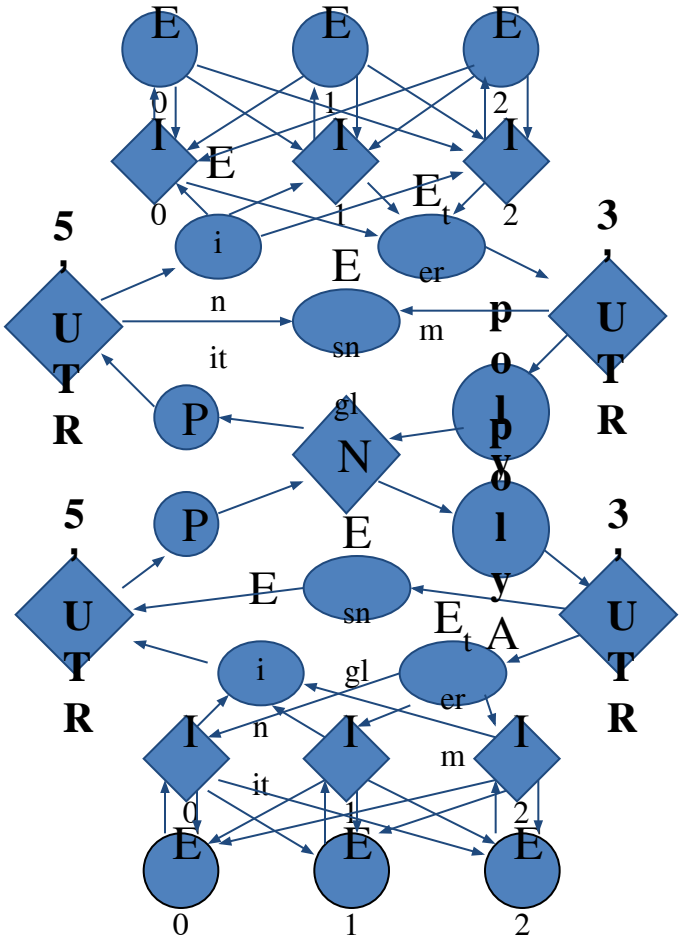
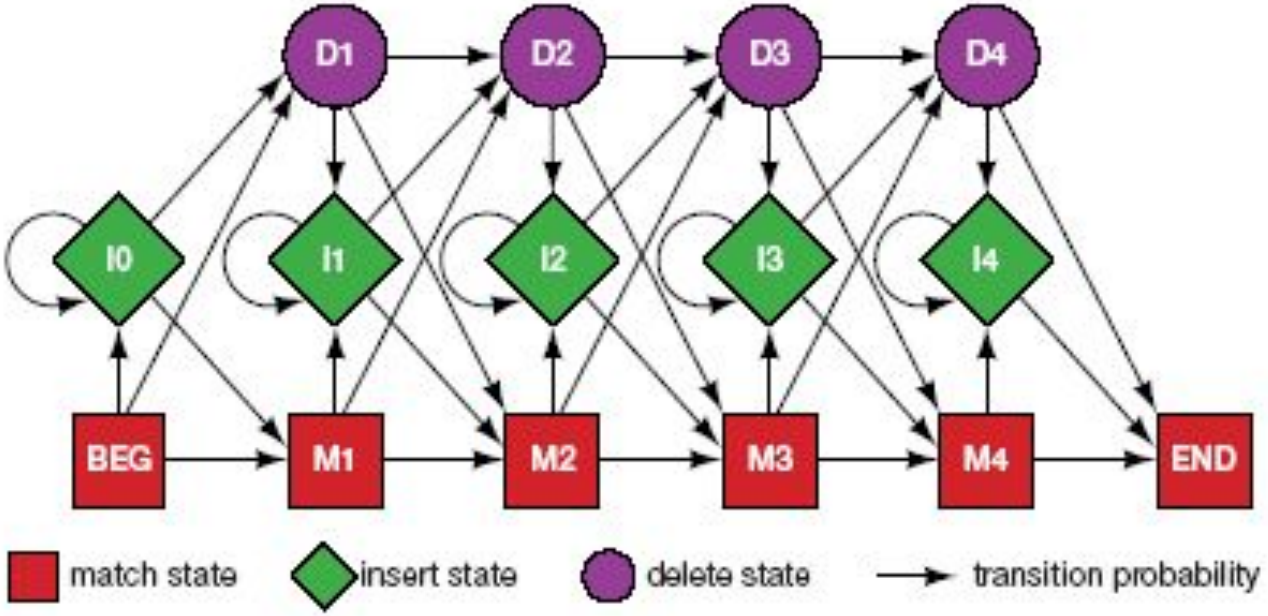
НММ, ПОИСК ГЕНОВ И ПРОФИЛЕЙ





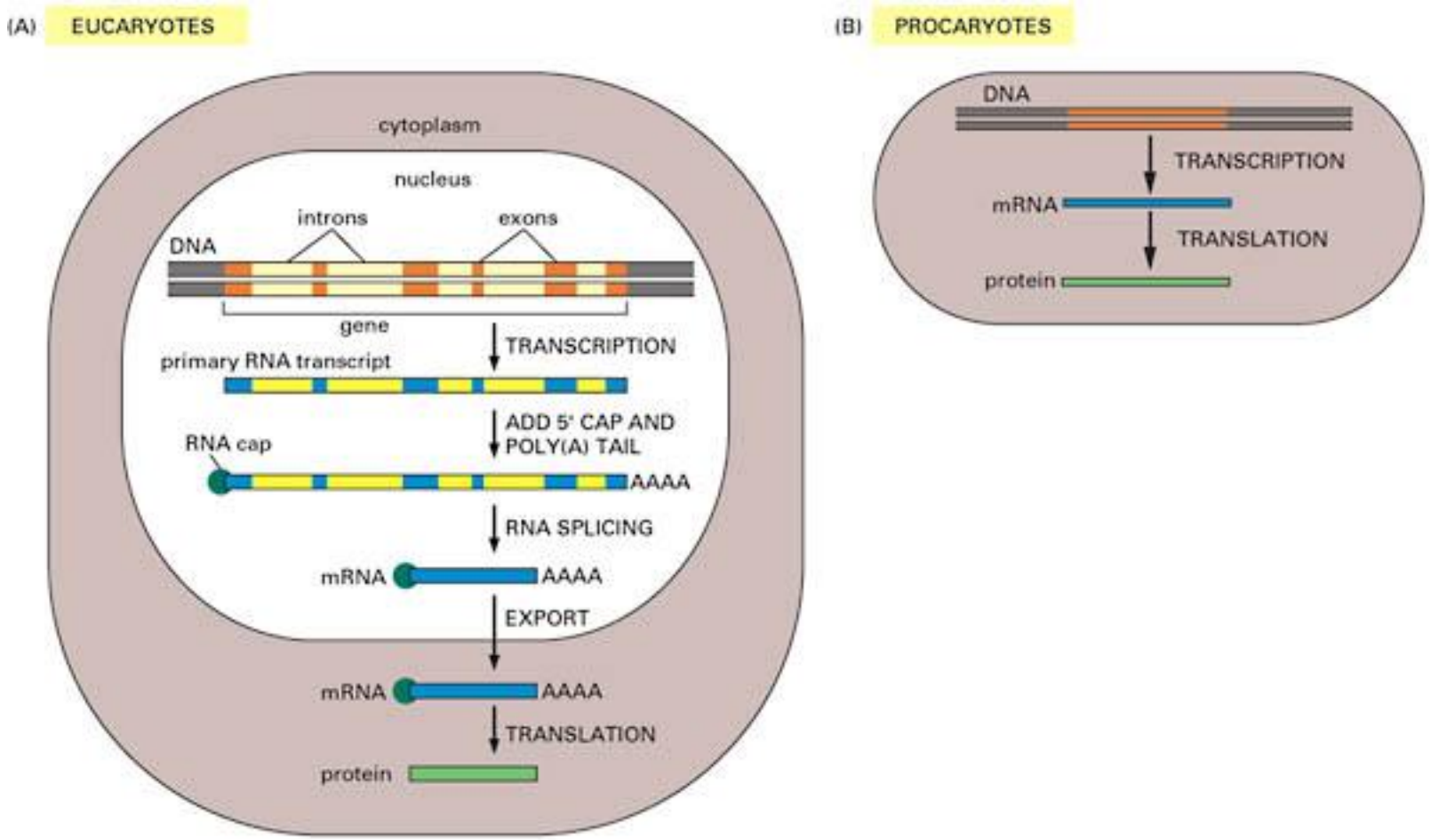
Поиск генов

Index of ftp://ftp.ncbi.nlm.nih.gov/genomes/Bacteria
/Escherichia_coli_K_12_substr__MG1655_uid57779/

Up to higher level directory

Name	Size	Last Modified
NC_000913.GeneMark-2.5m	1052 KB	3/5/13 5:10:00 AM
NC_000913.GeneMarkHMM-2.6r	263 KB	3/5/13 5:10:00 AM
NC_000913.Glimmer3	175 KB	3/5/13 5:09:00 AM
NC_000913.Prodigal-2.50	930 KB	3/5/13 5:11:00 AM
NC_000913.asn	19596 KB	3/5/13 2:02:00 AM
NC_000913.faa	1778 KB	3/5/13 2:02:00 AM
NC_000913.ffn	4498 KB	3/5/13 2:02:00 AM
NC_000913.fna	4596 KB	3/5/13 2:02:00 AM
NC_000913.frn	60 KB	3/5/13 2:02:00 AM
NC_000913.gbk	18078 KB	3/5/13 2:02:00 AM
NC_000913.gff	2265 KB	3/5/13 2:02:00 AM
NC_000913.ptt	387 KB	3/5/13 2:02:00 AM
NC_000913.rnt	10 KB	3/5/13 2:02:00 AM
NC_000913.rpt	1 KB	3/5/13 2:02:00 AM
NC_000913.val	8013 KB	3/5/13 2:02:00 AM

Gene Structure



What is it about genes that we can measure (and model)?

