

# Hidden Markov Models

Laboratory of Bioinformatics I  
Module 2

2 April, 2020

**Emidio Capriotti**

<http://biofold.org/>



**Biomolecules**  
**Folding and**  
**Disease**

Department of Pharmacy and  
Biotechnology (FaBiT)  
University of Bologna



# Formal Definition

A HMM is a stochastic generator of sequences characterized by:

- $N$  states
- A set of transition probabilities between two states  $\{a_{kj}\}$   
$$a_{kj} = P(\pi(i) = j \mid \pi(i-1) = k)$$
- A set of starting probabilities  $\{a_{0k}\}$   
$$a_{0k} = P(\pi(1) = k)$$
- A set of ending probabilities  $\{a_{k0}\}$   
$$a_{k0} = P(\pi(i) = \text{END} \mid \pi(i-1) = k)$$
- An alphabet  $\mathbf{C}$  with  $M$  characters.
- A set of emission probabilities for each state  $\{e_k(c)\}$   
$$e_k(c) = P(s^i = c \mid \pi(i) = k)$$
- Constraints:  
$$\sum_k a_{0k} = 1$$
$$a_{k0} + \sum_j a_{kj} = 1 \quad \forall k$$
$$\sum_{c \in C} e_k(c) = 1 \quad \forall k$$

$s$ : sequence,  $\pi$ : path through the states

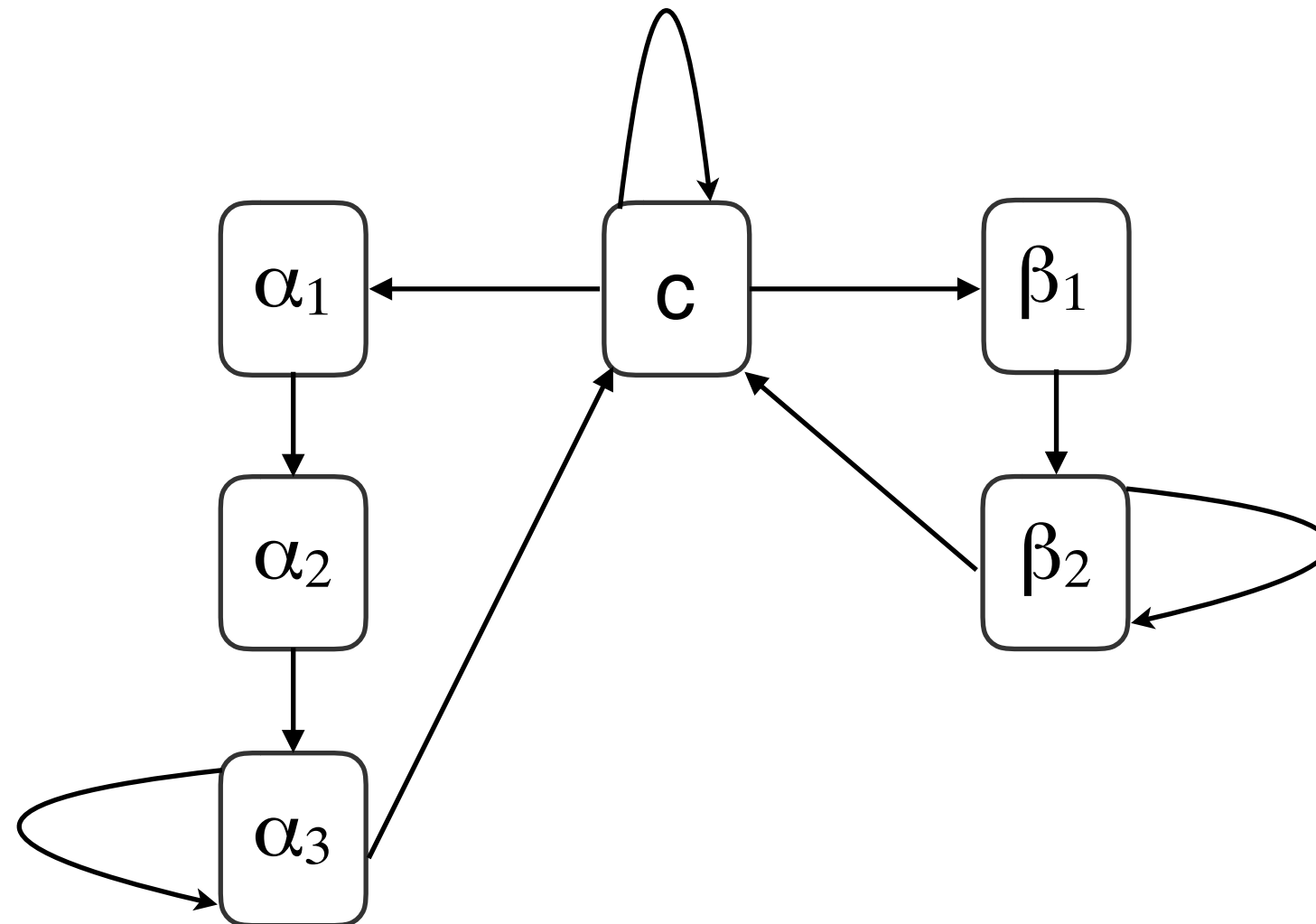
# Hidden Markov Models

HMMs **interpret an observable sequence** (residue sequence or DNA/RNA sequence) as «generated» by an underlying (hidden) process.

