Generalized Hidden Markov Models

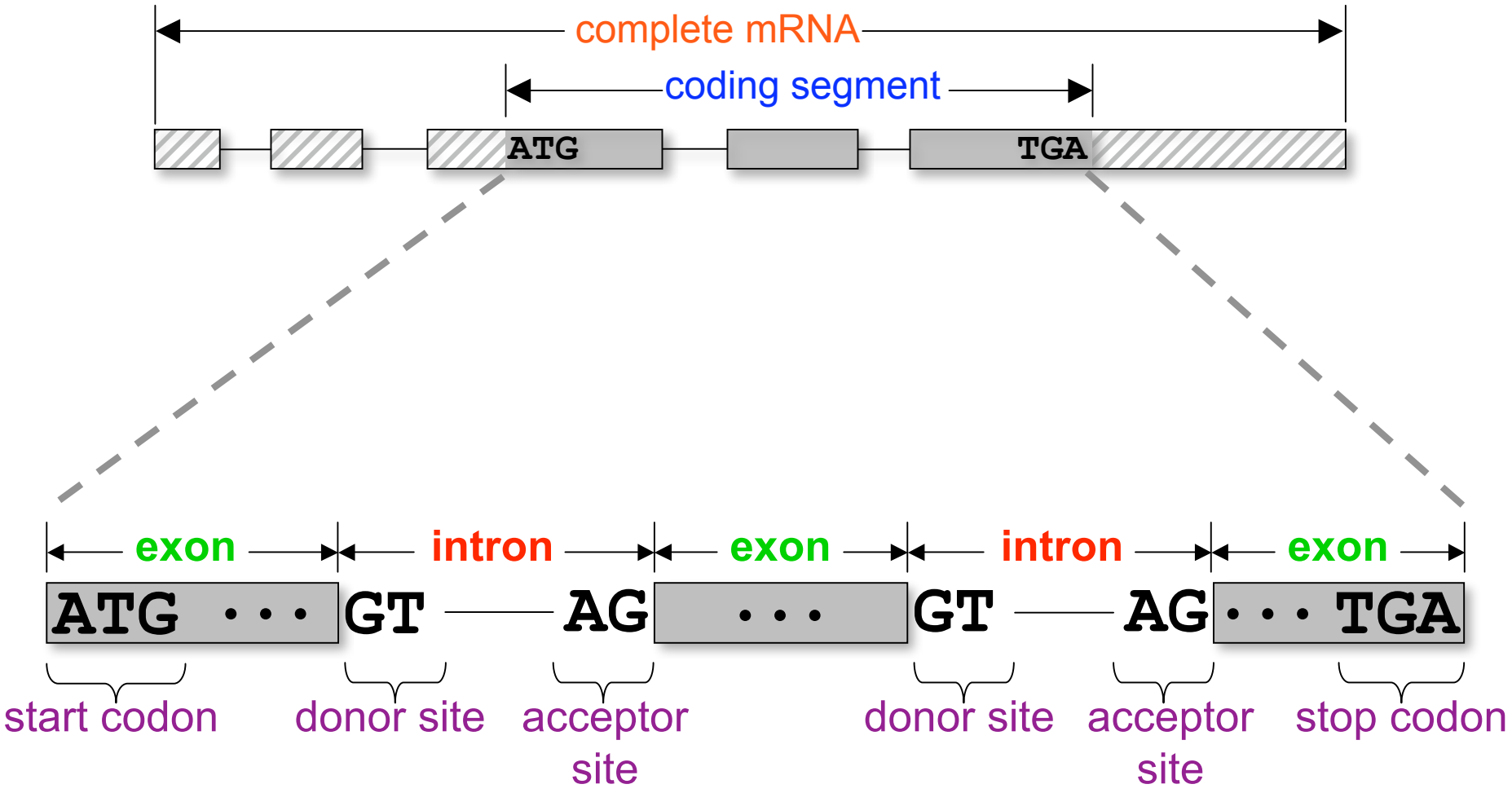
for Eukaryotic gene prediction

CBB 231 / COMPSCI 261

W.H. Majoros

Duke University

Recall: Gene Syntax



Recall: "Pure" HMMs

An HMM is a *stochastic machine* $M=(Q, \alpha, P_t, P_e)$ consisting of the following:

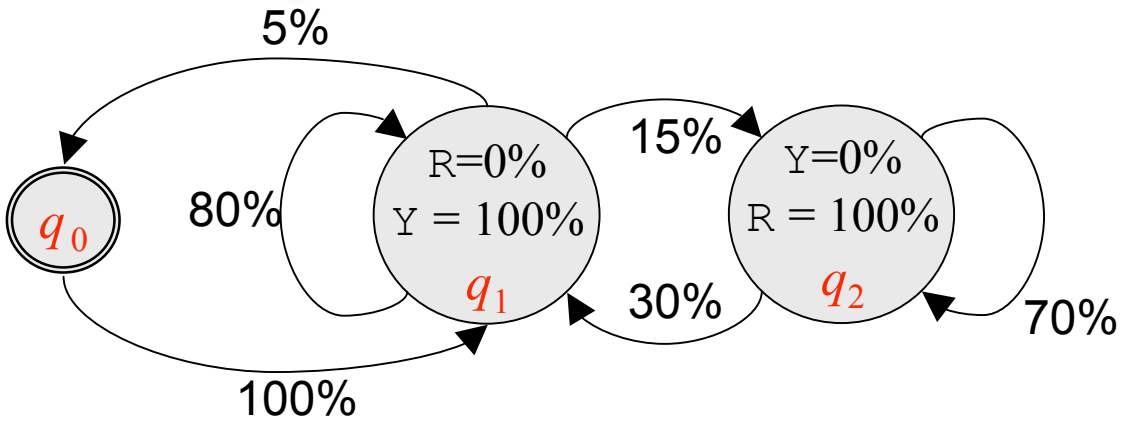
- a finite set of states, $Q=\{q_0, q_1, \ldots, q_m\}$
- a finite alphabet $\alpha=\{s_0, s_1, \ldots, s_n\}$
- a transition distribution $P_t : Q \times Q \mapsto \mathbb{R}$ i.e., $P_t(q_j \mid q_i)$
- an emission distribution $P_e : Q \times \alpha \mapsto \mathbb{R}$ i.e., $P_e(s_j \mid q_i)$

An Example

$M_1=(\{q_0,q_1,q_2\},\{\mathtt{Y},\mathtt{R}\},P_t,P_e)$

$P_t=\{(q_0,q_1,1), (q_1,q_1,0.8), (q_1,q_2,0.15), (q_1,q_0,0.05), (q_2,q_2,0.7), (q_2,q_1,0.3)\}$

$P_e=\{(q_1,\mathtt{Y},1), (q_1,\mathtt{R},0), (q_2,\mathtt{Y},0), (q_2,\mathtt{R},1)\}$



Generalized HMMs

A GHMM is a stochastic machine $M=(Q, \alpha, P_t, P_e, P_d)$ consisting of the following:

- a finite set of states, $Q=\{q_0, q_1, \ldots, q_m\}$
- a finite alphabet $\alpha=\{s_0, s_1, \ldots, s_n\}$
- a transition distribution $P_t : Q \times Q \mapsto \mathbb{R}$ i.e., $P_t(q_j \mid q_i)$
- an emission distribution $P_e : Q \times \alpha^* \times \mathbb{N} \mapsto \mathbb{R}$ i.e., $P_e(s_j \mid q_i, d_j)$
- a duration distribution $P_e : Q \times \mathbb{N} \mapsto \mathbb{R}$ i.e., $P_d(d_j \mid q_i)$

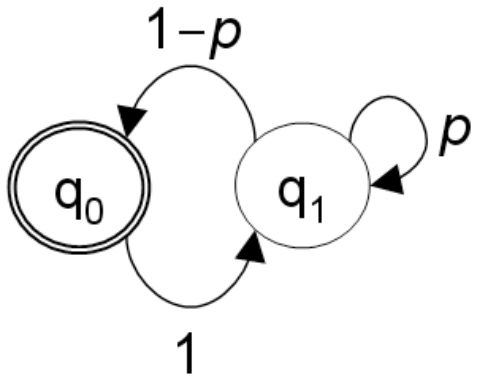
Key Differences

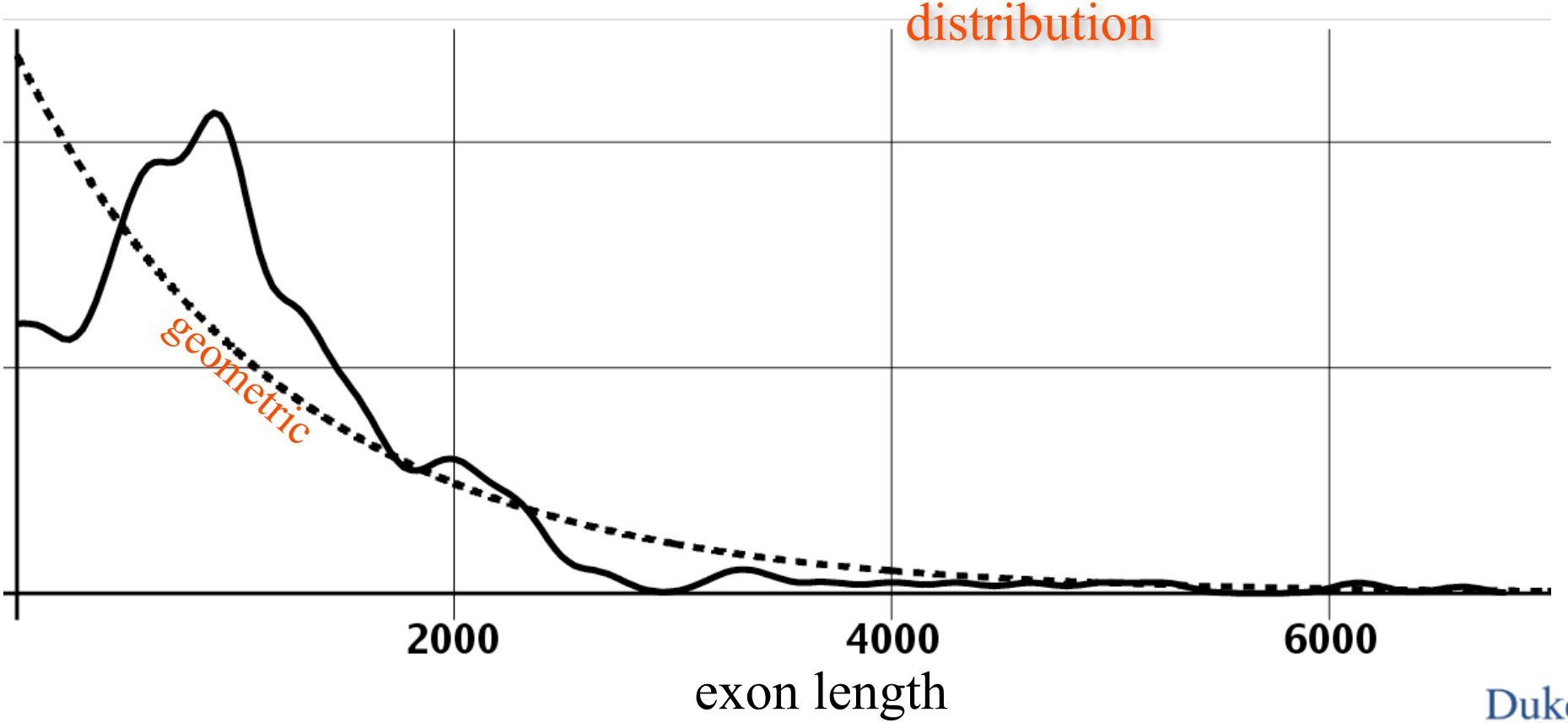
- each state now emits an entire subsequence rather than just one symbol
- feature lengths are now explicitly modeled, rather than implicitly geometric
- emission probabilities can now be modeled by any arbitrary probabilistic model
- there tend to be far fewer states => simplicity & ease of modification