- Most of our knowledge is biased towards protein-coding characteristics
  - ORF (Open Reading Frame): a sequence defined by in-frame AUG and stop codon, which in turn defines a putative amino acid sequence.
  - Codon Usage: most frequently measured by CAI (Codon Adaptation Index)
- Other phenomena
  - Nucleotide frequencies and correlations:
    - value and structure
  - Functional sites:
    - splice sites, promoters, UTRs, polyadenylation sites

Статистика кодирующей последовательности

- Неравное использование кодонов в кодирующих областях – универсальная характеристика геномов.
  - Неравное использование аминокислот в существующих белках
  - Неравное использование синонимичных кодонов (коррелирует с избытком соответствующих tRNAs)
- Эти характеристики могут быть использованы для разделения между кодирующими и некодирующими областями генома.
- Статистика кодирования – функция, которая для данной ДНК последовательности вычисляет правдоподобие (условную вероятность) того, что последовательность является кодирующей для белка

An Example of Coding Statistics

The Human Codon Usage Table

Gly	GGG	17.08	0.23	Arg	AGG	12.09	0.22	Trp	TGG	14.74	1.00	Arg	CGG	10.40	0.19
Gly	GGA	19.31	0.26	Arg	AGA	11.73	0.21	End	TGA	2.64	0.61	Arg	CGA	5.63	0.10
Gly	GGT	13.66	0.18	Ser	AGT	10.18	0.14	Cys	TGT	9.99	0.42	Arg	CGT	5.16	0.09
Gly	GGC	24.94	0.33	Ser	AGC	18.54	0.25	Cys	TGC	13.86	0.58	Arg	CGC	10.82	0.19
Glu	GAG	38.82	0.59	Lys	AAG	33.79	0.60	End	TAG	0.73	0.17	Gln	CAG	32.95	0.73
Glu	GAA	27.51	0.41	Lys	AAA	22.32	0.40	End	TAA	0.95	0.22	Gln	CAA	11.94	0.27
Asp	GAT	21.45	0.44	Asn	AAT	16.43	0.44	Tyr	TAT	11.80	0.42	His	CAT	9.56	0.41
Asp	GAC	27.06	0.56	Asn	AAC	21.30	0.56	Tyr	TAC	16.48	0.58	His	CAC	14.00	0.59
Val	GTG	28.60	0.48	Met	ATG	21.86	1.00	Leu	TTG	11.43	0.12	Leu	CTG	39.93	0.43
Val	GTA	6.09	0.10	Ile	ATA	6.05	0.14	Leu	TTA	5.55	0.06	Leu	CTA	6.42	0.07
Val	GTT	10.30	0.17	Ile	ATT	15.03	0.35	Phe	TTT	15.36	0.43	Leu	CTT	11.24	0.12
Val	GTC	15.01	0.25	Ile	ATC	22.47	0.52	Phe	TTC	20.72	0.57	Leu	CTC	19.14	0.20
Ala	GCG	7.27	0.10	Thr	ACG	6.80	0.12	Ser	TCG	4.38	0.06	Pro	CCG	7.02	0.11
Ala	GCA	15.50	0.22	Thr	ACA	15.04	0.27	Ser	TCA	10.96	0.15	Pro	CCA	17.11	0.27
Ala	GCT	20.23	0.28	Thr	ACT	13.24	0.23	Ser	TCT	13.51	0.18	Pro	CCT	18.03	0.29
Ala	GCC	28.43	0.40	Thr	ACC	21.52	0.38	Ser	TCC	17.37	0.23	Pro	CCC	20.51	0.33

Codon Adaptation Index (CAI)

$$CAI = \prod_{i=codons} \left[\frac{f_{codon_i}}{f_{(codon_i)_{\max}}} \right]$$

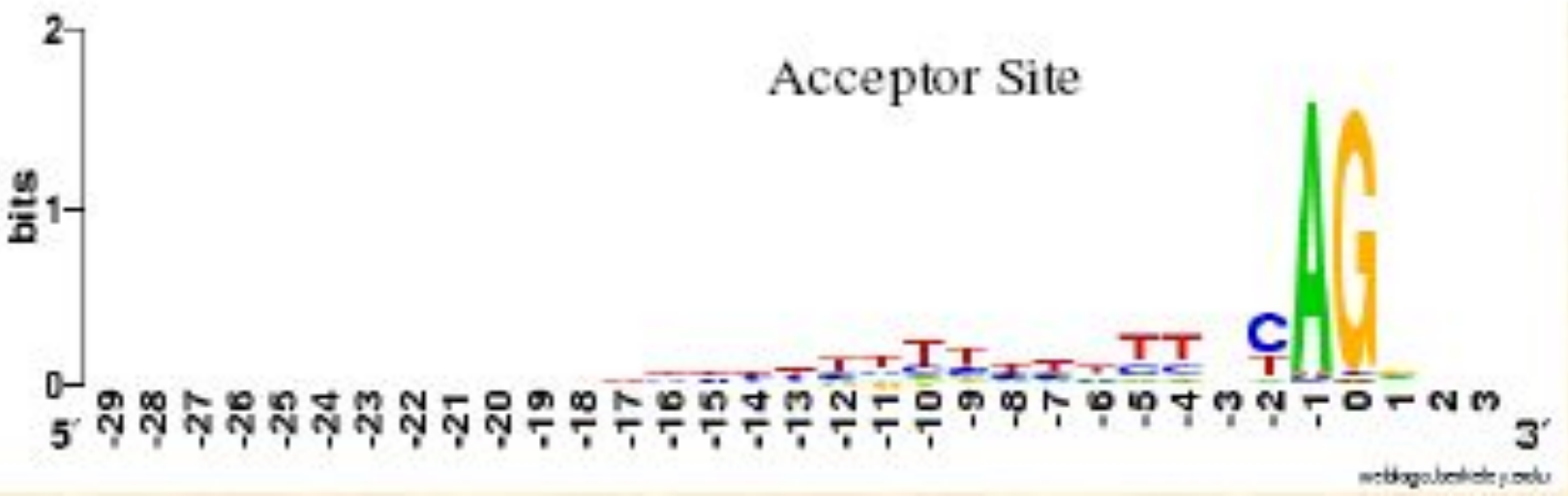
the geometric mean of the weight associated to each codon over the length of the gene sequence (measured in codons).

- This is not perfect
  - Genes sometimes have unusual codons for a reason
  - The predictive power is dependent on length of sequence

CAI Example: Counts per 1000 codons

		hs	sc			hs	sc			hs	sc			hs
UUU	Phe	16.6	26.0	UCU	Ser	14.5	23.6	UAU	Tyr	12.1	18.8	UGU	Cys	9.7
UUC	Phe	20.7	18.2	UCC	Ser	17.7	14.2	UAC	Tyr	16.3	14.7	UGC	Cys	12.4
UUA	Leu	7.0	26.3	UCA	Ser	11.4	18.8	UAA	stop	0.7	1.0	UGA	stop	1.3
UUG	Leu	12.0	27.1	UCG	Ser	4.5	8.6	UAG	stop	0.5	0.5	UGG	Trp	13.0
CUU	Leu	12.4	12.2	CCU	Pro	17.2	13.6	CAU	His	10.1	13.7	CGU	Arg	4.7
CUC	Leu	19.3	5.4	CCC	Pro	20.3	6.8	CAC	His	14.9	7.8	CGC	Arg	11.0
CUA	Leu	6.8	13.4	CCA	Pro	16.5	18.2	CAA	Gln	11.8	27.5	CGA	Arg	6.2
CUG	Leu	40.0	10.4	CCG	Pro	7.1	5.3	CAG	Gln	34.4	12.2	CGG	Arg	11.6
AUU	Ile	15.7	30.2	ACU	Thr	12.7	20.2	AAU	Asn	16.8	36.0	AGU	Ser	11.7
AUC	Ile	22.3	17.1	ACC	Thr	19.9	12.6	AAC	Asn	20.2	24.9	AGC	Ser	19.3
AUA	Ile	7.0	17.8	ACA	Thr	14.7	17.7	AAA	Lys	23.6	42.1	AGA	Arg	11.2
AUG	Met	22.2	20.9	ACG	Thr	6.4	8.0	AAG	Lys	33.2	30.8	AGG	Arg	11.1
GUU	Val	10.7	22.0	GCU	Ala	18.4	21.1	GAU	Asp	22.2	37.8	GGU	Gly	10.9
GUC	Val	14.8	11.6	GCC	Ala	28.6	12.6	GAC	Asp	26.5	20.4	GGC	Gly	23.1
GUA	Val	6.8	11.7	GCA	Ala	15.6	16.2	GAA	Glu	28.6	45.9	GGA	Gly	16.4
GUG	Val	29.3	10.7	GCG	Ala	7.7	6.1	GAG	Glu	40.6	19.1	GGG	Gly	16.5

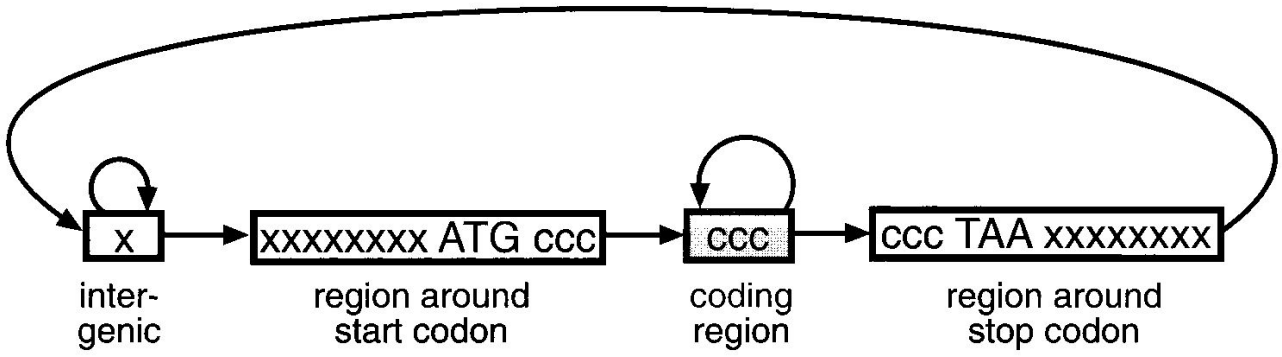
Splice signals (mice): GT , AG



HMMs and Prokaryotics Gene Structure

- Nucleotides *{A,C,G,T}* are the observables
- Different states generate nucleotides at different frequencies

A simple HMM for unspliced genes:



AAAGC A

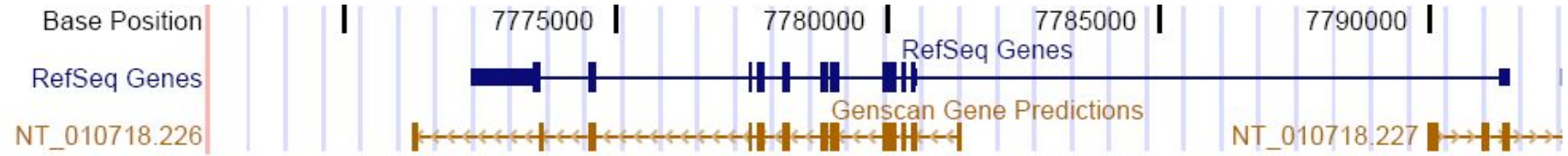
- The sequence of states is an annotation of the generated string – each nucleotide is generated in intergenic, start/stop, coding state
- This HMM has 4 states: x- non-coding, c- coding, start and stop

