемо разматрати када израчунавамо потенцијална стања поклапања биће логаритамски однос вероватноће модела и вероватноће стандардног случајног (енг. *random*) модела

$$P(x \mid R) = \Pi_i q_{xi}$$

где се са q_{xi} обележава фреквенција симбола x_i у ДНК секвенци.

Прелазак на логаритме неће променити резултат (однос), израчунавање ће бити јасније и ефикасније, а и избећи ћемо проблеме са израчунавањем веома малих вредности.

Формуле за Витерби алгоритам

$V_j^M(i)$ је логаритамски однос најбоље путање и случајне модел секвенце до стања j завршно емитовањем x_i из стања M_j . Слично $V_j^I(i)$ је логаритамски

однос завршно са емитовањем x_i из стања I_j , и $V_j^D(i)$ је логаритамски однос за најбољу путању која завршава у D_j . Ево једначина у компактном облику:

$$V_j^{\mathrm{M}}(i) = \log \frac{e_{\mathrm{M}_j}(x_i)}{q_{x_i}} + \max \begin{cases} V_{j-1}^{\mathrm{M}}(i-1) + \log a_{\mathrm{M}_{j-1}\mathrm{M}_j}, \\ V_{j-1}^{\mathrm{I}}(i-1) + \log a_{\mathrm{I}_{j-1}\mathrm{M}_j}, \\ V_{j-1}^{\mathrm{D}}(i-1) + \log a_{\mathrm{D}_{j-1}\mathrm{M}_j}; \end{cases}$$

$$V_j^{\mathrm{I}}(i) = \log \frac{e_{\mathrm{I}_j}(x_i)}{q_{x_i}} + \max \begin{cases} V_j^{\mathrm{M}}(i-1) + \log a_{\mathrm{M}_j\mathrm{I}_j}, \\ V_j^{\mathrm{I}}(i-1) + \log a_{\mathrm{I}_j\mathrm{I}_j}, \\ V_j^{\mathrm{D}}(i-1) + \log a_{\mathrm{D}_j\mathrm{I}_j}; \end{cases}$$

$$V_j^{\mathrm{D}}(i) = \max \begin{cases} V_{j-1}^{\mathrm{M}}(i) + \log a_{\mathrm{M}_{j-1}\mathrm{D}_j}, \\ V_{j-1}^{\mathrm{I}}(i) + \log a_{\mathrm{I}_{j-1}\mathrm{D}_j}, \\ V_{j-1}^{\mathrm{D}}(i) + \log a_{\mathrm{D}_{j-1}\mathrm{D}_j}. \end{cases}$$

У случају да почетак или крај секвенце не одговара првом или задњем стању преклапања, морамо да омогућимо поравнање из уметање или брисање стања.

Прво, почетно стање преименоваћемо у M_0 и биће $V_0^M(0)=0$. Омогућићемо транзицију до I_0 и D_0 . Слично ћемо могући завршетак у уметање и брисање стању сакупити у завршном стању, преименовати га у M_{L+1} и израчунати као $V_{L+1}^M(n)$ без емисионог сабирка (први сабирак) као коначан резултат.

Формуле за Унапред алгоритам

Формуле за Унапред алгоритам се разликују од Витерби формула по томе што је максимум замењен сабирањем. Мада на први поглед изгледају компликовано (коришћен је алгебарски трик $xy = \log(e^x+e^y)$), имплементација није тешка.

$$
\begin{aligned}
F_j^{\mathrm{M}}(i) &= \log \frac{e_{\mathrm{M}_j}(x_i)}{q_{x_i}} + \log\left[a_{\mathrm{M}_{j-1}\mathrm{M}_j} \exp\left(F_{j-1}^{\mathrm{M}}(i-1)\right)\right. \\
&\qquad \left. + a_{\mathrm{I}_{j-1}\mathrm{M}_j} \exp\left(F_{j-1}^{\mathrm{I}}(i-1)\right) + a_{\mathrm{D}_{j-1}\mathrm{M}_j} \exp\left(F_{j-1}^{\mathrm{D}}(i-1)\right)\right]; \\
F_j^{\mathrm{I}}(i) &= \log \frac{e_{\mathrm{I}_j}(x_i)}{q_{x_i}} + \log\left[a_{\mathrm{M}_j\mathrm{I}_j} \exp\left(F_j^{\mathrm{M}}(i-1)\right)\right. \\
&\qquad \left. + a_{\mathrm{I}_j\mathrm{I}_j} \exp\left(F_j^{\mathrm{I}}(i-1)\right) + a_{\mathrm{D}_j\mathrm{I}_j} \exp\left(F_j^{\mathrm{D}}(i-1)\right)\right]; \\
F_j^{\mathrm{D}}(i) &= \log\left[a_{\mathrm{M}_{j-1}\mathrm{D}_j} \exp\left(F_{j-1}^{\mathrm{M}}(i)\right) + a_{\mathrm{I}_{j-1}\mathrm{D}_j} \exp\left(F_{j-1}^{\mathrm{I}}(i)\right)\right. \\
&\qquad \left. + a_{\mathrm{D}_{j-1}\mathrm{D}_j} \exp\left(F_{j-1}^{\mathrm{D}}(i)\right)\right].
\end{aligned}
$$

Почетни и крајњи услови су као код Витерби алгоритма.

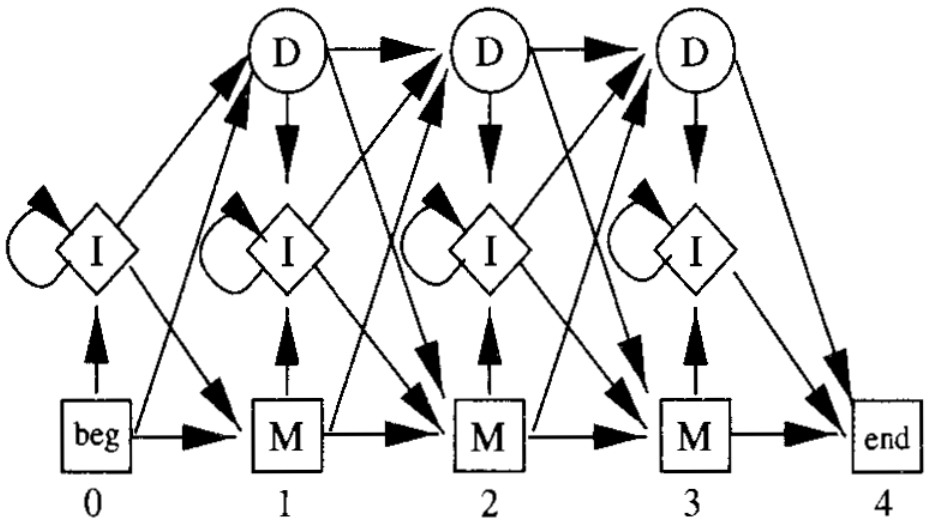
3.2.4. Пример конструкције ПСММ

Основни проблеми конструкције ПСММ су процена параметара вероватноћа и одређивање које колоне вишеструког поравнања секвенци (ВПС) ће одговарати поклапање стањима а које уметање стањима.

Пример:

```
                M M       M
Слепи Миш       A G - - - C
Пацов           A – A G - C
Мачка           A G – A A -
Комарац         - - A A A C
Коза            A G - - - C
                1 2       3
```

Графички, ПСММ модел представљен је на Слици 3.11.



Слика 3.11. ПСММ модел са три стања поклапања.

Колонама вишеструког поравнања секвенци означеним са M или 1,2,3 додељена су стања поклапања. У тим колонама симболи се додељују Поклапање стањима а празнине Брисање стањима. У неозначеним колонама симболи се додељују Уметање стањима а празнине се игноришу. Бројање транзиција и емисија по путањама стања користи се за процену параметара вероватноћа.

Има 2^L (L је дужина ВПС) могућих додељивања колона стањима Поклапање и Уметање (варијације са понављањем). То значи да има 2^L могућих ПСММ.

Избор се може урадити ручно, а може се користити нека хеуристика, на пример колона се може означавати према томе колико је празнина заступљена – одреди се неки праг (рецимо више од пола).

Постоји ефикасан алгоритам са динамичким програмирањем који може да додели колоне одговарајућим стањима и да при томе максимизира вероватноћу модела, а да истовремено оптимално одреди параметре вероватноћа транзиција и емисија. Опис алгоритма је дат у књизи [1].

Овде настављамо са претходним примером (почетак примера је из књиге [1]). Поред тога што је детаљније урађен, допуњен је поравнавањем секвенце AGAGC.

Логаритамском верзијом Витерби алгоритма поравнаћемо секвенцу AGAGC са претходно конструисаним ПСММ.

Графички ПСММ модел изгледа као на слици 3.11.

На сваку од горњих 5 секвенци применићемо следећи алгоритам :

1.Ако колона одговара Поклапање стању

1.1 симболи су прелазак у Поклапање стање

1.2 празнине су прелазак у Брисање стање

2.Ако колона или колоне одговарају Уметање стању

2.1 Ако у колонама које одговарају том Уметање стању **све празнине** прескаче се стање (игноришу се празнине)

2.2 Ако у колонама које одговарају том Уметање стању **има само један симбол** онда је то долазак у Уметање стање а одлазак је у Брисање стање (следећа је празнина у Поклапање стању) или Поклапање стање (следећи је симбол у Поклапање стању)

2.3 Ако у колонама које одговарају том стању **има више симбола** онда:

2.3.1 Први симбол значи да се прешло у Уметање стање

2.3.2 Сваки следећи симбол у том истом стању значи прелазак у самог себе

2.3.3 Када се од последњег симбола истог стања иде даље на следеће (поклапање) стање

2.3.3.1 Ако је следећи симбол прелази се у Поклапање стање.

2.3.3.2 Ако је следећа празнина прелази се у Брисање стање.

Примењено на горњи ВПС:

```
0       1       2       Уметање             3       4
M0--->M1--->M2---------------------------> M3--->M4
M0--->M1--->D2---> I2 ---> I2 ------------> M3--->M4
M0--->M1--->M2-----------> I2 ---> I2 ---> D3---> M4
M0--->D1---> D2---> I2 ---> I2 ---> I2 ---> M3--->M4
M0--->M1--->M2---------------------------> M3--->M4
```

Остаје само да се преброји.

Параметре за ПСММ можемо израчунати (проценити) користећи Лапласову методу – све се увећа за 1.

		0	1	2	3
Емисије из Поклапање стања + 1	A	-	5	1	1
	C	-	1	1	5
	G	-	1	4	1
	T	-	1	1	1
Емисије из Уметање стања + 1	A	1	1	7	1
	C	1	1	1	1
	G	1	1	2	1
	T	1	1	1	1
Транзиције +1	M-M	5	4	3	5
	M-D	2	2	1	-
	M-I	1	1	2	1
	I-M	1	1	3	1
	I-D	1	1	2	-
	I-I	1	1	5	1
	D-M	-	1	1	2
	D-D	-	2	1	-
	D-I	-	1	3	1

Коначно, добијамо емисионе и транзиционе вероватноће:

		0	1	2	3
Емисионе вероватноће за Поклапање стања	A	-	5/8	1/7	1/8
	C	-	1/8	1/7	5/8
	G	-	1/8	4/7	1/8
	T	-	1/8	1/7	1/8
Емисионе вероватноће за Уметање стања	A	1/4	1/4	7/11	1/4
	C	1/4	1/4	1/11	1/4
	G	1/4	1/4	2/11	1/4
	T	1/4	1/4	1/11	1/4
Транзиционе вероватноће	M-M	5/8	4/7	3/6	5/6
	M-D	2/8	2/7	1/6	-
	M-I	1/8	1/7	2/6	1/6
	I-M	1/3	1/3	3/10	1/2
	I-D	1/3	1/3	2/10	-
	I-I	1/3	1/3	5/10	1/2
	D-M	-	1/4	1/5	2/3
	D-D	-	2/4	1/5	-
	D-I	-	1/4	3/5	1/3

$V^M_0(0) = 0$.

Применом Витерби алгоритма за ПСММ на секвенцу AGAGC добићемо Витерби вредност -0.932558 и путању приказану у следећој табели:

	i=0	X_1 = A(i=1)	X_2 = G	X_3 = A	X_4 = G	X_5 = C
M_1	-1e+006	**0.446287**	-3.8712	-3.36038	-6.06843	-7.16704
M_2	-1e+006	-3.3322	**0.71335**	-3.15785	-2.87017	-5.35508
M_3	-1e+006	-4.38203	-3.10906	-0.672944	-1.34807	**-0.750236**
I_0	-1e+006	-2.07944	-3.17805	-4.27667	-5.37528	-6.47389
I_1	-1e+006	-2.77259	-1.49962	-2.59824	-3.69685	-4.79546
I_2	-1e+006	-1.65596	-1.63576	**0.549047**	**-0.462554**	-2.1673
I_3	-1e+006	-4.78749	-3.51453	-2.17702	-2.159	-2.85215
D_1	-1.38629	-3.17805	-4.27667	-5.37528	-6.47389	-7.5725
D_2	-2.07944	-0.806476	-2.59824	-3.69685	-4.79546	-5.89407
D_3	-3.68888	-2.41591	-1.07841	-1.06039	-2.07199	-3.77674

Витерби вредност -0.932558 се добија од $V^M_3(5)$ преко максимума од 3 вредности. Наставимо ли процедуру у назад добићемо следећи низ
$V^M_3(5) \leftarrow V^I_2(4) \leftarrow V^I_2(3) \leftarrow V^M_2(2) \leftarrow V^M_1(1) \leftarrow V^M_0(0)$
Дакле A одговара M1, G одговара M2, A и G одговарају I2 а C одговара M3.
Могући су следећи ВПС:

A	G	–	–	–	C
A	–	A	G	–	C
A	G	–	A	A	–
–	–	A	A	A	C
A	G	–	–	–	C
A	**G**	**A**	**G**	**–**	**C**

A	G	–	–	–	C
A	–	A	G	–	C
A	G	–	A	A	–
–	–	A	A	A	C
A	G	–	–	–	C
A	**G**	**A**	**–**	**G**	**C**

A	G	–	–	–	C
A	–	A	G	–	C
A	G	–	A	A	–
–	–	A	A	A	C
A	G	–	–	–	C
A	**G**	**–**	**A**	**G**	**C**

3.3. Вишеструко поравнање секвенци (ВПС) помоћу ПСММ *(енг: Mutiple Sequence Alignment (MSA) by profileHMM)*

3.3.1. Увод

ВПС мора обично да буде формирано на основу знања које имамо о свакој секвенци посебно. Биолози могу да направе ВПС ручно користећи своје експертско знање које долази са искуством. Међутим, ручна метода је веома напорна те се процес у пракси мора аутоматизовати и направити алгоритам. Биолошко знање треба да претворимо у нумеричке формуле. Тако ће сваки ВПС имати вредност (енг. *score*) и треба да бољи ВПС има бољу вредност.