**Transition topology and probabilities** define a global grammar

Emission probabilities cast the propensity of observable symbols in each state

# Secondary Structure



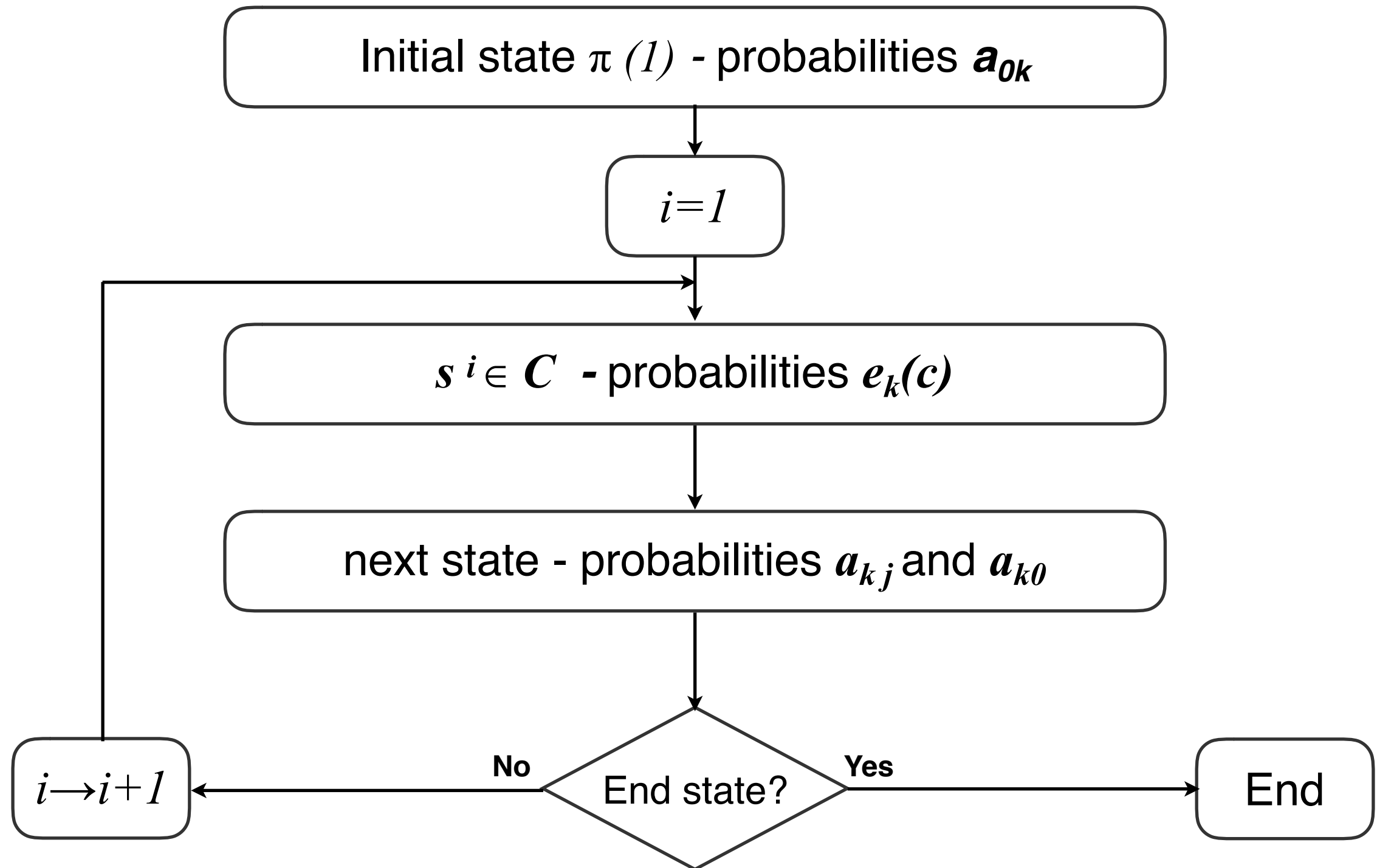
**S A L K M N Y T R E I M V A S N Q**  
 c  $\alpha_1$   $\alpha_2$   $\alpha_3$   $\alpha_3$   $\alpha_3$   $\alpha_3$  c c c c  $\beta_1$   $\beta_2$   $\beta_2$   $\beta_2$  c c  
 c  $\alpha$   $\alpha$   $\alpha$   $\alpha$   $\alpha$   $\alpha$  c c c c  $\beta$   $\beta$   $\beta$   $\beta$  c c