Parse

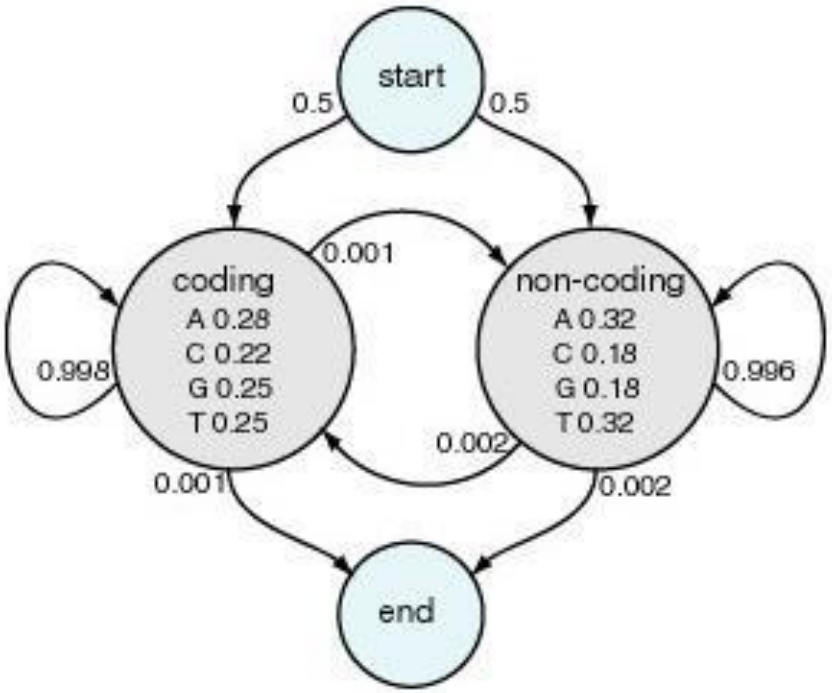
```
S =
ACTGACTACTACGACTACGATCTACTACGGGCGCGACCTATGCG
P =
IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIGGGGG

TATGTTTTGAACTGACTATGCGATCTACGACTCGACTAGCTAC
GGGGGGGGGGIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
```

- For a given sequence, a parse is an assignment of gene structure to that sequence.
- In a parse, every base is labeled, corresponding to the content it (is predicted to) belongs to.
- In our simple model, the parse contains only “I” (intergenic) and “G” (gene).
- A more complete model would contain, e.g., “-” for intergenic, “E” for exon and “I” for intron.



The HMM Matrixes: Φ and H



$$\Phi = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.5 & 0.998 & 0.002 & 0 \\ 0.5 & 0.001 & 0.996 & 0 \\ 0 & 0.001 & 0.002 & 0 \end{bmatrix}$$

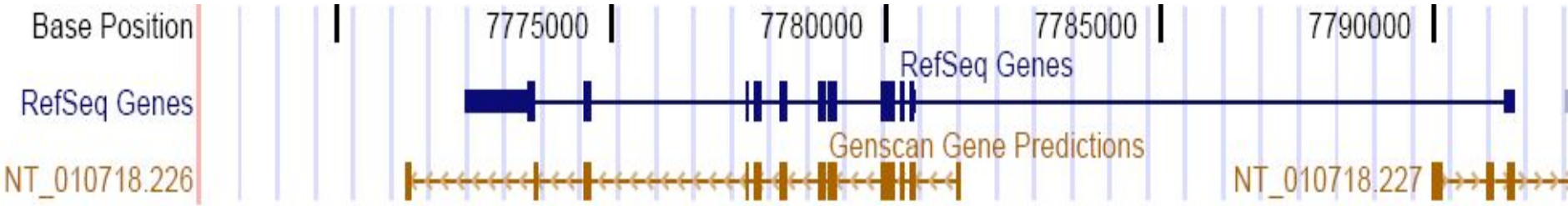
$$H = \begin{bmatrix} 0.28 & 0.32 \\ 0.22 & 0.18 \\ 0.25 & 0.18 \\ 0.25 & 0.32 \end{bmatrix}$$

$x_m(i)$ = probability of being in state m at position i;

$H(m,y_i)$ = probability of emitting character y_i in state m;

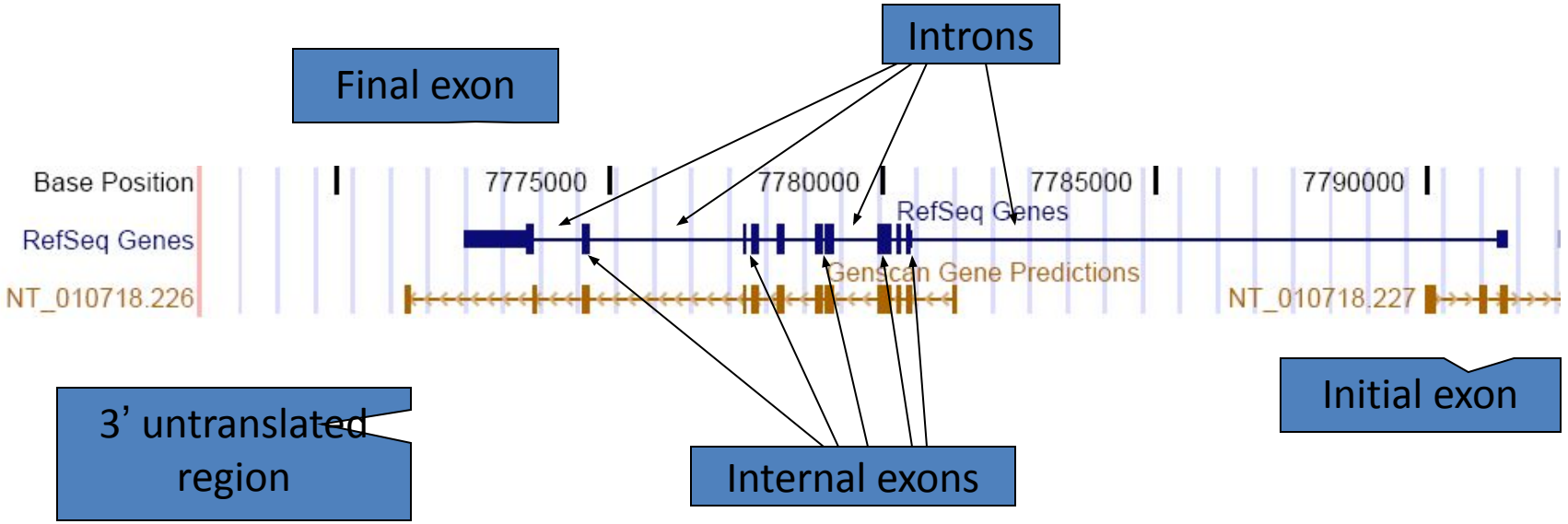
Φ_{mk} = probability of transition from state k to m.

A eukaryotic gene



- This is the human p53 tumor suppressor gene on chromosome 17.
- Genscan is one of the most popular gene prediction algorithms.

A eukaryotic gene



This particular gene lies on the reverse strand.

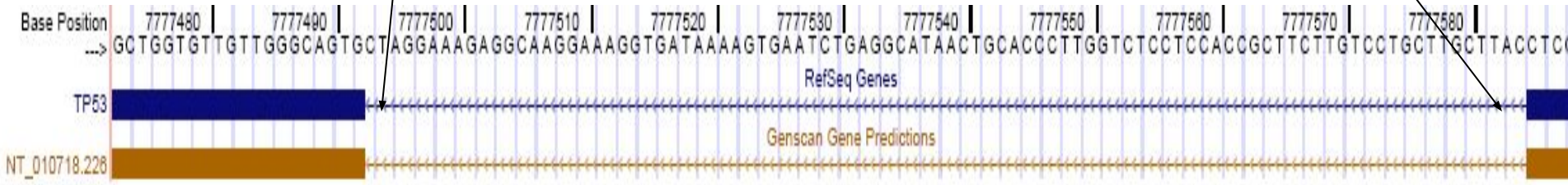
An Intron

revcomp(CT)=AG

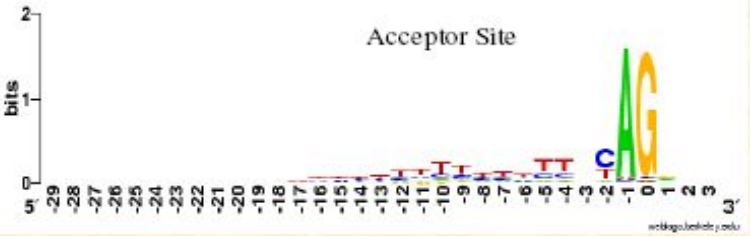
GT: signals start of intron

AG: signals end of intron

revcomp(AC)=GT



3' splice site



5' splice site

Signals vs contents

- In gene finding, a small pattern within the genomic DNA is referred to as a signal, whereas a region of genomic DNA is a content.
- Examples of signals: splice sites, starts and ends of transcription or translation, branch points, transcription factor binding sites
- Examples of contents: exons, introns, UTRs, promoter regions

Prior knowledge

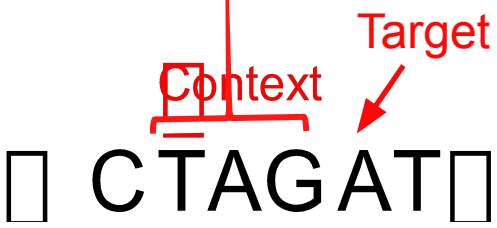
- We want to build a probabilistic model of a gene that incorporates our prior knowledge.
- E.g., the translated region must have a length that is a multiple of 3.

Prior knowledge

- The translated region must have a length that is a multiple of 3.
- Some codons are more common than others.
- Exons are usually shorter than introns.
- The translated region begins with a start signal and ends with a stop codon.
- 5’ splice sites (exon to intron) are usually GT;
- 3’ splice sites (intron to exon) are usually AG.
- The distribution of nucleotides and dinucleotides is usually different in introns and exons.

Цепи Маркова высокого порядка

- k^{th}-order Markov model bases the probability of an event on the preceding k events.
- Example: With a 3rd-order model the probability of this sequence:



would be:

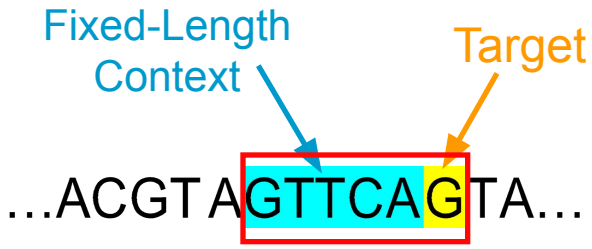
$$P(G \mid CTA) \cdot P(A \mid TAG) \cdot P(T \mid AGA)$$

Target

Context

Цепи Маркова высокого порядка

- Advantages:
  - Easy to train. Count frequencies of (*k*+1)-mers in training data.
  - Easy to compute probability of sequence.
- Disadvantages:
  - Many (*k*+1)-mers may be undersampled in training data.
  - Models data as fixed-length chunks.