Шта ВПС значи?

ВПС је поравнање више од две секвенце. У ВПС слични симболи (у случају протеина) су поравнати у колонама и у структурном и у еволуционом смислу. Идеално би било да колона поравнатих симбола (резидуала) заузима у

простору сличну позицију и да су сви дивергирали од истог заједничког претка резидуала (симбола).

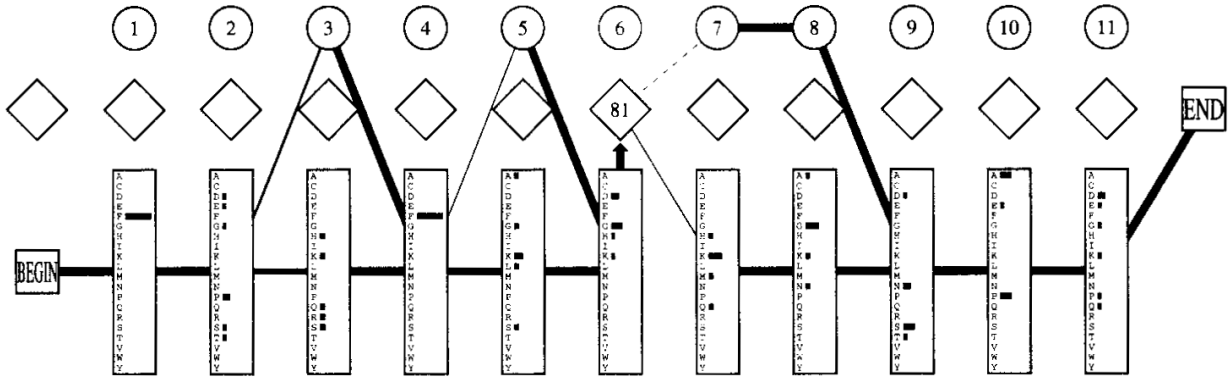
Осим у случају готово идентичних секвенци, није могуће одредити само један коректан ВПС. У принципу, постоји поравнање од кога су настале секвенце, али у пракси је такво поравнање еволуционо теже одредити него структурно. Одређивање структурног поравнања може се ослонити и на кристалографију икс зрацима (енг. *x-ray crystallography*) и нуклеарну магнетну резонанцу (енг- *nuclear magnetic resonance* (*NMR*)). Еволуциону историју резидуала секвенце никако не можемо сазнати, па ни од програма и алгоритама не треба очекивати да сви произведу исто поравнање. Треба се фокусирати на скупове колона резидуала и структурних елемената који се могу поуздано поравнати.

Једна од најчешће коришћених метода за одређивање ВПС је ПСММ.

3.3.2. ВПС ако се зна ПСММ *(енг. Multiple alignment with a known profile HMM)*

Овај проблем се јавља када имамо ВПС и модел од малог скупа секвенци фамилије, и желимо да тај модел користимо да поравнамо већи број секвенци те фамилије.

ВПС ћемо конструисати тако што ћемо наћи Витерби поравнање (путању) за сваку секвенцу. Резидуали (симболи) који су емитовани истим ПСММ поклапање стањем су поравнати и у колони ВПС. Ово ћемо појаснити на примеру.




```
FPHF-DLS-----HGSAQ
FESFGDLSTPDAVMGNPK
FDRFKHLKTEAEMKASED
FTQFAG-KDLESIKGTAP
FPKFKGLTTADQLKKSAD
FS-FLK-GTSEVPQNNPE
FG-FSG----AS---DPG
```

Слика 3.12. Пример за ВПС и одговарајући ПСММ.

Слика 3.12. приказује мањи ПСММ и ВПС . Осенчени резидуали (симболи) ће бити уметање у овом примеру, а осталих једанаест колона одговарају једанаест Поклапање стања. Седам секвенци је поново поравнато са моделом и добијене оптималне Витерби путање су приказане на слици 3.13.

1	2	3	4	5	6	УМЕТАЊЕ	7	8	9	10	11
F	P	H	F	–	D	LS	H	G	S	A	Q
F	E	S	F	G	D	LSTPDAV	M	G	N	P	K
F	D	R	F	K	H	LKTEAEM	K	A	S	E	D
F	T	Q	F	A	G	KDLESI	K	G	T	A	P
F	P	K	F	K	G	LTTADQL	K	K	S	A	D
F	S	–	F	L	K	GTSEVP	Q	N	N	P	E
F	G	–	F	S	G	AS	–	–	D	P	G

Слика 3.13. Поравнање секевенци са моделом

Од ових путања се прави ВПС као што је приказано на слици 3.14. лево где су симболи који одговарају уметање стању приказани малим словима, а они који одговарају поклапање стањима приказани великим словима.

```
FPHF-Dls.....HGSAQ
FESFGDlstpdavMGNPK
FDRFKHlkteaemKASED
FTQFAGkdlesi.KGTAP
FPKFKGlttadqlKKSAD
FS-FLKgtsevp.QNNPE
FG-FSGas.....--DPG
```

```
FS-FLKngvdptaai--NPK
FPHF-Dls.......HGSAQ
FESFGDlstpdav..MGNPK
FDRFKHlkteaem..KASED
FTQFAGkdlesi...KGTAP
FPKFKGlttadql..KKSAD
FS-FLKgtsevp...QNNPE
FG-FSGas.......--DPG
```

Слика 3.14 ВПС добијено са ПСММ (лево) и поравнање са новом секвенцом (десно)

Важно запажање овде је да су почетни ВПС и крајњи ВПС исти. ПСММ не покушава да поравна симболе написане малим словима који одговарају уметање стању (стањима). Они се поравнавају произвољно. Биолошки гледано, ти делови секвенци нису очувани и њихово поравнање нема

функционално значење. У случају протеинских секвенци, делови који се зову петље (енг. loops) су често структурно различити и не могу се поравнати.

Супротно од тога, многи други методи ВПС поравнавају целе секвенце, без обзира да ли поравнање неких деова има биолошки неко значење.

На слици 3.14. десно приказано је поравнање нове секвенце. Ова секвенца има више симбола емитованих уметање стањем него било која од 7 раније поравнатих.

3.3.3. Преглед ПСММ учења за непоравнате секвенце

Проблем је да се направи ПСММ и ВПС од непоравнатих секвенци.

Опис алгоритма:

Почетак: Изабрати дужину ПСММ и задати почетне параметре.

Учење: Проценити модел користећи Баум-Велч алгоритам или Витерби алгоритам. Обично је неопходно да се користи нека хеуристика да се избегне локални максимум.

ВПС: Поравнати све секвенце према коначном моделу користећи Витерби алгоритам и направити ВПС као у ранијем случају (случај кад је познат ПСММ).

<u>Почетни модел</u>

Најчешће коришћено правило је да дужина модела L (могли бисмо рећи број поклапања стања) буде просечна дужина секвенци за учење. Тада је број свих стања 3L+3.

Код одређивања вероватноћа транзиција треба бити веома опрезан. Тако би требало да транзиције у поклапање стања буду вероватније него остале транзиције.

Како Баум-Велч алгоритам налази локалне максимуме, треба да почињемо од различитих тачака (почетних параметара) и да видимо (проверимо) да ли ће конвергирати истом максимуму (глобалном). Дакле, пожељна је случајност (енг. *randomness*) у избору почетних параметара.

Баум-Велч алгоритам

Процена параметара се добија применом већ описаног алгоритма Баум-Велч.
Ево формула прилагођених нотацији ПСММ:

Унапред алгоритам за ПСММ:

ПОЧЕТАК: $f_{\mathrm{M}_0}(0) = 1.$

ИТЕРА-ЦИЈА:

$$f_{\mathrm{M}_k}(i) = e_{\mathrm{M}_k}(i)[f_{\mathrm{M}_{k-1}}(i-1)a_{\mathrm{M}_{k-1}\mathrm{M}_k} + f_{\mathrm{I}_{k-1}}(i-1)a_{\mathrm{I}_{k-1}\mathrm{M}_k} + f_{\mathrm{D}_{k-1}}(i-1)a_{\mathrm{D}_{k-1}\mathrm{M}_k}];$$

$$f_{\mathrm{I}_k}(i) = e_{\mathrm{I}_k}(i)[f_{\mathrm{M}_k}(i-1)a_{\mathrm{M}_k\mathrm{I}_k} + f_{\mathrm{I}_k}(i-1)a_{\mathrm{I}_k\mathrm{I}_k} + f_{\mathrm{D}_k}(i-1)a_{\mathrm{D}_k\mathrm{I}_k}];$$

$$f_{\mathrm{D}_k}(i) = f_{\mathrm{M}_{k-1}}(i)a_{\mathrm{M}_{k-1}\mathrm{D}_k} + f_{\mathrm{I}_{k-1}}(i)a_{\mathrm{I}_{k-1}\mathrm{D}_k} + f_{\mathrm{D}_{k-1}}(i)a_{\mathrm{D}_{k-1}\mathrm{D}_k}.$$

КРАЈ:

$$f_{\mathrm{M}_{M+1}}(L+1) = f_{\mathrm{M}_M}(L)a_{\mathrm{M}_M\mathrm{M}_{M+1}} + f_{\mathrm{I}_M}(L)a_{\mathrm{I}_M\mathrm{M}_{M+1}} + f_{\mathrm{D}_M}(L)a_{\mathrm{D}_M\mathrm{M}_{M+1}}.$$

Уназад алгоритам за ПСММ:

ПОЧЕТАК:

$$b_{\mathrm{M}_{M+1}}(L+1) = 1;$$
$$b_{\mathrm{M}_M}(L) = a_{\mathrm{M}_M\mathrm{M}_{M+1}};$$
$$b_{\mathrm{I}_M}(L) = a_{\mathrm{I}_M\mathrm{M}_{M+1}};$$
$$b_{\mathrm{D}_M}(L) = a_{\mathrm{D}_M\mathrm{M}_{M+1}}.$$

ИТЕРАЦИЈА:

$$b_{\mathrm{M}_k}(i) = b_{\mathrm{M}_{k+1}}(i+1)a_{\mathrm{M}_k\mathrm{M}_{k+1}}e_{\mathrm{M}_{k+1}}(x_{i+1}) + b_{\mathrm{I}_k}(i+1)a_{\mathrm{M}_k\mathrm{I}_k}e_{\mathrm{I}_k}(x_{i+1}) + b_{\mathrm{D}_{k+1}}(i)a_{\mathrm{M}_k\mathrm{D}_{k+1}};$$

$$b_{\mathrm{I}_k}(i) = b_{\mathrm{M}_{k+1}}(i+1)a_{\mathrm{I}_k\mathrm{M}_{k+1}}e_{\mathrm{M}_{k+1}}(x_{i+1}) + b_{\mathrm{I}_k}(i+1)a_{\mathrm{I}_k\mathrm{I}_k}e_{\mathrm{I}_k}(x_{i+1}) + b_{\mathrm{D}_{k+1}}(i)a_{\mathrm{I}_k\mathrm{D}_{k+1}};$$

$$b_{\mathrm{D}_k}(i) = b_{\mathrm{M}_{k+1}}(i+1)a_{\mathrm{D}_k\mathrm{M}_{k+1}}e_{\mathrm{M}_{k+1}}(x_{i+1}) + b_{\mathrm{I}_k}(i+1)a_{\mathrm{D}_k\mathrm{I}_k}e_{\mathrm{I}_k}(x_{i+1}) + b_{\mathrm{D}_{k+1}}(i)a_{\mathrm{D}_k\mathrm{D}_{k+1}}.$$

Добијене унапред и уназад променљиве се могу користити за поновну процену параметара.

$$E_{\mathrm{M}_k}(a) = \frac{1}{P(x)} \sum_{i|x_i=a} f_{\mathrm{M}_k}(i)b_{\mathrm{M}_k}(i);$$

$$E_{\mathrm{I}_k}(a) = \frac{1}{P(x)} \sum_{i|x_i=a} f_{\mathrm{I}_k}(i)b_{\mathrm{I}_k}(i).$$

$$A_{X_k M_{k+1}} = \frac{1}{P(x)} \sum_i f_{X_k}(i) a_{X_k M_{k+1}} e_{M_{k+1}}(x_{i+1}) b_{M_{k+1}}(i+1);$$

$$A_{X_k I_k} = \frac{1}{P(x)} \sum_i f_{X_k}(i) a_{X_k I_k} e_{I_k}(x_{i+1}) b_{I_k}(i+1);$$

$$A_{X_k D_{k+1}} = \frac{1}{P(x)} \sum_i f_{X_k}(i) a_{X_k D_{k+1}} b_{D_{k+1}}(i).$$

Уместо Баум-Велч алгоритма може се користити Витерби алгоритам.