s: sequence

$\pi$ : path

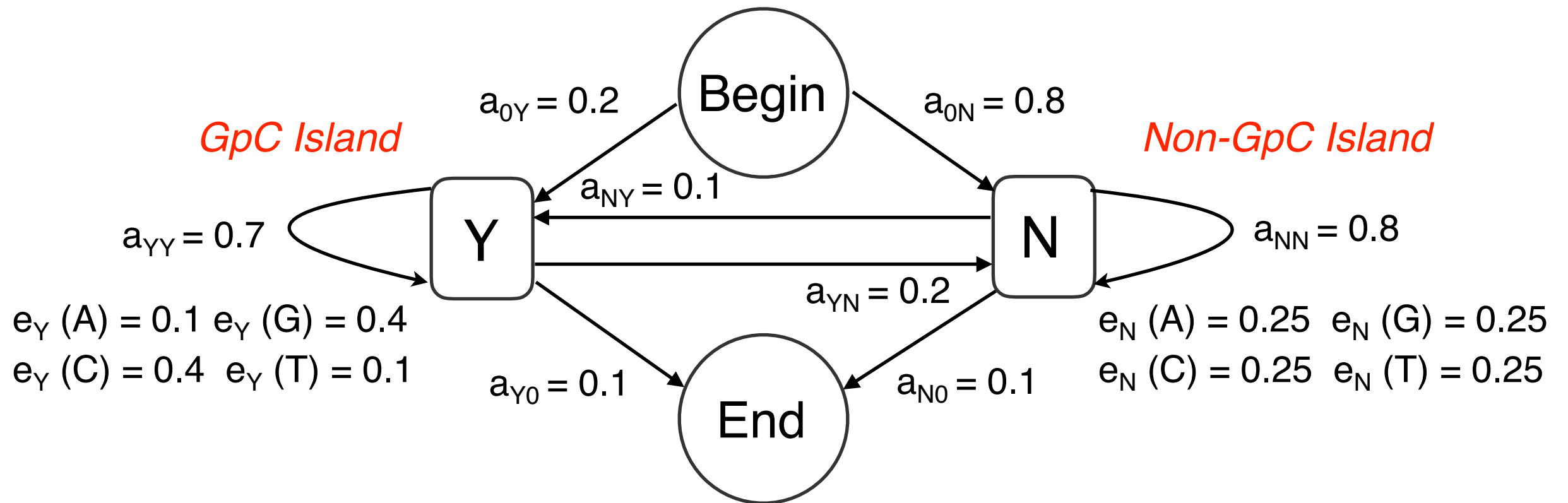
$Y(\pi)$ : labels

# Generating HMM Sequence



# GpC Islands Model

Probability of a sequence  $s$  with a given path  $\pi$



$s : \mathbf{A \quad G \quad C \quad G \quad C \quad G \quad T \quad A \quad A \quad T \quad C \quad T \quad G}$   
 $\pi : \mathbf{Y \quad Y \quad Y \quad Y \quad Y \quad Y \quad Y \quad N \quad N \quad N \quad N \quad N \quad N}$

Emission:  $0.1 \times 0.4 \times 0.4 \times 0.4 \times 0.4 \times 0.4 \times 0.1 \times 0.25 \times 0.25 \times 0.25 \times 0.25 \times 0.25 \times 0.25$   
 Transition:  $0.2 \times 0.7 \times 0.7 \times 0.7 \times 0.7 \times 0.7 \times 0.7 \times 0.2 \times 0.8 \times 0.8 \times 0.8 \times 0.8 \times 0.8 \times 0.1$

# Joint Probability

Calculate the joint probability of the sequence (s) and the path ( $\pi$ ) given the model (M)

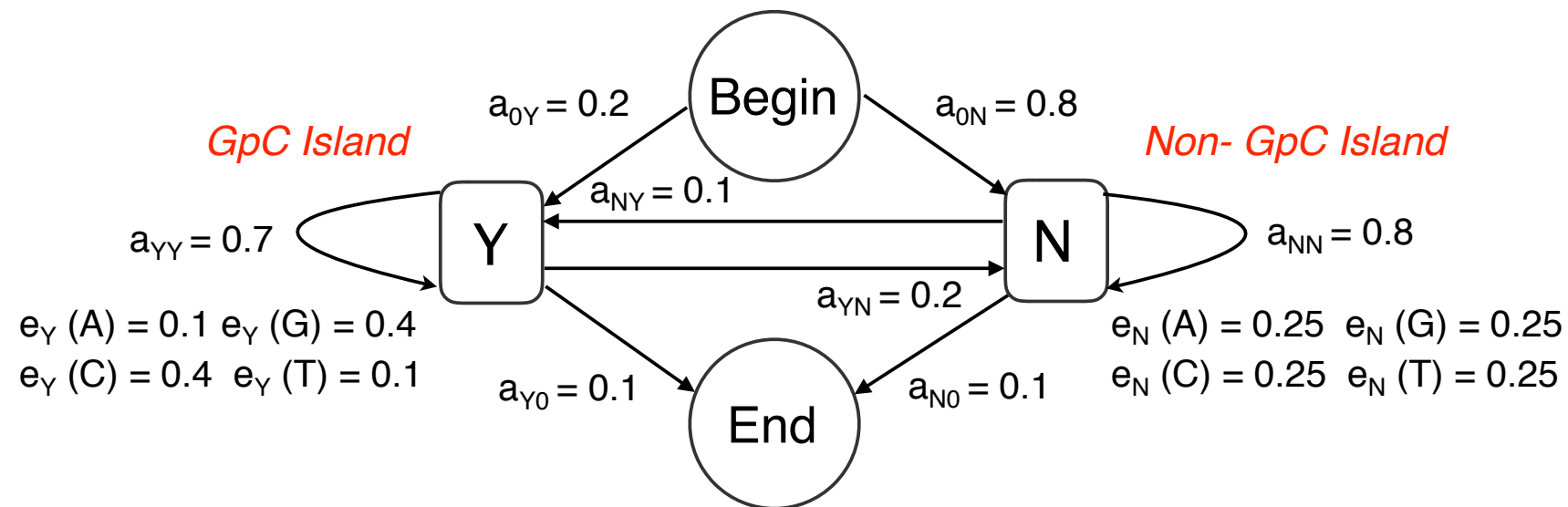
$$P(s, \pi \mid M) = P(s \mid \pi, M) \cdot P(\pi \mid M)$$