Journal of Molecular Biology

Volume 268, Issue 1, 25 April 1997, Pages 78–94

Genscan Example

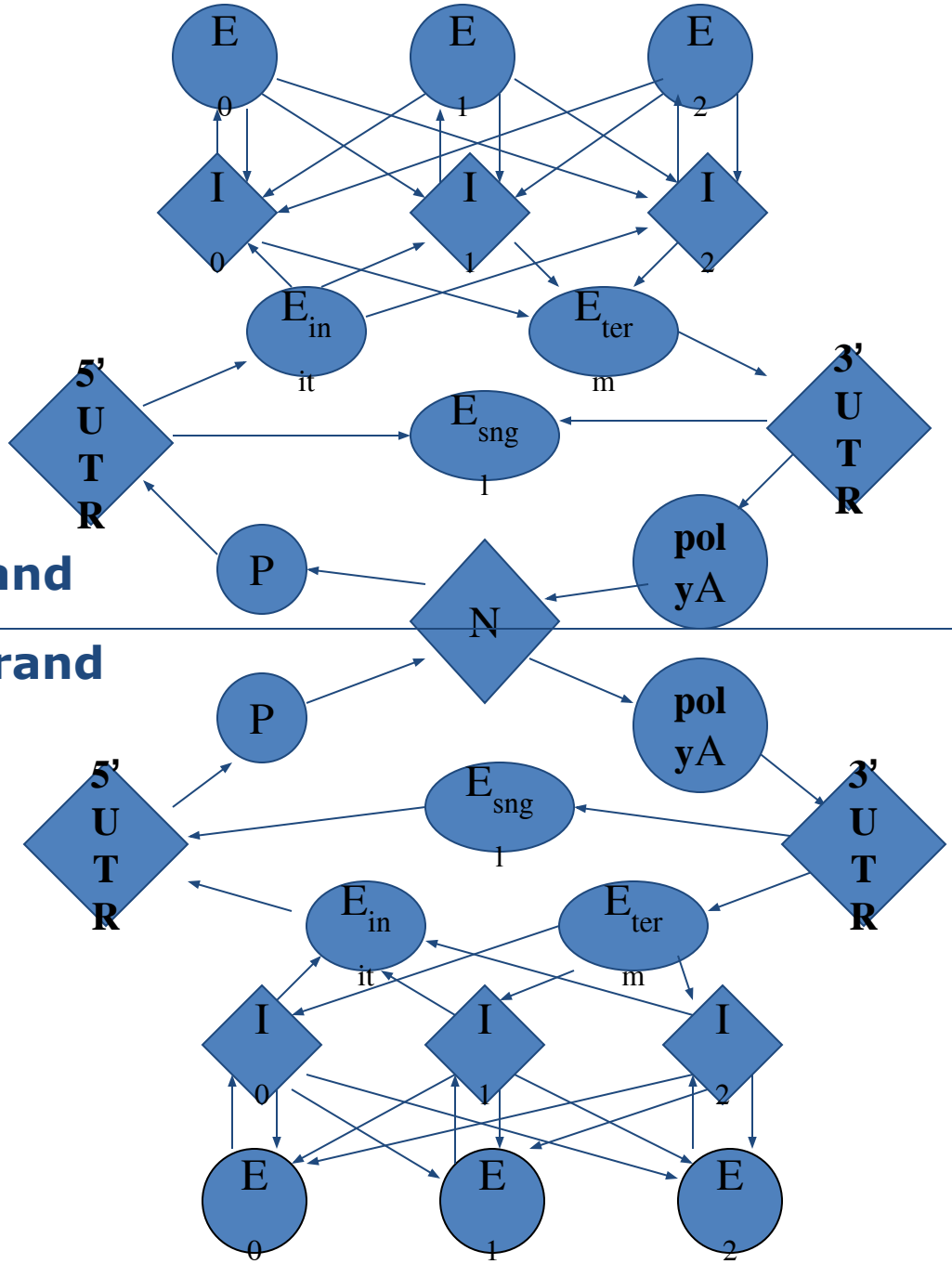
Prediction of complete gene structures in human genomic DNA

Chris Burge[a], , Samuel Karlin[a]

[a] Department of Mathematics Stanford University, Stanford CA, 94305, USA

http://dx.doi.org/10.1006/jmbi.1997.0951, How to Cite or Link Using DOI

- Uses explicit state duration HMM to model gene structure (different length distributions for exons)
- Different model parameters for regions with different GC content



E- exons
I- introns
single exon
5' UTRs
3' UTRs
P- promoter
region polyA site
N- intergenic
region

Nucleic Acids Research, 1998, Vol. 26, No. 4 1107–1115

GeneMark.hmm: new solutions for gene finding

Alexander V. Lukashin and Mark Borodovsky[1,*]

School of Biology and [1]Schools of Biology and Mathematics, Georgia Institute of Technology, Atlanta, GA 30332-0230, USA

Received August 14, 1997; Revised and Accepted December 30, 1997

GeneMark.hmm

http://nar.oxfordjournals.org/content/26/4/1107

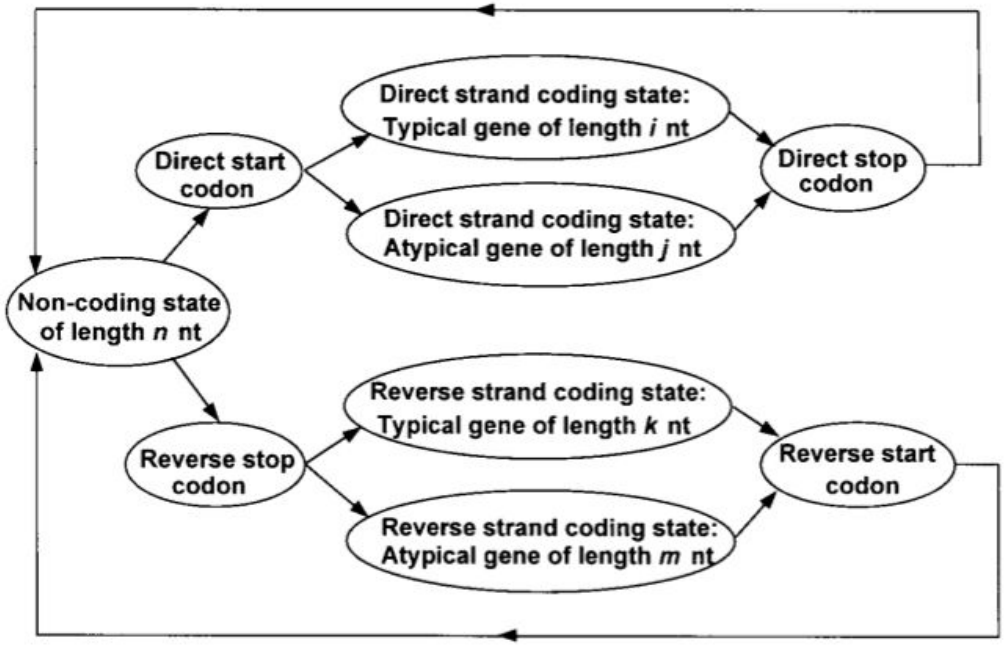


Figure 1. Hidden Markov model of a prokaryotic nucleotide sequence used in the GeneMark.hmm algorithm. The hidden states of the model are represented as ovals in the figure, and arrows correspond to allowed transitions between the states.

GeneMark

- Borodovsky & McIninch, *Comp. Chem* 17, 1993.
- Uses 5^{th}-order Markov model.
- Model is 3-periodic, *i.e.*, a separate model for each nucleotide position in the codon.
- DNA region gets 7 scores: 6 reading frames & non-coding—high score wins.
- Lukashin & Borodovsky, *Nucl. Acids Res.* 26, 1998 is the HMM version.

Interpolated Markov Models (IMM)

- Introduced in Glimmer 1.0

  Salzberg, Delcher, Kasif & White, *NAR* 26, 1998.
- Probability of the target position depends on a *variable* number of previous positions (sometimes 2 bases, sometimes 3, 4, etc.)
- How many is determined by the specific context.
- *E.g.*, for context ggtta the next position might depend on previous 3 bases tta .

  But for context catta all 5 bases might be used.

Real IMMs

- Model has additional probabilities, λ, that determine which parts of the context to use.
- *E.g.*, the probability of g occurring after context atca is:

$$\begin{aligned}
&\lambda(\text{atca})P(\text{g}\,|\,\text{atca}) \\
&+(1-\lambda(\text{atca}))[\lambda(\text{tca})P(\text{g}\,|\,\text{tca}) \\
&+(1-\lambda(\text{tca}))[\lambda(\text{ca})P(\text{g}\,|\,\text{ca}) \\
&+(1-\lambda(\text{ca}))[\lambda(\text{a})P(\text{g}\,|\,\text{a}) \\
&+(1-\lambda(\text{a}))P(\text{g})]]]
\end{aligned}$$

Real IMMs

- Result is a linear combination of different Markov orders:

$$b_4 P(\mathrm{g} \mid \mathrm{atca}) + b_3 P(\mathrm{g} \mid \mathrm{tca}) + b_2 P(\mathrm{g} \mid \mathrm{ca}) + b_1 P(\mathrm{g} \mid \mathrm{a}) + b_0 P(\mathrm{g})$$

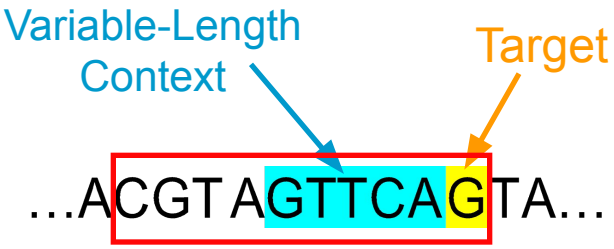
  where

$$b_0 + b_1 + b_2 + b_3 + b_4 = 1$$

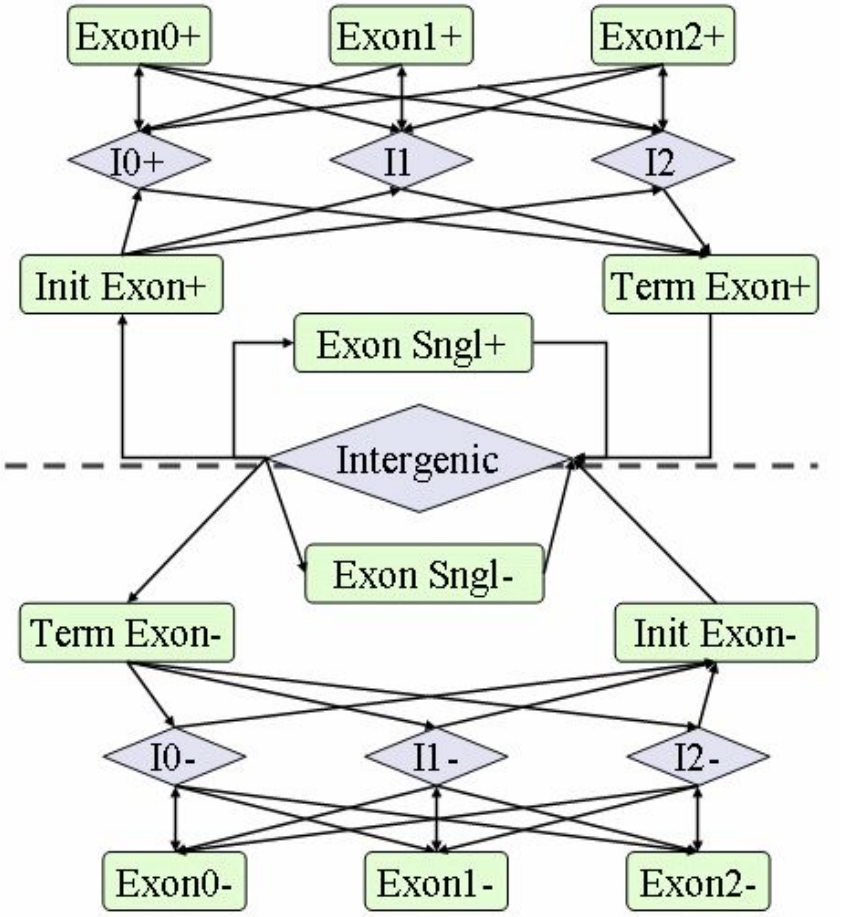
- Can view this as *interpolating* the results of different-order models.
- The probability of a sequence is still the probability of the bases in the sequence.