3.4. Проналажење гена

3.4.1 Увод

У биоинформатици проналажење (тражење) гена *(gene finding)* или предвиђање гена *(gene prediction)* је процес идентификације региона ДНК секвенце који кодирају гене. Овај процес укључује гене који кодирају протеине, РНК гене, као и функционалане елементе који су у вези са регулаторним регионима. Налажење гена је најважнији корак у разумевању генома после његовог секвенцирања.

У данашње време, са повећањем способности компјутера, тражење гена све више постаје рачунарски проблем . Одређивање да ли је нека секвенца функционална не значи и одређивање саме функције. Одређивање функције захтева лабораторијске експерименте, мада биоинформатика има све више успеха у предвиђању функције на основу познавања саме секвенце. Трошкови лабораторијских експеримената су велики, па је неопходно налажење гена тражењем неких знакова који указују да секвенца кодира протеине. Ове знакове можемо категорисати као сигнале или састав *(content).* Сигнали индицирају присуство гена у близини, а састав *(content)* је статистичка особина самог протеин-кодирајућег региона.

Тражење гена код прокариота и еукариота се доста разликује. Код прокариота већи део ДНК секвенце је кодирајући, промотер регион има доста познат облик, а кодирајући регион је непрекидна секвенца нуклеотида. Статистика стоп кодона и налажење отвореног оквира за читање(енг. *open reading frame (ORF))* је веома информативно. Све то чини налажење гена код прокариота доста једноставније и прецизније него код еукариота.

Тражење гена код еукариота, посебно код сложенијих организама као што је човек, је значајно теже из неколико разлога. Промотери и остали

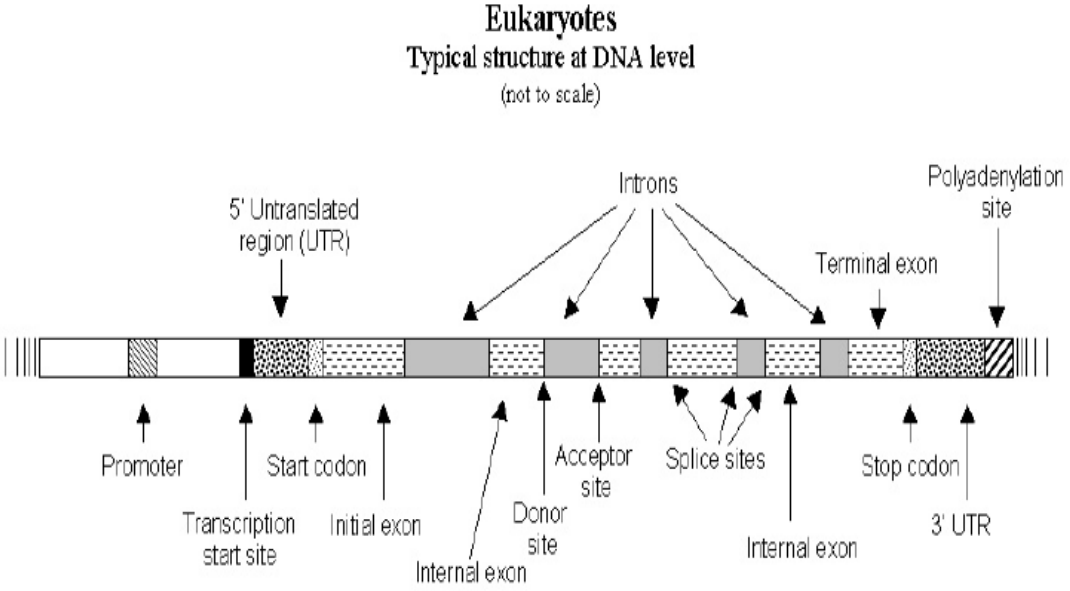
регулаторни сигнали су комплекснији и мање се о њима зна, што их чини тежим за препознавање. Два класична сигнала које програми за налажење гена препознају су CpG острва *(*енг. *CpG islands*) и место везивања поли (A) репа (енг. *poly(A) tail*).

Код еукариота регион кодирања није непрекидан (као код прокариота), већ је подељен на неколико делова – егзона *(*енг. *exons),* који су раздвојени некодирајућим секвенцама интронима *(*енг. *introns).*Региони раздвајања *(*енг. *splice sites)* су опет сами по себи сигнал који програми за нелажење гена траже. Типичан протеин кодирајући ген може бити подељен на десетак егзона који нису дужи од две стотине нуклеотида, док су неки кратки величине двадесет до тридесет нуклеотида. Стога је доста теже открити правила (периодичности) и статистике за налажење гена.

Напредни програми за налажење гена код прокариота и код еукариота користе сложеније статистичке моделе као што су скривени Марковљеви модели, који омогућавају комбиновање информација различитих сигнала и статистика. Два програма GLIMMER и GeneMark су највише у употреби и дају веома добре резултате у тражењу гена код прокариота. Код еукариота резултати нису тако добри. Најпознатији програм је GENSCAN. У новије време се комбинује више техника машинског учења да се поправи прецизност тражења гена код еукариота.

3.4.2. Структура гена код еукариота

Структура гена код еукариота је компликованија него код прокариота. Кодирајући регион није непрекидан и састављен је наизменично од егзона и интрона. После преписивања у РНК интрони се исецају и одбацују. Типичана структура гена која садржи више егзона приказана ја на слици 3.15.

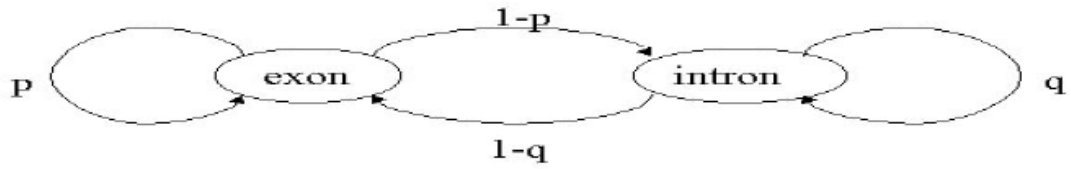


Слика 3.15. Структура гена код еукариота [5]

Ген почиње промотер регионом, иза кога следи некодирајући регион назван (енг. 5’ untranslated region (5’ UTR)). После стартног кодона долази први егзон. Наизменично се смењују интрони и егзони до последњег егзона иза кога је стоп кодон. Следи некодирајући 3’ UTR регион. На крају гена је поли А сигнал *(*енг. *polyadenylation)* који је састоји из неколико пута поновљеног аденин (енг. *Adenine*) нуклеотида. Границе између егзона и интрона – место сплајсинга (енг. *splice sites*) садрже сигнал дугачак два нуклеотида. На почетку интрона је секвенца донор (енг. *donor site*) GT, а на крају интрона је (енг. *acceptor site*) AG.

3.4.3 Генерализовани скривени Марковљеви модели (ГСММ)

Могући скривени Марковљев модел за тражење гена могао би да изгледа као на слици 3.16.



Слика 3.16.Једноставан СММ за тражење гена

Видимо да је вероватноћа остајања у егзон стању p а за прелазак у интрон стање је наравно 1 – p. Вероватноћа да генерисани егзон буде дужине k је p^k (1 – p) тј. има геометријску густину расподеле. Међутим, дужина егзона нема геометриску расподелу те се мора другачије моделовати.

Генерализовани скривени Марковљеви модели (енг. *Generalized Hidden Markov Model (GHMM)*) су уопштење скривених Марковљевих модела тако да неко стање емитује коначан низ симбола уместо само једног симбола. Густина расподеле емитовања за свако стање може бити различита. На пример, једно стање може користити тежински матрични модел (енг. *weight matrix model (WMM)*), док друго стање може користити скривени Марковљев модел .

Формалније, генерализовани скривени Марковљеви модел је скуп пет параметара:

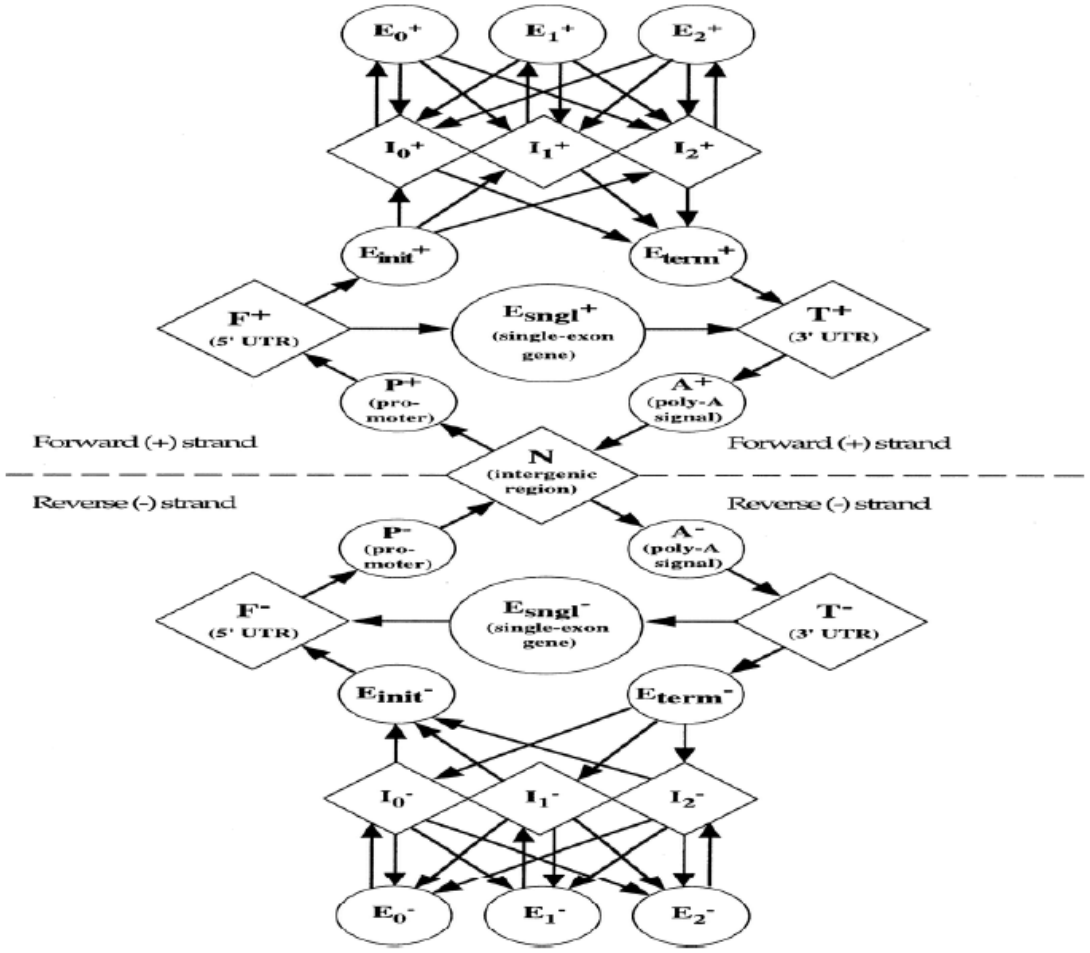
- Коначан скуп стања
- Вектор почетних вероватноћа
- Матрица транзиционих вероватноћа
- Вероватноћа дужине стања
- Модели за свако стање према коме ће се генерисати низови симбола

3.4.4 GENSCAN

Генерализовани Скривени марковљеви модели се користе у програму GENSCAN који су написали Chris Burge и Samuel Karlin [4].

Стања ГСММ одговарају различитим функционалним деловима гена као што су промотер регион, егзон, интрон итд. Транзиције између стања

обезбеђују да је пролажење кроз различита стања биолошки консистентно. Модел ћемо илустровати на слици 3.17.



Слика 3.17. GENSCAN

Може се уочити да се састоји из два симетрична дела (обележени са + и -). Делови служе да се истовремено предвиђају гени на оба комплементарна ланца ДНК секвенце. Стања E_i одговарају егзонима, стања I_i одговарају интронима, E_{init} је први егзон, E_{term} је задњи егзон, E_{sgnl} је једини егзон када нема интрона, F и T непреведени делови, P је промотер, A је поли A сигнал и коначно N је некодирајући део секвенце.

GENSCAN узима у обзир изохоре *(*енг. *isochore*). Изохоре су делови ДНК секвенце који су већи од 300000 нуклеотида и који имају доста униформан C+G садржај. Изохоре имају велики утицај на број гена, дужину гена итд. На пример, густина гена у (C+G) богатим деловима ДНК секвенце је пет пута већа него у деловима где је (C+G) садржај умерен, а десет пута је већа него у деловима богатим (A+T) садржајем. Према изохорама је скуп секвенци за обучавање модела подељен у четири категорије (C+G садржај мањи од 43%, C+G садржај већи или једнак 43% и мањи од 51%, C+G садржај већи или једнак од 51% и мањи од 57%, C+G садржај већи или једнал од 57%) и за сваку од њих су израчунате вероватноће почетних стања, транзиционе вероватноће и расподеле за дужину секвенце емитоване из неког стања.