$$P(\pi \mid M) = a_{0\pi(1)} \cdot \prod_{i=2}^T a_{\pi(i-1)\pi(i)} \cdot a_{\pi(T)0}$$

$$P(s \mid \pi, M) = \prod_{i=1}^T e_{\pi(i)}(s^i)$$

$$P(s, \pi \mid M) = a_{\pi(T)0} \cdot \prod_{i=1}^T a_{\pi(i-1)\pi(i)} \cdot e_{\pi(i)}(s^i)$$

# Sequence Probability



$$P(s | M) = \sum_{\pi} P(s, \pi | M)$$

**2<sup>13</sup> different paths**

Summing over all the path will give the probability of having the sequence

$s :$	<b>A</b>	<b>G</b>	<b>C</b>	<b>G</b>	<b>C</b>	<b>G</b>	<b>T</b>	<b>A</b>	<b>A</b>	<b>T</b>	<b>C</b>	<b>T</b>	<b>G</b>
$\pi_1 :$	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>
$\pi_2 :$	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>N</b>
$\pi_3 :$	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>N</b>	<b>Y</b>
$\pi_4 :$	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>N</b>	<b>N</b>
$\pi_5 :$	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>N</b>	<b>Y</b>	<b>Y</b>



# Forward Algorithm

On the basis of preceding observations the computation of  $P(s \mid M)$  can be decomposed in simplest problems

For each state  $k$  and each position  $i$  in the sequence, we compute:

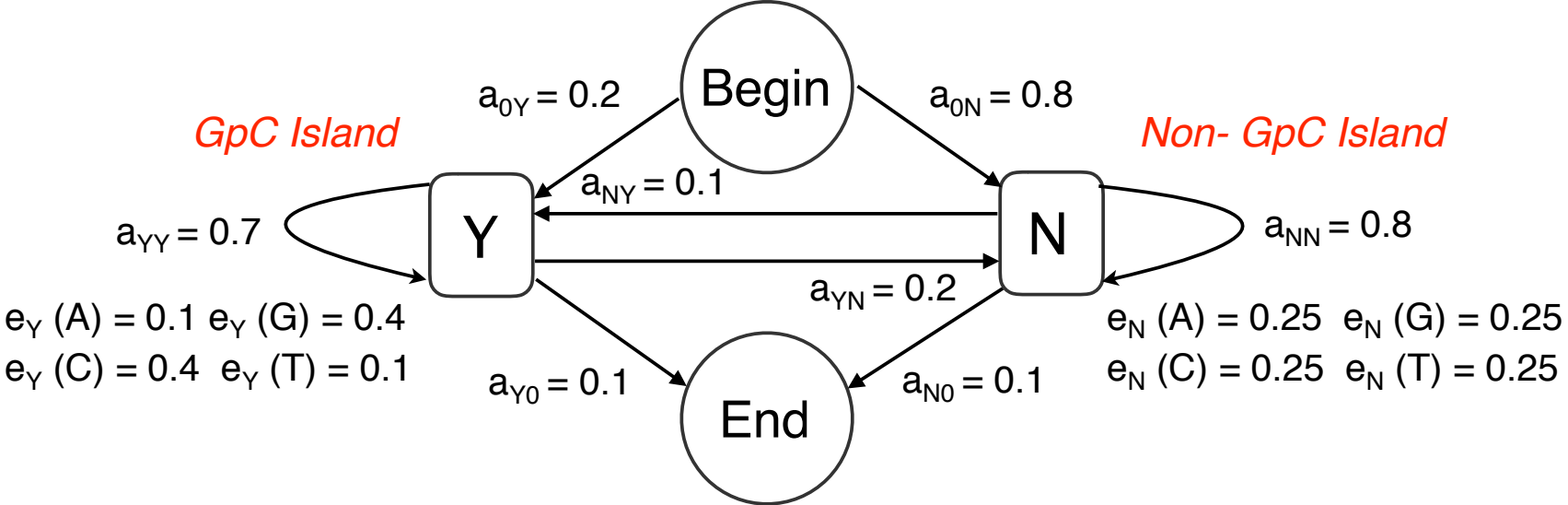
$$F_k(i) = P(s^1 s^2 s^3 \dots s^i, \pi(i) = k \mid M)$$

