隐马氏模型及其在生物信息学中的应用

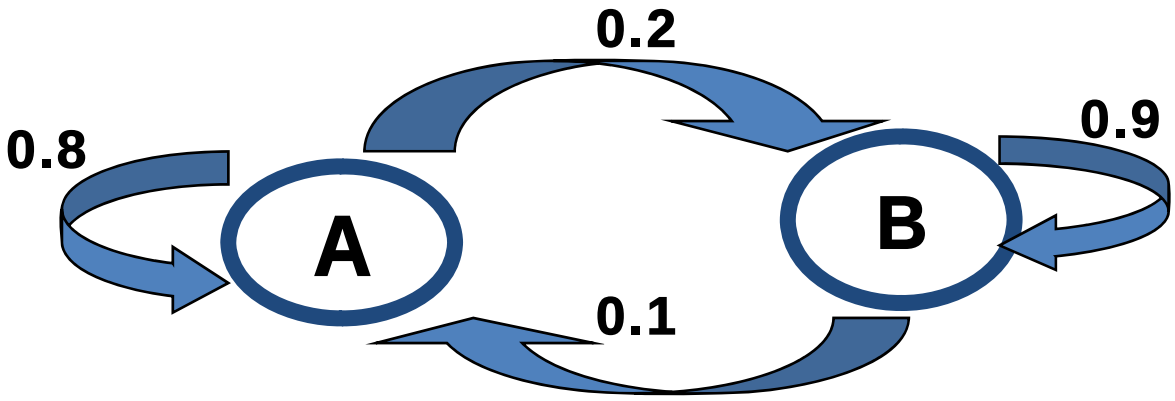
邓明华 dengmh@pku.edu.cn

内容提要

- 什么是隐马氏模型
- 隐马氏模型理论
- 生物信息学应用举例
  - Multiple sequence alignment (MSA)
  - 基因Finding
  - Gene expression clustering
  - CNV detection

韦小宝的骰子
（隐马氏模型）

- 两种骰子，开始以2/5的概率出千。
  - 正常A：以1/6的概率出现每个点
  - 不正常B：5,6出现概率为3/10,其它为1/10
- 出千的随机规律



韦小宝的骰子
（隐马氏模型）

- 观测到其一次投掷结果

$$O = (1, 3, 4, 5, 5, 6, 6, 3, 2, 6)$$

- 问题：请判断韦小宝什么时候出千了？

隐马氏模型理论

- 识别问题－ 已知若干个隐马氏模型及其参数，对一个观测样本，决定它来自哪一个模型 (如例子中的识别问题)。
- 解码问题－由观测样本得到隐状态；
- 学习问题－由观测样本得到参数组 λ；

隐马氏模型的数学模型

- 隐过程为$X=\{X_1,\cdots,X_T\}$
- 观察过程为$Y=\{Y_1,\cdots,Y_T\}$
- 模型参数$\lambda=\{\pi, A, B\}$
  - 初始分布$\pi=(\pi_i)$, $\pi_i=P\{X_1=i\}$
  - 转移矩阵$A=(a_{ij})$, $a_{ij}=P(X_{n+1}=j \mid X_n=i)$
  - 给定某个时间的隐状态的条件下，观测的分布矩阵$B=(b_{il})$, $b_{il}=P(Y_n=l \mid X_n=i)$。

识别问题

- 在已知若干个模型及其参数的情况下,识别问题就是一个对于给定样本进行 Bayesian 判决的问题。
- 判决步骤:
  - 根据参数求出在每一个模型中, 出现给定样本的概率P(Y | λ), 归一化就得到给定样本来自每个模型的概率 P (λ | Y) 。
  - 利用 Bayesian 原理， 就可以得到最好模型猜测。

观测序列的概率计算

$$Pr(Y=y|\lambda)=\sum_{X=x} Pr(Y=y|X=x,\lambda)Pr(X=x|\lambda)$$

$$=\sum_{x=(x_1,\cdots,x_T)} \pi(x_1)b_{x_1}(y_1)a_{x_1x_2}b_{x_2}(y_2)\cdots a_{x_{T-1}x_T}b_{x_T}(y_T)$$

