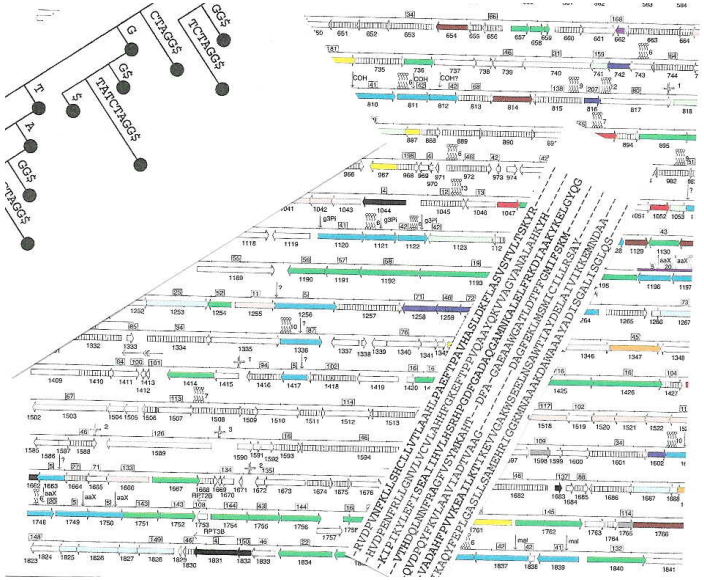
Lecture 6a: Introduction to Hidden Markov Models

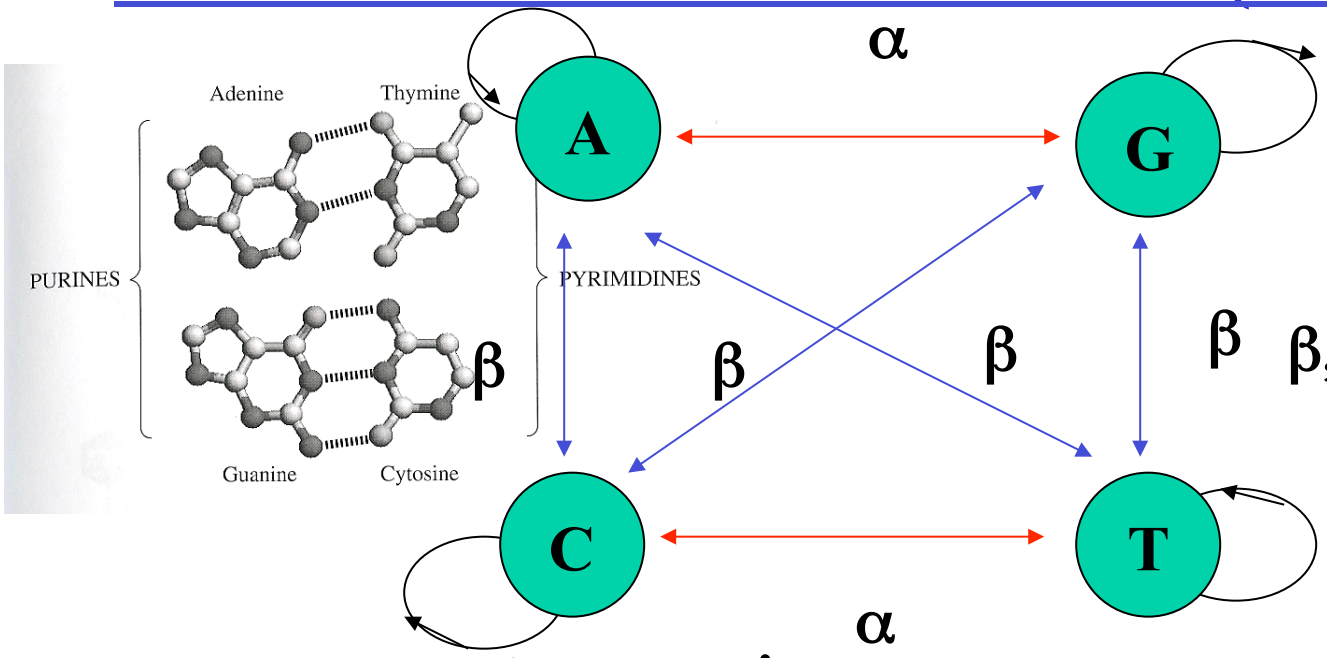
Introduction to Computational Biology

Instructor: Teresa Przytycka, PhD

Igor Rogozin PhD



First order Markov model (informal)



β,α –probability of given mutation in a unit of time

A random walk in this graph will generates a path;
say AATTCA….
For each such path we can compute the probability of the path
In this graph every path is possible (with different probability) but in general this does need to be true.

First order Markov model (formal)

Markov model is represented by a graph with set of vertices corresponding to the set of states Q and probability of going from state i to state j in a random walk described by matrix *a*:

a – n x n transition probability matrix

$$a(i,j) = P[q_{t+1}=j | q_t=i]$$

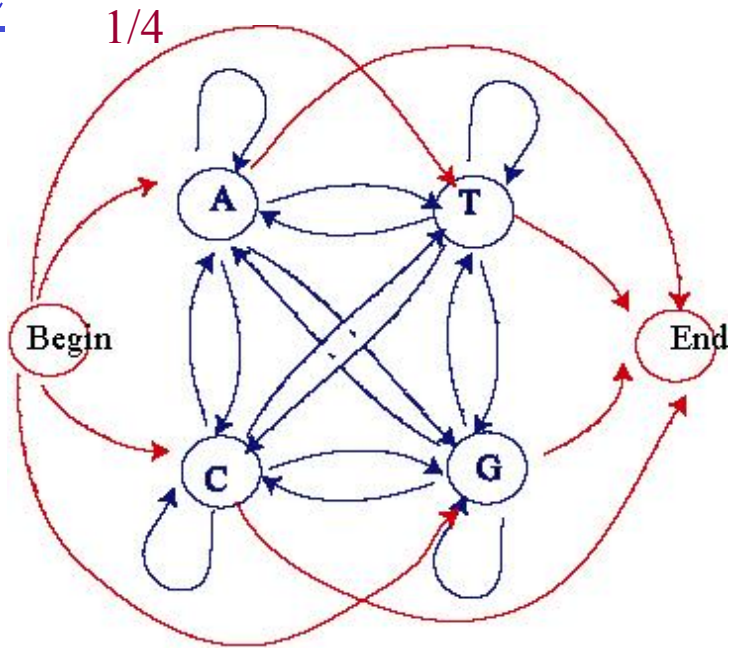
where q_t denotes state at time t

Thus Markov model M is described by Q and a

$$M = (Q, a)$$

Example

1/4



Transition probabilities for general DNA seq.

Set of states Q:

{Begin, End, A,T,C,G}

Transition probability matrix a:

a

–	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

Probability of a sequence of states $S=q_0,\ldots,q_m$ in model M

Assume a random walk that starts from state q_0 and goes for m steps. What is the probability that $S= q_0,\ldots,q_m$ is the sequence of states visited?

$$P(S|M) = a(q_0,q_1)\ a(q_1\ ,\ q_2)\ a(q_2\ ,\ q_3)\ \ldots\ a(q_{m-1}\ ,\ q_m)$$

P(S|M) = probability of visiting sequence of states S assuming Markov model M defined by a:

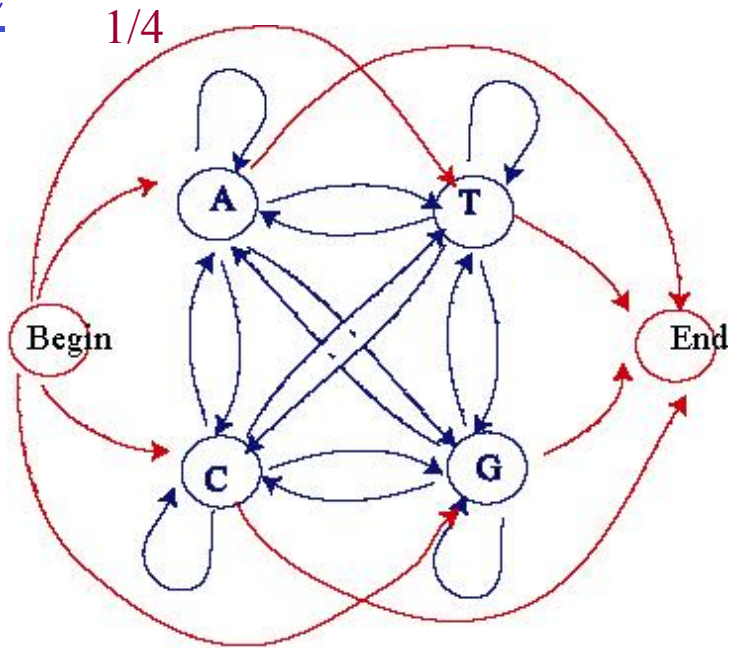
n x n transition probability matrix $a(i,j) = Pr[q_{t+1}=j|q_t=i]$