Ref: Kulp D, Haussler D, Reese M, Eeckman F (1996) A generalized hidden Markov model for the recognition of human genes in DNA. *ISMB '96*.

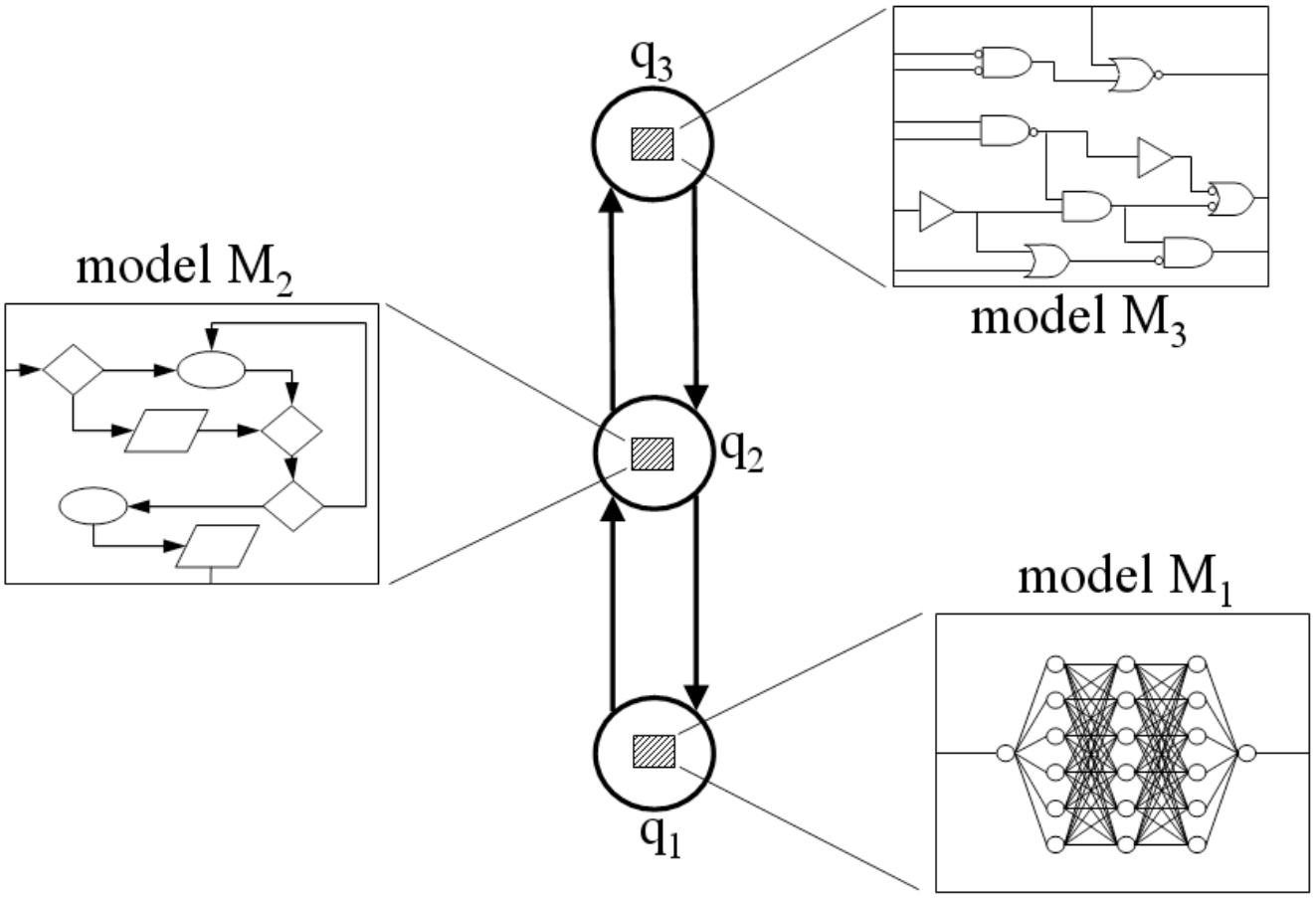
HMMs & Geometric Feature Lengths

$$P(x_0 \ldots x_{d-1} \mid \theta) = \left(\prod_{i=0}^{d-1} P_e(x_i \mid \theta) \right) \underbrace{p^{d-1}(1-p)}_{\text{geometric distribution}}$$





Model Abstraction in GHMMs



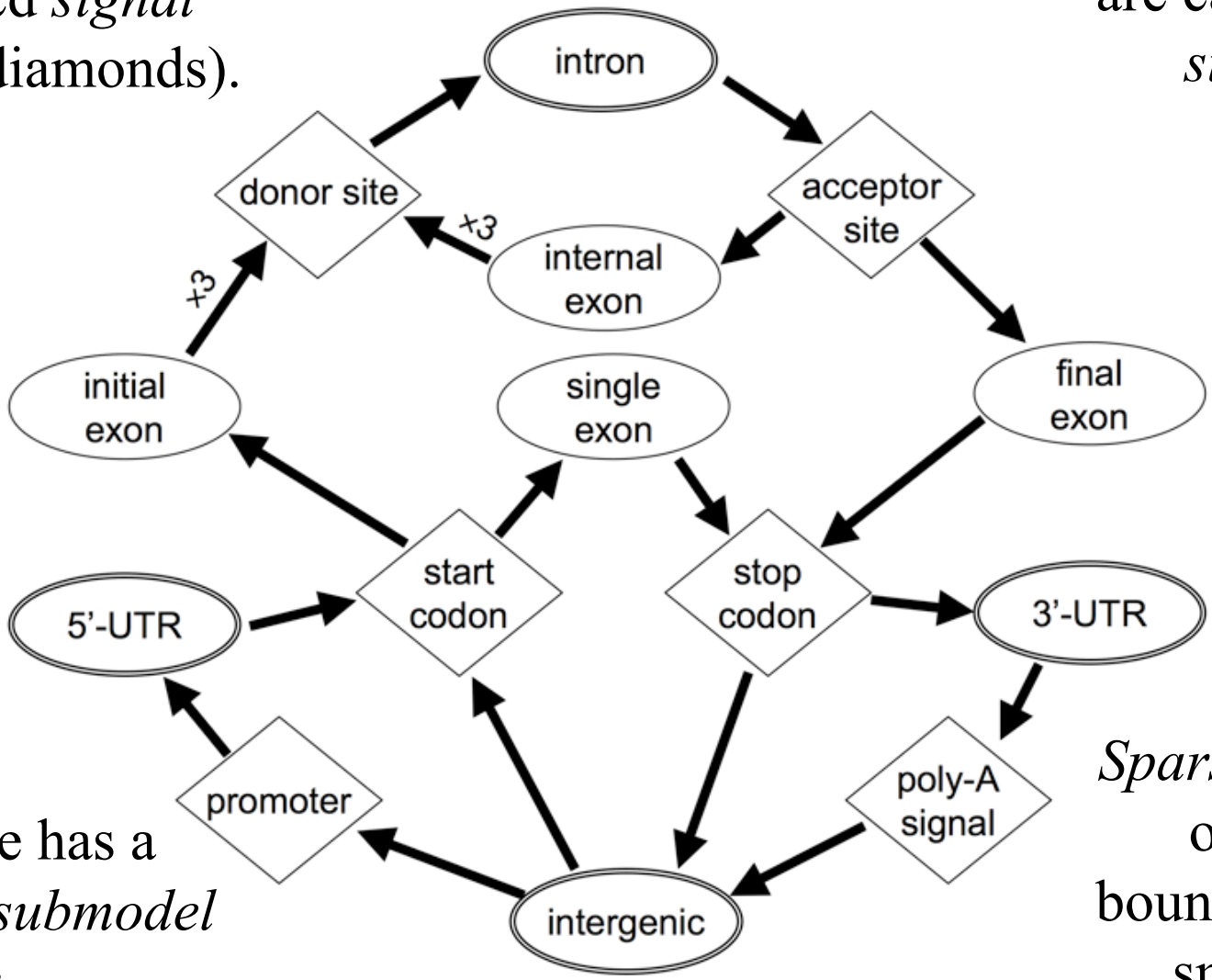
Advantages:
* Submodel abstraction
* Architectural simplicity
* State duration modeling

Disadvantages:
* Decoding complexity

Typical GHMM Topology for Gene Finding

Fixed-length states are called *signal states* (diamonds).

Variable-length states are called *content states* (ovals).



Each state has a separate *submodel* or *sensor*

Sparse: in-degree of all states is bounded by some small constant

Some GHMM Submodel Types

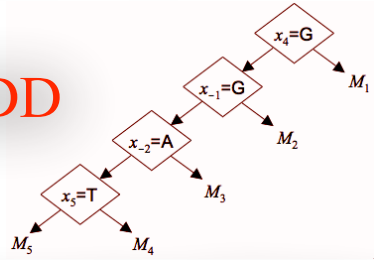
1. WMM (Weight Matrix) $\prod_{i=0}^{L-1} P_i(x_i)$

2. Nth-order Markov Chain (MC) $\prod_{i=0}^{n-1} P(x_i \mid x_0 \ldots x_{i-1}) \prod_{i=n}^{L-1} P(x_i \mid x_{i-n} \ldots x_{i-1})$

3. Three-Periodic Markov Chain (3PMC) $\prod_{i=0}^{L-1} P_{(f+i)(\mathrm{mod}\,3)}(x_i)$

5. Codon Bias $\prod_{i=0}^{n-1} P(x_{\alpha+3i} x_{\alpha+3i+1} x_{\alpha+3i+2})$

6. MDD



Ref: Burge C (1997) Identification of complete gene structures in human genomic DNA. *PhD thesis.* Stanford University.