IMMs *vs* Fixed-Order Models

- Performance
  - IMM generally should do at least as well as a fixed-order model.
  - Some risk of overtraining.
- IMM result can be stored and used like a fixed-order model.
- IMM will be somewhat slower to train and will use more memory.



GLIMMER-HMM



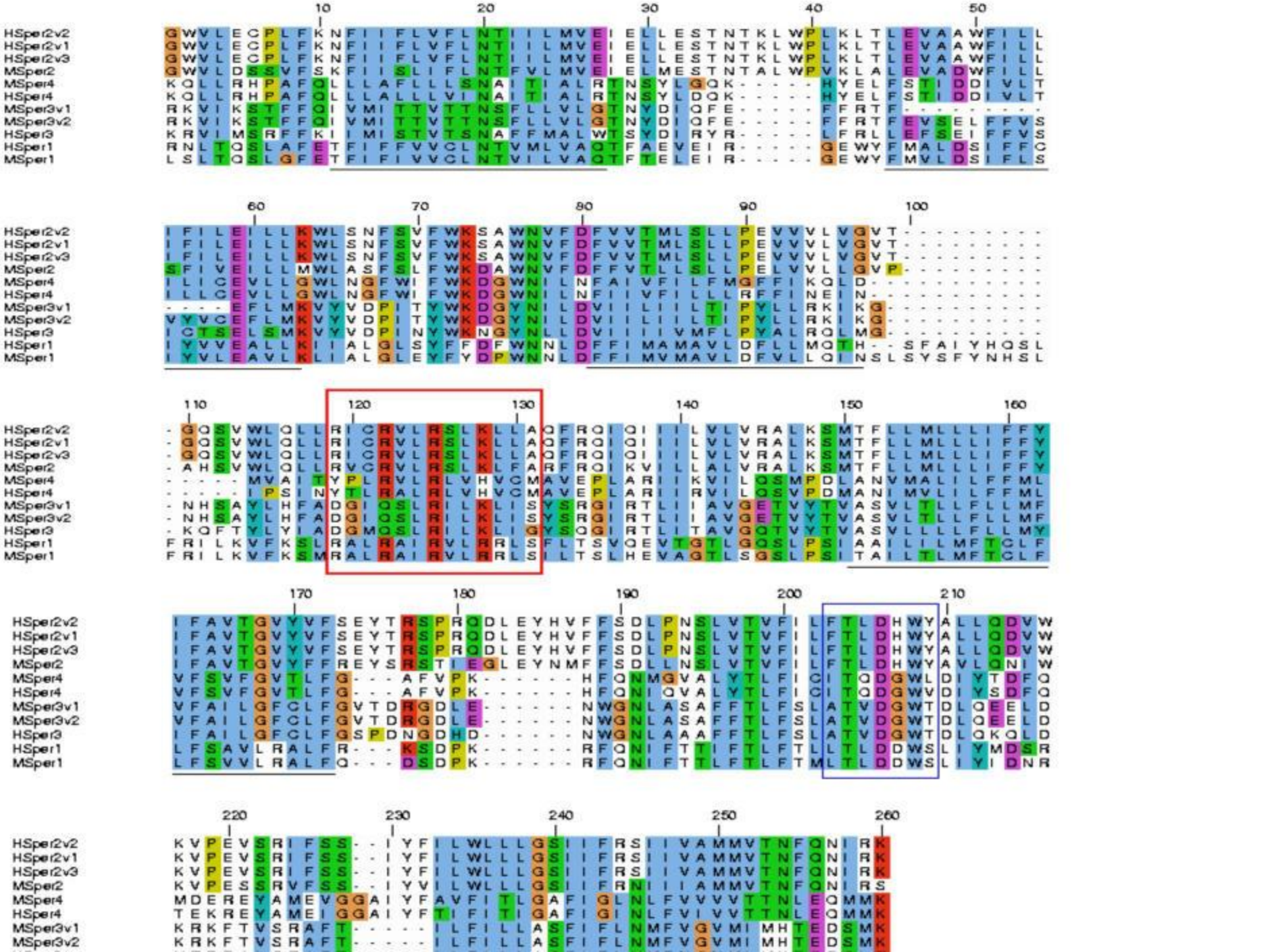
Nth-order *interpolated Markov models* (*IMM*) *(N=8)*

General Things to Remember about (Protein-coding) Gene Prediction Software

- It is, in general, organism-specific
- It works best on genes that are *reasonably* similar to something seen previously
- It finds protein coding regions far better than non-coding regions
- In the absence of external (direct) information, alternative forms will not be identified
- It is imperfect! (It's biology, after all…)

Профильные HMM
Profile HMM

- Берем множественное выравнивание и делаем из него статистическую модель.


HSper2v2
HSper2v1
HSper2v3
MSper2
MSper4
HSper4
MSper3v1
MSper3v2
HSper3
HSper1
MSper1

Profile HMMs

- Моделирует семейство последовательностей
- Вычисляется из множественного выравнивания семейства
- Вероятности переходов состояний и испускания данных зависят от позиции выравнивания (position-specific)
- Надо установить параметры модели такими, чтобы полная вероятность достигала максимума для членов семейства.
- Последовательности могут быть протестированы на принадлежность семейству, используя алгоритм Витерби для оценки совпадения с профилем

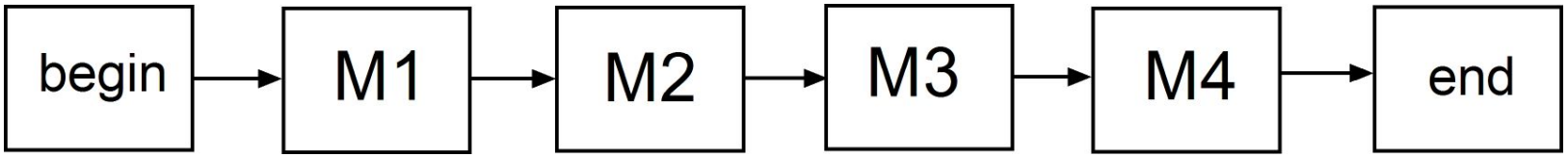
Строим модель: состояния совпадения (Match States)

- Если нам нужно выполнить выравнивание без пропусков, то мы можем использовать простую, неразветвленную HMM, где из каждого состояния совпадения можно перейти в другое состояния совпадения

- Для каждого состояния существует вероятность испускания аминокислоты, которые зависят от состояния совпадения

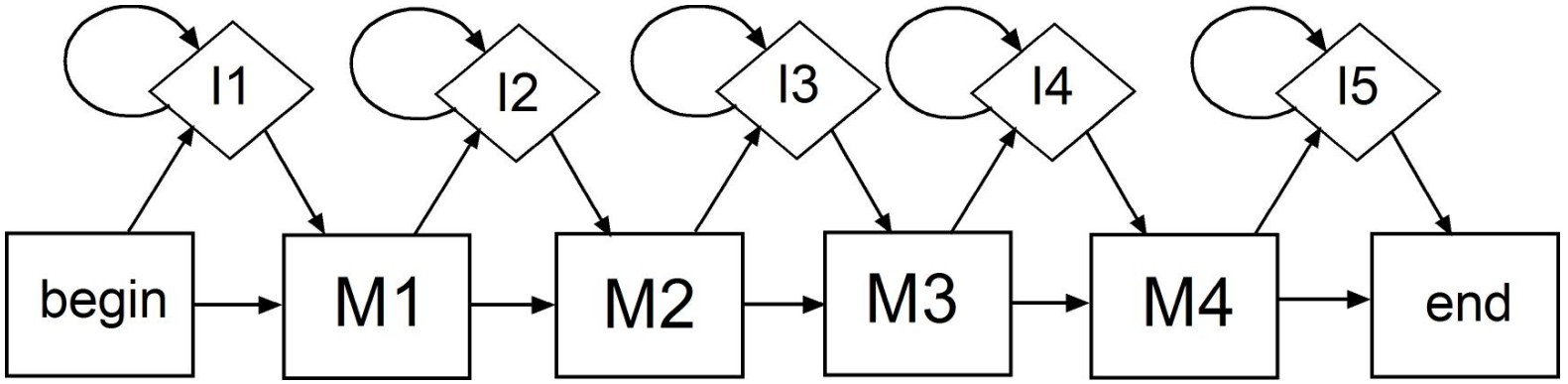
  По существу это PSSM (Position Specific Scoring Matrix): вес каждой колонки PSSM может быть отмасштабирован от 0 до 1 в соответствии с вероятностями испускания.

  Все вероятности переходов назначаются 1: существует только один выбор – двигаться в следующее состояния совпадения.



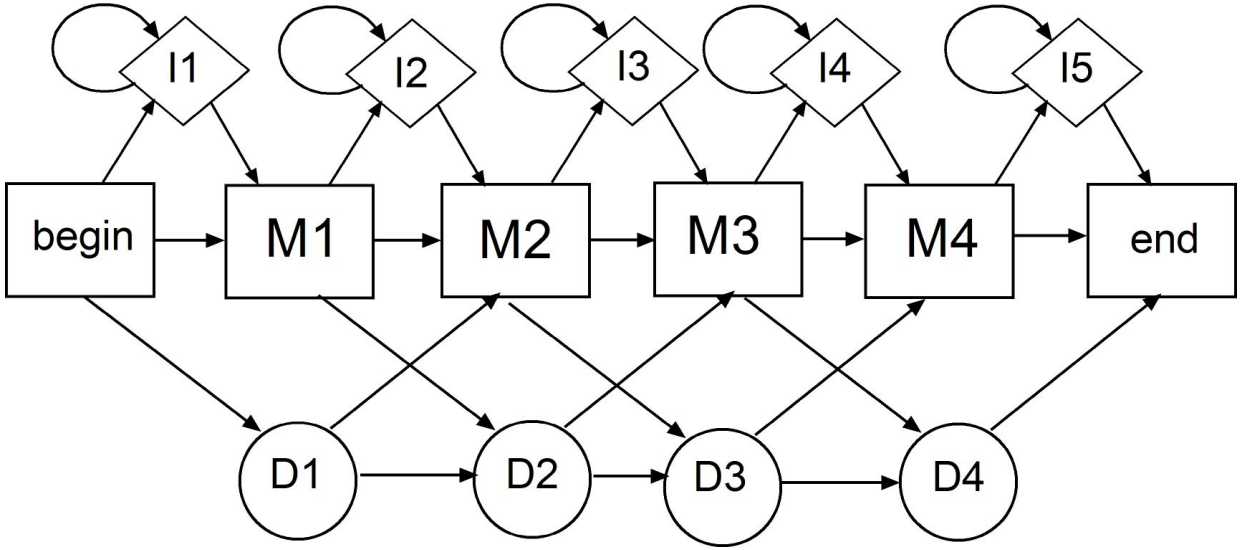
Состояния вставки
Insertion States

- Во множественном выравнивании часто встречаются колонки, являющиеся пропусками в большинстве последовательностях, но содержащие аминокислоты в некоторых.
  - Такие колонки лучше обозначать как состояния вставки.
- По мере продвижения по модели и генерирования искомой последовательности, состояния вставки генерируют экстра аминокислоты, находящиеся в этих колонках.
- Состояния вставки обладают вероятностями испускания, которые обычно такие же, как и общая пропорция каждой аминокислоты в базе данных.
- Состояния вставки замыкаются на себя, что означает, что множество позиций может быть испущено в этом состоянии.
- В состояние вставки можно войти из одного состояния совпадения, но выход происходит уже в следующее: вставка происходит между соседними аминокислотами.