Example

Transition probabilities for general DNA seq. (human)

a1

−	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



Transition probabilities for CpG island

a2

+	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

(Assume that any state can be the end state)

Probability of sequence Begin AACGC:

in model M1: P(S|M1) =

¼ x0.3x 0.205 x0.078 x 0.246

in model M2: P(S|M2)

¼ x 0.18 x 0.274 x 0.274 x 0.339

Choosing Markov model that is most likely to generate given sequence

In is not simply which of the two P(S|M1) and P(S|M2) is bigger (unless we make additional assumption)

P(Model|Sequence) =
P(Sequence| Model)P(Model) / P(Sequence)

assuming probability of sequence is the same we need to compare:

P(S|M1)P(M1) < P(S|M2)P(M2)

If often assume that both models are equally likely and then it boils down to comparing

P(S|M1) < P(S|M2)

But we may know that in a given genome CpG island are less frequent than non CpG seqences and we can use this prior knowledge to define P(M1) and P(M2)

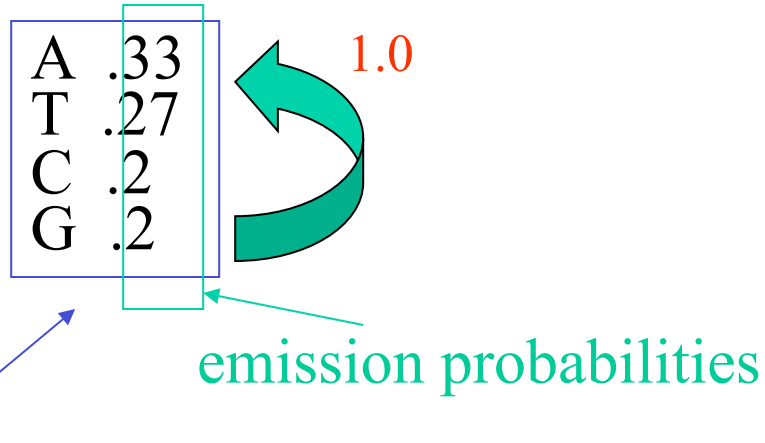
Introducing HMM

- Assume that we not only perform random walk on the graph but also in each state print out (emit) one of a finite number of symbols

Introducing emission probabilities

- Assume that at each state a Markov process emits (with some probability distribution) a symbol from alphabet Σ.
- Hidden Markov Model: Rather than observing a sequence of states we observe a sequence of emitted symbols.

Example:
Σ ={A,C,T,G}.
Generate a sequence where A,C,T,G have frequency p(A) =.33, p(G)=.2, p(C)=.2, p(T) = .27 respectively



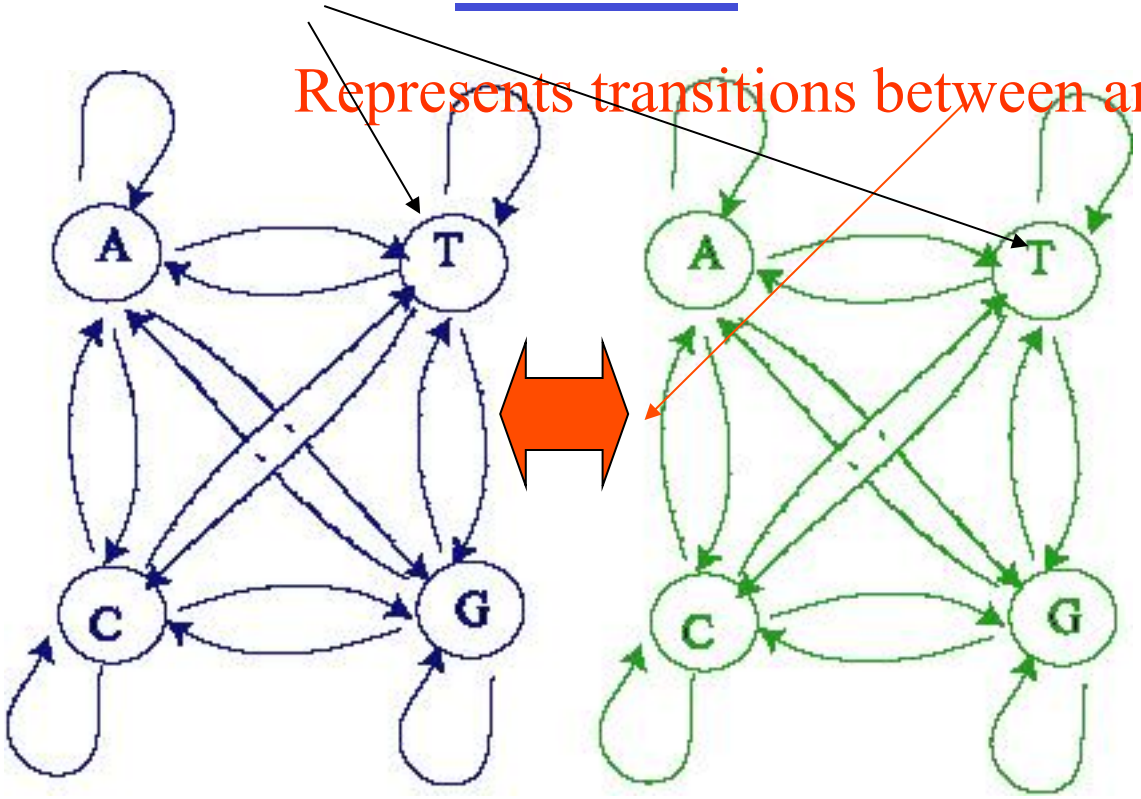
First, back to Markov

- A Markov model of DNA sequence where regular sequence is interrupted with CpG islands

generating DNA sequence with CpG islands

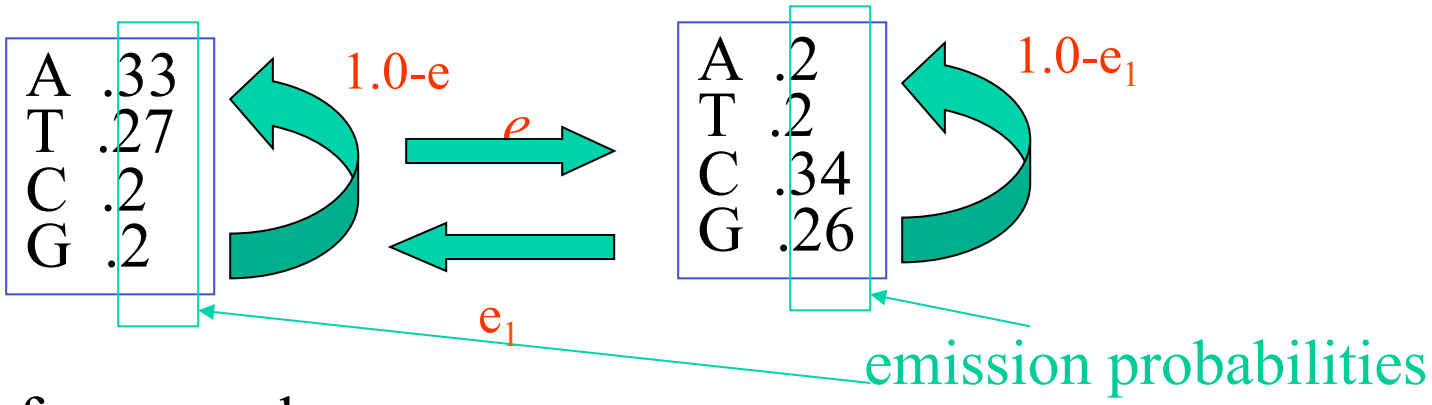
Distinguishable states

Represents transitions between any pair of states



We do not have a 1-1 mapping between states and generated letters thus if we observe a sequence, say GCGCAGC generated by such model we cannot be sure which state generated which letter.

The same with HHM: Example CpG island-continued



Model for general sequence

Model for CpG island

(the numbers are for illustration only)

What is different comparing Markov model?

- Note that if there each letter can be generated in a two different states.
- Thus is we have a sequence of letters we cannot uniquely identify the path used in the generation - the path of states is “hidden”
- Rather than observing sequence of states we observe a sequence of emitted letters.
- We move it even further by giving each sate a choice for emitting one of several possible letters.

HMM – formal definition

HMM is a Markov process that at each time step generates a symbol from some alphabet, Σ, according to emission probability that depends on state.