7. Interpolated Markov Model

$$P_e^{IMM}(s \mid g_0 \ldots g_{k-1}) = \begin{cases} \lambda_k^G P_e(s \mid g_0 \ldots g_{k-1}) + (1 - \lambda_k^G) P_e^{IMM}(s \mid g_1 \ldots g_{k-1}) & \text{if } k > 0 \\ P_e(s) & \text{if } k = 0 \end{cases}$$

Ref: Salzberg SL, Delcher AL, Kasif S, White O (1998) Microbial gene identification using interpolated Markov models. *Nucleic Acids Research* 26:544-548.

Duke UNIVERSITY

Recall: Decoding with an HMM

$$\phi_{\max} = \underset{\phi}{argmax}\, P(\phi \mid S) = \underset{\phi}{argmax}\, \frac{P(\phi \wedge S)}{P(S)}$$

$$= \underset{\phi}{argmax}\, P(\phi \wedge S)$$

$$= \underset{\phi}{argmax}\, P(S \mid \phi) P(\phi)$$

$$P(S \mid \phi) = \prod_{i=0}^{L-1} P_e(x_i \mid y_{i+1})$$

emission prob.

$$P(\phi) = \prod_{i=0}^{L} P_t(y_{i+1} \mid y_i)$$

transition prob.

$$\phi_{\max} = \underset{\phi}{argmax}\, P_t(q_0 \mid y_L) \prod_{i=0}^{L-1} P_e(x_i \mid y_{i+1}) P_t(y_{i+1} \mid y_i)$$

Duke UNIVERSITY

Decoding with a GHMM

$$\phi_{\max} = \underset{\phi}{argmax}\, P(\phi \mid S) = \underset{\phi}{argmax}\, \frac{P(\phi \wedge S)}{P(S)}$$

$$= \underset{\phi}{argmax}\, P(\phi \wedge S)$$

$$= \underset{\phi}{argmax}\, P(S \mid \phi) P(\phi)$$

$$P(S \mid \phi) = \prod_{i=1}^{|\phi|-2} P_e(S_i \mid y_i, d_i)$$

emission prob.

$$P(\phi) = \prod_{i=0}^{|\phi|-2} P_t(y_{i+1} \mid y_i) P_d(d_i \mid y_i)$$

transition prob.

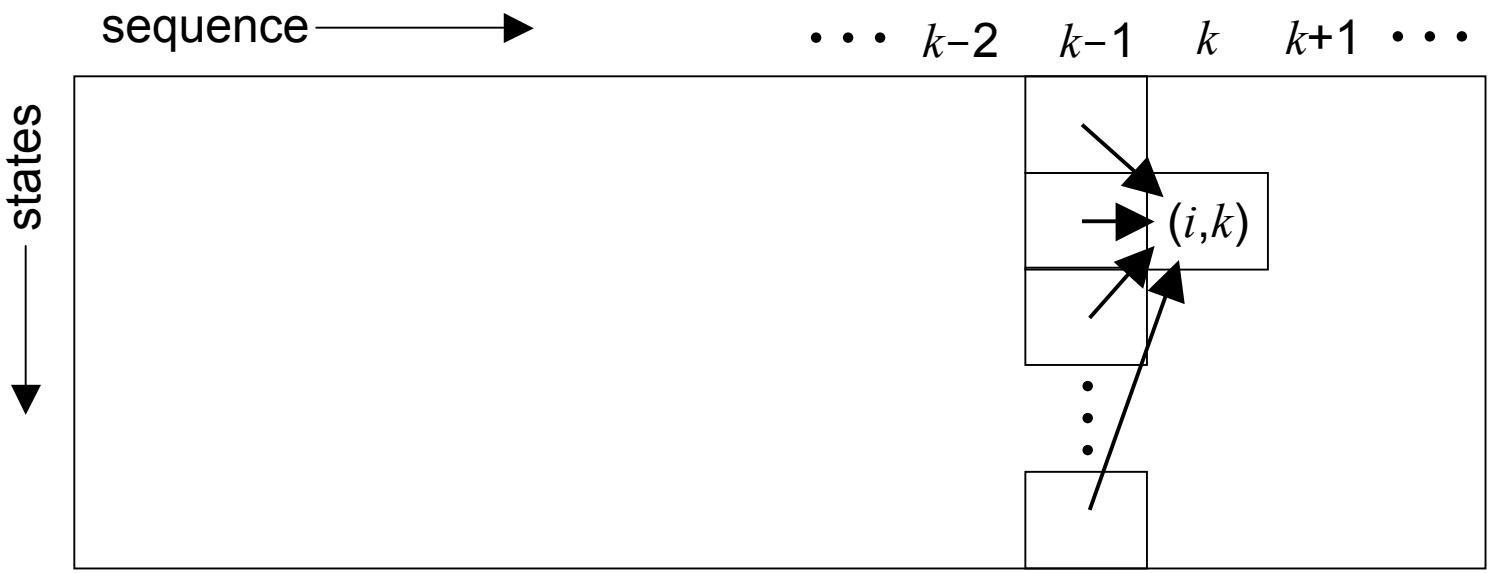
duration prob.

$$\phi_{\max} = \underset{\phi}{argmax} \prod_{i=0}^{|\phi|-2} P_e(S_i \mid y_i, d_i) P_t(y_{i+1} \mid y_i) P_d(d_i \mid y_i)$$

Duke UNIVERSITY

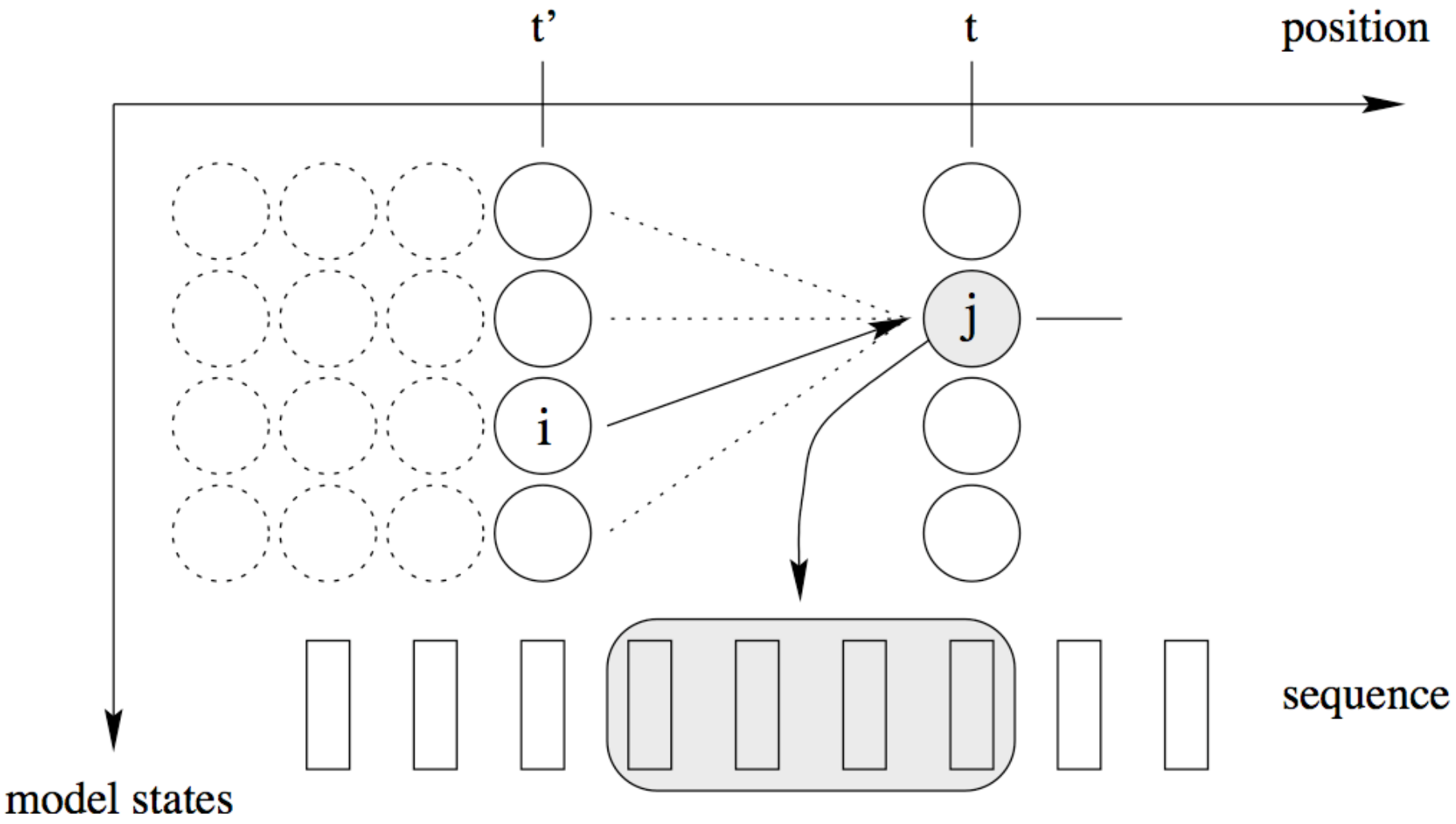
Recall: Viterbi Decoding for HMMs

$$V(i,k) = \begin{cases} \max_j V(j,k-1) P_t(q_i \mid q_j) P_e(x_k, q_i) & \text{if } k > 0, \\ P_t(q_i \mid q_0) P_e(x_0 \mid q_i) & \text{if } k = 0. \end{cases}$$



run time: $\mathbb{O}(L \times |Q|^2)$

Naive GHMM Decoding



run time: $\mathbb{O}(L^3 \times |Q|^2)$

Assumptions in GHMM Decoding for Gene Finding

(1) The emission functions P_e for variable-length features are *factorable* in the sense that they can be expressed as a product of terms evaluated at each position in a putative feature—i.e.,