Состояние делиции
Deletion States

- Делициями во множественном выравнивании называют позиции, в которых большинство последовательностей имеют аминокислоты, и только небольшое количество – пропуски.
- Состояния делиции используются для того, чтобы перескочить между состояниями.
  - Допускается пропуск состояний совпадения, переходя из одного состояния делиции в другое.
  - Состояния делиции действуют как афинные штрафы: вероятности перехода из состояния совпадения в состояния делиции равнозначно штрафу за открытие разрыва, и переход из одного состояния делиции в другое равнозначно штрафу за продолжения разрыва.
- В противоположность состояниям совпадения и состояниям вставки, состояния делиций являются молчащими, они ничего не испускают.



Profile HMMs

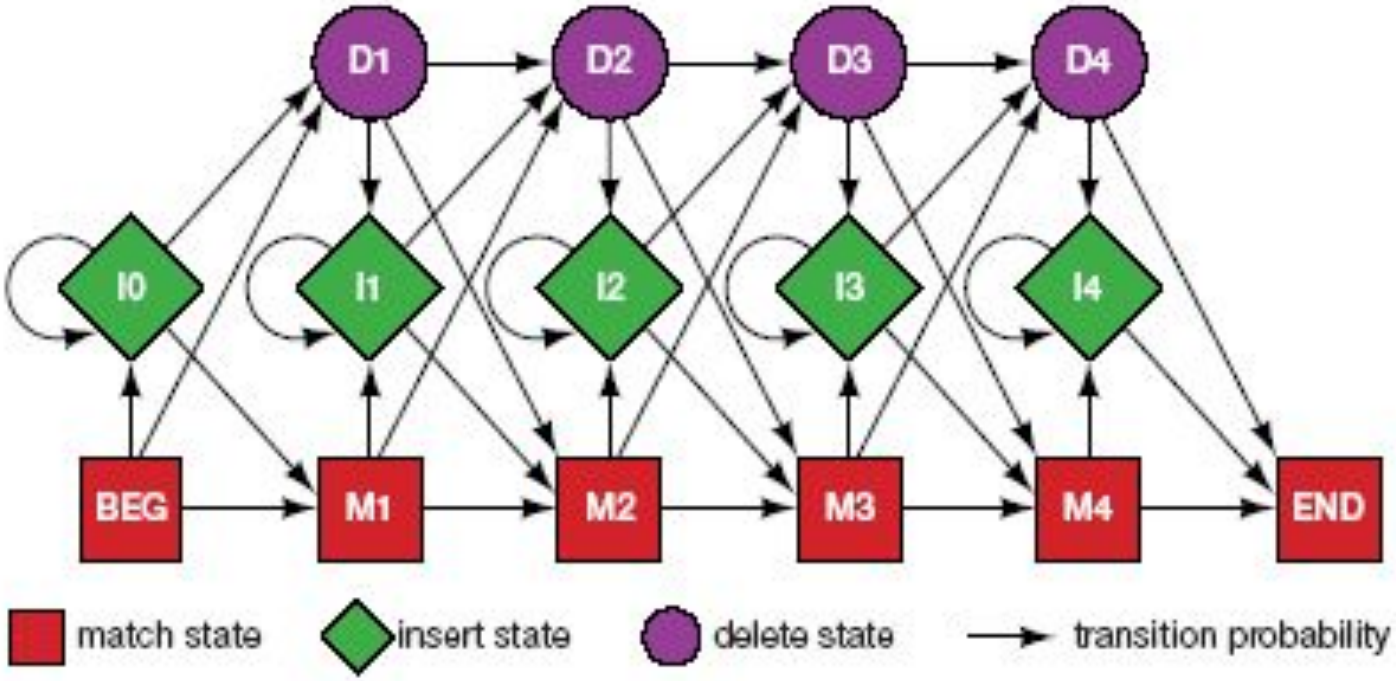
A. Sequence alignment

N	•	F	L	S
N	•	F	L	S
N	K	Y	L	T
Q	•	W	–	T

RED POSITION REPRESENTS ALIGNMENT IN COLUMN
GREEN POSITION REPRESENTS INSERT IN COLUMN
PURPLE POSITION REPRESENTS DELETE IN COLUMN

Существует также переход из состоянии вставки в состояние делиции, но такие переходы считаются маловероятными, и их существование помогает при построении модели

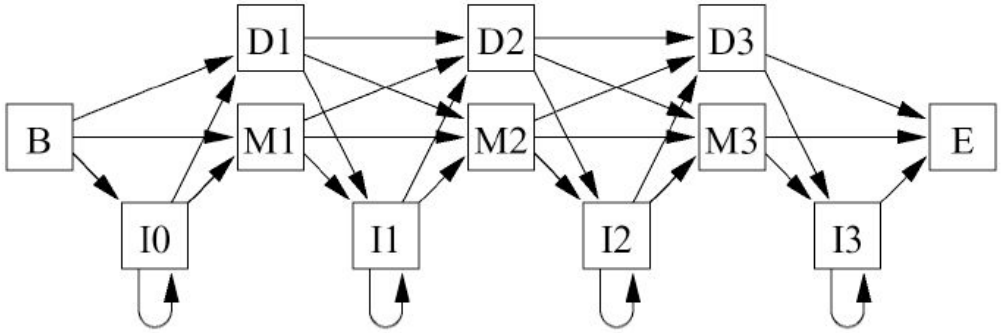
B. Hidden Markov model for sequence alignment



Profile HMMs: Example

An alignment of proteins from the HMM:

-	E	G	-	K	-
-	E	A	-	K	-
P	D	-	-	K	L
-	E	G	I	W	-



The states giving this alignment:

B → M1 → M2 → M3 → E

B → M1 → M2 → M3 → E

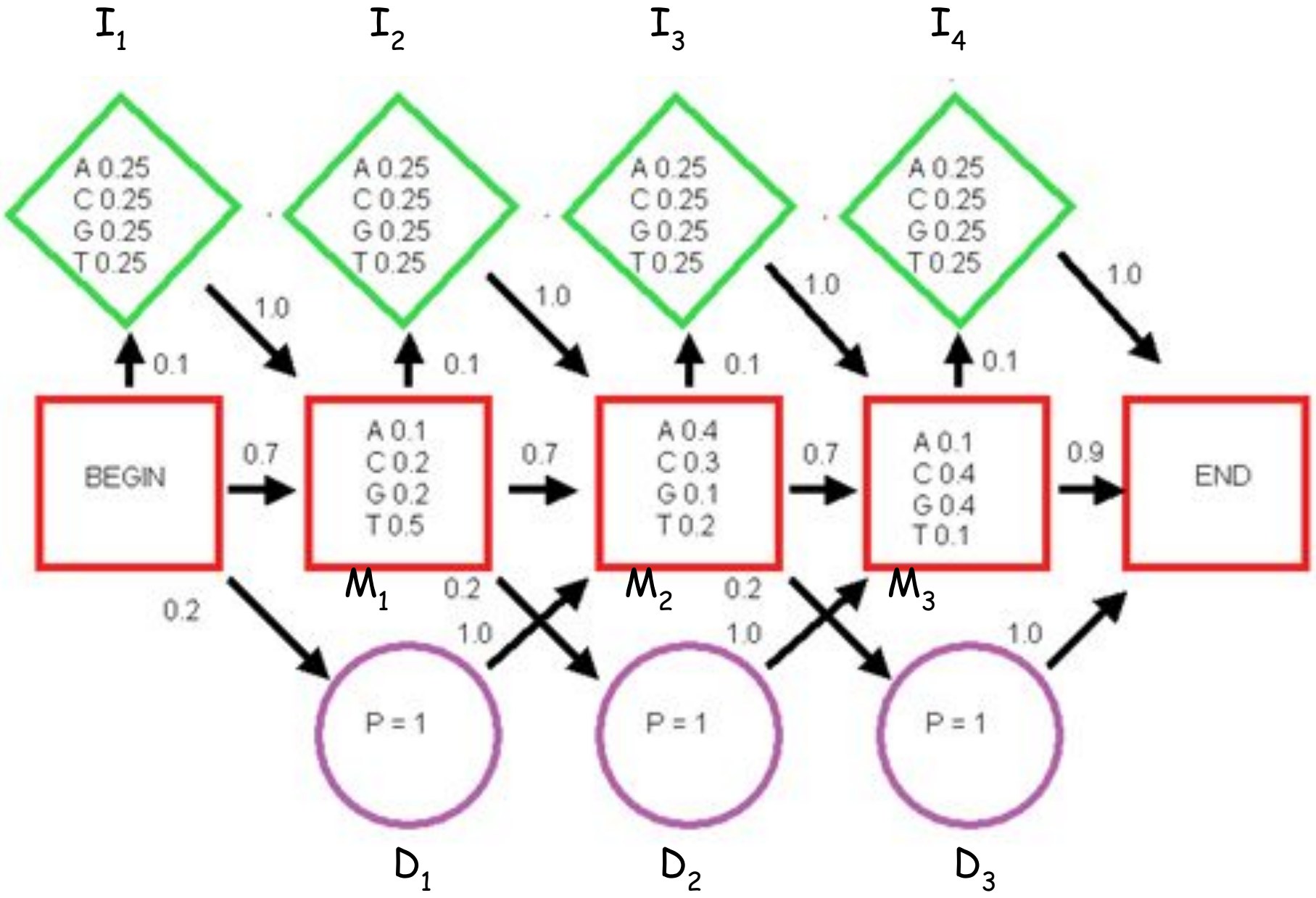
B → I0 → M1 → D2 → M3 → I3 → E

B → M1 → M2 → I2 → M3 → E

Note: These sequences could lead to other paths.