$M = (Q, \Sigma, a, e)$

Q – finite set of states, say n states $=\{1,\ldots n\}$

a – n x n transition probability

matrix $a(i,j) = Pr[q_{t+1}=j | q_t=i]$

q_t = *state in position t in the sequence (t^{th} time step)*

$\Sigma = \{\sigma_1, \ldots, \sigma_k\}$

$e(i,j)$ == probability of generating symbol σ_j in state q_i =

$P[a_t=\sigma_j | q_t = i]$; where a_t is t^{th} element of generated sequence

Typical applications of HMM

Given is a family of bio-sequences. Assume that it was generated using a HMM.

Goal: construct HMM that models these sequences.

Markov Chain/Hidden Markov Model

Both are based on the idea of random walk in a directed graph, where probability of next step is defined by edge weight.

In HMM additionally, at step a symbol from some fixed alphabet is emitted.

Markov Chain – the result of the experiment (what you observe) is a sequence of state visited.

HMM – the result of the experiment is the sequence of symbols emitted. The states are hidden from the observer.

Example from Jones/Pevzner book

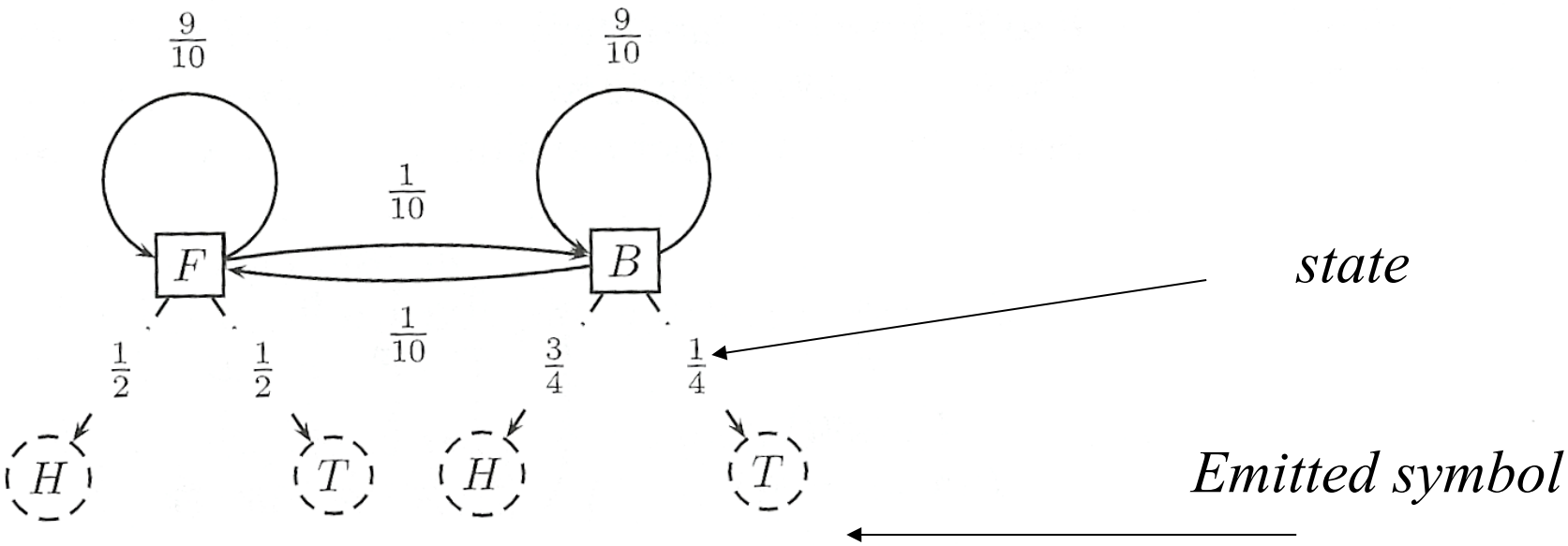


Figure 11.1 The HMM designed for the Fair Bet Casino problem. There are two states: F (fair) and B (biased). From each state, the HMM can emit either heads (H) or tails (T), with the probabilities shown. The HMM will switch between F and B with probability 1/10.

HTHHTHTTTHHHTHHHHTHHHTHH

You may guess that somewhere here the cassino switched to biased coin

HMM – formal definition

HMM is a Markov process that at each time step generates a symbol from some alphabet, Σ, according to emission probability that depends on state.

$M = (Q, \Sigma, a, e)$

Q – finite set of states, say n states $=\{1,\ldots n\}$

a – n x n transition probability

matrix $a(i,j) = Pr[q_{t+1}=j | q_t=i]$

q_t = *state in position t in the sequence (t^{th} time step)*

$\Sigma = \{\sigma_1, \ldots, \sigma_k\}$

$e(i,j)$ == probability of generating symbol σ_j in state q_i =

$P[a_t=\sigma_j | q_t = i]$; where a_t is t^{th} element of generated sequence

Recall from Markov chain: Probability of a sequence of states $S=q_0,\ldots,q_m$ in Markov model M

Assume a random walk that starts from state q_0 and goes for m steps. What is the probability that $S= q_0,\ldots,q_m$ is the sequence of states visited?

$$P(S|M) = a(q_0,q_1)\, a(q_1, q_2)\, a(q_2, q_3) \ldots a(q_{m-1}, q_m)$$

P(S|M) = probability of visiting sequence of states S assuming Markov model M defined by a:

n x n transition probability matrix $a(i,j) = Pr[q_{t+1}=j|q_t=i]$

In HMM: P(S|M) = probability of generating a sequence of symbols

$$P[S|M] = \sum_{\pi \in Q^{n+1}} P[\pi|M]P[S|\pi,M]$$

(sum over all possible paths $p=q_0,\ldots q_n$ of product: probability of the path in the model times the probability of generating given sequence assuming given path in the model.)

$$P[\pi|M] = a(q_0,q_1)\, a(q_1,q_2)\, a(q_2,q_3) \ldots a(q_{n-1},q_n)$$

$$P[S|\pi,M] = e(q_0,\sigma_1)\, e(q_1,\sigma_2)\, e(q_2,\sigma_3) \ldots e(q_n,\sigma_n)$$

Computing P[S|M] from the definition (enumerating all sequences p) would take exponential time. We will do it by dynamic programming.

π– sequence of states

$S = \sigma_0, \ldots, \sigma_n$

$\pi = q_0, \ldots, q_n$

$P(\sigma_i | q_i)$

$P(q_{i-1} | q_i)$

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ F & F & F & B & B & B & B & B & F & F & F \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} & \frac{3}{4} & \frac{3}{4} & \frac{3}{4} & \frac{1}{4} & \frac{3}{4} & \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{9}{10} & \frac{9}{10} & \frac{1}{10} & \frac{9}{10} & \frac{9}{10} & \frac{9}{10} & \frac{9}{10} & \frac{1}{10} & \frac{9}{10} & \frac{9}{10} \end{pmatrix}$$

$P[S| \pi,M] = ½ * ½ * ½ * ¾ * ¾ * \ldots * ½$

$P[\pi|M] = ½ * 9/10 * 9/10 * 1/10 * \ldots$

$P[S|M] = ½ * ½ * 9/10 * ½ * 9/10 * \ldots$

Typical context for using HMM in computational biology

- Select a basic method to model a sequence family – what are the states, what are the emitted symbols.
- Using a training set – a set of known sequences in the family – estimate the parameters of the model: emission and transition probabilities.

Algorithmic problems

- Given a sequence, and several HMMs we would like to decide which of HMM models in most likely to generate it
- Given a HMM model M and a sequence S we would like to be able find the most likely path in M that generates S (example – predicting CpG island).

Finding the most likely path

- Most likely path for a sequence S is a path p in M that maximizes the probability of generating the sequence S along this path: $P(S|\pi)$.
- Why it is often interesting to know most likely path:
  - Frequently HMM are designed in such a way that one can assign biological relevance to a state: e.g. position in protein sequence, beginning of coding region, beginning/ end of an intron.

Recall: P(S|M) = probability of generating a sequence S by HMM M

$M = (Q, \Sigma, a, e)$

$S = \sigma_0 \ldots \sigma_n$

$$P[S|M] = \sum_{\pi \in Q^{n+1}} P[\pi|M] P[S|\pi, M]$$

(sum over all possible paths $\pi = q_0, \ldots q_n$ of product: probability of the path in the model times the probability of generating given sequence assuming given path in the model.)

$$P[\pi|M] = a(q_0, q_1)\, a(q_1, q_2)\, a(q_2, q_3) \ldots a(q_{n-1}, q_n)$$

$$P[S|\pi, M] = e(q_0, \sigma_1)\, e(q_1, \sigma_2)\, e(q_2, \sigma_3) \ldots e(q_n, \sigma_n)$$

Computing P[S|M] from the definition (enumerating all sequences p) would take exponential time. We will do it by dynamic programming.