$$factorable(P_e) \Leftrightarrow \exists_f P_e(S)=\prod_i f(S,i)$$

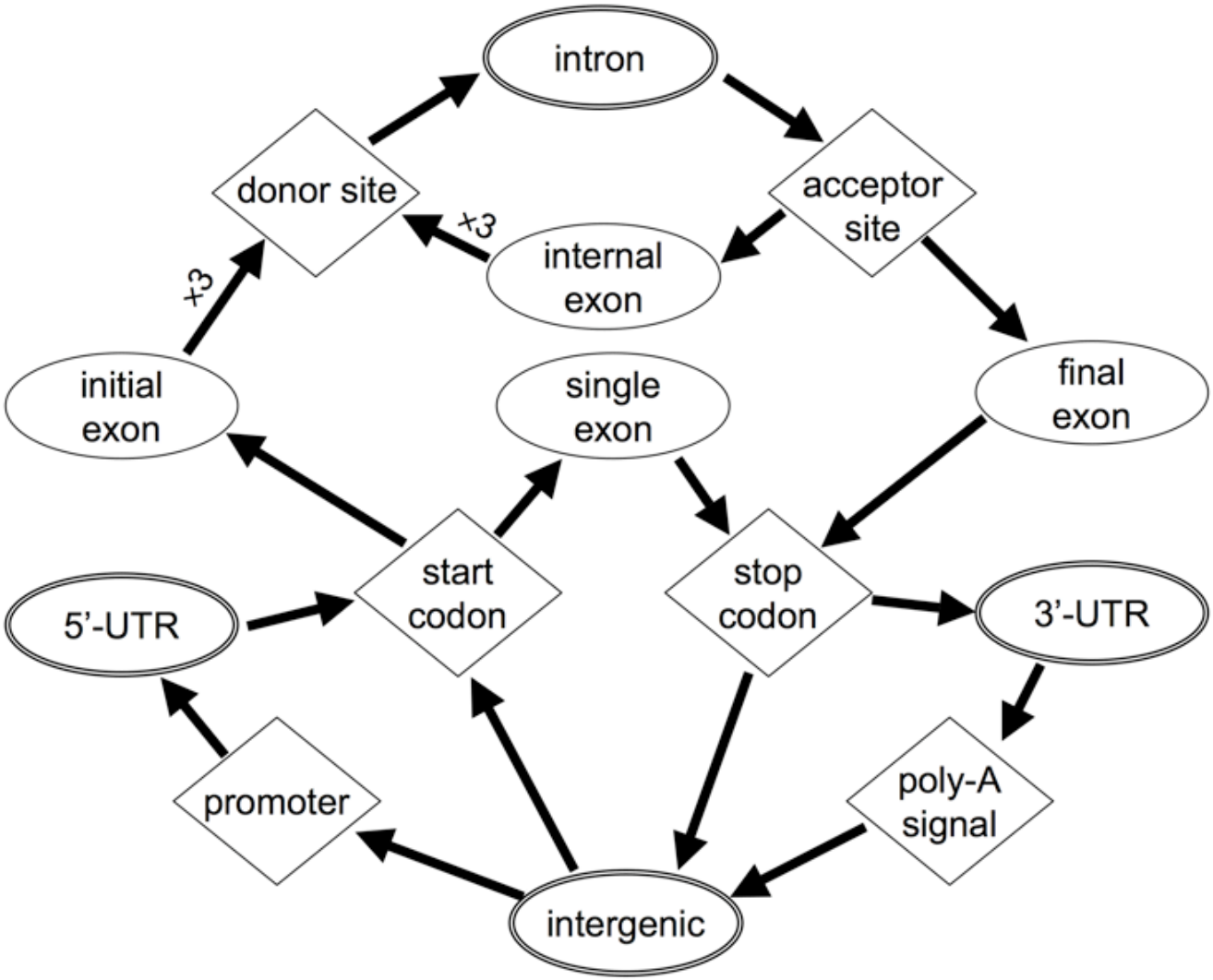
for $0 \leq i < |S|$ over any putative feature S.

(2) The lengths of noncoding features in genomes are geometrically distributed.

(3) The model contains no transitions between variable-length states.

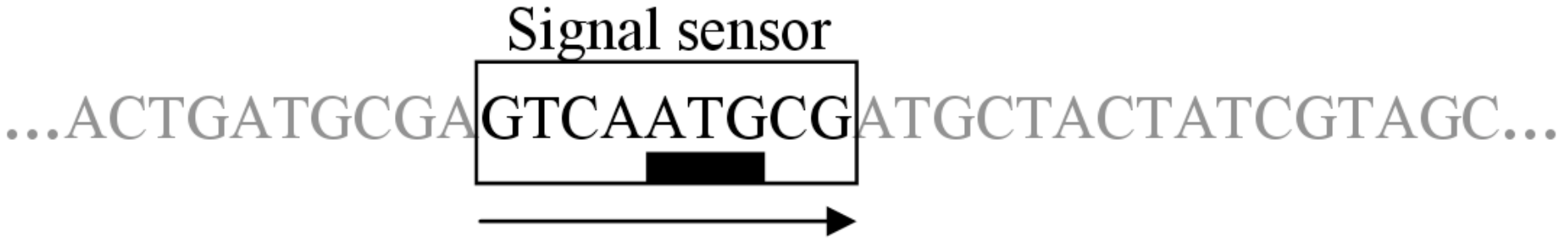
Assumption #3

Variable-length states cannot transition directly to each other.

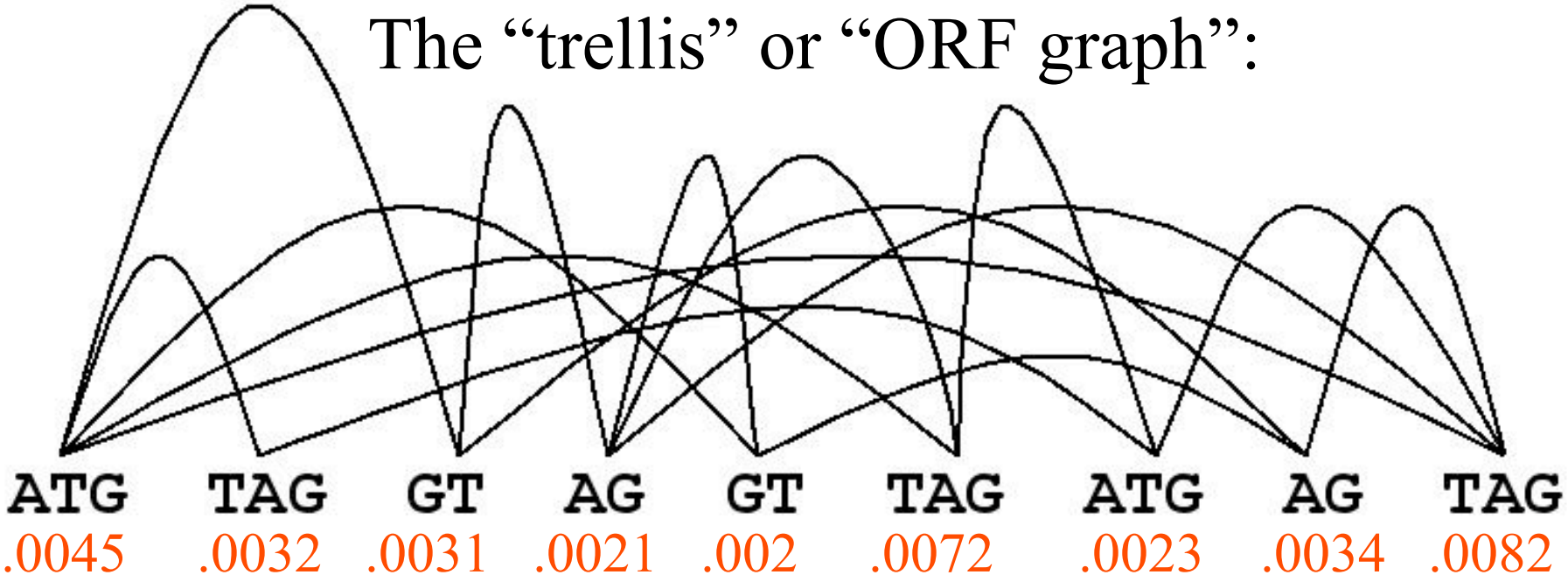


Efficient Decoding via Signal Sensors

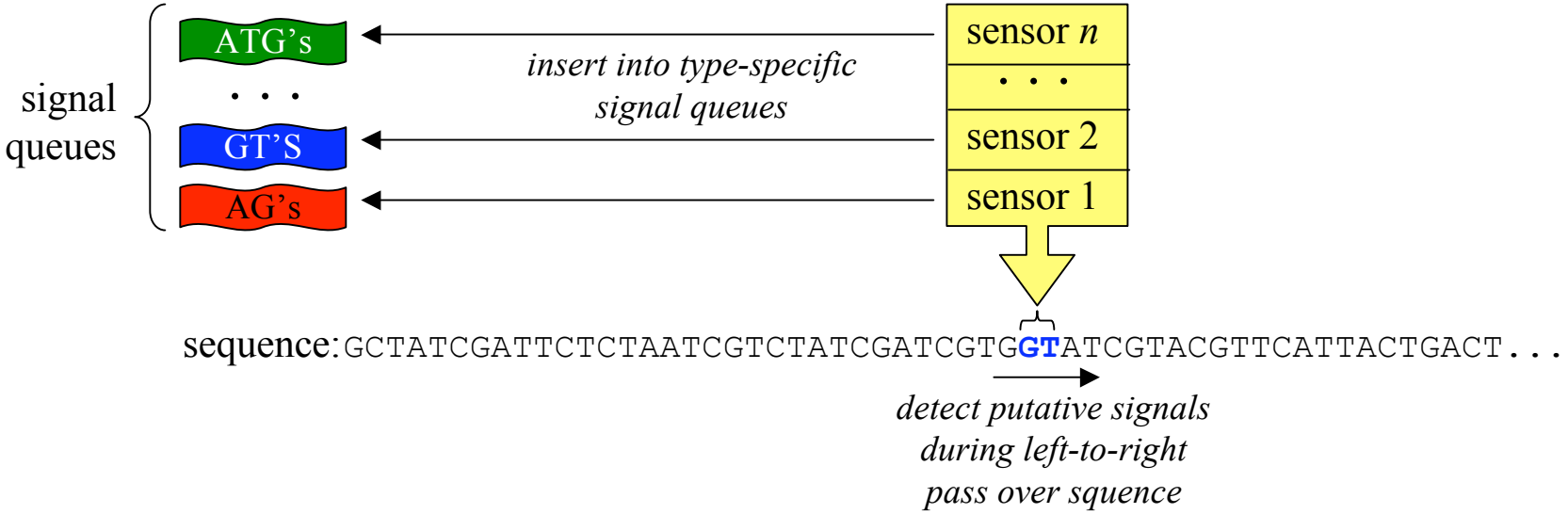
Each signal state has a *signal sensor*:

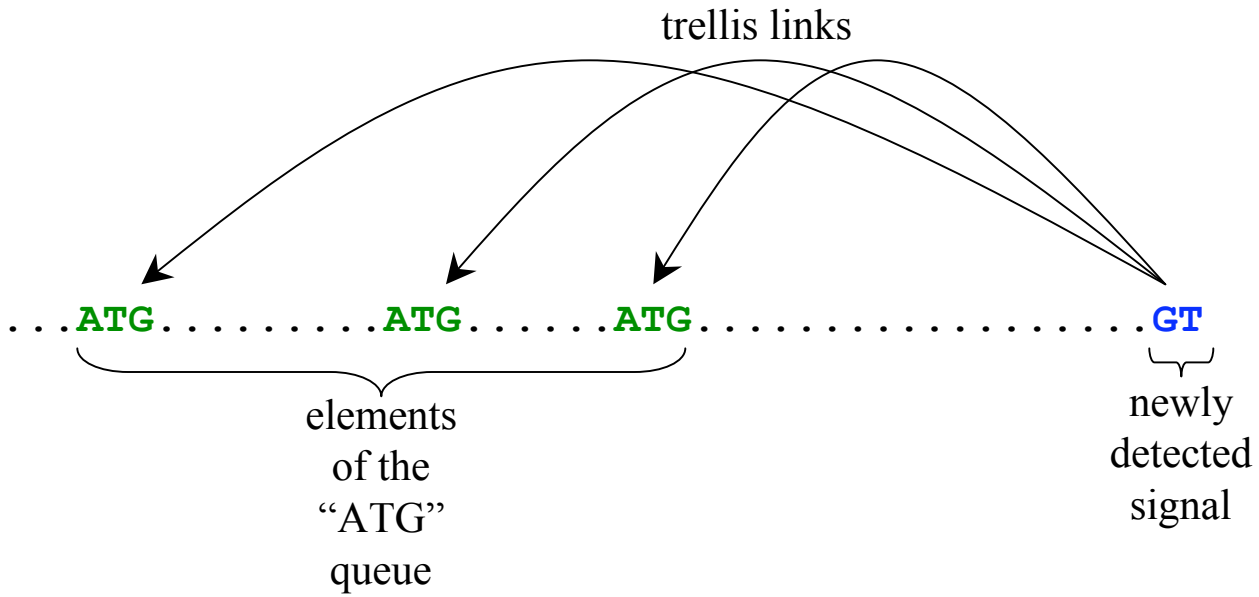


The "trellis" or "ORF graph":



Efficient Decoding via Signal Sensors





Duke UNIVERSITY

Decoding with an ORF Graph

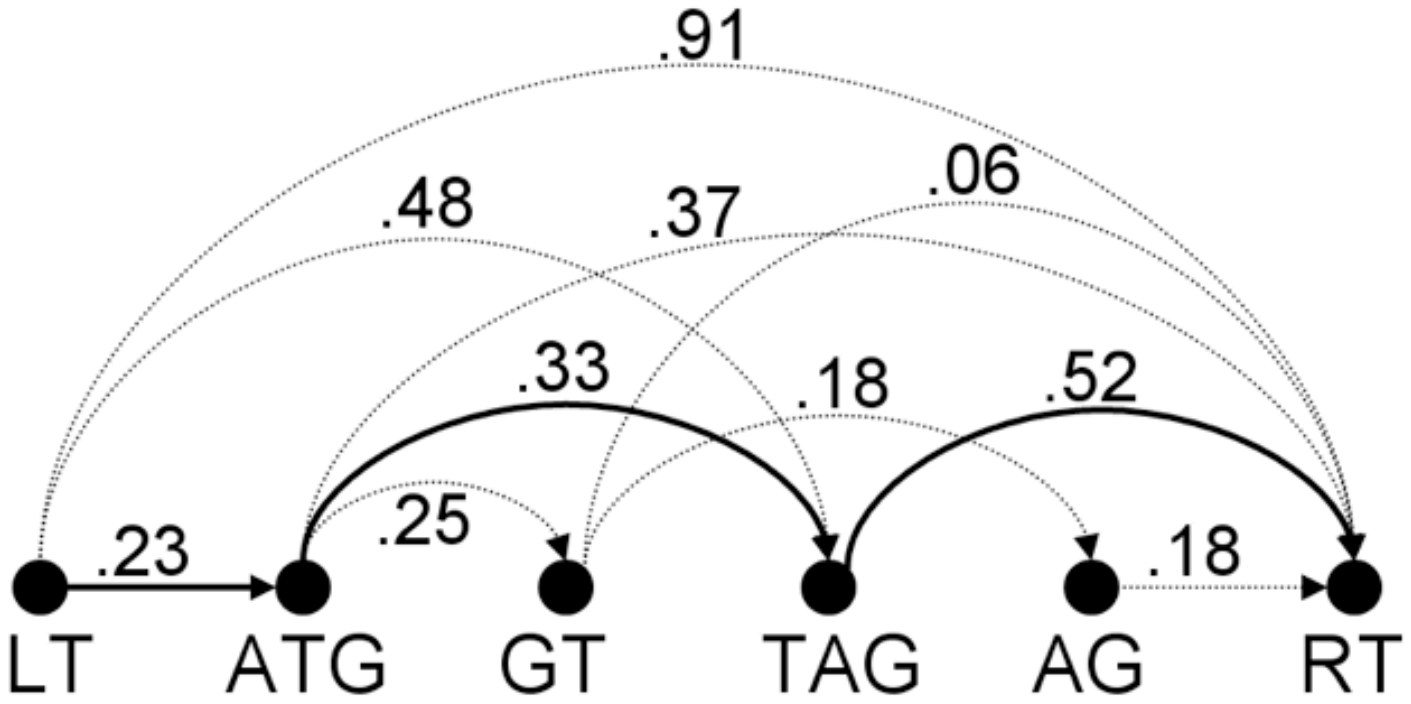
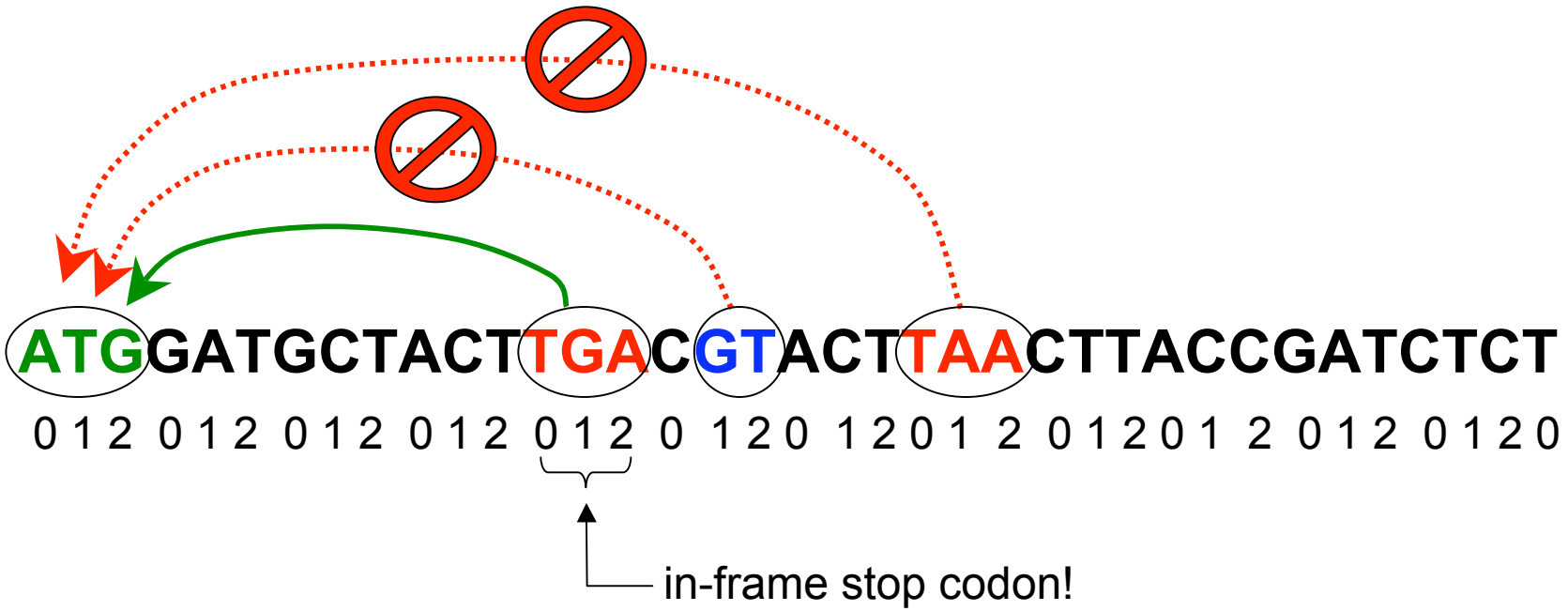


***Figure 8.17**: Operation of the highestScoringPath() algorithm for a small portion of a weighted ORF graph. The highest scoring path is shown in bold, and represents the optimal gene parse for this portion of the sequence. LT=left terminus, RT=right terminus.*

The Notion of “Eclipsing”



Eclipsing of Signals

Algorithm 8.1 Eclipsing signals in coding queue `G` when a stop codon has been encountered at position `p`. `pos(s)` is the position of the first base of the signal's consensus sequence (e.g., the `A` in `ATG`). `len(s)` is the length of the signal's consensus sequence (e.g., 3 for `ATG`).

```
procedure eclipse(ref G,p)
1. foreach s∈G do
2.    ω←(pos(s)+len(s)-p)mod3;
3.    eclipsed_s[ω]←true;
4.    if eclipsed_s[(ω+1)mod3] and
5.       eclipsed_s[(ω+2)mod3]
6.      then drop(s,G);
```