**Initialization:**  $F_{BEGIN}(0) = 1 \quad F_i(0) = 0 \quad \forall i \neq BEGIN$

**Recurrence:** 
$$\begin{aligned} F_l(i+1) &= P(s^1 s^2 \dots s^i s^{i+1}, \pi(i+1) = l) = \\ &= \sum_k P(s^1 s^2 \dots s^i, \pi(i) = k) \cdot a_{kl} \cdot e_l(s^{i+1}) = \\ &= e_l(s^{i+1}) \cdot \sum_k F_k(i) \cdot a_{kl} \end{aligned}$$

**Termination:** 
$$\begin{aligned} P(s) &= P(s^1 s^2 s^3 \dots s^T, \pi(T+1) = END) = \\ &= \sum_k P(s^1 s^2 \dots s^T, \pi(T) = k) \cdot a_{k0} \\ &= \sum_k F_k(T) \cdot a_{k0} \end{aligned}$$

# Forward Algorithm: Example



*S:* ATGCG *Initialization:*  $F_{BEGIN}(0) = 1$   $F_i(0) = 0 \forall i \neq BEGIN$

*Recurrence:*  $F_l(i+1) = e_l(s^i) \cdot \sum_k F_k(i) \cdot a_{kl}$  *Termination:*  $P(s) = \sum_k F_k(T) \cdot a_{k0}$

	-	A	T	G	C	G	-
Begin	1	0	0	0	0	0	0
Y	0	$0.2 \times 0.1$	$2e-2 \times 0.7 \times 0.1 + 0.2 \times 0.1 \times 0.1 = 3.4e-3$	$3.4e-3 \times 0.7 \times 0.4 + 4.1e-2 \times 0.1 \times 0.4 = 2.59e-3$	$2.59e-3 \times 0.7 \times 0.4 + 8.37e-3 \times 0.1 \times 0.4 = 1.06056e-3$	$1.06056e-3 \times 0.7 \times 0.4 + 1.8036e-3 \times 0.1 \times 0.4 = 3.691008e-4$	
N	0	$0.8 \times 0.25$	$2e-2 \times 0.2 \times 0.25 + 0.2 \times 0.8 \times 0.25 = 4.1e-2$	$3.4e-3 \times 0.2 \times 0.25 + 4.1e-2 \times 0.8 \times 0.25 = 8.37e-3$	$2.59e-3 \times 0.2 \times 0.25 + 8.37e-3 \times 0.8 \times 0.25 = 1.8036e-3$	$1.06056e-3 \times 0.2 \times 0.25 + 1.8036e-3 \times 0.8 \times 0.25 = 4.13748e-4$	
End	0	0	0	0	0	0	$3.69e-4 \times 0.1 + 4.13e-4 \times 0.1 = 7.82e-5$

# Backward Algorithm

Similar to the Forward algorithm: it computes  $P(s | M)$ , reconstructing the sequence from the end

For each state  $k$  and each position  $i$  in the sequence, we compute:

$$B_k(i) = P(s^{i+1}s^{i+2}s^{i+3}\dots s^T | \pi(i) = k)$$

**Initialization:**  $B_k(T) = P(\pi(T+1) = \text{END} | \pi(T) = k) = a_{k0}$

**Recurrence:**  $B_l(i-1) = P(s^i s^{i+1} \dots s^T | \pi(i-1) = l) =$   
 $= \sum_k P(s^{i+1} s^{i+2} \dots s^T | \pi(i) = k) \cdot a_{lk} \cdot e_k(s^i) =$   
 $= \sum_k B_k(i) \cdot e_k(s^i) \cdot a_{lk}$

**Termination:**  $P(s) = P(s^1 s^2 s^3 \dots s^T | \pi(0) = \text{BEGIN}) =$   
 $= \sum_k P(s^2 \dots s^T | \pi(1) = k) \cdot a_{0k} \cdot e_k(s^1) =$   
 $= \sum_k B_k(1) \cdot a_{0k} \cdot e_k(s^1)$

# Computational Complexity

## *Naïf method*

$$P(s | M) = \sum_{\pi} P(s, \pi | M)$$

There are  $N^T$  possible paths.

Each path requires about  $2 \cdot T$  operations.