Finding most likely path for sequence S in model M :Viterbi algorithm

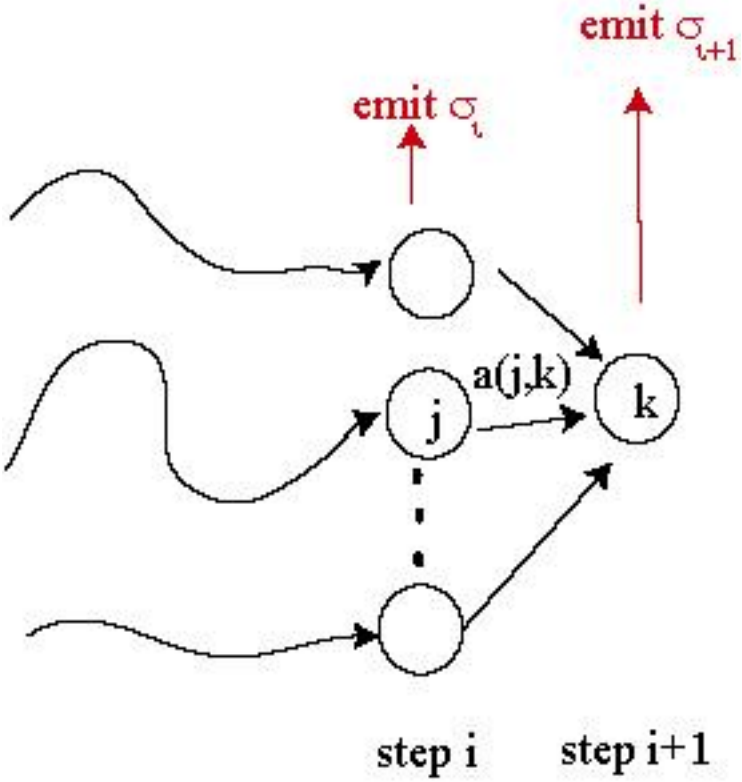
Idea: Use dynamic programming to compute for every state k and every position i in the sequence the value:

Let $v_j(i)$ = the probability of generating the prefix S_i using the most likely path subject to the restriction that the path ends at state j at step i

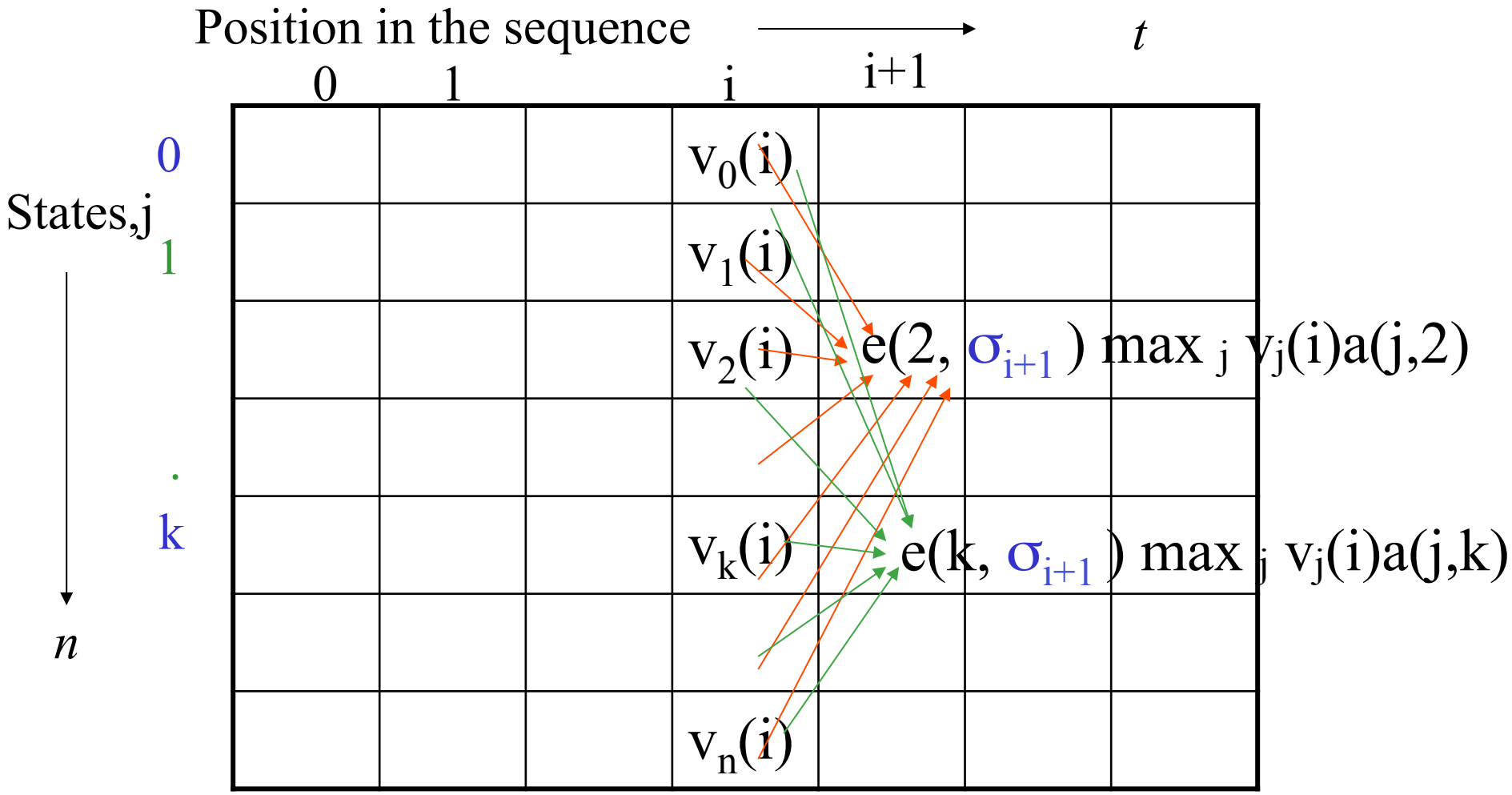
Let $S = \sigma_0 \ldots \sigma_n$ and let

σ_{i+1} = s = symbol emitted in time i+1

$v_k(i+1) = e(k,s) \max_j v_j(i) a(j,k)$



Dynamic programming table



Cost = $O(n^2 t)$

The algorithm for most likely path generating sequence $S = \sigma_0 \ldots \sigma_t$

- Initialization: $v_0(0)=1$; $v_k(0)=0$; for $k>0$;
- Recursion:

  for i= 1 to t

  for all states k

  $v_k(i)= e(k,\sigma_i) \max_j v_j(i-1)a(j,k)$

  trace-back for path determination:

  trace(k,i) = record the source of the arrow that that gives $\max_j v_j(i-1)a(j,k)$

  P*(t) =trace(stop,t); *P(t) is predecessor of state t*
- Trace-back:

  for i = t to 1 do P*(i-1) = trace(P*(i),i)

So far:

- Now we can compute quickly the most likely path (and the probability of this path).
- But a given sequence can be generated in many different ways (using other paths) except that with a smaller probability
- If we are interested in computing the probability of generating a sequence my a HMM must account for all possibilities