Bounding the Number of Coding Predecessors

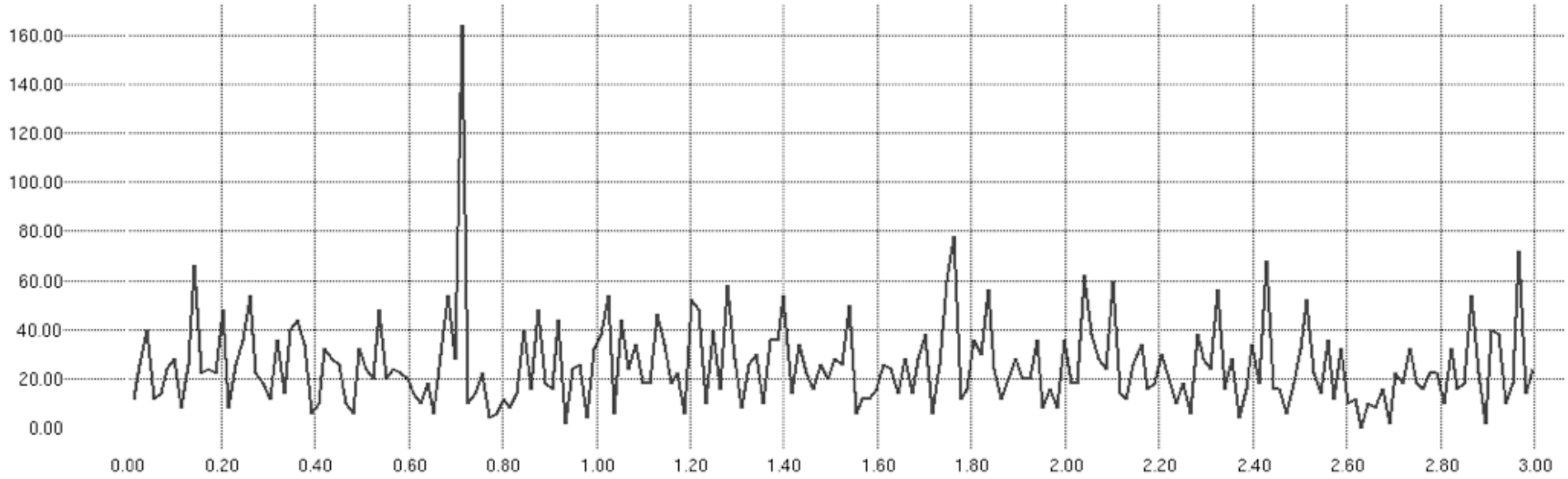


Figure 8.6 *Number of potential coding predecessors (Γ_{bc}) as a function of sequence length (in megabases). Data are from a 3 Mb human DNA sequence. Linear regression resulted in a slope of −0.000001, demonstrating that the number of coding predecessors does not increase without bound; the intercept indicated 27.3 predecessors on average.*

Geometric Noncoding Lengths (Assumption #2)

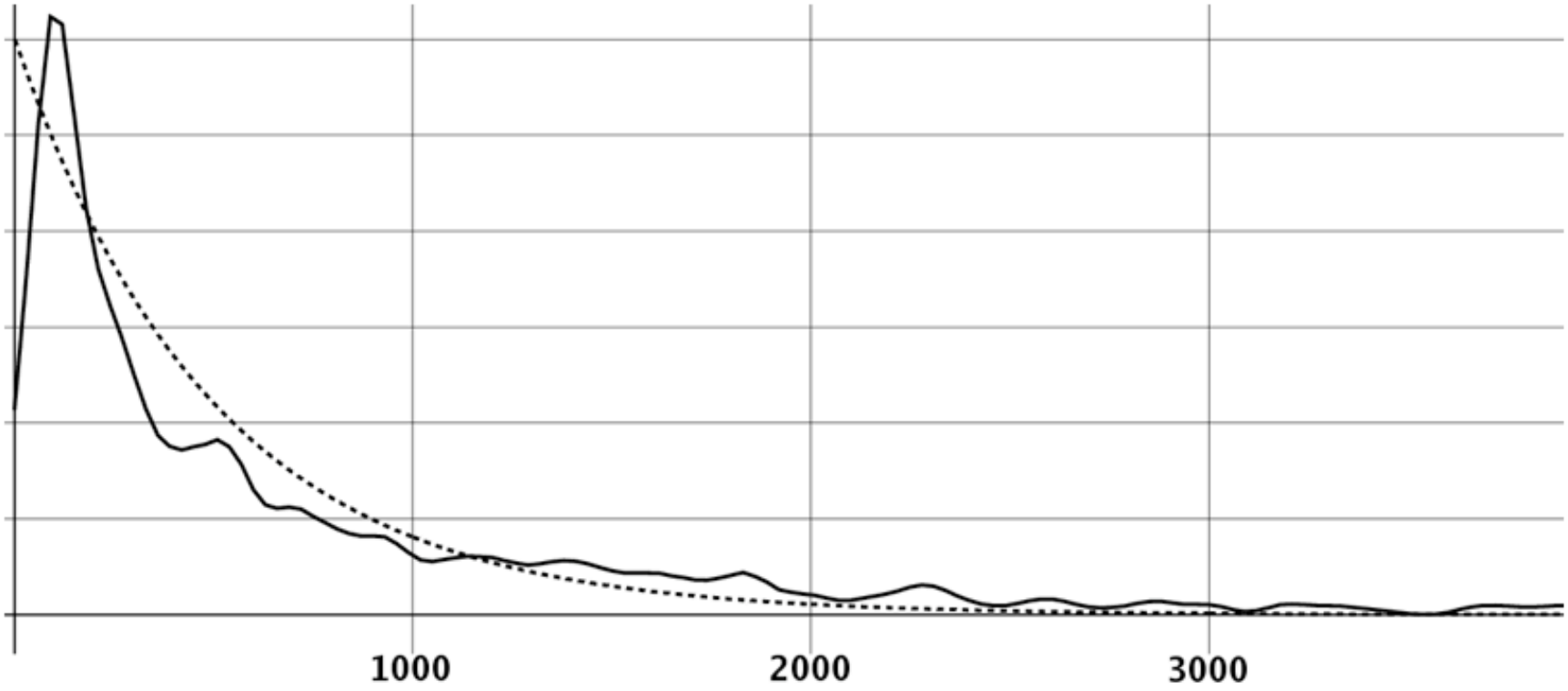
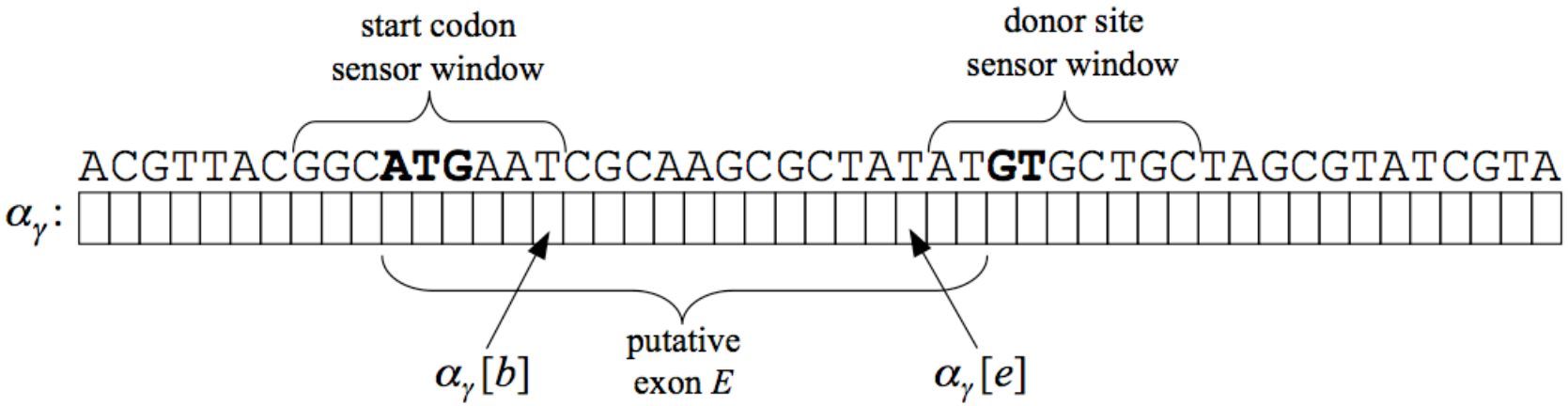


Figure 8.5 *Intron length distribution for Homo sapiens (solid line) and a geometric distribution (dashed line) with parameter p=0.002.*

Prefix Sum Arrays for Emission Probabilities

(Assumption #1: factorable content sensors)



$$\log P_e(S_{b+1..e} \mid exon, \omega) = \alpha_\gamma[e] - \alpha_\gamma[b]$$

Figure 8.9 *Using a prefix sum array to compute the emission probability for a putative exon. Subtraction is performed between array elements at either end of the [b,e] interval, which covers only the portion of the exon not covered by either signal sensor.*

Bounding the Number of Noncoding Predecessors

Theorem 8.1 *(Burge's noncoding predecessor theorem) Suppose that for some signal* $s_j \in \Gamma_{en}$, *the optimal noncoding predecessor was found to be some* $s^* \in \Gamma_{bn}$. *If the next signal encountered after* s_j *is some* $s_i \in \Gamma_{en}$, *then under assumptions (1) and (2) above,* s^* *is also the optimal predecessor (of its type and phase) for* s_i.

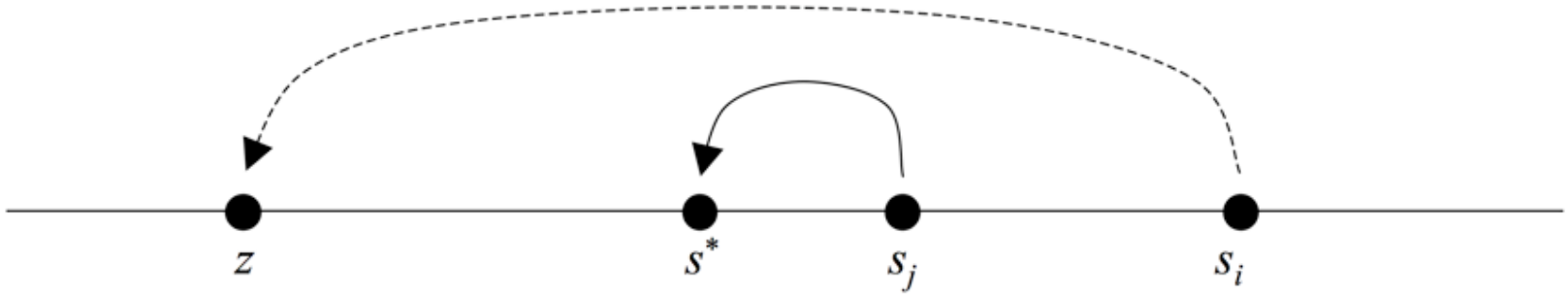


Figure 8.7 *Once an optimal noncoding predecessor* s^* *is identified for signal* s_j, *no signal* z *preceding* s^* *can be selected as a predecessor of a later signal* s_i, *assuming a geometric length distribution and a factorable content scoring function.*

Ref: Burge C (1997) Identification of complete gene structures in human genomic DNA. *PhD thesis.* Stanford University.