The time for the computation is  $O(T \cdot N^T)$

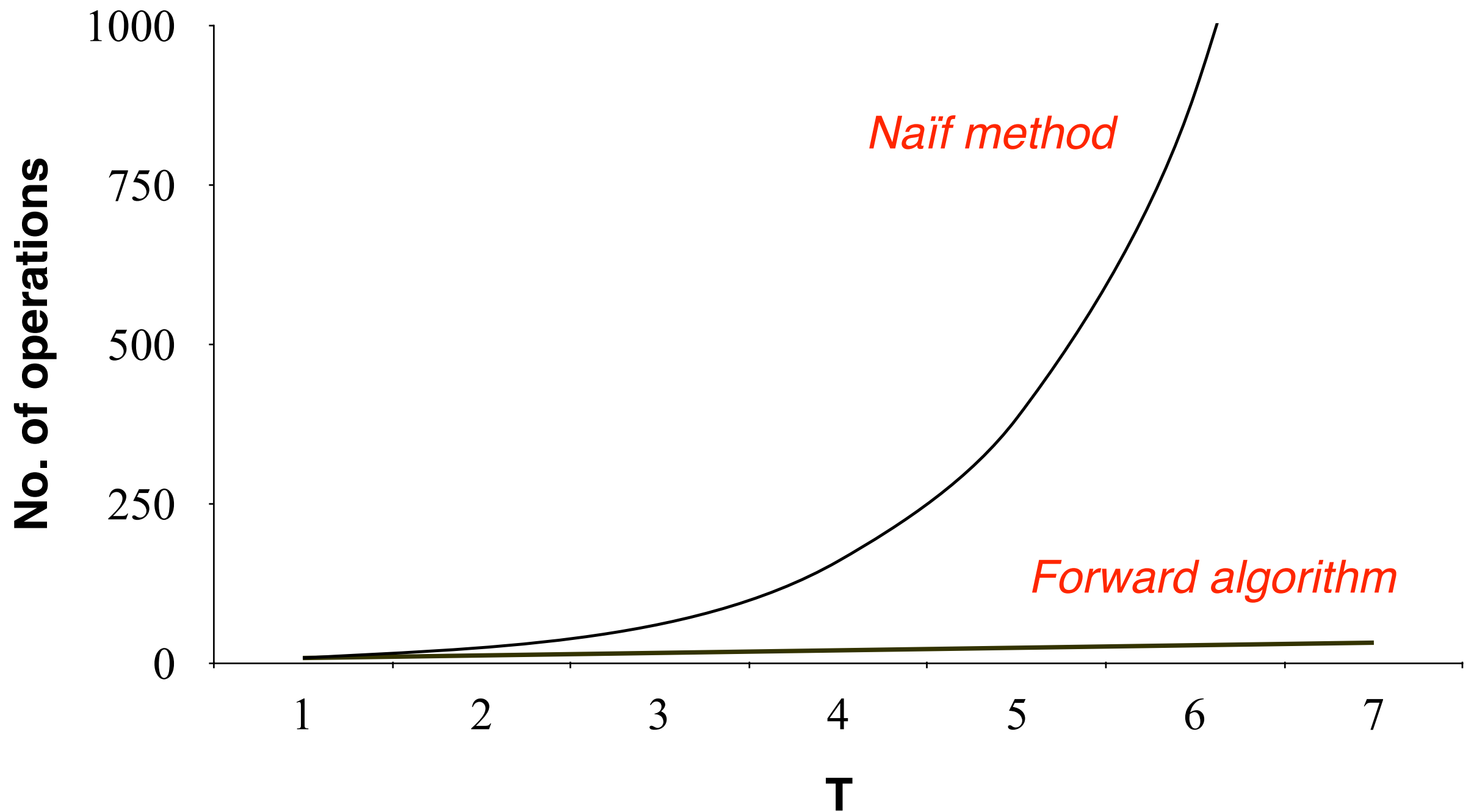
## *Forward Algorithm*

$T$  positions,  $N$  values for each position

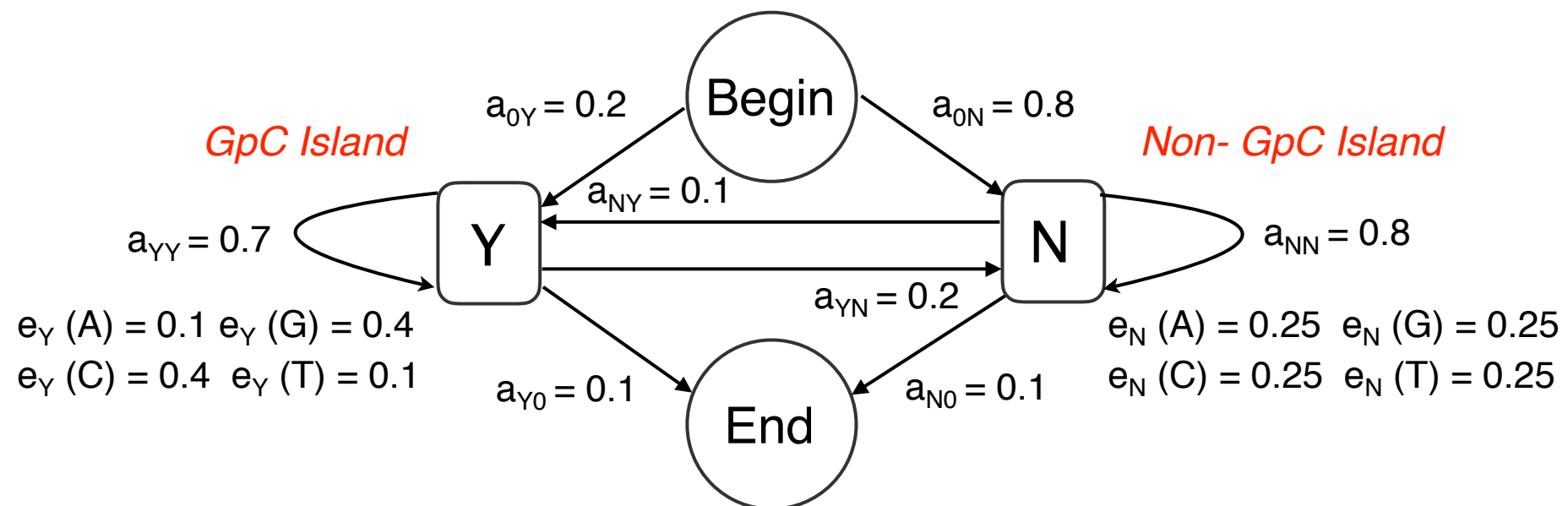
Each element requires about  $2 \cdot N$  product and 1 sum