Computing probability of the sequence in HMM: forward algorithm

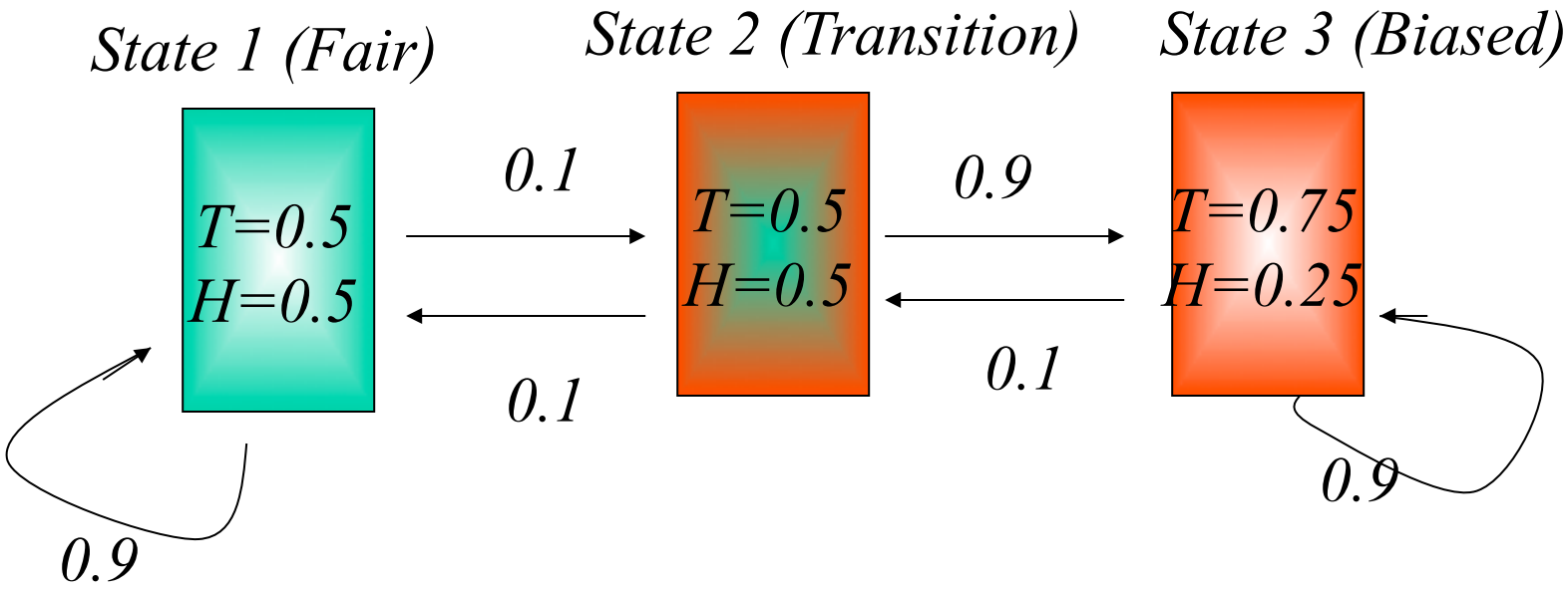
$f_k(i)$ - probability of generating the subsequence 1,…i using a path that ends at state k (forward variable)

$$f_k(i+1)=e(k,\sigma_i)\Sigma_j f_j(i)a(j,k)$$

f_{n+1}(stop)- probability of generating the sequence

Another important measurement: probability of the fact that that a symbol at position *i* in the sequence was generated in state *k*

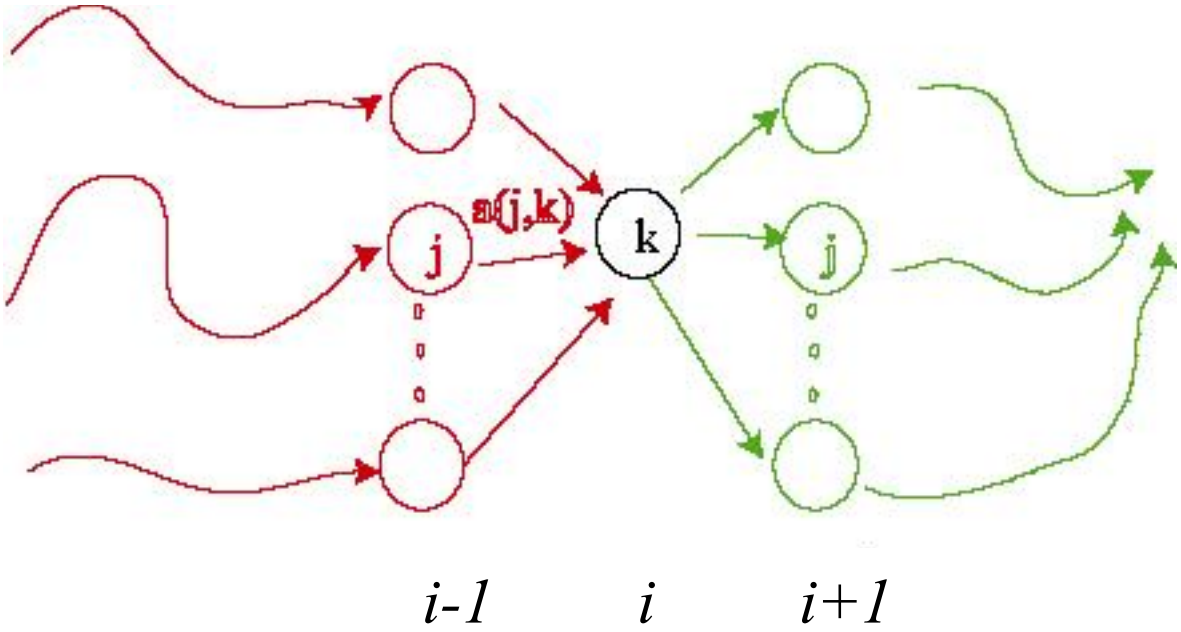
Example, state 2 is designed to be a "decision point"



Probability that observation σ_i came from state k

$S=\sigma_1,\ldots\ \sigma_i\ldots\sigma_t$ - sequence

Compute : $P(\sigma_i = k|S)$ the posterior probability



Method:

$$P(S|\sigma_i = k) =$$

$$P(\sigma_1,\ldots\ \sigma_{i-1}|\sigma_i =k)$$

x

$$P(\sigma_{i+1},\ldots\sigma_t\ |\sigma_i =k)$$

The backward algorithm

$P(\sigma_1,\ldots\sigma_i = k)$ is computed as "forward variable" in in the previous algorithm

$P(\sigma_{i+1},\ldots\sigma_t \mid \sigma_i = k)$ run the algorithm "backward" that is assume that stop state is the start state.

This is almost exactly the same as the forward with one caveat: a slight modification is required not to count the emission probability at state k twice (going from both directions)

Designing HMM

- Model structure design (deciding on number of states, connections between the states) – no theory is developed.
- Parameters estimation – given topology of the model assign transition and emission probabilities
  - Gather statistics and compute the model in an explicit way (what we have done so far).
  - Have a computer algorithm that given training set as the input constricts best fitting model well developed methods.

Parameter Estimation from a training set

- Problem: Given a training set $S_1, ..S_n$ and a topology of HMM find emission and transition probabilities M that maximize the likelihood that $S_1, ..S_n$ are generated by the model.
- Assumption: $S_1, ..S_n$ are independent(!):

$$P(S_1, ..S_n|M) = \Pi_i P(S_i|M)$$

- As usual, substituting likelihood with score:

$$\text{Score}(S_1, ..S_n|M) = \log P(S_1, ..S_n|M)$$
$$= \Sigma_i \log P(S_i|M)$$

Two variants of parameter estimation method

1. For each sequence in the training set we know the path in HMM used to generate this sequence.
2. Paths is unknown.

Estimation when the path is known

- Examples: features of DNA sequence intrins/ exons etc – we may know the path from experimental data
- Estimation method:

  A_{kl} , E_{kb} # of times given transitions/ emission is chosen *(we know them since we know the paths!)*

  Set:

  $$a(k,l) = (A_{kl})/(\Sigma_{l'} A_{kl'}) \; ; \; e(k,b) = (E_{kb})/(\Sigma_{b'} E_{kb'})$$

Adding pseudocounters (Bayesian approach)

- Potential problem – insufficient data; what if some state is never used in training?
- Extend the statistics by adding "pseudocounters" r_{kl}, r'_{kb} :

  A_{kl} # of times given transitions + r_{kl};

  E_{kb} # of times given emission is chosen+ r'_{kb}
- Choosing pseudocounters:
  - no prior knowledge – small total values Σr_{kl} $\Sigma r'_{kl}$
  - large total values – reflection of prior knowledge.

Parameter estimation when paths are unknown: Baum-Welsh method

1. Start with some initial parameters (say uniform probability distribution)
2. For a given training set compute expected number of times, A_{kl}, E_{kl} , each transition/emission is used.
3. Estimate parameters of the model from A_{kl}, E_{kl} *(use these values in the same way as the corresponding values for the algorithm when the paths are known)*
4. Repeat steps 2 and 3 until a convergence criterion is reached.

QUESTIONS:

1. How to compute A_{kl}, E_{kl} ?
2. Does the parameter estimation converges ?
3. What it converges to?

Calculating A_{kl}, E_{kl} in Baum Welch (training

Dynamic programming algorithm using method similar to the algorithm for computing probability most likely state that generated i^{th} symbol.

convergence

- B-W training is an example example of EM (Expectation maximization) method
- Each iteration is an improvement (for expectation of generating a given training set) (unless maximum is reached). *(In next model sequences are generated with higher probability than on the model form previous iteration)*
- Converges to local maximum *(but not necessarily in a finite number of steps)*
- Sample termination criterions :
  - The improvement is small
  - Maximum number of iterations is reached
- Problem with the method: once transition probability is set to zero it will remain to be so in subsequent iterations. Slow.