枚举复杂度 $2TN^T$

多项式复杂度算法：前传算法和后传算法

前传概率

$$\alpha_t(i) = Pr(y_1, y_2, \cdots, y_t, x_t = i|\lambda)$$

$$\begin{aligned} a_{t+1}(i) &= Pr(y_1, y_2, \cdots, y_{t+1}, x_{t+1} = i|\lambda) \\ &= \sum_j Pr(y_1, y_2, \cdots, y_{t+1}, x_t = j, x_{t+1} = i|\lambda) \\ &= \sum_j Pr(y_1, y_2, \cdots, y_t, x_t = j|\lambda) Pr(y_{t+1}, x_{t+1} = i|x_t = j, \lambda) \\ &= \sum_j \alpha_t(j) a_{ji} b_i(y_{t+1}) \end{aligned}$$

前传算法 (Forward Algorithm)

- 初始化

$$\alpha_1(i) = \pi_i b_i(y_1), i = 1, 2, \cdots, N.$$

- 迭代

$$\alpha_{t+1}(i) = \sum_{j=1}^{N} \alpha_t(j) a_{ji} b_i(y_{t+1})$$

$$i = 1, 2, \cdots, N, \quad t = 1, 2, \cdots, T-1.$$

- 结果

$$Pr(Y \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i).$$

后传概率

$$\beta_t(i) = Pr(y_{t+1}, y_{t+2}, \cdots, y_T | x_t = i, \lambda)$$

$$\begin{aligned}
\beta_t(i) &= \sum_j Pr(y_{t+1}, y_{t+2}, \cdots, y_T, x_{t+1} = j | x_t = i, \lambda) \\
&= \sum_j Pr(y_{t+2}, \cdots, y_T | x_{t+1} = j, \lambda) Pr(y_{t+1}, x_{t+1} = j | x_t = i, \lambda) \\
&= \sum_j \beta_{t+1}(j) a_{ij} b_j(y_{t+1})
\end{aligned}$$

后传算法(Backward Algorithm)

- 初始化

$$\beta_T(i) = 1, i = 1, 2, \cdots, N;$$

- 迭代

$$\beta_t(i) = \sum_{j=1}^{N} \beta_{t+1}(j) a_{ij} b_j(y_{t+1})$$

$$1 \leq i \leq N, \quad t = T-1, \cdots, 1;$$

- 结果

$$Pr(Y|\lambda) = \sum_{i=1}^{N} \beta_1(i) \pi_i b_i(y_1).$$

解码问题(I)

- 要解决的问题： 给定观测序列$Y=(y_1,y_2,\cdots,y_T)$, 如何给出隐状态序列$X^0=(x^0_1,x^0_2,\cdots,x^0_T)$
- 单点最优

$$\gamma_t(i) = Pr(X_t = i|Y)$$
$$X'_t = \operatorname*{Argmax}_i \gamma_t(i)$$

- 路径最优指：对任意的$X=(x_1,x_2,\cdots,x_T)$ 有

$$Pr(x'_1, x'_2, \cdots, x'_T|y_1, \cdots, y_T)$$
$$\geq Pr(x_1, x_2, \cdots, x_T|y_1, \cdots, y_T)$$

解码问题(II)

- 由Bayesian公式有

$$Pr(x'_1, x'_2, \cdots, x'_T | y_1, \cdots, y_T) = \frac{Pr(x_1, x_2, \cdots, x_T, y_1, \cdots, y_T)}{Pr(y_1, \cdots, y_T)}$$

- 又由于序列 **Y** 给定, 问题等价于找最优的X^0使联合概率$Pr(x_1,x_2,\cdots,x_T,y_1,y_2,\cdots,y_T)$}最大。

最优单点确定

$$\gamma_t(i) = Pr(X_t = i|y_1, y_2, \cdots, y_T, \lambda)$$
$$= \frac{Pr(X_t = i, y_1, \cdots, y_T|\lambda)}{Pr(y_1, \cdots, y_T|\lambda)}$$
$$= \frac{Pr(X_t = i, y_1, y_2, \cdots, y_T|\lambda)}{\sum_i Pr(X_t = i, y_1, y_2, \cdots, y_T|\lambda)}$$
$$= \frac{Pr(y_{t+1}, \cdots, y_T|x_t = i, y_1, \cdots, y_t, \lambda) Pr(x_t = i, y_1, \cdots, y_t|\lambda)}{\sum_i Pr(X_t = i, y_1, y_2, \cdots, y_T|\lambda)}$$
$$= \frac{\alpha_t(i)\beta_t(i)}{\sum_i \alpha_t(i)\beta_t(i)}$$