The time for the computation is  $O(T \cdot N^2)$

# Complexity Plot



# Hidden Paths



$$\pi^* = \operatorname{argmax}_{\pi} [ P( \pi \mid s, M ) ]$$

$$= \operatorname{argmax}_{\pi} [ P( \pi, s \mid M ) ]$$

**$2^{13}$  different paths**  
 Viterbi path: path that gives  
 the best joint probability

$s :$	<b>A</b>	<b>G</b>	<b>C</b>	<b>G</b>	<b>C</b>	<b>G</b>	<b>T</b>	<b>A</b>	<b>A</b>	<b>T</b>	<b>C</b>	<b>T</b>	<b>G</b>
$\pi_1 :$	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>
$\pi_2 :$	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>N</b>
$\pi_3 :$	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>N</b>	<b>Y</b>
$\pi_4 :$	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>N</b>	<b>N</b>
$\pi_5 :$	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>Y</b>	<b>N</b>	<b>Y</b>	<b>Y</b>

# Searching the Hidden Path

## *Viterbi decoding*

Among all the possible path, choose the path  $\pi^*$  that maximizes the  $P(\pi \mid s, M)$

$$\pi^* = \operatorname{argmax}_{\pi} [ P(\pi \mid s, M) ] = \operatorname{argmax}_{\pi} [ P(\pi, s \mid M) ]$$

## *A Posteriori decoding*

For each position choose the state  $\underline{\pi}(i)$ :

$$\underline{\pi}(i) = \operatorname{argmax}_k [ P(\pi(i) = k \mid s, M) ]$$

The contribution to this probability derives from all the paths that go through the state  $k$  at position  $i$ .

The A posteriori path can be a non-sense path (it may not be a legitimate path if some transitions are not permitted in the model)

# Viterbi Algorithm

$$\pi^* = \operatorname{argmax}_{\pi} [ P( \pi , s \mid M ) ]$$

The computation of  $P(s, \pi^* \mid M)$  can be decomposed in simplest problems

Let  $V_k(i)$  be the probability of the most probable path for generating the subsequence  $s^1 s^2 s^3 \dots s^i$  ending in the state  $k$  at iteration  $i$ .

*Initialization:*  $V_{BEGIN}(0) = 1 \quad V_i(0) = 0 \quad \forall i \neq BEGIN$

*Recurrence:*  $V_l(i+1) = e_l(s^{i+1}) \cdot \operatorname{Max}_k ( V_k(i) \cdot a_{kl} )$