Pfam

- “A comprehensive collection of protein domains and families, with a range of well-established uses including genome annotation.”
- Each family is represented by two multiple sequence alignments and two profile-Hidden Markov Models (profile-HMMs).
- A. Bateman et al. ***Nucleic Acids Research*** (2004) Database Issue 32:D138-D141


I1
I2
I3
I4
A 0.25
C 0.25
G 0.25
T 0.25
1.0
0.1
BEGIN
0.7
A 0.1
C 0.2
G 0.2
T 0.5
A 0.4
C 0.3
G 0.1
T 0.2
A 0.1
C 0.4
G 0.4
T 0.1
0.9
END
M1
M2
M3
0.2
P = 1
D1
D2
D3

A Profile HMM Example

- This is a section of a repeated sequence in *Bacillus megaterium.*
- 15 последовательностей, и выравнивание имеет длину 16 оснований.
- Сначала параметризуем модель, то есть оцениваем вероятности переходов и испускания.
- После этого модель может использоваться для оценки разных последовательностей.

```
GG-GGAAAAACGTATT
TG-GGACAAAAGTATT
TG-GAACAAAAGTATG
TACGGACAAAATTATT
T--GAAGAAAAGTATG
TA-GAACAAAAGTAGG
TG-GAACAAACGCATT
CGGGACAAA-AGTATT
TGGGGTAAA-AGTATT
TGAGACAAA-AGTAGT
TGAGACAAA-AGTATA
TGGGACAAAGAGTATT
TG-AAACAAAGATATT
CG-GAACAAAAGTATT
TA-GGACAAAAGTGTT
```

Создание модели

- Что называть вставками, что делициями?
  - >50% пропусков -> вставка
  - <50% пропусков -> делиция
- 9 последовательностей имеют разрыв в третьей колонке и одна последовательность имеет разрыв в колонке 2.
  - По определенному правилу колонка 3 должна быть вставкой, а колонка 2 – делицией, но это означает, что у нас будет переход сразу от делиции ко вставке, а этого следует избегать.
  - Пусть колонка 2 и 3 будут делициями.
- У четырех последовательностей разрывы в колонке 10. Это должна быть делиция, но мы сделаем это вставкой, чтобы иметь хотя бы одну вставку.

```
GG-GGAAAAACGTATT
TG-GGACAAAAGTATT
TG-GAACAAAAGTATG
TACGGACAAAATTATT
T--GAAGAAAAGTATG
TA-GAACAAAAGTAGG
TG-GAACAAACGCATT
CGGGACAAA-AGTATT
TGGGGTAAA-AGTATT
TGAGACAAA-AGTAGT
TGAGACAAA-AGTATA
TGGGACAAAGAGTATT
TG-AAACAAAGATATT
CG-GAACAAAAGTATT
TA-GGACAAAAGTGTT
```

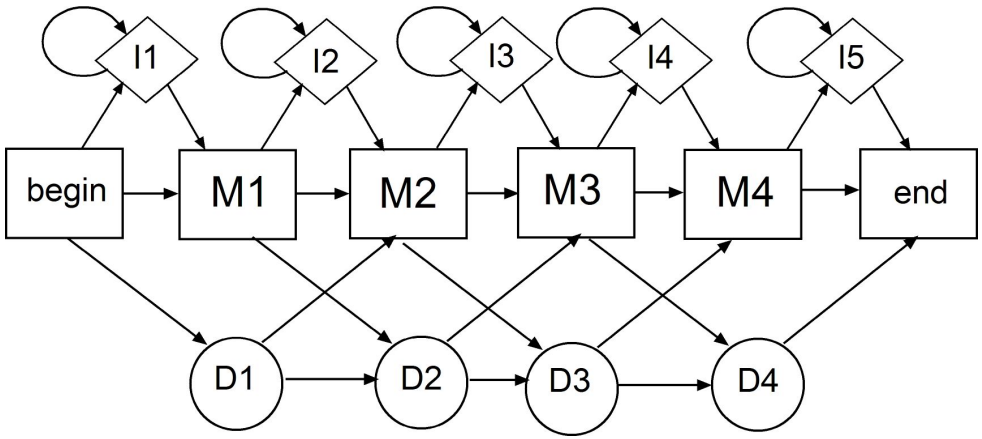
More Set Up

- Колонки 2 и 3- состояния делиции, но в других последовательностях – состояния совпадения.
- Колонка 10 – состояние вставки – основания других последовательностей испускаются из состояния вставки, поэтому для этой колонки нет состояния совпадения.
- Окончательная модель имеет 15 состояний совпадений с соответствующими состояниями вставок и делиций.
  - Большинство состояний вставок и делиций не используются в нашей последовательности, поэтому у них будут низкие вероятности. Но, тем не менее, они должны быть включены в модель.

column	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
state	M1	M2 / D1	M3 / D2	M4	M5	M6	M7	M8	M9	I9	M10	M11	M2	M13	M14	M15

Параметризация

- Какие параметры нам нужны?
- Эмиссионные:
  - В каждом состояние надо задать вероятности эмиссии для всех 4 оснований
  - Состояние вставки также нуждается в вероятностях эмиссии для всех 4 оснований.
    - Обычно берутся фоновые вероятности из всего генома или базы данных
- Переходные:
  - Для колонок 2 и 3 нам нужны вероятности перехода совпадения -> делиция match -> delete (M->D), и делиция -> делиция (D->D).
  - Для колонки 10, нам нужна вероятность M->I, и I->I (для которой у нас нет данных).
  - Нам также нужны общие вероятности M->M, M->D, and M->I для других колонок
  - Другие вероятности будут вычислены из условия, что все вероятности переходов из данного состояния должны суммироваться в 1.



Эмиссионные вероятности

- Фоновый уровень (вероятности оснований, если бы они были выбраны случайным образом)
  - Используются для состояний вставки.
  - Можно взять частоты из целого генома *B. Megaterium*. GC=38%.
    - G = C = 0.19 и A = T = 0.31.
- Специфические эмиссионные вероятности для каждого состояния совпадения

.

  - Посчитать частоты каждого основания (без пробелов) в каждой колонки
  - Но еще нужны псеводочастоты.

Эмиссионные псевдочастоты

- The simplest way to do pseudocounts is the Laplace method: adding 1 to the numerator and 4 (i.e. total types of base) to the denominator:
  - Freq(C in column 1) = (count of C's + 1) / (total number of bases + 4)
  - = (2 + 1) / (15 + 4) = 0.158
  - As compared to actual frequency = 2/15 = 0.133
  - There are no A's in column 1, so the probability of A from column 1 = 1/19 = 0.052
- A somewhat more sophisticated method is to use overall base frequencies for each base.
  - Freq(C in column 1) = (count of C's + 0.19) / (total number of bases + 1) = 2.19/16 = 0.137
  - Freq(A in column 1) = 0.31/16 = 0.019
- The base frequency method could be altered by multiplying the pseudocounts by some constant, as an estimate of our uncertainty of how likely we are to find a sequence with an A first.
  - For example, to be more equivalent to the Laplace method, multiply by 4:
  - Freq(C in column 1) = (count of C's + (4 * 0.19)) / (total number of bases + 4) = 2.76/19 = 0.145
  - Freq(A in column 1) = (4 * 0.31)/19 = 0.065
  - Note how different the probabilities are for A.
- We will just say that how to apply pseudocounts is an area of heuristics and active research.
- We will use the overall base frequency method.

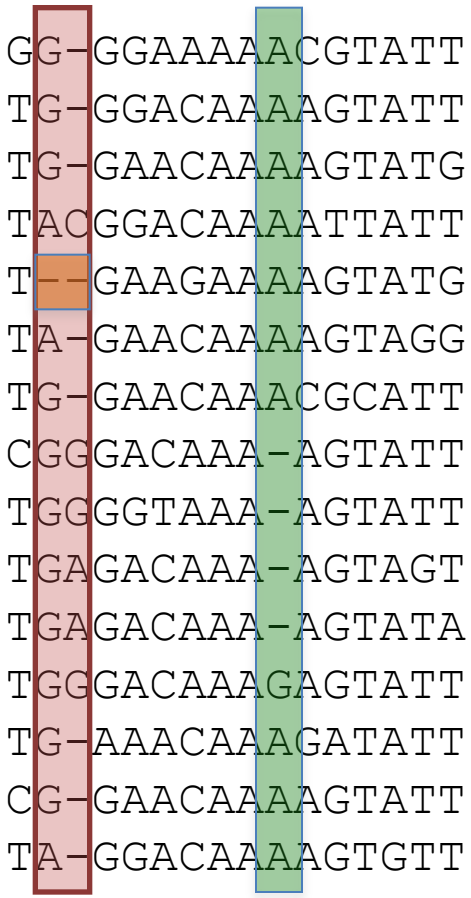
Частоты переходов

- Всего 225 переходов, и только 9 M->D.
  P(M->D) = 9/225 = 0.040.
  - Для D->D, есть 1 случай из 9 делиций, когда последовательность продложает быть делицией, поэтому P(D->D)=1/9 = 0.111. Тогда
    P(D->M) = 1 – (D->D) = 0.888

- Всего 11 M->I переходов. (колонка 10).
  P(M->I)= 11/225 = 0.044.
  - Нет случаев I->I, поэтому мы произвольно решаем сделать эту вероятность, равной D->D (0.111), поскольку мы произовльным образом решили, какие колонки трактовать как вставки, а какие как делиции.
  - P(I->M)= 0.888

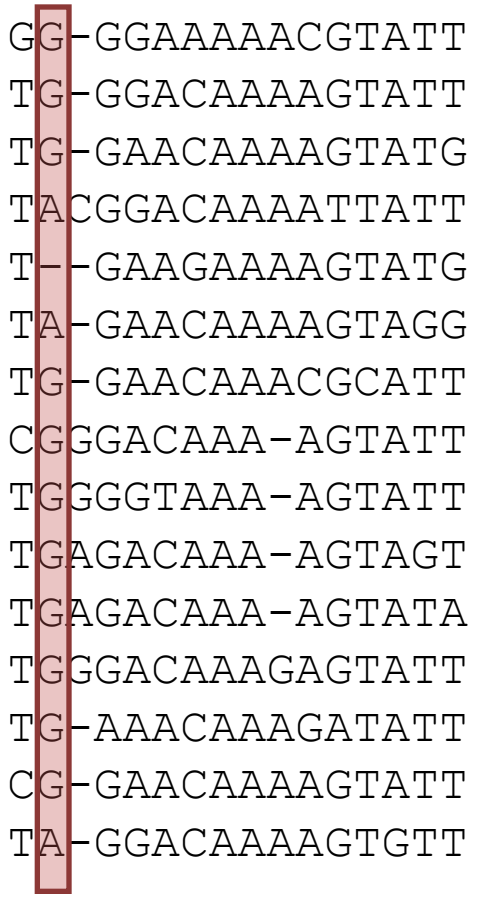
- Тогда фоновые переходы P(M->M)= 1 – (P(M->I) + P(M->D)) = 1 – (0.040 + 0.044) = 0.916.