PSA Decoding

Algorithm 8.3 Overview of the PSA decoding algorithm. See text for details.

```
procedure PSA(S, θ)
1.    Initialize arrays via Equations 8.9 & 8.10
2.    At each position along the sequence do:
3.      Perform eclipsing via Algorithm 8.1
4.      Apply signal sensors at current location
5.      If a putative signal s_i is detected then:
6.        Link s_i back to optimal predecessors
7.                                    via Eq. 8.14
8.        Append s_i to appropriate signal queues
9.    Form the optimal parse φ* via Algorithm 8.2
10.   Convert φ* to a set of gene predictions
```

run time: $\mathbb{O}(L \times |Q|)$

(assuming a sparse GHMM topology)

Duke UNIVERSITY

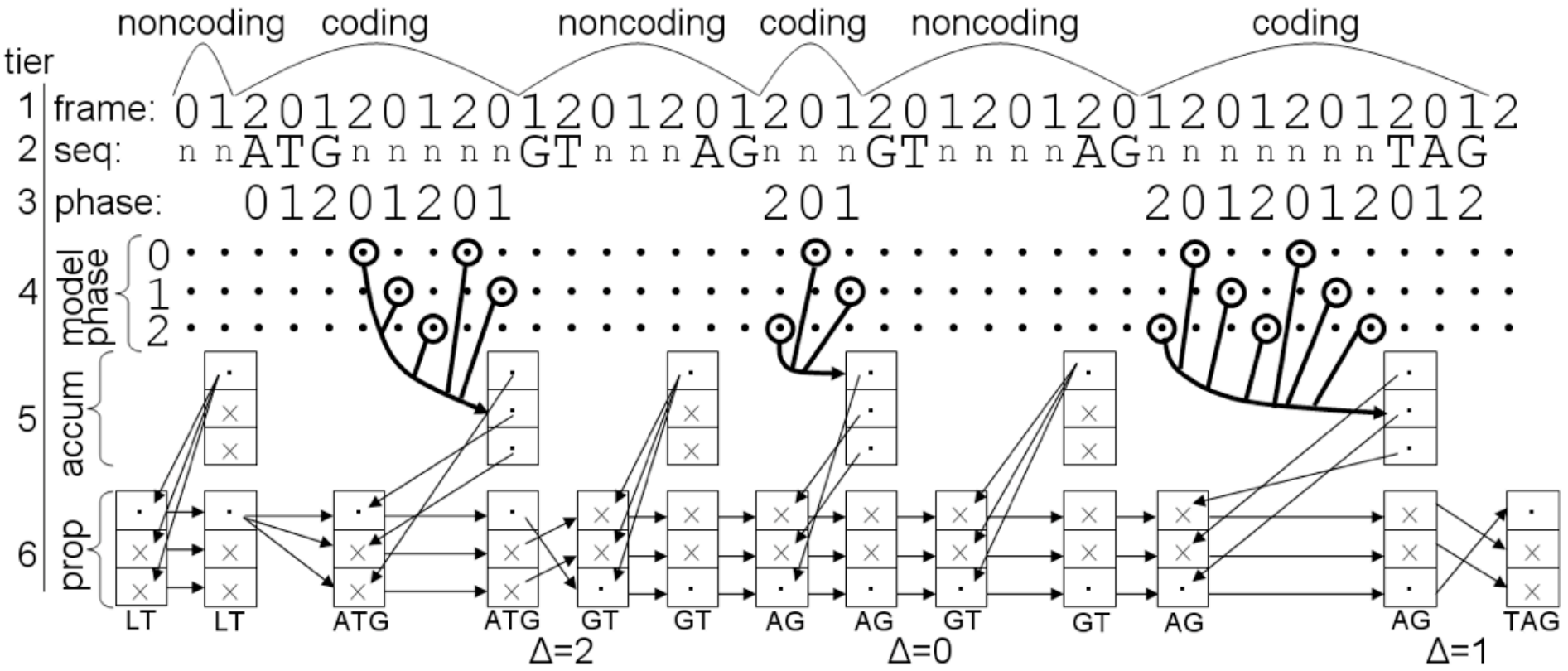
Traceback for GHMMs

Algorithm 8.2 Reconstruction of the optimal parse by tracing back through trellis links. Parameters are the selected right-terminus signal `s` and its chosen phase ω. Returns a stack of signals constituting the optimal parse, with the top signal at the beginning of the parse and the bottom signal at the end. `exon_length(p,s)` denotes the number of coding nucleotides between signals `p` and `s`.

```
procedure traceback(s,ω)
1.     stack K;
2.     push K,s;
3.     while ¬left_terminus(s) do
4.       p←pred(s,ω);
5.       push K,p;
6.       if type(p)∈{ATG,TAG} then ω←0;
7.       elsif type(p)=AG then
8.         ω←(ω-exon_length(p,s))mod3;
9.       s←p;
10.    return K;
```

Duke UNIVERSITY

DSP Decoding



DSP = Dynamic Score Propagation: we incrementally propagate the scores of all possible partial parses up to the current point in the sequence, during a single left-to-right pass over the sequence.

Ref: Majoros WM, Pertea M, Delcher AL, Salzberg SL (2005) Efficient decoding algorithms for generalized hidden Markov model gene finders. *BMC Bioinformatics* 6:16.

Duke UNIVERSITY

DSP Decoding

Algorithm 8.4 Overview of the DSP decoding algorithm. See text for details.

```
procedure DSP(S, θ)
1.    At each position along the sequence do:
2.      Evaluate all content sensors
3.      Update accumulators with content scores
4.      Perform eclipsing via Algorithm 8.1
5.      Allow mature signals to graduate from
6.                              their holding queues
7.      Apply signal sensors at current location
8.      If a putative signal s_i is detected then:
9.        Link s_i back to optimal predecessors
10.       Propagate appropriate queue elements
11.       Append s_i to appropriate holding queues
12.   Form the optimal parse φ* via Algorithm 8.2
13.   Convert φ* to a set of gene predictions
```

run time: $\mathbb{O}(L \times |Q|)$

(assuming a sparse GHMM topology)

PSA vs. DSP Decoding

Theorem 8.2 *(Equivalence of PSA and DSP) Let S be a sequence and θ a set of model parameters for a GHMM. Then given parses* $\phi^*_{PSA}=PSA(S,\theta)$ *and* $\phi^*_{DSP}=DSP(S,\theta)$ *selected under the PSA and DSP decoding algorithms, respectively, we have* $\phi^*_{PSA}=\phi^*_{DSP}$.

Space complexity:

PSA: $\mathbb{O}(L\times|Q|)$

DSP: $\mathbb{O}(L+|Q|)$

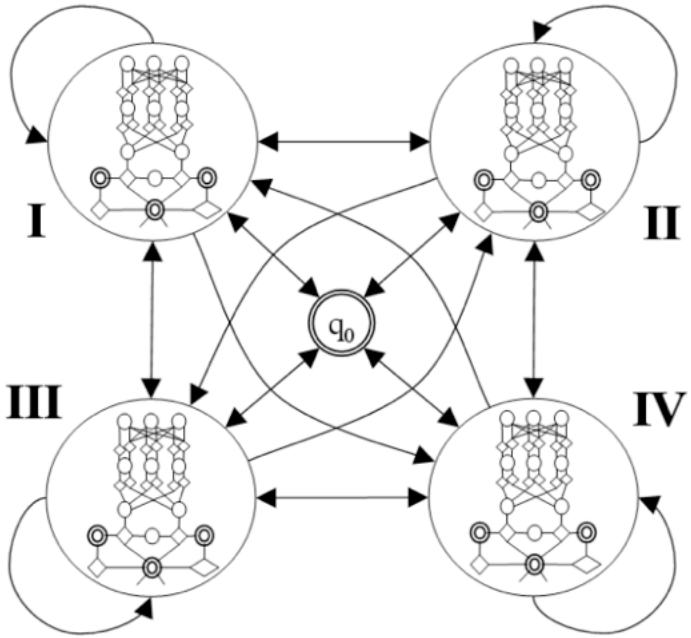
	RAM/state (Mb)	*seconds/ state*
DSP	0.95	2.8
PSA	14	2.8

Table 8.2 *Comparison of memory and time requirements of the PSA vs. DSP decoding algorithms on a sample 922 kb sequence. DSP requires far less memory, while achieving the same speed as PSA. Adapted from from (Majoros et al., 2005a).*

Duke UNIVERSITY

Modeling Isochores

I: {G,C} density $\in [0\%, 43\%]$,
II: {G,C} density $\in (43\%, 51\%]$,
III: {G,C} density $\in (51\%, 57\%]$,
IV: {G,C} density $\in (57\%, 100\%]$.



"inhomogeneous GHMM"

Ref: Allen JE, Majoros WH, Pertea M, Salzberg SL (2006) JIGSAW, GeneZilla, and GlimmerHMM: puzzling out the features of human genes in the ENCODE regions. *Genome Biology 7(Suppl 1):S9*.