Viterbi算法(I)

- 算法的思想动态规划的递推算法。
- 递推变量为

$$\delta_t(i) = \max_{x_1,\cdots,x_{t-1}} Pr(x_1,\cdots,x_{t-1},x_t=i,y_1,\cdots,y_t|\lambda)$$

- 我们有递推公式

$$\begin{aligned}\delta_{t+1}(i) &= \max_{x_1,\cdots,x_t} Pr(x_1,\cdots,x_t,x_{t+1}=i,y_1,\cdots,y_{t+1}|\lambda) \\ &= \left(\max_j \delta_t(j)a_{ji}\right) b_i(y_{t+1})\end{aligned}$$

- 以$\psi_t(i)$记录t时刻时使$\delta_t(j)a_{ji}$最大的状态j。

Viterbi算法(II)

- 初始化

$$\delta_1(i) = \pi_i b_i(y_1),$$
$$\psi_1(i) = 0, \quad i = 1, 2, \cdots, N.$$

- 迭代

$$\delta_t(j) = \left(\max_{1 \le i \le N} \delta_{t-1}(i) a_{ij} \right) b_j(y_t)$$
$$\psi_t(j) = \operatorname*{Argmax}_{1 \le i \le N} \left(\delta_{t-1}(i) a_{ij} \right)$$
$$t = 2, \cdots, T; \; j = 1, \cdots, N.$$

Viterbi算法(III)

- 终止

$$p^\star = \max_{1\le i\le N} \delta_T(i)$$
$$x_T^\star = \operatorname*{Argmax}_{1\le i\le N} \delta_T(i)$$

- 后推

$$x_t^\star = \psi_{t+1}(x_{t+1}^\star)$$
$$t = T-1, T-2, \cdots, 1.$$

Viterbi算法实例(I)

- 转移概率以及初概率

	A	B
A	0.8	0.2
B	0.1	0.9
初概率	0.6	0.4

- 条件概率(Emission Probability)

	V_1	V_2	V_3	V_4	V_5	V_6
A	1/6	1/6	1/6	1/6	1/6	1/6
B	0.1	0.1	0.1	0.1	0.3	0.3

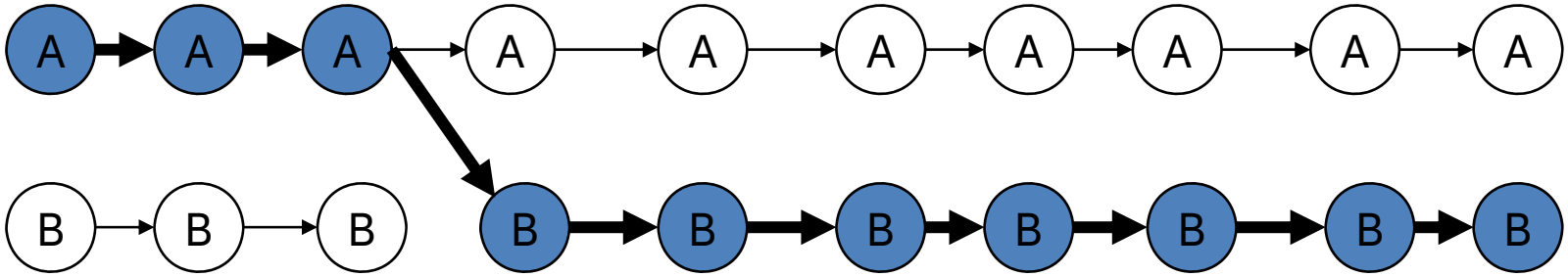
Viterbi算法实例(II)

	y_t	$\delta_t(A)$	$\psi_t(A)$	$\delta_t(B)$	$\psi_t(B)$
t=1	1	1.000×10^{-1}	-	4.000×10^{-2}	-
t=2	3	1.333×10^{-2}	A	3.600×10^{-3}	B
t=3	4	1.778×10^{-3}	A	3.240×10^{-4}	B
t=4	5	3.370×10^{-4}	A	1.067×10^{-4}	A
t=5	5	3.161×10^{-4}	A	2.880×10^{-5}	B
t=6	6	4.214×10^{-6}	A	7.776×10^{-6}	B
t=7	6	5.619×10^{-7}	A	2.100×10^{-6}	B
t=8	3	7.492×10^{-8}	A	1.890×10^{-7}	B
t=9	2	9.989×10^{-9}	A	1.701×10^{-8}	B
t=10	6	1.322×10^{-9}	A	4.592×10^{-9}	B