- Нам также нужны низкие вероятности для переходов I->D и D->I, которые не должны происходить, так что мы их ставим равными 0.00001



Специфические переходы

- Колонки вставок и делиций.
- Колонка 2 содержит 1 M->D и14 M->M.
  - Need to add in pseudocounts from the overall data, so:
    P(M->D| column 2) =
    (M->D count + 0.04) / (total transitions in column 2 + 1) =
    1.04/16 = 0.065.
  --M->I in column 2 is the background level, 0.044
  - M->M for column 2 is 1 – 0.065 - 0.044 = 0.891
- Колонка 3 содержит 8 M->D и 6 M->M (еще есть D->D, но мы его посчитали).
  - Prob(M->D in column 3) = 8.04/15 = 0.536
  - Prob (M->M in column 3) = 1 – 0.536 - 0.044 = 0.420
- Колонка 10 содержит вставку M->I и 5 переходов M->M
  - Prob(M->I in column 10) = 10.044/16 = 0.628
  - Prob (M->D in column 10) = 0.04 (background)
  - Prob (M->M in column 10 is 1 – 0.628 – 0.04 = 0.332



Emission Probability Tables

match	A	C	G	T
1	0.028	0.130	0.078	0.764
2	0.229	0.005	0.750	0.015
3	0.349	0.154	0.464	0.033
4	0.090	0.005	0.890	0.014
5	0.653	0.005	0.328	0.014
6	0.653	0.255	0.015	0.077
7	0.403	0.505	0.078	0,014
8	0.965	0.005	0.015	0.014
9	0.965	0.005	0.015	0.014
10	0.778	0.130	0.078	0.014
11	0.090	0.005	0.828	0.077
12	0.028	0.067	0.015	0.889
13	0.903	0.005	0.078	0.014
14	0.028	0.005	0.140	0.827
15	0.090	0.005	0.203	0.702

	A	C	G	T
overall	0.31	0.19	0.19	0.31

Transitions

Specific		
M1□M2	0.891	
M1->D	0.065	
M2->M3	0.420	
M2->D	0.536	
M9->M10	0.332	
M9->I	0.628	

Default	
M->M	0.916
M->I	0.044
M->D	0.040
D->M	0.888
D->I	0.0001
D->D	0.111
I->M	0.888
I->I	0.111
I->D	0.0001

Scoring a Sequence

- Whew! We have now estimated parameters for all transitions and emissions.
- Scoring a sequence. We are going to use both the Viterbi algorithm and the forward algorithm to determine the most likely path through the model and the overall probability of emitting that sequence.
  - Note that we really should convert everything to logarithms
  - Also, it is standard practice to express emission probabilities as odds rations, which means dividing them by the overall base frequencies.
  - We are not going to do either of these things here, in the interest of simplification and clarity.

- Let's just score the first sequence in the list:
  - GG-GGAAAAACGTATT
  - Remove the gap, since a sequence derived from real data is not going to come with a gap (which came from a multiple alignment program)
  - GGGGAAAAACGTATT

Scoring

- GGGGAAAAACGTATT
- Base 1 is G. To start the global model off, we are going to require that this be a match state.
  - The emission probability for G in M1 is 0.078, so this is the initial overall probability and Viterbi probability.
- Base 2 is also G. There are 3 possibilities for this base: it might be a match state (M2), or it might the result of an insert state, or it might be the result of entering a delete state (and thus match a later base. We choose the most likely:
  - M1->M2 has a 0.891 probability, and the probability of emitting a G in column 2 is 0.750. So, this probability is 0.891 * 0.750 = 0.668
  - M1->D = 0.065
  - M1->I, then emitting a G from the insert state = 0.044 * 0.19 = 0.008
  - M1->M2 is most likely.
    - So, Viterbi probability = previous prob * this prob = 0.078 * 0.668 = 0.052.
    - Overall prob = 0.078 * (0.668 + 0.065 + 0.008) = 0.078 * 0.741 = 0.058

More Scoring

- Base 3 is also a G.
  - M2->M3 has 0.420 probability and 0.464 chance of emitting a G. 0.420 * 0.464 = 0.195
  - M2->D has 0.536 probability
  - M1->I, then emitting a G from the insert state = 0.044 * 0.19 = 0.008
  - Choose M2->D. Viterbi = 0.052 * 0.536 = 0.028.
  - Overall = 0.058 * (0.195 + 0.536 + 0.008) = 0.058 * 0.739 = 0.043.
- We are now in a delete state between M2 and M4; we skipped the M3 state. Since delete states are silent, the G in position 3 hasn't been emitted yet.
  - From the delete state we can either move to another delete state (skipping the M4 state in addition to M3) or we can move to M4 and emit the G.
  - D->M4 = 0.888 and M4 emitting a G = 0.890, so prob = 0.888 * 0.890 = 0.790
  - D->D = 0.111
  - Move to M4. Viterbi = 0.028 * 0.790 = 0.022.
  - Overall = 0.043 * (0.790 + 0.111) = 0.043 * 0.901 = 0.039.
- We can now move on to base 4 (another G)
- Our path so far: M1->M2->D->M4. We have emitted the first 3 bases.
- GGGGAAAAACGTATT

Still More Scoring

- GGG GAAAA ACGTATT
- The next several bases are easy. Since the probability of moving to a delete or insert state is low, we just have to be sure that the M->M probability times the emission probability stays above 0.044.
- M4->M5 : G prob = 0.916 * 0.328 = 0.300
  - Viterbi prob = 0.022 * 0.300 = 0.0066
  - Overall prob = 0.039 * (0.300 + 0.040 + (0.044 * 0.19)) = 0.039 * 0.3484 = 0.0136
- M5->M6 : A prob = 0.916 * 0.653 = 0.598
  - Viterbi prob = 0.0066 * 0.598 = 0.00395
  - Overall prob = 0.0136 * (0.598 + 0.040 + (0.044 * 0.31)) = 0.0136 * 0.6516 = 0.0089
- M6->M7 : A prob = 0.916 * 0.403 = 0.369
  - Viterbi prob = 0.00395 * 0.369 = 0.00146
  - Overall prob = 0.0089 * (0.369 + 0.040 + (0.044 * 0.31)) = 0.0089 * 0.423 = 0.00376
- M7->M8 : A prob = 0.916 * 0.965 = 0.884
  - Viterbi prob = 0.00146 * 0.884 = 0.00129
  - Overall prob = 0.00376 * (0.884 + 0.040 + (0.044 * 0.31)) = 0.00376 * 0.938 = 0.00353
- M8->M9 : A prob = 0.916 * 0.965 = 0.884
  - Viterbi prob = 0.00129 * 0.884 = 0.00114
  - Overall prob = 0.00353 * (0.884 + 0.040 + (0.044 * 0.31)) = 0.00353 * 0.938 = 0.00331

Yet More

- At this point we have emitted positions 1- 8, and the most probable path is M1->M2->D->M4->M5->M6->M7->M8->M9
- GGG GAAAA ACGTATT
- Since the transition out of M9 is not the standard one, we need to pause and think it through.
  - M9->M10 = 0.332. Emission prob for A from M10 is 0.778. 0.332 * 0.778 = 0.258
  - M9->I = 0.628. Emission prob for A from an insert state (i.e. background probability) is 0.31 0.628 * 0.31 = 0.195.
  - Thus our best choice, the most probable path, is M9->M10. However, looking at the aligned sequences we can see that this is the **wrong** choice.
    - Don't despair: correction occurs in the next step.
  - Viterbi prob = 0.00114 * 0.258 = 0.000294
  - Overall prob = 0.00331 * (0.258 + 0.195 + 0.040) = 0.00331 * 0.493 = 0.00163

```
GG-GGAAAAACGTAT
T
TG-GGACAAAAGTAT
T
TG-GAACAAAAGTAT
G
TACGGACAAAATTAT
T
T--GAAGAAAAGTAT
G
TA-GAACAAAAGTAG
G
TG-GAACAAACGCAT
T
CGGGACAAA-AGTAT
T
TGGGGTAAA-AGTAT
T
TGAGACAAA-AGTAG
T
TGAGACAAA-AGTAT
A
TGGGACAAAGAGTAT
T
```

Yet Still More

- At this point we have emitted positions 1- 8, and the most probable path is M1->M2->D->M4->M5->M6->M7->M8->M9->M10
- GGG GAAAAA CGTATT
- At M10, we can:
  - move to M11 and emit a C. Prob = 0.916 * 0.005 = 0.0046
  - Move to an insert state and emit a C. Prob = 0.044 * 0.19 = 0.0083.
  - Move to a delete state. Prob = 0.04. This would be the best choice, but it leads to a mess: delete all the remaining match states, then inserting all the remaining bases in the query sequence at the end. It clearly shows the need for dynamic programming.
    - And while we are at it, switching to logarithms at the beginning would greater ease calculations.
  - So, to continue our example, we move from M10 to an insert state and emit a C.
    - Viterbi prob = 0.000294 * 0.0083 = 2.44 x 10^{-6}
    - Overall prob = 0.00163 * (0.0046 + 0.0083) = 2.10 x 10^{-5}

To the End…

- Our path so far:
  - M1->M2->D->M4->M5->M6->M7->M8->M9->M10->I
  - GGG GAAAAAC GTATT
- From the insert state we can:
  - I->I and emit a G, with probability 0.111 * 0.19 = 0.0211
  - I->M11, with prob 0.888 * 0.828 = 0.735
    - Viterbi prob = $2.44 \times 10^{-6} * 0.735 = 1.79 \times 10^{-6}$
    - Overall prob = $2.10 \times 10^{-5} * (0.0211 + 0.735) = 1.58 \times 10^{-5}$
- The remaining steps are all match states, so we skip the calculations:
  - Final Viterbi probability = 4.46×10^{-7}
  - Final overall prob = 6.79×10^{-6}

Final probability

- We need to know what the probability would be for the random model, with every base inserted according to its overall frequency in the genome.
- GGGGAAAAACGTATT has 6 G/C and 9 A/T, so the random probability is:

  $(0.19)^6 * (0.31)^9 = 1.24 \times 10^{-9}$

- We compare to the overall probability of 6.79×10^{-6} by dividing, giving 5459. This means that the overall score for this sequence is 5459 times more likely than chance to match the model.

Profile Hidden Markov Models

- Вычисление веса последовательности по профильным HMM
  - Имея профильную HMM, любой путь по модели «испускает» последовательность с некоторой вероятностью.

Вероятность пути – это произведение всех вероятностей переходов и испускания данных вдоль пути.

Profile Hidden Markov Models

- Вычисление веса последовательности по профильным HMM
- **Алгоритм Витерби**:
  - Имея исходную последовательность, мы можем посчитать **наиболее вероятный путь**, который сгенерирует («испустит») эту последовательность.

Profile Hidden Markov Models

- Вычисление веса последовательности по профильным HMM
  - **Алгоритм прогона вперед**:
    - Другой интересный вопрос: Какова вероятность, что данная последовательность **могла быть сгенерирована этой скрытой Марковской моделью?**
    - Решение: Можно посчитать, **суммируя по всем возможным путям**, которые сгенерировали данную последовательность