Viterbi learning

- Replace the calculation involving all possible paths with calculation using most likely path.
- Faster algorithm – no backward computation is needed.
- Viterbi path is interpretable in terms of the sequence features.
- Meta-MEME (Motif-based Hidden Markov Models of Protein, San Diego) trains using the Viterbi algorithm

Profile HMM

- Goal: to model sequence similarities
- Idea – extending the idea of sequence profile
- Simplified goal: Assume that the sequences are aligned without gaps (e.g.. Blocks).
- Method: Have each state corresponding to one position of the alignments.Each state emits amino-acids with probability equal to the frequency of an amino-acid at this position.

Building Profile HMM
Model 1: No gaps-same sequence length

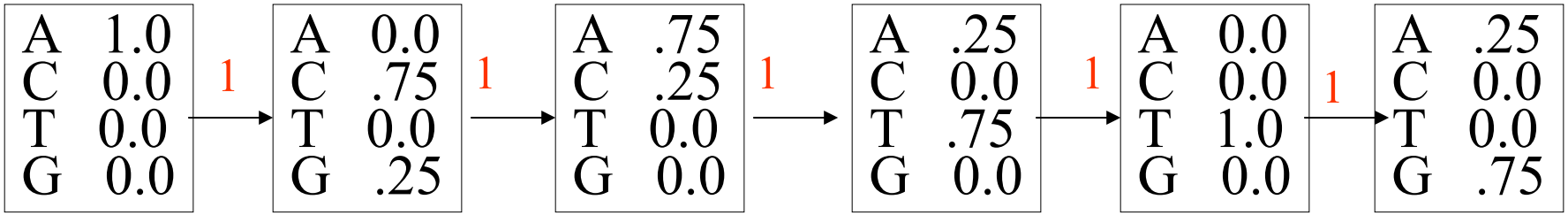
aligned sequences :

ACATTC
ACCTTC
ACATTC
AGAATA

Profile

Position/ letter	1	2	3	4	5	6
A	1.0	0.0	.75	0.25	0.0	0.25
C	0.0	0.75	0.25	0.0	0.0	0.0
T	0.0	0.0	0.0	0.75	1.0	0.0
G	0.0	.25	0.0	0.0	0.0	.75

HMM:

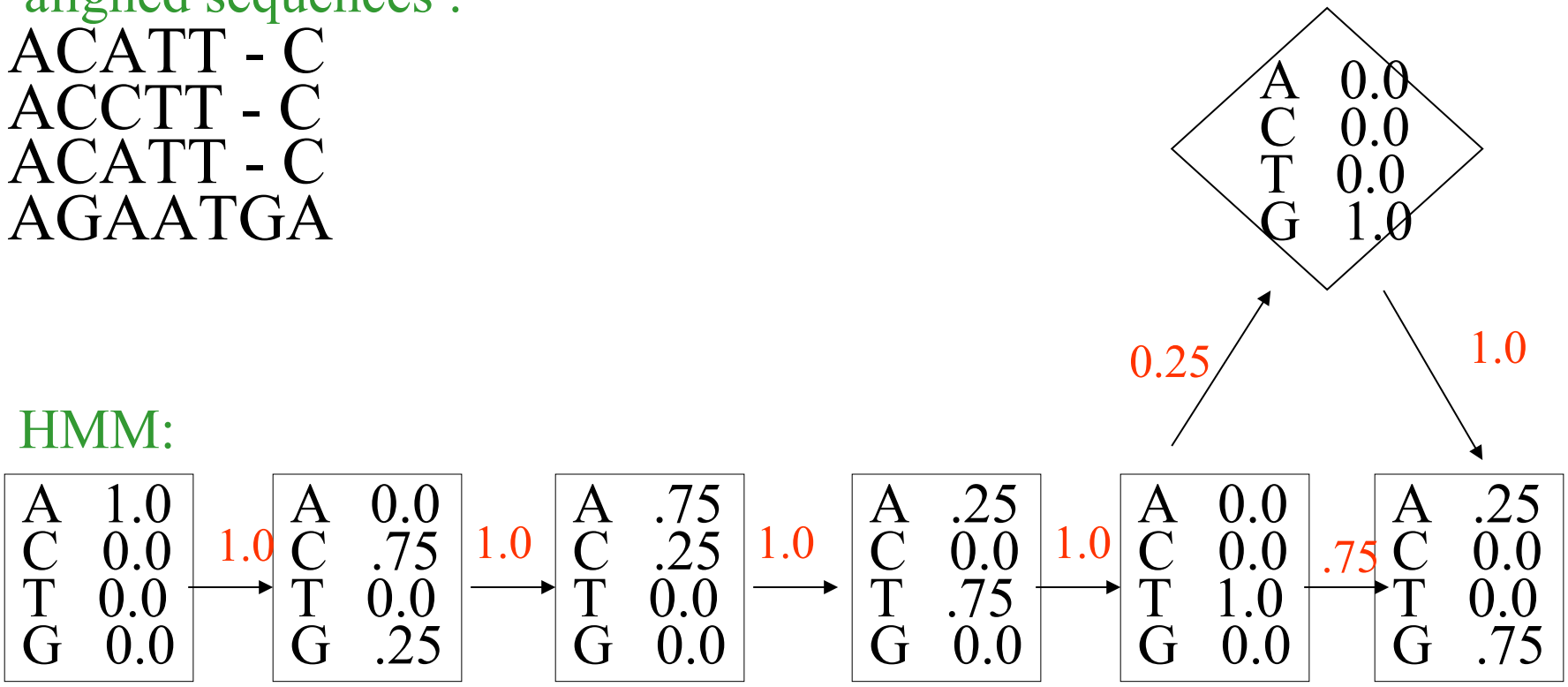


6 sates, linear topology, transition probability form i to i+1 equal 1
Emission probability defined by profile

Building Profile HMM
Model 2: Insertions of length 1

aligned sequences :

ACATT - C
ACCTT - C
ACATT - C
AGAATGA



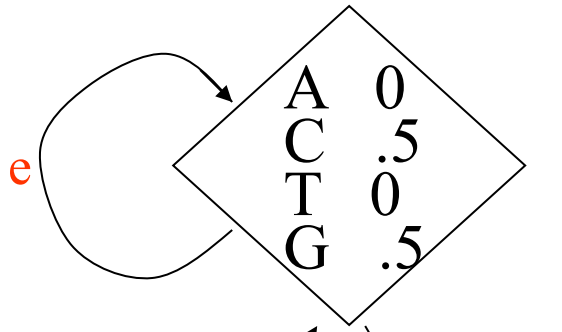
"Insert" state added

Building Profile HMM
Model 3: Insertions of variable length gap

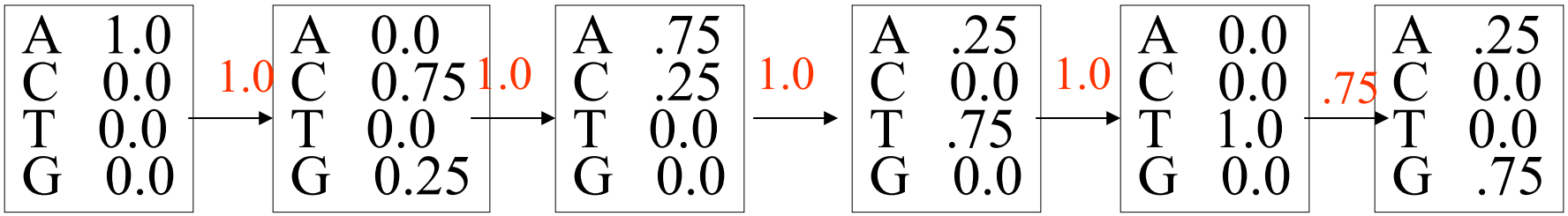
aligned sequences :

ACATT - - - - C
ACCAT - - - - C
ACAAT - - - - C
AGAATGCGCA

e – gap extension probability: 3/12 since there are 12 possibilities 3 of them extensions



HMM:

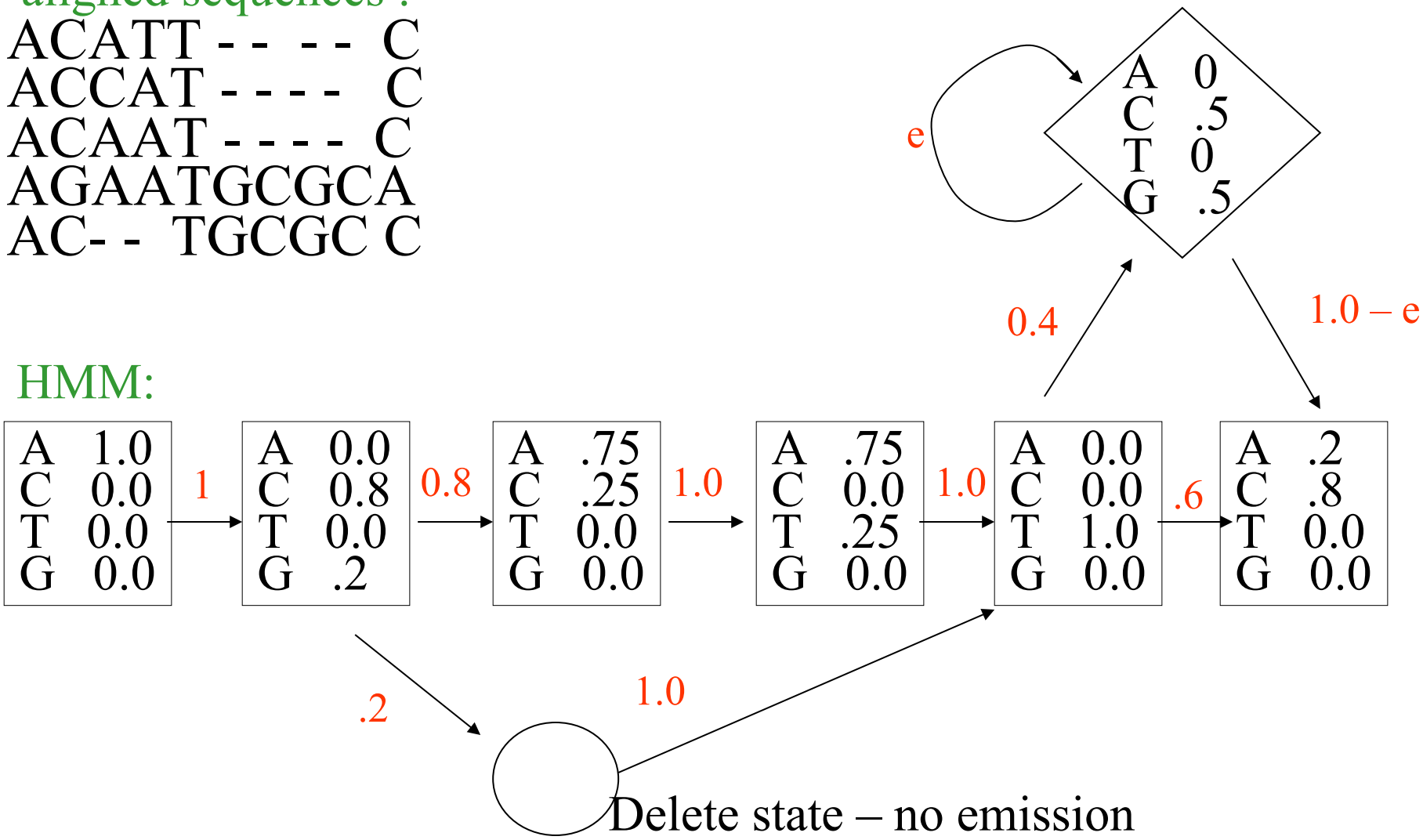


Problems: The insert model is to weak to capture GCGC pattern in the insert and the extension probability does not depend on gap length.

Building Profile HMM
Model 4: Constant length deletion

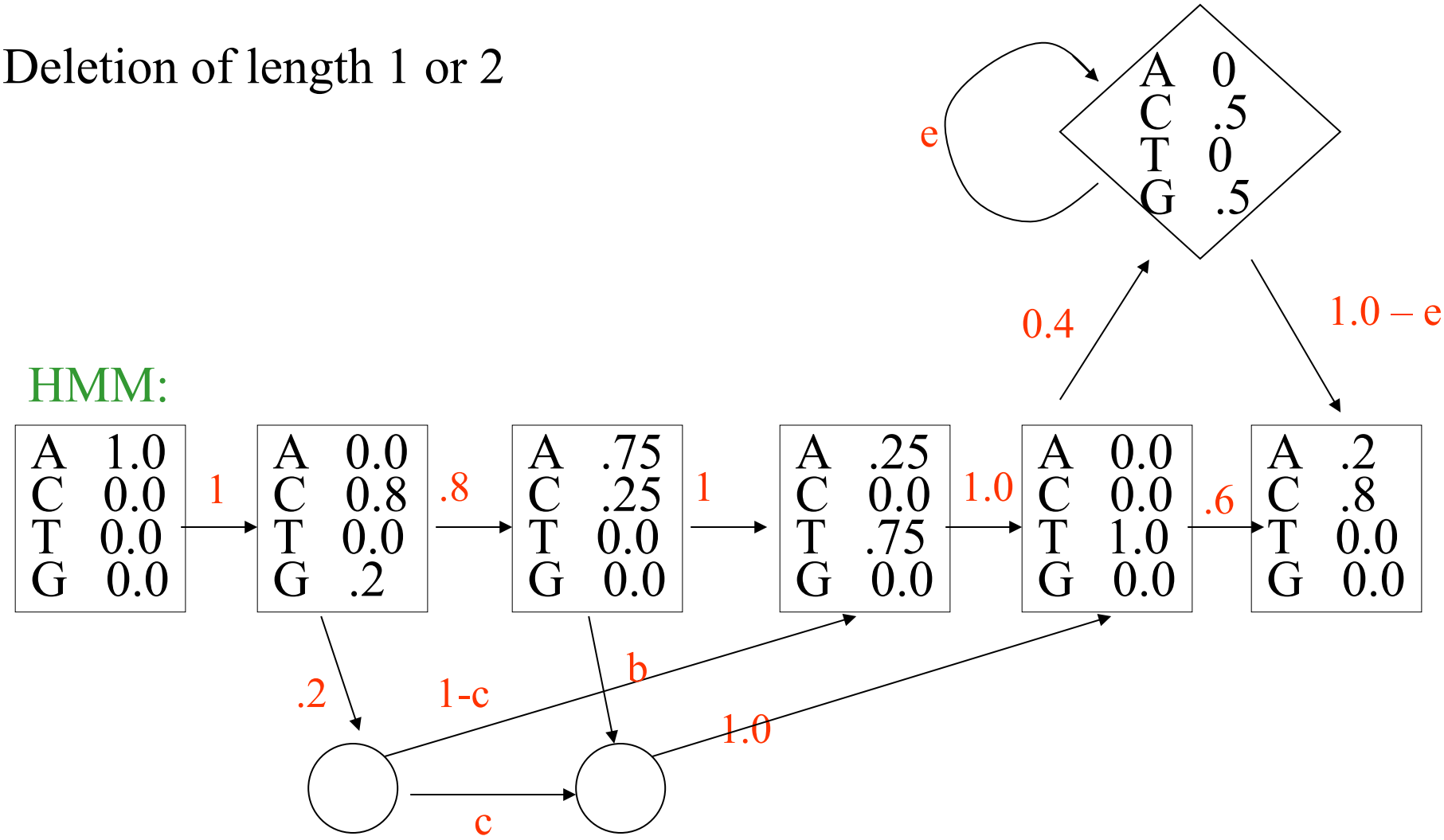
aligned sequences :