Viterbi算法实例(III)

观测序列为：

1 3 4 5 5 6 6 3 2 6



解码出来的状态序列为：

A A A B B B B B B B

HMM学习问题

- 学习问题:
  - 就是由观测估计模型参数.
- 学习的两种情况:
  - 观测链相应的状态链已知;
  - 观测链相应的状态链未知。

学习原则

- 极大似然估计(MLE)

$$\widehat{\lambda} = \underset{\lambda}{\text{Argmax}}\, Pr(y_1, \cdots, y_T|\lambda)$$

- 状态链已知时

$$Pr(y_1, \cdots, y_T, X_1, \cdots, X_T|\lambda)$$

- 状态链未知时

$$\sum_{(X_1,\cdots,X_T)} Pr(y_1, \cdots, y_T, X_1, \cdots, X_T|\lambda)$$

状态链已知时的MLE

$$
\begin{aligned}
& Pr(y_1, \cdots, y_T; X_1, \cdots, X_T | \lambda) \\
&= \pi_{X_1} \prod_{t=1}^{T-1} a_{X_t X_{t+1}} \prod_{t=1}^{T} b_{X_t}(y_t) \\
&= \prod_i \pi_i^{1_i(X_1)} \prod_{i,j} a_{ij}^{\sum_{t=1}^{T-1} 1_i(X_t) 1_j(X_{t+1})} \prod_{i,l} b_i(l)^{\sum_{t=1}^{T} 1_i(X_t) 1_l(y_t)} \\
&= \prod_i \pi_i^{C_i} \prod_{i,j} a_{ij}^{A_{ij}} \prod_{i,l} b_i(l)^{B_{il}}
\end{aligned}
$$

简单优化问题

$$\text{Max:} \sum_k z_k \log x_k$$

$$\text{subject to:} \sum_k x_k = 1$$

$$\text{Estimation: } x_i = \frac{z_i}{\sum_k z_k}, i = 1, \cdots, N.$$

参数估计(状态已知)

- 把从状态 *i* 到下一个时刻转移为状态 *j* 的频数记为 A_{ij}，可估计 a_{ij}为

$$\widehat{a}_{ij} = \frac{A_{ij}}{\sum_{j=1}^{N} A_{ij}}$$

- 同样记从状态 i 到同一时刻的观测转移的频数为 B_{il}，则可估计b_{il}为

$$\widehat{b}_{il} = \frac{B_{il}}{\sum_{j=1}^{M} B_{ij}}$$

参数估计评价

- 隐Markov模型的状态链要有充分长的样本(大数定律，以频率代替概率)。

- 不幸的是状态链往往并不知道, 而只是可以得到估计, 不修正地使用频率估计会增加误差, 且这种估计不稳健。

参数估计的EM思想

- 当状态链未知时，由于似然函数的计算中包含了对所有可能的状态链的求和，计算过大，在实际中是不可能被采用的．为此，人们采取折衷的方案，构造一个递推算法，使之能相当合理地给出模型参数的粗略估计．
- 其核心思想是：在当前参数下，用期望值当成频数“数数”，并用“频率”估计概率。这实际上是一种EM迭代算法思想。

EM算法

- 实际上是 ‘**E**’ (期望) 与 ‘**M**’ (最大化) 两个步骤合起来构成的算法，称为**EM算法**.
- EM算法是针对测量数据不完全时, 求参数的一种近似于最大似然估计的统计方法。
- HMM 模型参数的估计, 是EM算法的一个最常见且极有用的一种典型例子.