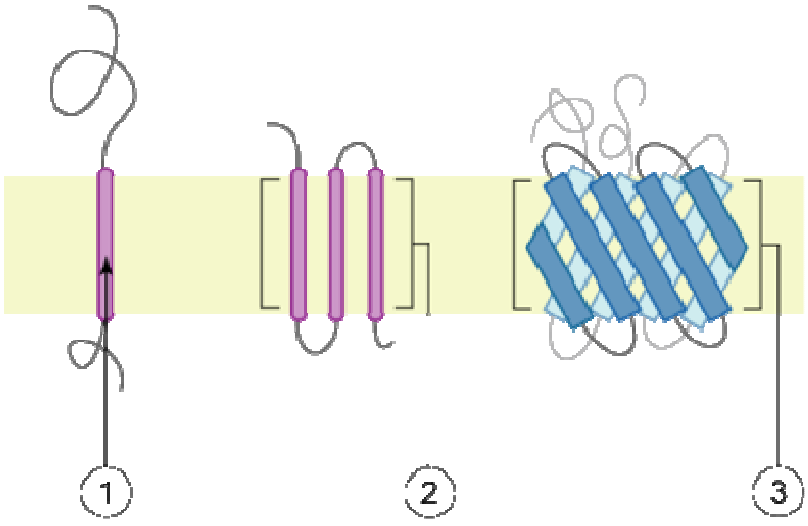
GENSCAN користи различите моделе за детектовање сигнала различитих функционалних јединица. Један од модела је тежински матрични модел *(*енг. *weight matrix model (WMM))* који се користи за поли А сигнал(енг. *polyadenylation signal*), почетни сигнал превођења (енг. *translation initiation signal*), завршни сигнал превођења (енг. *translation termination signal*) и за промотере. Тежински низ модел (енг. *weighted array model (WAM)*) је генерализација тежинског матричног модела која омогућава зависност између суседних позиција, а користи се за препознавање места спајања (енг. *splice sites*) и даје програму побољшање перформанси .

3.5. Скривени Марковљеви модели за трансмембранске протеине

Трансмембрански протеин (енг. *transmembrane protein*) је протеин који пролази кроз мембрану(слика 3.18). Он може једном или више пута да прође кроз мембрану. Њихова функција је да служе као капија где се дозвољава или одбија пролаз супстанци.

Постоје две основне врсте трансмембранског протеина(ТМ): алфа-хеликс (енг. *alpha-helix*) и бета-буре (енг. *beta-barrel*). Најчешћи су ТМ алфа-хеликси и они чине 27% свих протеина код људи.

На слици 3.18. [6] протеин је приказан шематски: лево(под један) садржи само један алфа-хеликс; у средини(под два) садржи три алфа-хеликса; десно (под три) је бета-буре које се састоји из бета-трака (енг.*beta-sheets*). Мембрана је приказана светло (браон) бојом.

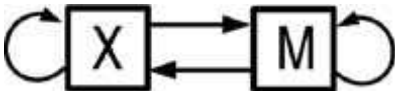


Слика 3.18. Шематски приказ трансмембранског протеина

Одлика трансмембранских протеина је да су више хидрофобни од просека. Према томе, да би се пронашли ТМ алфа хеликси у секвенци амино

киселина, неке методе користе неку врсту хидрофобне скале, рачунају просек у прозору одређене дужине и ако он прелази одговарајући праг предвиђају да је то ТМ алфа хеликс.

Уместо тога, могуће је направити једноставан скривени Марковљев модел приказан на слици 3.19.



Слика 3.19. Једноставан СММ за ТМ протеине

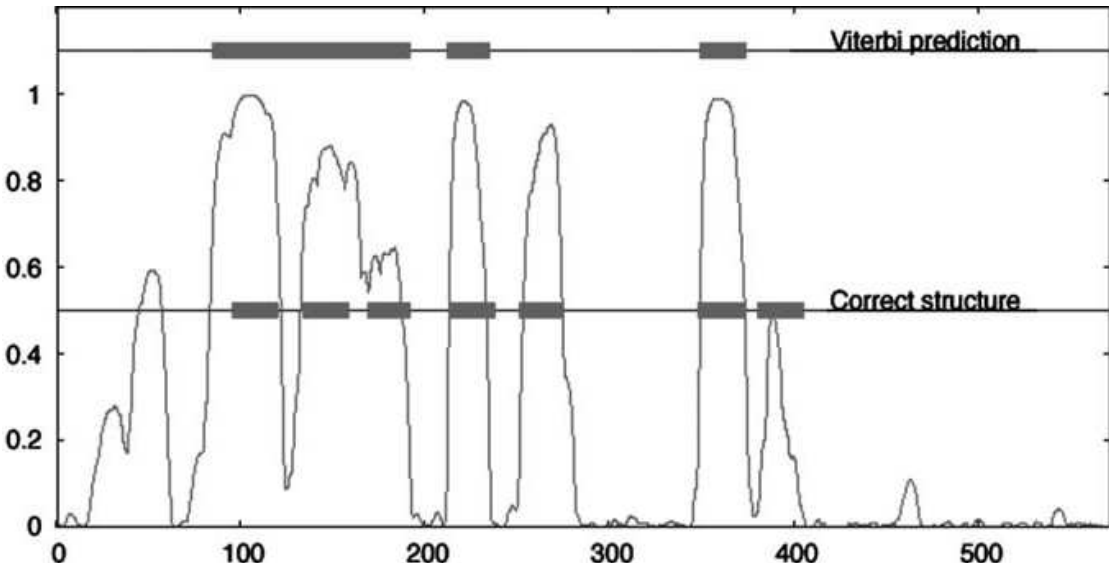
Модел се састоји из два стања, једно је за алфа-хеликсе и обележено је са М а друго је за све остало и обележено је са Х. Стању М придружићемо емисионе вероватноће које су фреквенције амино киселина у алфа хеликсима већ познатих ТМ протеина. Слично, емисионе вероватноће стања Х ће бити фреквенције амино киселина у остатку познатих ТМ протеина. Следећа табела показује како се рачунају емисионе вероватноће из скупа од 160 познатих ТМ протеина.

Амино киселина	Број (ТМхеликс)	Фреквенција (ТМ хеликс)	Број (остатак)	Фреквенција (остатак)	Однос фрекв.
I	1826	0,120	2187	0,046	2,61
F	1370	0,090	1854	0,039	2,31
L	2562	0,168	4156	0,087	1,93
V	1751	0,115	2935	0,061	1,89
M	616	0,040	1201	0,025	1,60
W	414	0,027	819	0,017	1,59
A	1657	0,109	3382	0,071	1,54
Y	615	0,040	1616	0,034	1,18
G	1243	0,082	3352	0,070	1,17
C	289	0,019	960	0,020	0,95
T	755	0,050	2852	0,060	0,83
S	806	0,053	3410	0,071	0,75
P	423	0,028	2640	0,055	0,51
H	121	0,008	1085	0,023	0,35
N	250	0,016	2279	0,048	0,33
Q	141	0,009	2054	0,043	0,21
D	104	0,007	2551	0,053	0,13
E	110	0,007	2983	0,062	0,11
K	78	0,005	2651	0,055	0,09
R	83	0,005	2933	0,061	0,08
УКУПНО	15214	1,000	47900	1,000	

Фреквенције одговарају емисионим вероватноћама два стања у горњем скривеном Марковљевом моделу. Амино киселине у табели су поређане према односу фреквенција (последња колона). Може се приметити да је већина амино киселина где је тај однос преко један хидрофобно.
Ова табела и овај крађи преглед је према [7].

Транзиционе вероватноће се изводе из следећих података. Укупно је посматрано 160 секвенци. Оне садрже 696 ТМ хеликса. Укупно је било 15214 амино киселина у ТМ хеликсима. Ван хеликса је било 47900 амино киселина.
Из стања М (15214 амино киселине) можемо прећи у стање Х 696 пута а то је на крају алфа хеликса. Вероватноћа преласка из М у Х је према томе 696 / 15214, а вероватноћа останка у стању М је (15214 – 696) / 15214. Да одредимо транзиције из стања Х од укупног броја амино киселина које нису у хеликсима (47900) морамо прво одузети 160 амино киселина које су на крају посматраних секвенци. Број прелазака из стања Х у М једнак је броју алфа хеликса а то је 696. Вероватноћа преласка из стања Х у М је према томе 696 / 47740, а вероватноћа останка (47740 – 696) / 47740.

На слици 3.20. приказан је резултат примене горњег модела на један од 160 ТМ протеина. Овај протеин има седам ТМ алфа хеликса. На х оси су позиције амино киселина у протеину чија је дужина 572. Танка крива линија приказује вероватноћу да нека амино киселина припада ТМ хеликсу добијена постериор декодирањем. Стварни ТМ хеликси приказани су као црне подебљане линије у средини графика. Највероватнија путања добијена Витерби алгоритмом је на врху графика. Може се приметити да је предвиђање постериор декодирањем значајно боље од предвиђања Витерби алгоритмом, који чини неколико значајних грешака.

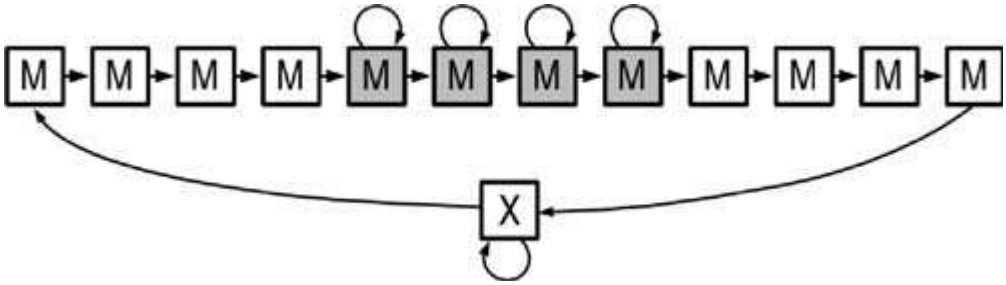


Слика 3.20. Резултат примене једноставног модела на један од 160 ТМ протеина .

Горњи модел је најпростији могући. Предвиђање ТМ хеликса се може поправити усложњавањем модела.

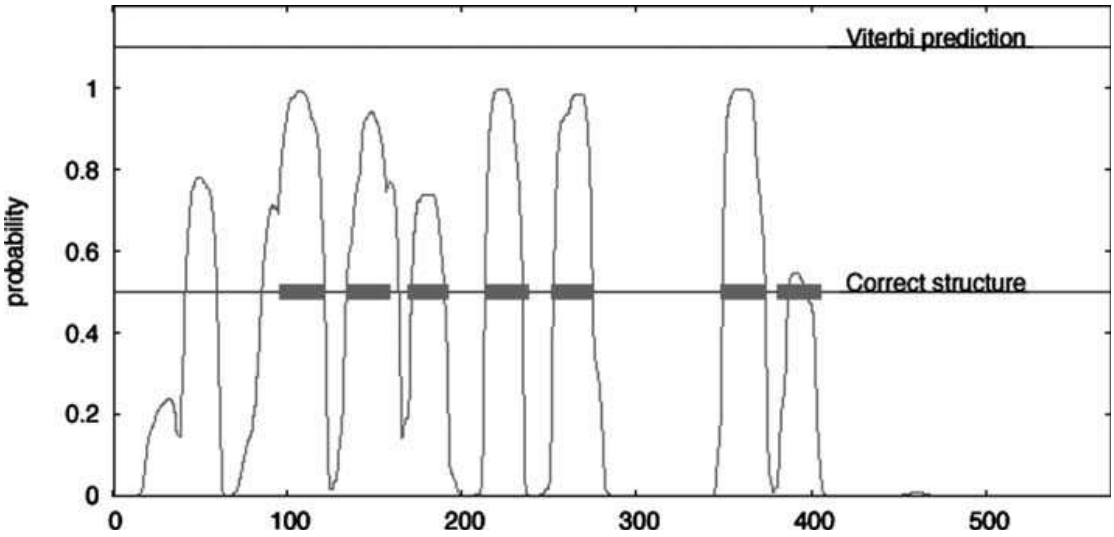
У горњем моделу садржана је погрешна претпоставка о расподели вероватноћа за дужину хеликса. Стање M има транзицију на самог себе и рецимо да је та вероватноћа p. Вероватноћа остајања у стању M за n амино киселина је p^n. Ова геометријска расподела је веома далеко од стварне расподеле вероватноћа за дужину хеликса, која би требало да има вероватноћу 0 ако је хеликс краћи од 15 амино киселина, максималну вероватноћу око двадесет друге амино киселине и да опада до 0 око тридесете амино киселине. Могуће је моделовати боље расподелу густина додавањем нових стања.

На слици 3.21 приказан је модел чија топологија стања даје расподелу вероватноћа облика звона и која се зове негативна биномна расподела. Овај модел користи четири стања која имају транзицију на себе и на тај начин моделују променљиву дужину хеликса (на слици су осенчена). Транзиционе и емисионе вероватноће су им исте. На почетку и на крају модела су додата по четири стања која дају моделу минималну дужину од осам амино киселина. После процене параметара овакав модел може да одговара жељеној густини расподела вероватноћа.