ACATT - - - - C
ACCAT - - - - C
ACAAT - - - - C
AGAATGCGCA
AC- - TGCGC C

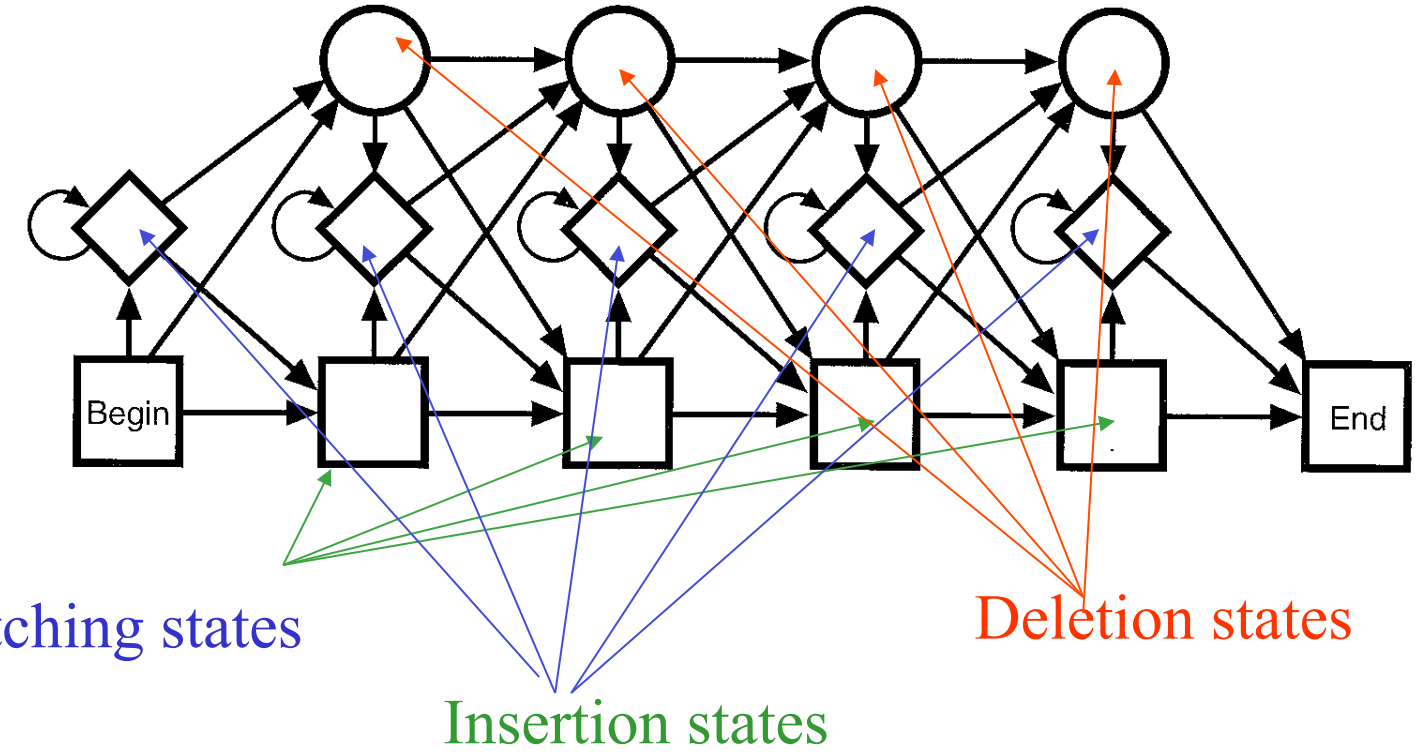


Building Profile HMM
Model 5: Variable length deletion

Deletion of length 1 or 2



Building Profile HMM:Complete Model



Parameters:

number of matching states = average sequence length in the family

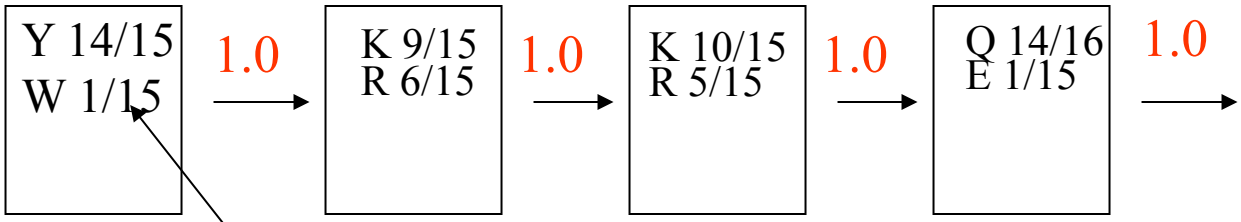
emission and transition probabilities – estimated form training set

Example

Recall example:

Subcluster of block BL00094F

YKQAGNSVVV
YKQLGNSVTV
YRQFGNSVAV
YKQAGNSITV
YKQAGNSITV
YKQTGNSITV
YRQFGNSVSV
WRQFGNSVPV
YRQFGNSVVV
YKQFGNSVVI
YKQFGNSVAV
YRQMGNSVVV
YRQFGNSVCV
YKQFGNSVAV



With pseudcounters set to 1

$(14 + 1) / (14 + 20) = 15/34$

Better with smaller pseudcounters set to .1:

$(14 + .1) / (14 + 2.0) = 15/16$

Comments on training set

- Size of training set must be large relatively to the number of states to avoid possibility over fitting Usually we partition randomly the data into training set and test set
- It may be necessary to weight training set.

Subcluster of block BL00094F
YKQAGNSVVV
YKQLGNSVTV
YRQFGNSVAV
YKQAGNSITV
YKQAGNSITV
YKQTGNSITV
YRQFGNSVSV
WRQFGNSVPV
YRQFGNSVVV
YKQFGNSVVI
YKQFGNSVAV
YRQMGNSVVV
YRQFGNSVCV
YKQFGNSVAV

Representative of another sub cluster

RKQIGMAVPP

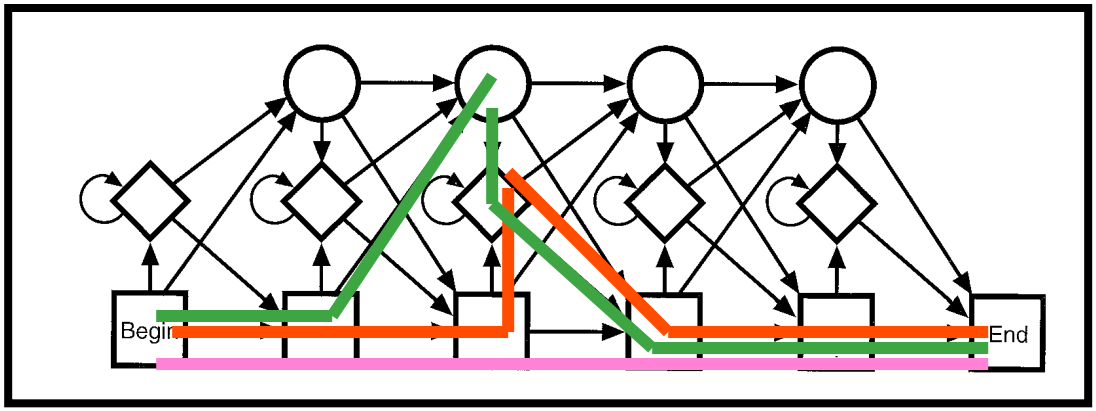
If the two sub-clusters are of different sizes one may consider to assign weights to each training sequence reverse proportional to the size of the cluster.

Sample applications of HMM

- Representing protein families (Profile HMM)
- Multiple sequence alignment
- Database searches
- Fold recognition
- Gene finding
- Genomic region discrimination (CpG islands)

Sequence alignment and HMM

1. Construct profile HMM for your family.
2. For each sequence find most likely path.
3. The matching/insert/delete states define the alignment.



In point 1 often a “seed” alignment is used

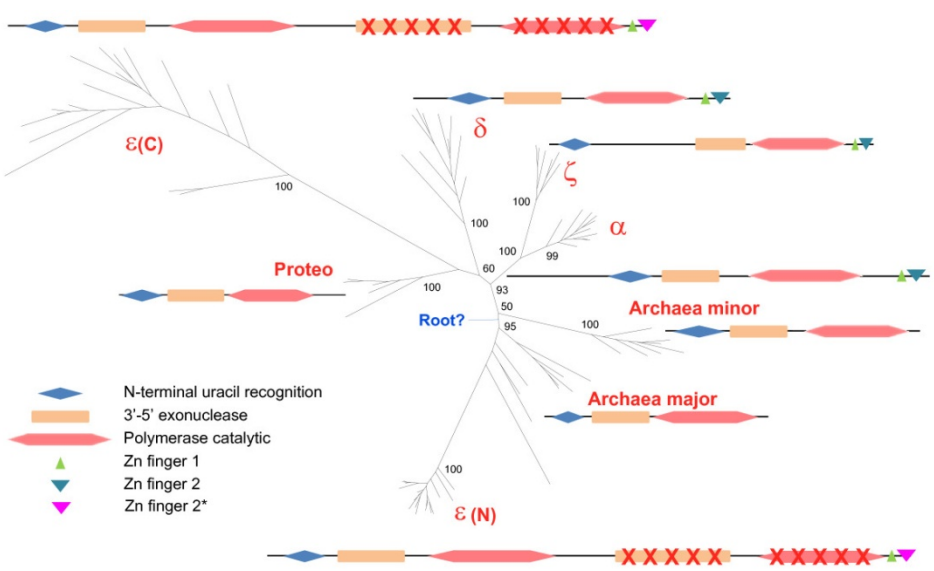
Pfam database

The Pfam database is a large collection of protein (domain) families, each represented by multiple sequence alignments and hidden Markov models (HMMs).

http://pfam.sanger.ac.uk/

Proteins are generally composed of one or more functional regions, commonly termed ***domains****.*

DNA polymerase, B family



Pfam database

There are two components to Pfam: Pfam-A and Pfam-B. Pfam-A entries are high quality, manually curated families. Although of lower quality, Pfam-B families can be useful for identifying functionally conserved regions when no Pfam-A entries are found.

Pfam also generates higher-level groupings of related families, known as clans. A clan is a collection of Pfam-A entries which are related by similarity of sequence, structure or profile-HMM.