EM算法基本框架

- 观测数据Y
- 缺失数据 X
- 完全数据 Z=(Y, X).
- E-Step (取期望).

$$\widehat{Z} = E(Z \mid Y, \theta^{(t-1)})$$

- M-step (取极大).

$$\theta^{(t)} = \underset{\theta}{\operatorname{Argmax}}\, L(\theta \mid \widehat{Z}, \theta^{(t-1)})$$

期望频数(状态未知)

$$\xi_t(i,j) = Pr(X_t = i, X_{t+1} = j | y_1, \cdots, y_T, \lambda)$$

$$= \frac{Pr(X_t = i, X_{t+1} = j, y_1, \cdots, y_T | \lambda)}{Pr(y_1, \cdots, y_T | \lambda)}$$

$$= \frac{Pr(X_t = i, X_{t+1} = j, y_1, \cdots, y_T | \lambda)}{\sum_i \sum_j Pr(X_t = i, X_{t+1} = j, y_1, \cdots, y_T | \lambda)}$$

$$= \frac{Pr(x_{t+1} = j, y_{t+1}, \cdots, y_t | x_t, y_1, \cdots, y_t, \lambda) Pr(x_t = i, y_1, \cdots, y_t | \lambda)}{Pr(y_1, \cdots, y_T) | \lambda}$$

$$= \frac{\alpha_t(i) a_{ij} b_j(y_{t+1}) \beta_{t+1}(j)}{\sum_i \sum_j a_{ij} b_j(y_{t+1}) \beta_{t+1}(j)}$$

$$= \frac{\alpha_t(i) a_{ij} b_j(y_{t+1}) \beta_{t+1}(j)}{\sum_i \alpha_t(i) \beta_t(i)}$$

期望频数(状态未知)

$$\begin{aligned}\gamma_t(i) &= Pr(x_t = i|y_1, \cdots, y_T, \lambda) \\ &= \sum_{j=1}^{N} Pr(x_t = i, x_{t+1} = j|y_1, \cdots, y_T, \lambda) \\ &= \sum_{j=1}^{N} \xi_t(i, j)\end{aligned}$$

Baum-Welch公式

$$
\begin{cases}
\overline{\pi}_i = r_1(i) \\
\overline{a}_{ij} = \dfrac{\sum\limits_{t=1}^{T-1} \xi_t(i,j)}{\sum\limits_{t=1}^{T-1} \gamma_t(i)} \\
\overline{b}_j(l) = \dfrac{\sum\limits_{t=1}^{T} \gamma_t(j)\delta(y_t, V_l)}{\sum\limits_{t=1}^{T} \sum\limits_{j} \gamma_t(j)}
\end{cases}
$$

Baum-Welch公式的推导(1)

- 我们定义一个描述模型“趋势”的量，以衡量参数估计前后的概率分布的差异。相对熵是最好的选择

$$Q(\overline{\lambda}|\lambda) = \sum_{X} P(X, Y|\lambda) \log Pr(X, Y|\overline{\lambda})$$

Baum-Welch公式的推导(2)

$$
\begin{aligned}
& Q(\overline{\lambda}|\lambda) - Q(\lambda|\lambda) \\
&= \sum_X Pr(X,Y|\lambda) \log \frac{Pr(X,Y|\overline{\lambda})}{Pr(X,Y|\lambda)} \\
&\leq \sum_X Pr(X,Y|\lambda)(\frac{Pr(X,Y|\overline{\lambda})}{Pr(X,Y|\lambda)} - 1) \\
&= Pr(Y|\overline{\lambda}) - Pr(Y|\lambda)
\end{aligned}
$$

- 说明只要依据Q函数增大来更新参数，就能够使得似然函数朝变大的方向改进，而且一个观测Y对应了一次改进。

Baum-Welch公式的推导(3)

- 于是要想得到参数修改的递推公式, 只要把模型λ_m修改为更好的模型λ_{m+1}, 即只需将它取得使下式成立,

$$\lambda_{m+1} = \underset{\lambda}{\mathrm{Argmax}}\, Q(\lambda|\lambda_m)$$

Baum-Welch公式的推导(4)

- 写出Q函数的表达式，我们有

$$
\begin{aligned}
Q(\overline{\lambda}|\lambda) &= \sum_X Pr(X,Y|\lambda)\log Pr(X,Y|\overline{\lambda}) \\
&= \sum_X Pr(X,Y|\lambda)\left(\log \overline{\pi}_{x_1} + \sum_{t=1}^{T-1}\log \overline{a}_{x_t x_{t+1}} + \sum_{t=1}^{T}\log \overline{b}_{x_t}(y_t)\right) \\
&= \sum_i \left(\sum_X Pr(X,Y|\lambda)1_i(x_1)\right)\log \overline{\pi}_i \\
&+ \sum_i\sum_j \left(\sum_{t=1}^{T-1}\sum_X Pr(X,Y|\lambda)1_i(x_t)1_j(x_{t+1})\right)\log \overline{a}_{ij} \\
&+ \sum_i\sum_l \left(\sum_{t=1}^{T}\sum_X Pr(X,Y|\lambda)1_i(x_t)1_l(y_t)\right)\log \overline{b}_i(l)
\end{aligned}
$$