Слика 3.21. CMM модел са бољом густином расподеле за дужину хеликса

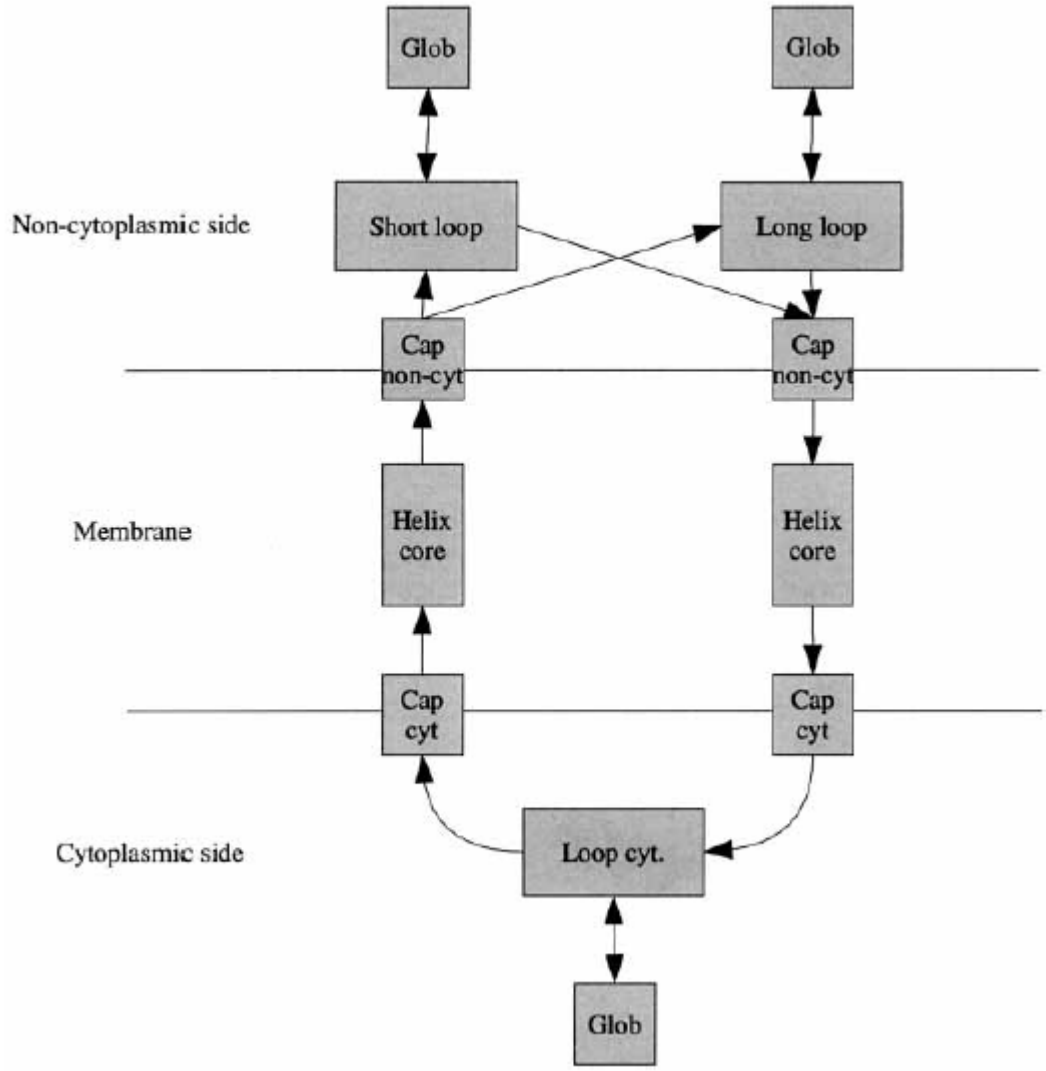
После процене параметара можемо користити Витерби и постериор декодирање на исти протеин коме смо предвиђали TM хеликсе најједноставнијем моделом. Резултати се могу видети на слици 3.22



Слика 3.22. Резултати примене побољшаног модела на исти протеин са слике 3.20.

Видимо да постериор декодирање одлично одговара овом моделу и да се предвиђање у односу на претходни модел значајно поправило. На пример, овај модел разликује други и трећи хеликс, док је претходни модел предвиђао погрешно да је то један хеликс. Мало се лошије показао код погрешне предикције ТМ хеликса на почетку протеина, али је код последњех хеликса дао јасан сигнал.

Даље побољшање модела за ТМ протеине могуће је додавањем нових стања. Могу се додати стања за моделовање хеликс капа (на почетку и крају хеликса – енг. *helix caps*), стања за унутарћелијски и ванћелијски део ланца амино киселина, стања за дуже или краће петље (енглески: *loops*) итд. Слика 3.23 илуструје једну такву могућност:



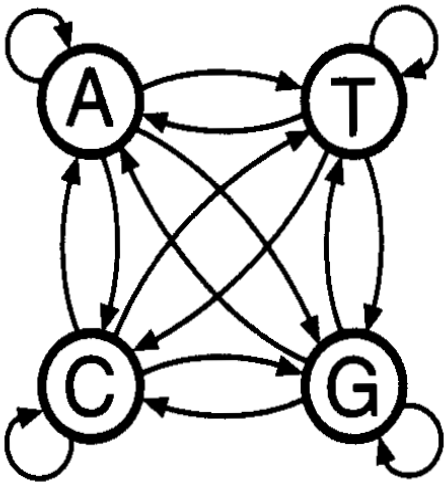
Слика 3.23. Један сложенији модел за ТМ протеине. *Glob* је глобуларан; *short loop* је краћа петља; *long loop* је дужа петља; *cap non-cyt* је капа хеликса ван ћелије; *helix core* је језгро хеликса; *cap cyt* је хеликс капа у ћелији; *loop cyt* је петља у ћелији.

Скривени Марковљеви модели одлично одговарају предвиђању ТМ протеина, а значај тога је већи с обзиром да су експерименти са ТМ протеинима скупи и тешко изводљиви.

4.Једна примена СММ на обележавање CpG острва

4.1. CpG острва, Марковљеви ланци и Скривени Марковљеви модели

О Марковљевим ланцима постоји много литературе. Ми ћемо само основну идеју искористити за изградњу модела за идентификацију CpG острва. Представимо Марковљев ланац графички (на слици 4.1) као скуп стања, где су стања A, C, G, T (нуклеотиди – ДНК азбука). Лукови означавају прелазе који се догађају са одређеном вероватноћом.



Слика 4.1 Марковљев ланац за CpG острва

Вероватноће прелаза (у нашем моделу) су вероватноће да, један нуклеотид следи за другим, на пример да нуклеотид T следи за нуклеотидом A.

Вероватноћа секвенце $x_1x_2...x_N$ ће тада бити

$P(x) = P(x_1)\, P(x_2 \mid x_1)\, P(x_3 \mid x_2)\, \ldots\ldots P(x_N \mid x_{N-1})$

Према [1] узето је 48 секвенци људског ДНК са обележеним CpG острвима. Направљена су два модела Марковљевих ланаца, један за обележена CpG острва (+ модел), а други за остатак секвенци (- модел).

Вероватноће прелаза из стања k у стање m се рачунају према формули (посебно за + и - модел):

$a_{km} = A_{km} / \Sigma_d A_{kd}$ где је A_{km} број динуклеотида km, а k, m, d су из скупа { A, C, G, T }.

Добијене су следеће табеле:

+	A	C	G	T
A	0.180	0.274	0.426	0.120
C	0.171	0.368	0.274	0.188
G	0.161	0.339	0.375	0.125
T	0.079	0.355	0.384	0.182

-	A	C	G	T
A	0.300	0.205	0.285	0.210
C	0.322	0.298	0.078	0.302
G	0.248	0.246	0.298	0.208
T	0.177	0.239	0.292	0.292

Први ред су вероватноће (фреквенције) да нуклеотид A буде праћен нуклеотидом A, C, G, T.

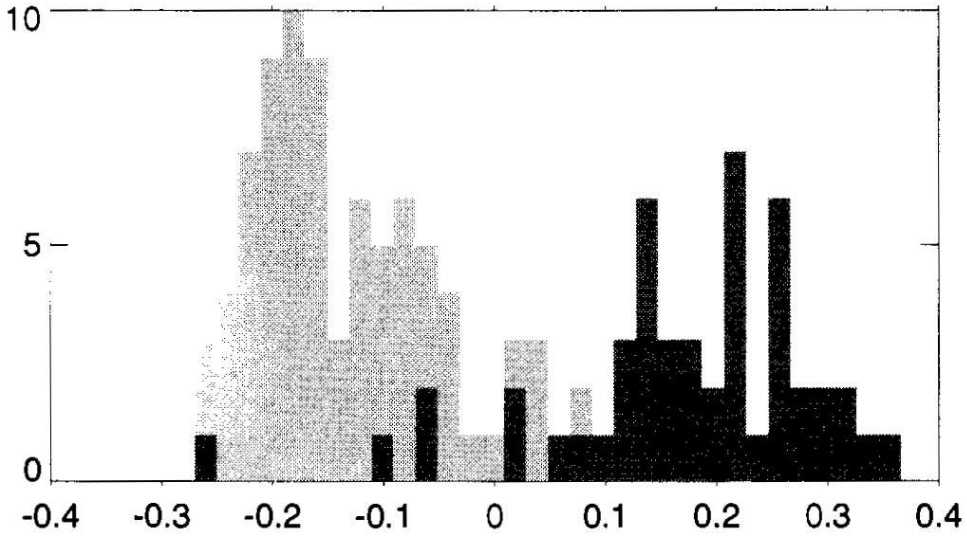
Да бисмо ове моделе користили за дискриминацију (CpG острво / не-CpG острво), израчунаћемо вредност

$$S(x) = \log (P(x \mid model +) / P (x \mid model -)) = \sum \log (a^{+}_{x_{i-1}x_i} / a^{-}_{x_{i-1}x_i}) = \sum \beta_{x_{i-1}x_i}$$

где је x секвенца и $\beta_{x_{i-1}x_i}$ логаритамски однос одговарајућих транзиционих вероватноћа. Табела за β је:

β	A	C	G	T
A	-0.740	0.419	0.580	-0.803
C	-0.913	0.302	1.812	-0.685
G	-0.624	0.461	0.331	-0.730
T	-1.169	0.573	0.393	-0.679

На слици 4.2 [1] је приказана расподела за вредност S(x) нормализована дељењем дужином секвенце.



Слика 4.2 Расподела за вредност S(x) нормализована дељењем дужином секвенце

CpG острва су приказана црно. Види се дискриминација између CpG острва и других секвенци.

Модел Марковљевих ланаца је добар за одговор на питање да ли је дата секвенца CpG острво или не; али није тако добар за тражење CpG острва у некој дужој секвенци. Морао би да се користи прозор рецимо дужине 100. Али да ли је то баш најбоље? Промена дужине прозора би увећала време израчунавања.

Боље је направити један модел који ће садржати оба Марковљева ланца и који ће са неком малом вероватноћом прелазити из једног у други. Сада је проблем што имамо два стања која емитују исти симбол (A у CpG, A изван CpG и слично за остале нуклеотиде). Решење је да стања једног ланца обележимо са + а другог са - . Тако долазимо до CMM где ћемо узети сличне транзиционе вероватноће (као у претходно описаном MM) између стања из истог ланца и мале транзиционе вероватноће за прелазак у стања из другог ланца. Ове вероватноће ће бити мало веће код преласка из + стања у – стања него обрнуто.

Тако добијамо скуп стања { q_0, A^+, C^+, G^+, T^+, A^-, C^-, G^-, T^- } где је q_0 почетно стање. Матрица транзиционих вероватноћа би изгледала као на слици 4.3:

a_{ij}	A+	C+	G+	T+	A-	C-	G-	T-
A+								
C+			0.27					
G+								
T+								
A-								
C-							0.08	
G-								
T-								

Слика 4.3 Шематски приказ основне идеје за матрицу транзиционих вероватноћа

Плаве и жуте транзиционе вероватноће су сличне одговарајућим вероватноћама из + и - Марковљевих ланаца, респективно.

На слици 4.3 је наглашена основна идеја модела: појављивање CpG динуклеотида у CpG острву неколико пута веће него у остатку секвенце.

Како се из X стања ($X \in \{A,C,G,T\}$) емитује X симбол, емисионе вероватноће у том случају ће бити 1, а за остале симболе 0. То није уобичајено код CMM. Према томе:

$e_{x+}(X) = e_{x-}(X) = 1$

$e_{x+}(Y) = e_{x-}(Y) = 0$ за Y различито од X.

Сада можемо да нађемо CpG острва у секвенцама.

Помоћу Витерби алгоритма може да се нађе највероватнија путања; када она пролази кроз + стања, ту су CpG острва (видети слику 4.4). Слично је са алгоритмом постериор декодирања.

A	T	A	T	A	C	G	A	C	G	C	G	T	A	T	A	G	A	C	T
A-	T-	A-	T-	A-	C+	G+	A+	C+	G+	C+	G+	T-	A-	T-	A-	G-	A-	C-	T-
					CpG ostrvo (island)														

Слика 4.4. Обележавање CpG острва

На слици 4.4 прва врста представља секвенцу, а друга врста предствља путању добијену Витерби алгоритмом или алгоритмом постериор декодирања. Део путање који пролази кроз плус стања је CpG острво.

У [1] је описан експеримент са 41 означеном секвенцом.

Применом Витерби алгоритма само два CpG острва нису пронађена, а предвиђено је 121 лажно позитивно острво.

Чест је случај да алгоритми на једном дужем CpG острву (просек је око 1000 нуклеотида) предвиђају неколико краћих, па су зато после предвиђања урађена следећа два поступка: острва између којих је мање од 500 нуклеотида су спојена и затим су одбачена она која су краћа од 500 нуклеотида. Број лажно позитивних се смањио на 67.

Када је на исти скуп секвенци примењено постериор декодирање, иста два острва нису предвиђена а било је 236 лажно позитивних. Применом два горе споменута поступка тај број се смањио на 83 .

Аутори закључују да нема веће разлике у резултатима између ова два алгоритма, сем што постериор декодирање предвиђа више краћих острва. Могуће је да су нека од лажно позитивних у ствари стварна (још неозначена) CpG острва. Кажу да је могуће да су два лажно негативна острва – у ствари погрешно обележена као и да је можда потребан софистицираније модел како би се боље ухватио сигнал.