$$\operatorname{ptr}_i(l) = \operatorname{argmax}_k ( V_k(i) \cdot a_{kl} )$$

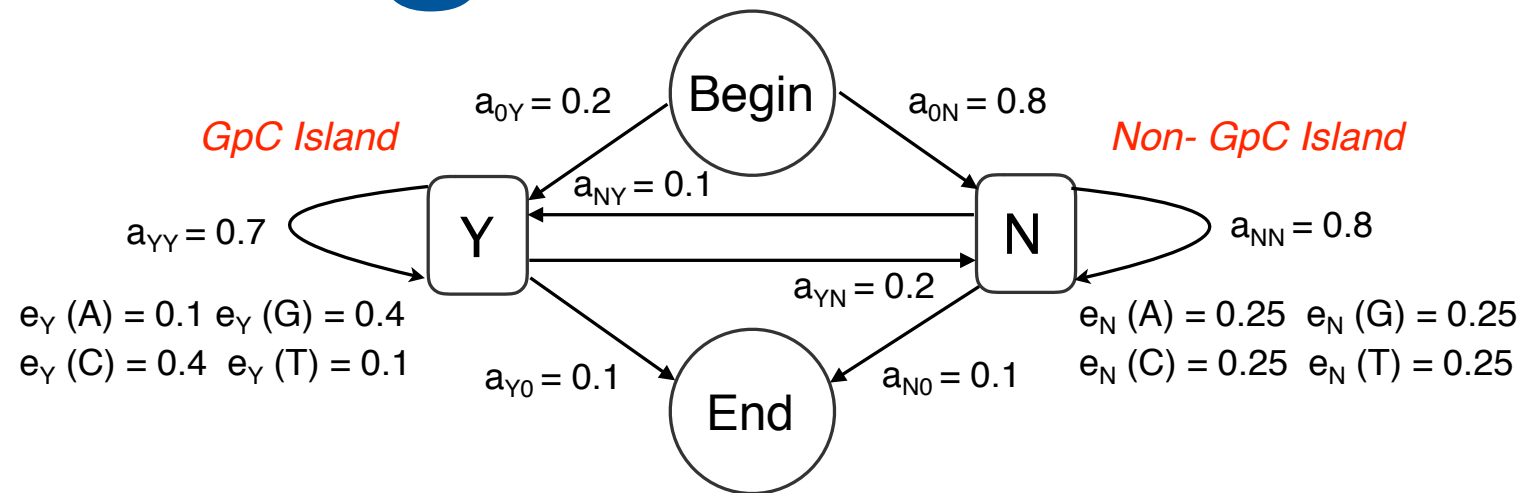
*Termination:*  $P(s, \pi^*) = \operatorname{Max}_k ( V_k(T) \cdot a_{k0} )$

$$\pi^*(T) = \operatorname{argmax}_k ( V_k(T) \cdot a_{k0} )$$

*Traceback:*  $\pi^*(i-1) = \operatorname{ptr}_i(\pi^*(i))$



# Viterbi Algorithm: Example



**S:** ATGCG **Initialization:**  $V_{BEGIN}(0) = 1$   $V_i(0) = 0 \forall i \neq BEGIN$

**Recurrence:**  $V_l(i) = e_l(s^i) \cdot \text{Max}_k (V_k(i-1) \cdot a_{kl})$  —  $ptr_i(l) = \text{argmax}_k (V_k(i-1) \cdot a_{kl})$

**Termination:**  $P(s, \pi^*) = \text{Max}_k (V_k(T) \cdot a_{k0})$  —  $\pi^*(T) = \text{argmax}_k (V_k(T) \cdot a_{k0})$

**Traceback:**  $\pi^*(i-1) = ptr_i(\pi^*(i))$

	-	A	T	G	C	G	-
Begin	1	0	0	0	0	0	0
Y	0	$0.2 \times 0.1 = 2e-2$ ptr=Begin	$\text{Max}(2e-2 \times 0.7 \times 0.1; 0.2 \times 0.1 \times 0.1)$ $2e-3$ ; ptr=N	$\text{Max}(2e-3 \times 0.7 \times 0.4; 1.6e-2 \times 0.1 \times 0.4)$ $6.4e-4$ ; ptr=N	$\text{Max}(6.4e-4 \times 0.7 \times 0.4; 3.2e-4 \times 0.1 \times 0.4)$ $1.79e-4$ ; ptr=Y	$\text{Max}(1.79e-4 \times 0.7 \times 0.4; 6.4e-5 \times 0.1 \times 0.4)$ $5.02e-5$ ; ptr=Y	
N	0	$0.8 \times 0.25 = 0.2$ ptr=Begin	$\text{Max}(2e-2 \times 0.2 \times 0.25; 0.2 \times 0.8 \times 0.25)$ $1.6e-2$ ; ptr=N	$\text{Max}(2e-3 \times 0.2 \times 0.25; 1.6e-2 \times 0.8 \times 0.25)$ $3.2e-4$ ; ptr=N	$\text{Max}(6.4e-4 \times 0.2 \times 0.25; 3.2e-4 \times 0.8 \times 0.25)$ $6.4e-5$ ; ptr=N	$\text{Max}(1.79e-4 \times 0.2 \times 0.25; 6.4e-5 \times 0.8 \times 0.25)$ $1.28e-5$ ; ptr=N	
End	0	0	0	0	0	0	$\text{Max}(5.01e-5 \times 0.1; 1.28e-5 \times 0.1)$ $5.02e-6$ ; ptr=Y

# A Posteriori Decoding

For each position choose the state  $\underline{\pi}(t)$  :

$$\underline{\pi}(i) = \operatorname{argmax}_k [ P(\pi(i) = k | s, M) ]$$

How to compute  $P(\pi(i) = k | s, M)$  for any state  $k$  and any position  $i$ ?

$$P(\pi(i) = k | s, M) = \frac{P(\pi(i) = k, s | M)}{P(s | M)}$$

$$\begin{aligned} P(\pi(i) = k, s | M) &= P(s^1 s^2 \dots s^i, \pi(i) = k | M) \cdot P(s^{i+1}, s^{i+2}, \dots s^T | \pi(i) = k, M) = \\ &= F_k(i) \cdot B_k(i) \end{aligned}$$

$$P(\pi(i) = k | s, M) = \frac{F_k(i) \cdot B_k(i)}{P(s | M)}$$

Elements of the Forward and  
Backward matrices

Computed with Forward or Backward  
algorithm termination steps