Explicit Modeling of Noncoding Lengths

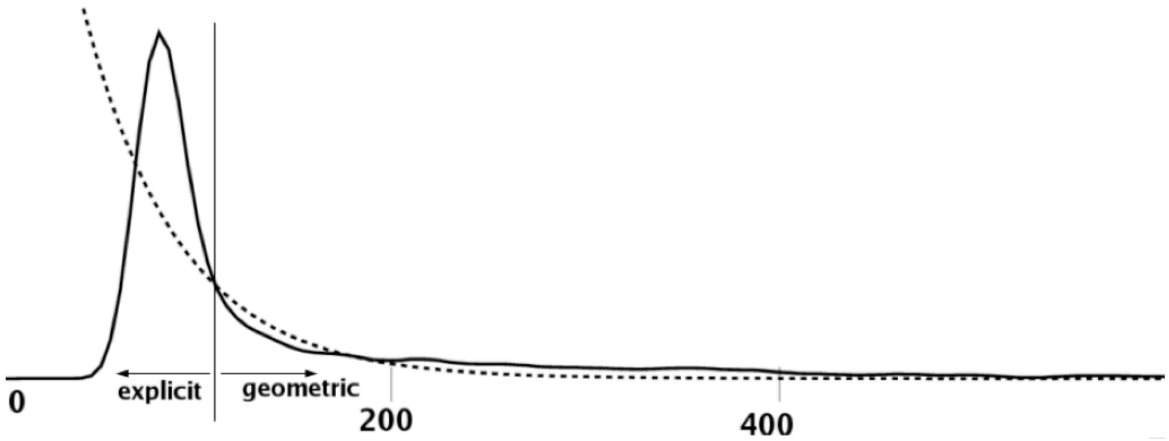


***Figure 8.14**: Explicit length modeling for noncoding features. The observed distribution of Arabidopsis intron lengths (solid line) is shown with a geometric distribution (dashed line). The geometric distribution is a reasonably good fit for lengths>100 bp. Below 100 bp we can use an explicit length distribution, as long as the intron queue stores all donor sites fewer than 100 bp away from the current position.*

still $\mathbb{O}(L \times |Q|)$, but slower by a constant factor

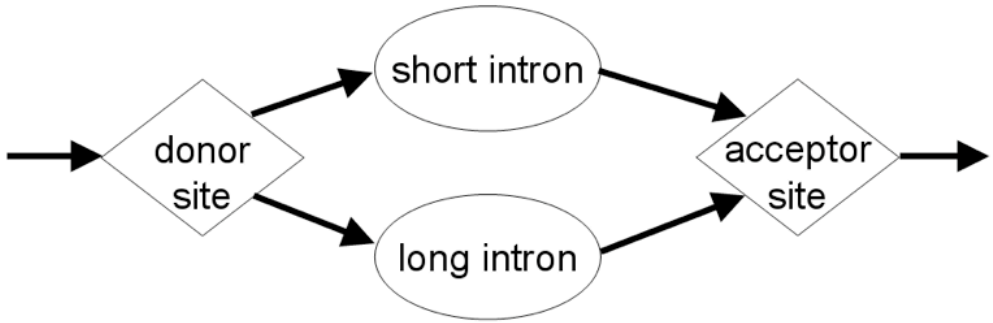


***Figure 8.15**: Explicit length modeling for introns. The short intron model is capable of generating introns of length up to some maximum, L_{short}, whereas the long intron model can generate only introns longer than this.*

Ref: Stanke M, Waack S (2003) Gene prediction with a hidden Markov model and a new intron submodel. *Bioinformatics* 19:II215-II225.

Duke UNIVERSITY

MLE Training for GHMMs

$$\theta_{MLE} = \arg\max_{\theta} \left(\prod_{(S,\phi)\in T} P(S,\phi) \right)$$

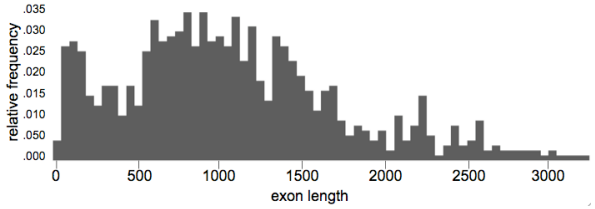
$$= \arg\max_{\theta} \left(\prod_{(S,\phi)\in T} \prod_{y_i \in \phi} P_e(S_i \mid y_i, d_i) P_t(y_i \mid y_{i-1}) P_d(d_i \mid y_i) \right)$$

$$= \arg\max_{\theta} \left(\prod_{(S,\phi)\in T} \prod_{y_i \in \phi} P_t(y_i \mid y_{i-1}) P_d(d_i \mid y_i) \prod_{j=0}^{|S_i|-1} P_e(x_j \mid y_i) \right)$$

estimate via labeled training data, as in HMM

$$a_{i,j} = \frac{A_{i,j}}{\sum_{h=0}^{|Q|-1} A_{i,h}}$$

construct a histogram of observed feature lengths



estimate via labeled training data, as in HMM

$$e_{i,k} = \frac{E_{i,k}}{\sum_{h=0}^{|\alpha|-1} E_{i,h}}$$

Duke UNIVERSITY

Maximum Likelihood vs. Conditional Maximum Likelihood

$$\theta_{MLE} = \arg\max_{\theta} \left(\prod_{(S,\phi)\in T} P(S,\phi) \right)$$

$$= \arg\max_{\theta} \left(\prod_{(S,\phi)\in T} \prod_{y_i \in \phi} P_e(S_i \mid y_i, d_i) P_t(y_i \mid y_{i-1}) P_d(d_i \mid y_i) \right)$$

maximizing P_e, P_t, P_d separately maximizes the entire expression

$$\theta_{CML} = \arg\max_{\theta} \left(\prod_{(S,\phi)\in T} P(\phi \mid S) \right)$$

$$= \arg\max_{\theta} \left(\prod_{(S,\phi)\in T} \frac{\prod_{y_i \in \phi} P_e(S_i \mid y_i, d_i) P_t(y_i \mid y_{i-1}) P_d(d_i \mid y_i)}{P(S)} \right)$$

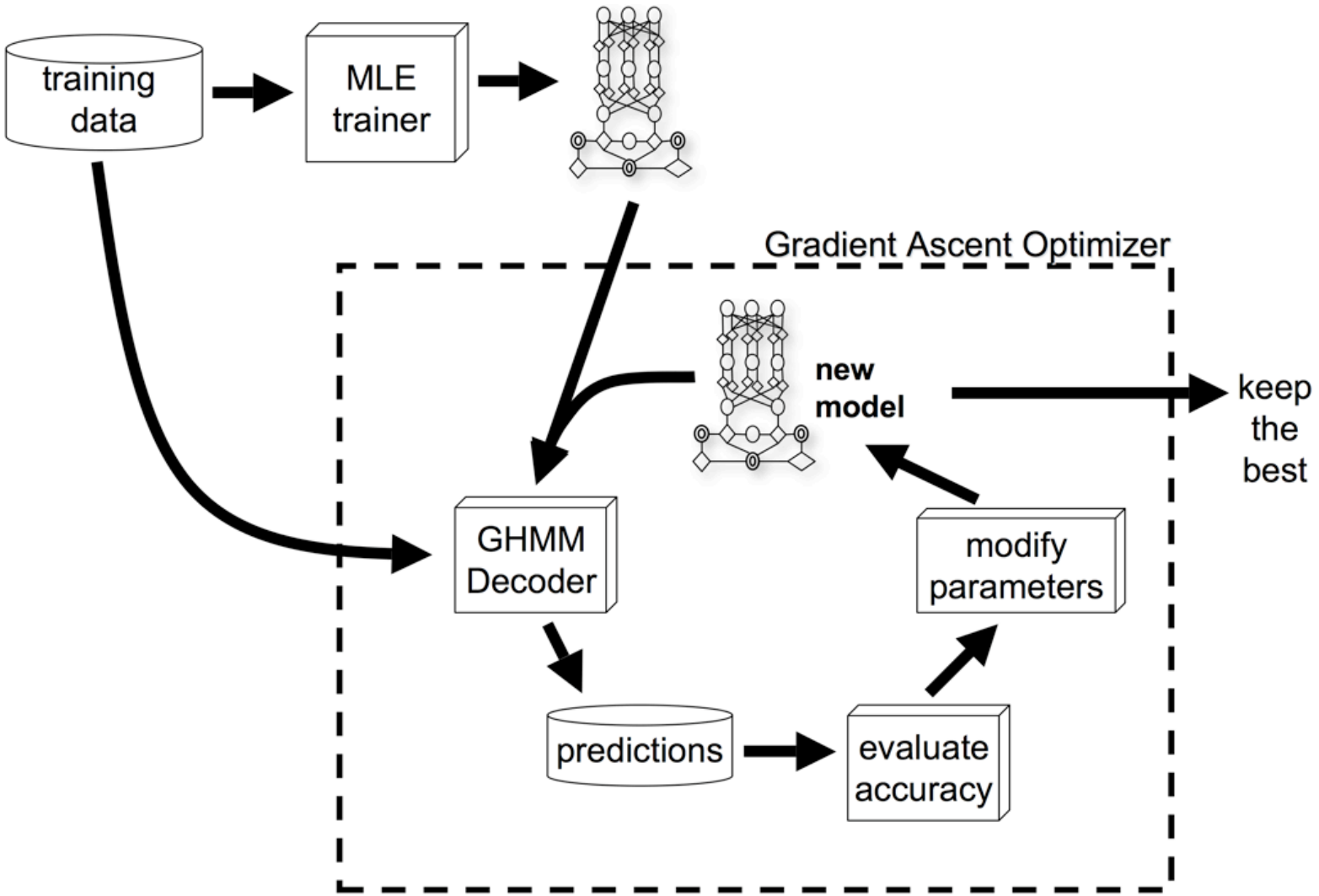
maximizing P_e, P_t, P_d separately DOES NOT maximize the entire expression

$\therefore$ MLE $\neq$ CML

Duke UNIVERSITY

Discriminative Training of GHMMs

$$\theta_{discrim} = \arg\max_{\theta} \left(accuracy\ on\ training\ set \right)$$



Discriminative Training of GHMMs

Ref: Majoros WM, Salzberg SL (2004) An empirical analysis of training protocols for probabilistic gene finders. *BMC Bioinformatics* 5:206.

- mean intron, intergenic, and UTR lengths
- transition probabilities
- exon "optimism" (section 7.3.4)
- sizes of all signal sensor windows
- locations of consensus regions within signal sensor windows
- emission orders for Markov chains and other models
- sensitivity of signal thresholds (when thresholding is used—section 8.3.1)
- number of signal *boosting* iterations to utilize during signal training (section 11.1)
- *skew* and *kurtosis* of exon length distributions (section 2.6)

	nucleotide	*exon*	*gene*
gradient ascent	94%	81%	48%
MLE	90%	71%	33%

***Table 8.3**: Gradient ascent vs. maximum likelihood estimation for GHMM training. Both protocols were applied to 1000 training and 1000 (distinct) test genes from Arabidopsis thaliana. Metrics are, left to right: nucleotide SMC, exon F-measure, and gene sensitivity. Results from (Majoros and Salzberg, 2004).*

Duke UNIVERSITY

Summary

- GHMMs generalize HMMs by allowing each state to emit a *subsequence* rather than just a single symbol
- Whereas HMMs model all feature lengths using a *geometric distribution*, coding features can be modeled using an arbitrary *length distribution* in a GHMM
- Emission models within a GHMM can be any arbitrary probabilistic model ("*submodel abstraction*"), such as a neural network or decision tree
- GHMMs tend to have many *fewer states* => simplicity & modularity
- When used for parsing sequences, HMMs and GHMMs should be trained *discriminatively*, rather than via MLE, to achieve optimal predictive accuracy