4.2 CpG анотатори

CpG анотатори су програми или подаци који обележавају CpG острва. CpG острва се обележавају експериментално и коришћењем рачунара. Експериментално обележавање је скупо и захтева доста времена. За обележавање рачунарима користе се програми од којих већину можемо сврстати у две групе: програми који користе традиционалну технику покретног прозора и програми који итеративно комбинују суседне CpG кластере чије је физичко растојање и густина испод или изнад раније дефинисаног прага према CpG особинама ДНК секвенци. Укратко ћемо описати CpG анотаторе који су коришћени у овом раду.

У [11] је дата формална дефиниција CpG острва која се најчешће користи. CpG острва су региони дугачки најмање 200 нуклеотида, GC садржај (број G и C нуклеотида) је већи од 50% и уочено/очекивано однос већи него 60%. Уочено/очекивано однос се рачуна према формули
(број CpG динуклеотида / (број C нуклеотида × број G нуклеотида)) × Укупан број нуклеотида у секвенци.

EMBL анотатор чине подаци са веб сајта Европског Биоинформатичког Института (енг. European Bioinformatics Institute(EBI)) који су добијени експерименталним путем. Подаци се могу наћи на ftp://ftp.ebi.ac.uk/pub/databases/cpgisle.

CpG-Discover је програм описан у [8] и заснован на CMM обученом на EMBL бази података. Параметри модела су одређени у неколико корака. За сваку секвенцу су почетни параметри одређени пребројавањем фреквенција и

онда је примењен Баум-Велч алгоритам на ту секвенцу до конвергенције. Тако су добијени параметри за сваку секвенцу посебно. Коначни параметри модела су добијени техником просечног већинског гласања (енг. voted averaged technique) од 90 секвенци као и нормализацијом дужином сваке секвенце. Овај програм процењује (учи) и емисионе вероватноће.

Тако добијеним моделом обележавају се CpG острва.

Обележавање се побољшава применом поступака постпроцесирања. Ако је растојање између острва мање од 20 нуклеотида, она се спајају у веће острво. Још се захтева да је CpG уочено/очекивано однос већи од 60% и да острва нису краћа од 140 нуклеотида.

CpGIE [9] је програм из групе програма који користе традиционалну технику покретног прозора (енг. *sliding window*) за тражење CpG острва. Типично, за сваки прозор, израчуна се вредност (скор) S(x) из одељка 3.1. Прозори који дају позитиван скор су потенцијална CpG острва. Мања острва, која се преклапају или нису међусобно удаљена, могу се удруживати у већа. Неки проблеми са овим приступом су:

1. Није јасно унапред коју величину прозора треба користити као ни величину корака (енг. *step*). На пример, ако је прозор превише велик, неколико мањих острва може бити обележено као једно.
2. CpG острва генерално не почињу и не завршавају се CpG динуклеотидом.
3. Дуго време извршавања.

CpGIF [10] је програм из групе програма који итеративно комбинују суседне CpG кластере чије је физичко растојање и густина испод или изнад раније дефинисаног прага према CpG особинама ДНК секвенци. У првом пролазу тражи делове ДНК секвенце са великом CpG густином који су названи семена (енг. seed). Семена се даље проширују и групишу у коначна CpG острва. При томе се рачуна GC садржај и CpG уочено/очекивано однос. Ово доводи до проблема искључивања краћих CpG острва (нарочито оних краћих од 210 нуклеотида).

4.2.1 MGcpg анотатор

У овом раду развијен је и сопствени CpG анотатор заснован на CMM. Улаз у програм су две текст датотеке model.txt и kinezi9.txt. У датотеци model.txt налазе се почетни параметри CMM, а у датотеци kinezi9.txt су секвенце нуклеотида.

На почетку програм нуди избор да ли да се мењају емисионе вероватноће код CMM или да остану непромењене као што су описане у одељку 3.1.

Имплементиран је алгоритам Витерби учења. Помоћу CMM из датотеке model.txt анотирају се секвенце. Анотиране секвенце се користе за нову процену параметара CMM. Поступак се понавља до конвергениције (док се не добије исти модел).

Када се добије исти модел као у претходној процени и Витерби учење заврши, програм анотира све секвенце, примене се три поступка постпроцесирања (GC садржај већи од 60%, CpG уочено/очекивано однос већи од 0.6 и острва дужа од 200 нуклеотида) и резултат смешта у текст датотеку

CpGostrva.txt. Резултат је низ редова где су записани почеци и крајеви CpG острва.

У односу на раније описане програме у овом одељку (3.2), овај програм има сличности са програмом CpG-Discover, јер оба користе CMM. CpG-Discover користи Баум-Велч алгоритам и технику просечног већинског гласања за процену параметара модела, док програм MGcpg користи Витерби учење. Програм CpG-Discover мења (процењује) емисионе вероватноће. MGcpg програм је у почетку радио без промене емисионих вероватноћа и оне су биле као у одељку 3.1. Касније је додата и могућност промене емисионих вероватноћа, али то није дало боље резултате(видети одељак 3.3 Експерименти).

Програм је писан у програмском језику C, оперативни систем је XP и коришћена је DB2 база података.

4.3. Експерименти

4.3.1 Експеримент са 9 секвенци

Први експеримент изведен у оквиру овог рада је проширење дела екперимента описаног у раду [8]. Посматрано је 9 секвенци (приступни NCBI бројеви AB000275, AB000276, AB000277, AB001915, AB011100, AB014544, AB017019, AB017365, AB020631) које су анотиране и приказане у табели 4.1. Анотатори су (EMBL[12], CpG-Discover[8], CpGIE[9],CpGIF[10]), као и MGcpg развијен у овом раду и описан у претходној тачки.

Подаци из табеле из рада [8], као и подаци из датотеке CpGostrva.txt (излазна датотека из програма MGcpg) смештени су у базу података ради даље обраде. Овде ћемо их приказати у табели 4.1.

	AB000275	AB000276	AB000277
EMBL	274 - 524 1472 - 1684	436 - 713 1836 - 2301	308 - 975 1390 - 1667 2790 - 3255
CPG Discover	471 - 598 1483 - 1740	482 - 667 1843 - 2294	1445 - 1531 1569 - 1621 2823 - 3248
CPGIE	210 - 1739	341 - 774 1767 - 2392	1 - 1083 1295 - 1728 2721 - 3346
CPGIF	3 - 1734	338 - 784 1890 - 2409	10 - 2567 2826 - 3345
MGcpg	283 - 1084 1478 - 1740	351 - 701 1843 - 2300	1 - 1030 1305 - 1655 2797 - 3254

	AB001915	AB011100	AB014544
EMBL	185 - 642	51 - 285	48 - 537 552 - 1094 1171 - 1642 1688 - 2696
CPG Discover	195 - 616		1 - 2834 4277 - 4797
CPGIE	89 - 694		1 - 2834
CPGIF	178 - 768		23 – 2949
MGcpg	172 - 630		20 – 2818

	AB017019	AB017365	AB020631
EMBL	47 - 381 39 - 930	47 - 289 293 - 681 691 - 1066 1155 - 1730	49 - 484
CPG Discover	1 - 896	1 - 1821 2096 - 2406	1 - 495
CPGIE	1 - 984	1 – 1829 2020 - 2235	1 - 505
CPGIF	6 - 1044 1757 - 1959	11 - 2387	6 - 519
MGcpg	1 - 890	1 - 1840	1 - 467

Табела 4.1 Предвиђање CpG острва за 9 секвенци анотаторима EMBL, CpG-Discover, CpGIE, CpGIF, MGcpg

4.3.2 Евалуација (статистичка оцена резултата) експеримента

Перформанса система за предвиђање CpG острва мери се поређењем предвиђене вредности нуклеотида са тачном вредности (да ли је или није у CpG острву). Статистичке мере којима можемо мерити прецизност система за предвиђање CpG острва су тачност, специфичност (енг. specificity), осетљивост (енг. sensitivity), коефицијент корелације итд. Подсетимо се основних термина и дефиниција:

TP (енг. *true positive*) број нуклеотида и у познатом ("стварном") CpG острву и у предвиђеном CpG острву
FN (енг. *false negative*) број нуклеотида који су у познатом CpG острву, а нису у предвиђеном CpG острву
FP (енг. *false positive*) број нуклеотида који нису у познатом CpG острву, а јесу у предвиђеном CpG острву
TN (енг. *true negative*) број нуклеотида који нису у познатом CpG острву и нису ни у предвиђеном CpG острву

Осетљивост (енг. sensitivity; ознака SN) теста је вероватноћа да нуклеотид буде у предвиђеном CpG острву ако је у познатом CpG острву.
SN = TP / (TP + FN)

Специфичност (енг. specificity; ознака SP) теста је вероватноћа да нуклеотид није у предвиђеном CpG острву, ако није у познатим CpG острвима.
SP = TN / (TN + FP)

Коефицијент корелације (енг. correlation coefficient):
MCC = (TP · TN – FN · FP) / $\sqrt{(TP+FN)(TN+FP)(TP+FP)(TN+FN)}$

Тачност (енг. Accuracy)
(TP + TN) / (TP + TN + FP + FN)
Прецизност (енг. Precision)
TP / (TP + FP)

После обележавања (анотирања) секвенци, MGcpg примењује поменута три поступка постпроцесирања (на пример, захтева се GC садржај већи од 60%, CpG уочено/очекивано однос већи од 0.6 и острва дужа од 200 нуклеотида). Поред тога, испробане су и друге вредности за GC садржај, CpG уочено/очекивано и дужину острва.

Програм MGcpg омогућава поређење са једним анотатором (рецимо EMBL) или са више њих. У другом случају се користи просечно већинско гласање.

4.3.2.1 Поређење MGcpg анотатора са EMBL анотатором

Поређење са једним анотатором (EMBL) дало је резултате као у табели 4.2. У табелама и на сликама скраћеница GC се односи на GC садржај, док се скраћеница o/e се односи на CpG уочено/очекивано однос. Подразумева се да су вредности веће или једнаке од датих бројева.

GC	o/e	Прецизност	Осетљивост	Специфичност	Тачност	К.Корелације
		0.695	**0.963**	**0.828**	**0.867**	**0.731**
		0.672	0.97	0.807	0.854	0.714
0.5	**0.5**	**0.774**	**0.959**	**0.885**	**0.907**	**0.798**
0.5	0.5	0.733	0.964	0.856	0.888	0.766
0.6	**0.5**	**0.823**	**0.959**	**0.916**	**0.928**	**0.839**
0.6	0.5	0.783	0.964	0.891	0.912	0.809
GC + o/e >= 1		**0.743**	**0.959**	**0.865**	**0.892**	**0.772**
GC + o/e >= 1		0.703	0.964	0.834	0.872	0.74
0.6	**0.6**	**0.823**	**0.959**	**0.916**	**0.928**	**0.839**
0.6	0.6	0.783	0.964	0.891	0.912	0.809

Табела 4.2 Поређење MGcpg са EMBL анотатором.

Свака врста у табели представља уствари део експеримента(или можемо рећи да се ради о 10 експеримената(у табели 4.2 има 10 врста)) где се пореде анотације добијене MGcpg и EMBL анотаторима. Експерименти се разликују по различитим вредностима параметара поступака постпроцесирања.

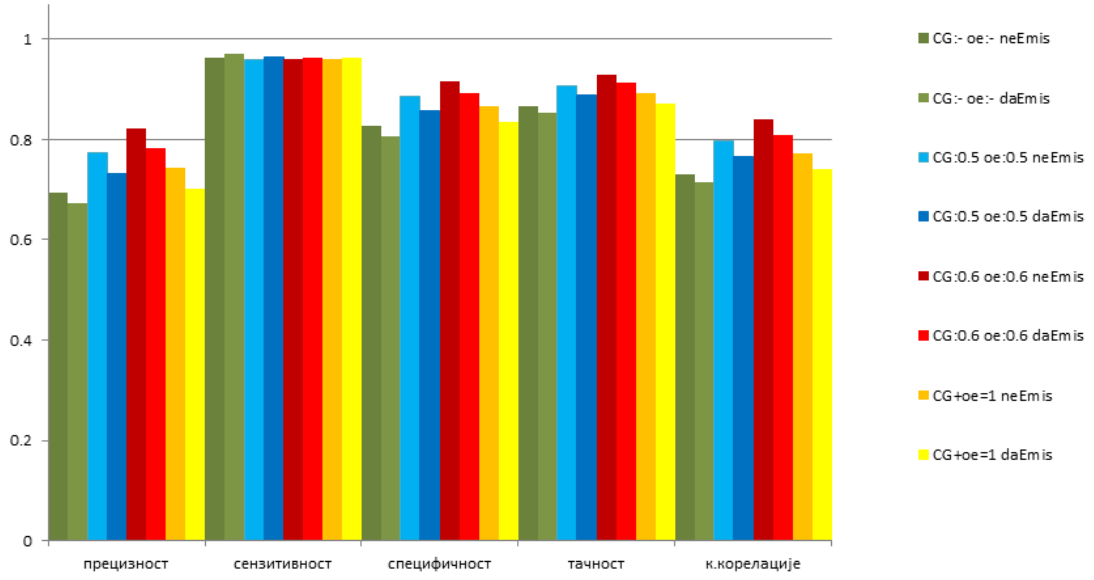
Масним словима су приказани резултати експеримената добијених СММ код којих се не мењају емисионе вероватноће. Видимо да мењање емисионих вероватноћа не даје боље резултате и у ствари је чешће за нијансу лошије. То се може графички приказати као на сликама 4.5. и 4.6

Прве две врсте садрже резултате екперимента добијене без примене три поступка постпроцесирања(само са СММ – без провере да ли је дужина острва већа од 200, да ли је GC садржај већи рецимо од 0.5 и да ли је CpG уочено/очекивано однос већи рецимо од 0.6) и видимо да су они нешто лошији тј. да је употреба поступака постпроцесирања оправдана. Прва врста (масна слова) је резултат експеримента где се не мењају емисионе вероватноће, а друга представља експеримент где се мењају.

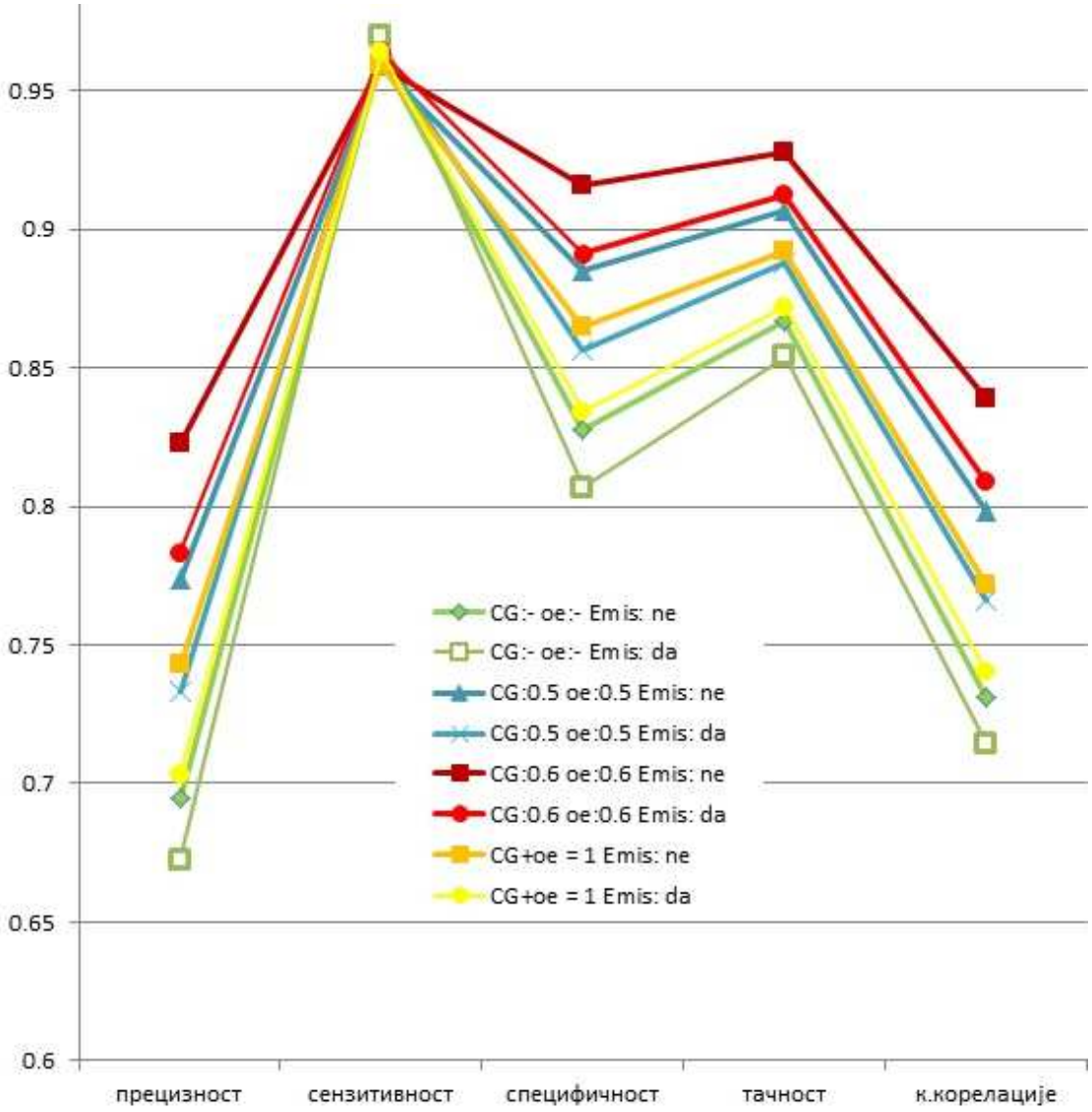
Од треће до десете врсте представљени су екперименти код којих су коришћени поступци постпроцесирања. У свим се захтева дужина острва већа од 200 нуклеотида. У првој и другој колони су приказане вредности за GC садржај и CpG уочено/очекивано однос респективно.

У седмој и осмој врсти су резултати експеримената где се захтевало да је збир GC садржаја и CpG уочено/очекивано односа најмање 1. Ово даје добар али не и најбољи резултат.

Резултати су исти ако за GC садржај користимо 0,6 без обзира да ли је CpG уочено/очекивано однос 0,5 или 0,6 (пета и девета врста су исте, као и шеста и десета). То је и најбољи резултат у табели.



Слика 4.5 Прецизност, осетљивост, специфичност, тачност и коефицијент корелације за различите вредности CG садржаја, CpG уочено/очекивано односа и у зависности од тога да ли се мењају емисионе вероватноће.



Слика 4.6 Прецизност, осетљивост, специфичност, тачност и коефицијент корелације за различите вредности CG садржаја, CpG уочено/очекивано односа и у зависности од тога да ли се мењају емисионе вероватноће.

4.3.2.2 Поређење MGcpg анотатора са више анотатора

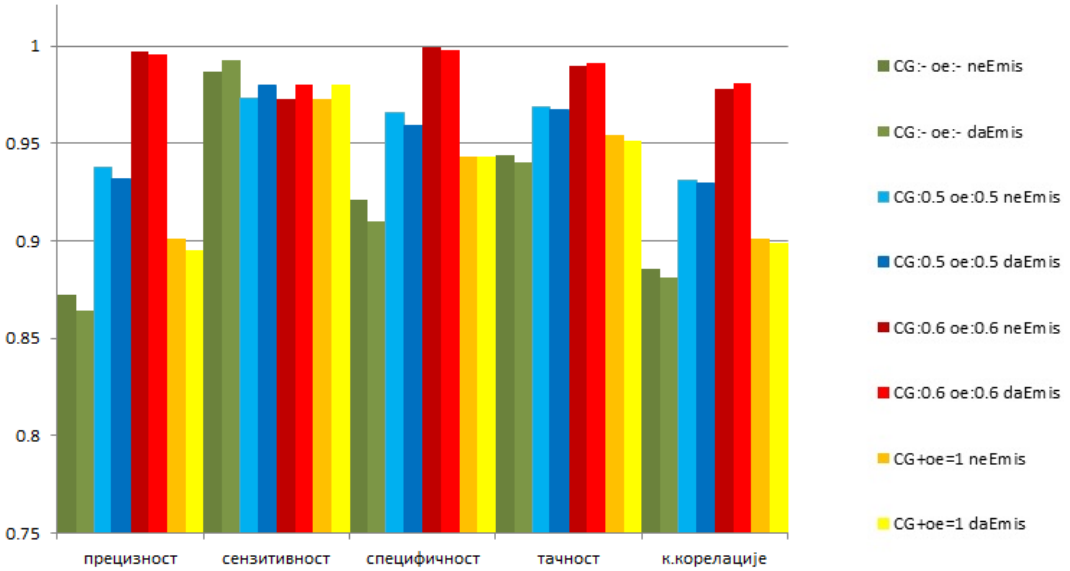
Поређење MGcpg са више анотатора даје резултате приказане у табели 4.3.

GC	o/e	Прецизност	Осетљивост	Специфичност	Тачност	К.Корелације
		0.872	**0.987**	**0.921**	**0.944**	**0.886**
		0.864	0.993	0.91	0.94	0.881
0.5	**0.5**	**0.938**	**0.973**	**0.966**	**0.969**	**0.931**
0.5	0.5	0.932	0.98	0.959	0.967	0.92945
0.6	**0.5**	**0.997**	**0.973**	**0.999**	**0.99**	**0.978**
0.6	0.5	0.996	0.98	0.998	0.991	0.981
GC + o/e = 1		**0.901**	**0.901**	**0.973**	**0.943**	**0.954**
GC + o/e = 1		0.895	0.895	0.98	0.943	0.951
0.6	**0.6**	**0.997**	**0.973**	**0.999**	**0.99**	**0.978**
0.6	0.6	0.996	0.98	0.998	0.991	0.981

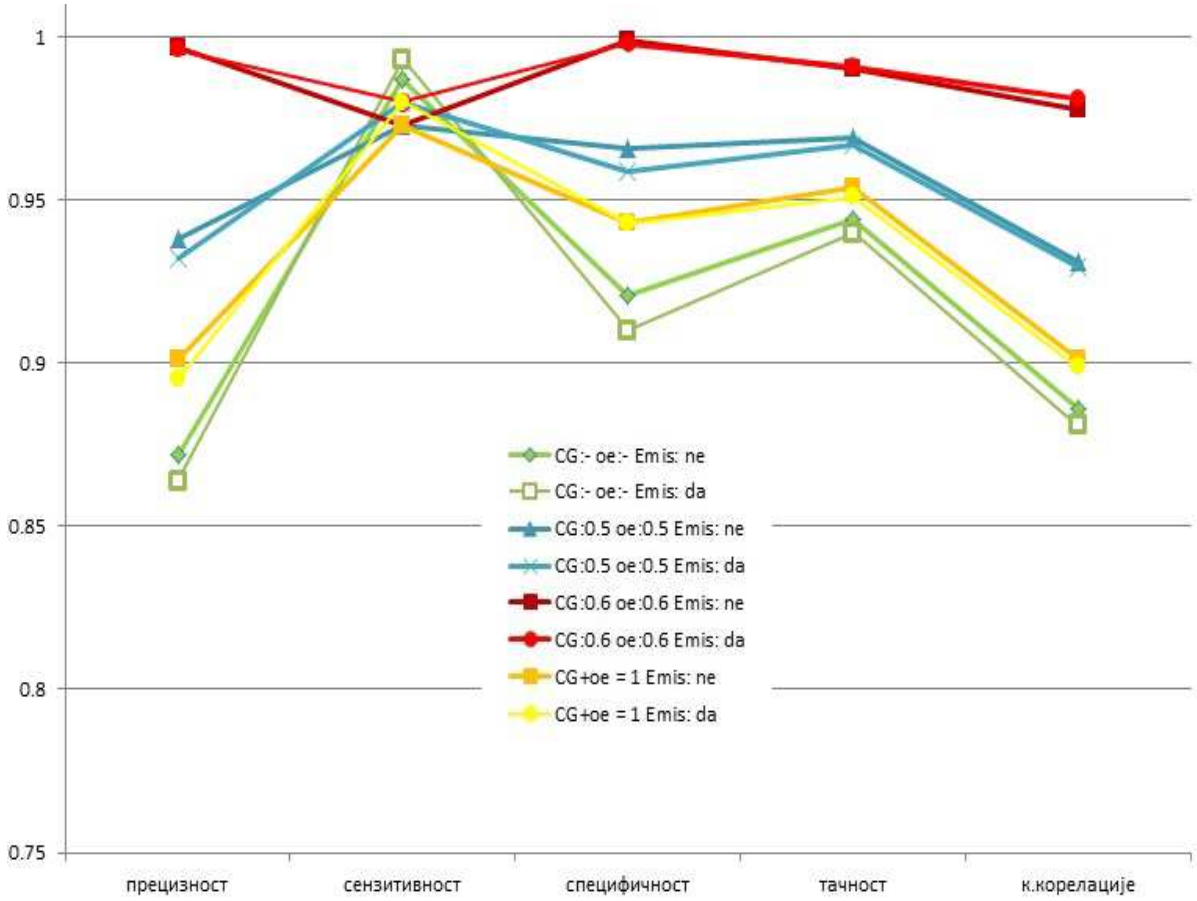
Табела 4.3 Поређење MGcpg са више анотатора

Видимо да су резултати бољи, што указује да је анотација CpG острва Витерби алгоритмом сличнија другим програмима него анотацији у EMBL бази података.

Графичка интерпретација резултата поређења MGcpg са више анотатора из табеле 4.3 се види на сликама 4.7 и 4.8.



Слика 4.7. Прецизност, осетљивост, специфичност, тачност и коефицијент корелације за различите вредности CG садржаја, CpG уочено/очекивано односа и у зависности од тога да ли се мењају емисионе вероватноће.



Слика 4.8 Прецизност, осетљивост, специфичност, тачност и коефицијент корелације за различите вредности CG садржаја, CpG уочено/очекивано односа и у зависности од тога да ли се мењају емисионе вероватноће.

4.3.3 Експеримент са 100 секвенци

Други експеримент изведен у оквиру овог рада односи се на 100 секвенци од којих је 9 као у првом експерименту а анотација 91 секвенце је преузета из EMBL базе тако да се секвенце разликују од првих 9.

Витерби учењем на 91 секвенци, без мењања емисионих вероватноћа, после 11 итерација добија се следећа матрица транзиционих вероватноћа:

	A+	C+	G+	T+	A-	C-	G-	T-
A+	0.15923	0.28959	0.42089	0.13016	0.00003	0.00003	0.00003	0.00003
C+	0.21661	0.36504	**0.18956**	0.22872	0.00002	0.00002	0.00002	0.00002
G+	0.19425	0.33461	0.33731	0.13376	0.00002	0.00002	0.00002	0.00002
T+	0.07526	0.33370	0.43647	0.14801	0.00266	0.00004	0.00004	0.00383
A-	0.00003	0.00003	0.00003	0.00003	0.31390	0.19179	0.26295	0.23123
C-	0.00004	0.00004	0.00004	0.00004	0.33234	0.27509	**0.06692**	0.32550
G-	0.00312	0.00004	0.00004	0.00004	0.29117	0.21129	0.26487	0.22944
T-	0.00145	0.00003	0.00003	0.00003	0.18019	0.21832	0.28316	0.31681

Витерби учењем на 91 секвенци, са мењањем емисионих вероватноћа, добијају се следеће матрице транзиционих и емисионих вероватноћа.

Матрица транзиционих вероватноћа (после 24 итерације):

	A+	C+	G+	T+	A-	C-	G-	T-
A+	0.16923	0.28116	0.41226	0.13711	0.00003	0.00014	0.00003	0.00003
C+	0.22468	0.36017	**0.18136**	0.23373	0.00002	0.00002	0.00002	0.00002
G+	0.20268	0.32360	0.33519	0.13845	0.00002	0.00002	0.00002	0.00002
T+	0.08061	0.32780	0.42907	0.15695	0.00250	0.00003	0.00003	0.00301
A-	0.00003	0.00003	0.00003	0.00003	0.32900	0.18459	0.24572	0.24055
C-	0.00005	0.00005	0.00005	0.00005	0.33977	0.26544	**0.05811**	0.33649
G-	0.00164	0.00005	0.00005	0.00005	0.29768	0.20652	0.25049	0.24354
T-	0.00336	0.00003	0.00003	0.00003	0.18845	0.20715	0.26794	0.33301

Матрица емисионих вероватноћа:

	A	C	G	T
A+	0.99991	0.00003	0.00003	0.00003
C+	0.00002	0.99995	0.00002	0.00002
G+	0.00002	0.00002	0.99994	0.00002
T+	0.00003	0.00003	0.00003	0.99990
A-	0.99989	0.00003	0.00003	0.00003
C-	0.00005	0.99986	0.00005	0.00005
G-	0.00005	0.00005	0.99986	0.00005
T-	0.00003	0.00003	0.00003	0.99990

У претходном експерименту коришћено је 9 секвенци и Витерби учење без мењања емисионих вероватноћа је дало, после 10 итерација, следећу матрицу транзиционих вероватноћа:

	A+	C+	G+	T+	A-	C-	G-	T-
A+	0.15569	0.29553	0.39644	0.14080	0.00048	0.01009	0.00048	0.00048
C+	0.19404	0.35114	**0.24725**	0.20665	0.00023	0.00023	0.00023	0.00023
G+	0.19595	0.39379	0.28259	0.12659	0.00027	0.00027	0.00027	0.00027
T+	0.08198	0.38238	0.38187	0.15071	0.00051	0.00051	0.00051	0.00153
A-	0.00018	0.00018	0.00018	0.00018	0.34737	0.15909	0.24739	0.24544
C-	0.00030	0.00030	0.00030	0.00030	0.38493	0.24263	**0.06222**	0.30902
G-	0.00404	0.00025	0.00025	0.00025	0.34352	0.16216	0.24678	0.24274
T-	0.00099	0.00020	0.00020	0.00020	0.20292	0.19306	0.27095	0.33149

Исти експеримент (Витерби учење на 9 секвенци) са мењањем емисионих вероватноћа даје после 6 итерација следеће матрице транзиционих и емисионих вероватноћа.

Матрица транзиционих вероватноћа:

	A+	C+	G+	T+	A-	C-	G-	T-
A+	0.17240	0.28986	0.39482	0.14113	0.00045	0.00045	0.00045	0.00045
C+	0.19714	0.35041	**0.24390**	0.20765	0.00022	0.00022	0.00022	0.00022
G+	0.20363	0.38366	0.28379	0.12789	0.00026	0.00026	0.00026	0.00026
T+	0.08134	0.37555	0.38090	0.15149	0.00341	0.00049	0.00049	0.00633
A-	0.00018	0.00018	0.00018	0.00018	0.34608	0.16111	0.24376	0.24832
C-	0.00031	0.00031	0.00031	0.00031	0.38707	0.24000	**0.06062**	0.31108
G-	0.00394	0.00026	0.00026	0.00026	0.34182	0.16330	0.24416	0.24600
T-	0.00040	0.00020	0.00020	0.00020	0.20474	0.19249	0.26937	0.33239

Матрица емисионих вероватноћа:

	A	C	G	T
A+	0.99865	0.00045	0.00045	0.00045
C+	0.00022	0.99932	0.00022	0.00022
G+	0.00026	0.00026	0.99921	0.00026
T+	0.00049	0.00049	0.00049	0.99853
A-	0.99945	0.00018	0.00018	0.00018
C-	0.00031	0.99907	0.00031	0.00031
G-	0.00026	0.00026	0.99921	0.00026
T-	0.00020	0.00020	0.00020	0.99939

Ова четири дела експеримента дају четири СММ којима је обележено 9 секвенци, а евалуација је са више анотатора. Добијени резултати су приказани у табели 4.4 (графички су приказани на слици 4.9.):

Број секвенци (учење)	Мењање емисионих вероватноћа	Прецизност	Сензитивност	Специфичност	Тачност	К.корелације
9	Не	0.997	0.973	0.999	0.99	0.978
9	Да	0.996	0.98	0.998	0.991	0.981
91	Не	0.995	0.972	0.997	0.988	0.974
91	Да	0.995	0.978	0.997	0.99	0.979

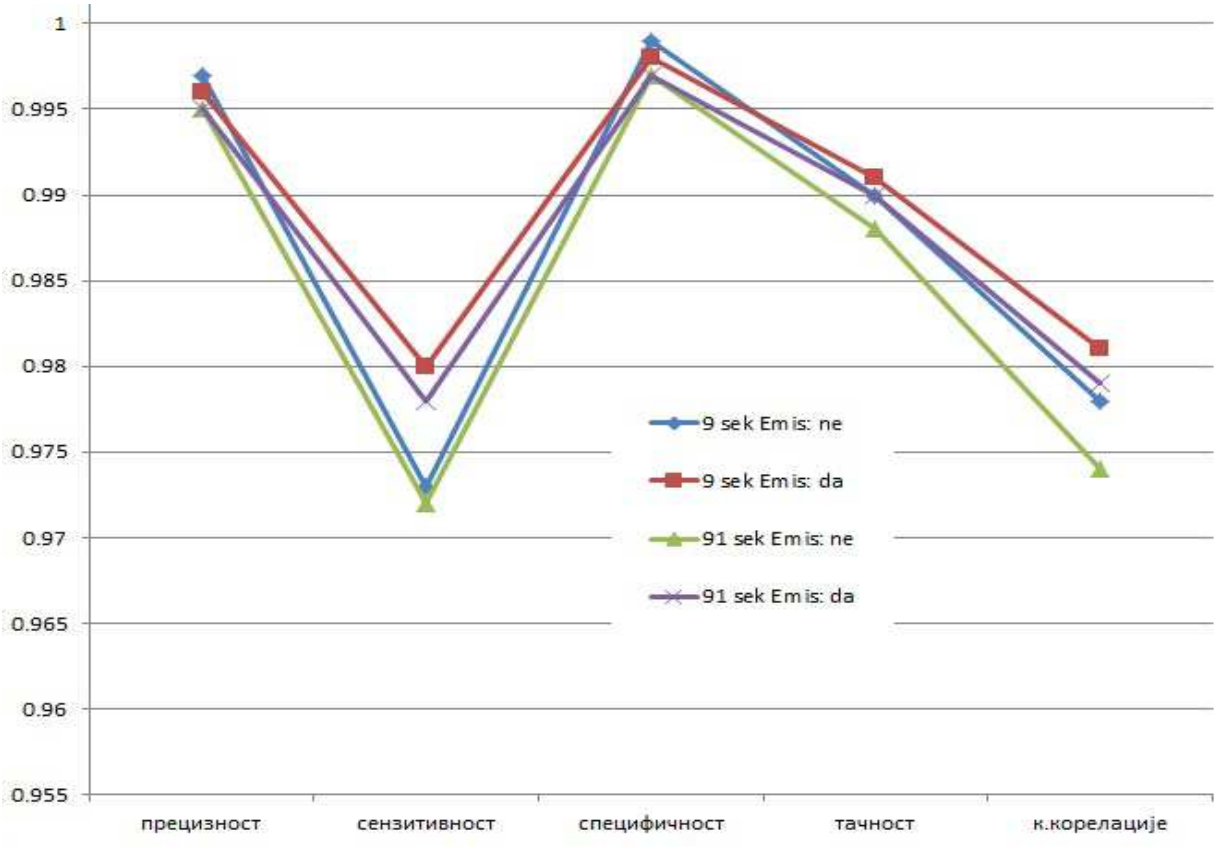
Табела 4.4 Прецизност, осетљивост, специфичност, тачност и коефицијент за четири дела експеримента са 100 секвенци

У свим деловима експеримента, после анотација са СММ, примењени је исти поступци простпроцесирања: острва су дужа од 200 нуклеотида, GC садржај је већи од 0,6 и CpG уочено/очекивано однос је већи од 0,6.

У прве две врсте су резултати добијени СММ који су учили на 9 секвенци, обележаване су исте те секвенце, а евалуација је према више анотатора(експеримент из 3.3.1 и 3.3.2). Прва врста је без мењања емисионих вероватноћа, а друга је са мењањем.

У трећој и четвртој врсти су резултати добијени СММ који су учили на 91 секвенци, обележавано је 9 секвенци које су различите од њих (исте оне

коришћене у експерименту из 3.3.1 и 3.3.2), а евалуација је према више анотатора. Трећа врста је без мењања емисионих вероватноћа, а четврта је са мењањем.



Слика 4.9 Прецизност, осетљивост, специфичност, тачност и коефицијент за четири дела експеримента са 100 секвенци.

Резултати су слични, а говоре и да обележавање CpG острва Витерби учењем и Витерби алгоритмом даје добре резултате.

5. Закључна запажања и даљи рад

У овом раду је показано да се Витерби учење параметара и Витерби алгоритам могу ефикасно користити за обележавање CpG острва и да развијени анотатор (програм) MGcpg даје добре резултате у поређењу са осталим анотаторима.

Применом CMM и одговарајућих метода учења избегава се коришћење технике покретног прозора и смањује ниво људске интервенције (одређивање прага густине и растојања CpG кластера), што даје предност у односу на остале анотаторе. С друге стране, анотатори засновани на алгоритмима који комбинују суседне CpG кластере на основу физичког растојање и густине могу бити рачунарски мање захтевни.

У овом раду вршени су експерименти са и без поступака постпроцесирања, где су примењиване различите вредности за GC садржај и CpG уочено/очекивано однос. Експерименти са поступцима постпроцесирања дали су боље резултате и тако је показано да је њихова употреба оправдана. Најбољи резултати су добијени за вредност GC садржаја 0.6, CpG уочено/очекивано однос од 0.5 или 0.6 и да је дужина острва већа од 200 нуклеотида. То одговара најчешће коришћеној дефиницији CpG острва [11] Gardiner-Garden M, Frommer M (1987): дужина већа од 200 нуклеотида, GC садржај 0.6 и CpG уочено/очекивано однос 0.5.

Као што је у раду изложено, у процесу учења могуће је мењати само транзиционе вероватноће или мењати и транзиционе и емисионе вероватноће. У раду [8] мењане су и транзиционе и емисионе вероватноће. Наши експерименти су показали да мењање емисионих вероватноћа није потребно јер не даје боље резултате.

Даљи рад би могао бити заснован на хибридном коришћењу разних техника машинског учења (једна од њих би наравно била CMM и Витерби учење са одговарајућим алгоритмима). Већина програма користи једну технику и неки поступак за побољшање резултата.

Природно се као добра идеја намеће мета учење. Системи мета учења су системи који обједињују резултате добијене различитим моделима. Најједноставнији облик је већинско гласање (енг. majority voting) који се широко користи у машинском учењу. Када више различитих модела каже да је CpG острво нађено, нема разлога да им не верујемо.

Литература

[1] Richard Durbin, Sean R. Eddy, Anders Krogh, Graeme Mitchison. “Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids”, 1998.

[2] Serafim Batzoglou. http://ai.stanford.edu/~serafim/

[3] Lloyd R. Welch. “Hidden Markov Models and the Baum-Welch Algorithm”. IEEE Information Theory Society Newsletter, Vol. 53, No.4, December 2003.

[4] Burge, C. and Karlin, S. “Prediction of complete gene structures in human genomic DNA”. J. Mol. Biol. **268,** 78-94, 1997.
http://genes.mit.edu/GENSCANinfo.html

[5] Ron Shamir, www.cs.tau.ac.il/~rshamir/

[6] http://en.wikipedia.org/wiki/Transmembrane_protein

[7] “Protein Structure Prediction” edited by Mohammed Zaki, Christopher Bystroff, Rensselaer Polytechnic Institute, Troy, New York, USA, 2008.

[8] Man LAN, Yu XU, Lin LI, Fei WANG, Ying ZUO, Yuan CHEN, Chew Lim TAN and Jian SU, “CpG-Discover: A Machine Learning Approach for CpG Islands Identification from Human DNA Sequence”, Neural Networks, 2009. IJCNN 2009. International Joint Conference on, June 2009.

[9] Wang, Y; Leung, F. “An evaluation of new criteria for CpG islands in the human genome as gene markers”. Bioinformatics, 20(7):1170-1177, 2004.

[10] Ye Sujuan, Asai Asaithambi, and Yunkai Liu. “CpGIF: an algorithm for the identification of CpG islands”. Bioinformation, 2(8): 335-338, 2008.

[11] Gardiner-Garden M, Frommer M (1987). "CpG islands in vertebrate genomes". Journal of Molecular Biology 196 (2): 261–282.

[12] European Molecular Biology Laboratory, ftp://ftp.ebi.ac.uk/pub/databases/cpgisle.