Baum-Welch公式的推导(5)

$$= \sum_i (Pr(x_1 = i, Y|\lambda)) \log \overline{\pi}_i$$

$$+ \sum_i \sum_j \left(\sum_{t=1}^{T-1} Pr(x_t = i, x_{t+1} = j, Y|\lambda) \right) \log \overline{a}_{ij}$$

$$+ \sum_i \sum_l \left(\sum_{t=1}^{T} \sum_X Pr(x_t = i, Y|\lambda) 1_l(y_t) \right) \log \overline{b}_i(l)$$

- 对每个变量可以分别取最大值。

初概率的重估计

$$\text{Max: } Q_\pi(\overline{\pi}|\lambda) = \sum_i P(x_1 = i, Y|\lambda) \log \overline{\pi_i}$$

$$\text{Subject to: } \sum_i \overline{\pi_i} = 1$$

$$\pi_i^{(m+1)} = \frac{P(x_1 = i, Y|\lambda^{(m)})}{P(Y|\lambda^{(m)})} = \gamma_1(i)$$

转移概率重估计

$$\text{Max: } Q_{a_i}(\overline{a_i}|\lambda) = \sum_j \Big(\sum_{t=1}^{T-1} P(x_t = i, x_{t+1} = j, Y|\lambda) \Big) \log \overline{a_{ij}}$$

$$\text{Subject to: } \sum_j \overline{a_{ij}} = 1$$

$$a_{ij}^{(m+1)} = \frac{\sum_{t=1}^{T-1} P(x_t = i, x_{t+1} = j, Y|\lambda^{(m)})}{\sum_{t=1}^{T-1} P(x_t = i, Y|\lambda^{(m)})}$$

$$= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

观测概率重估计

$$\text{Max: } Q_{b_i}(\overline{b_i}|\lambda) = \sum_k \Big(\sum_{t=1}^{T} P(x_t = i, Y|\lambda) 1_k(y_t) \Big) \log \overline{b_{ik}}$$

$$\text{Subject to: } \sum_k \overline{b_{ik}} = 1$$

$$b_{ik}^{(m+1)} = \frac{\sum_{t=1}^{T} \big(P(x_t = i, Y|\lambda^{(m)}) \delta(y_t, k) \big)}{\sum_{t=1}^{T} P(x_t = i, Y|\lambda^{(m)})}$$

$$= \frac{\sum_{t=1}^{T} \gamma_t(i) \delta(y_t, k)}{\sum_{t=1}^{T} \gamma_t(i)}$$

几点说明

- 更详细的内容可参见《应用随机过程》 第十章 HMM.
- 在上面所述的算法中，初始值 λ_0 的设置会直接影响到估计的好坏．为此，常用的一种方法是, 根据先验知识设置一条较长的“标准虚拟” 状态链, 再用前面讲的已知观测链相应的状态链的情况下参数估计的方法, 得到一个对参数的粗估计，并以它作为 λ_0 的取值。

隐马氏模型的生物信息学应用

- Multiple sequence alignment (MSA)
- 基因Finding
- Gene expression clustering
- CNV detection

应用I: Multiple Sequence Alignment

- Krogh A. et al, Hidden Markov model in computational biology. Applications to protein modeling. Journal of Molecular Biology 235:1501-1531, 1994.
- Eddy SR. Hidden Markov models. Current Opinion in Structure Biology 6:361-365, 1996.
- Eddy SR. Profile hidden Markov models. Bioinformatis 14:755-763, 1998.

This part is modified from Colin Cherry's slides download from *webdocs.cs.ualberta.ca/~**colin**c/cmput606/**HMM**Oct25.**ppt***

Methods for Characterizing a Protein Family

- Objective: Given a number of related sequences, encapsulate what they have in common in such a way that we can recognize other members of the family.
- Some standard methods for characterization:
  - Multiple Alignments
  - Regular Expressions
  - Consensus Sequences
  - Hidden Markov Models

A Characterization Example

A C A - - - A T G

T C A A C T A T C

A C A C - - A G C

A G A - - - A T C

A C C G - - A T C

Example borrowed from Salzberg, 1998

How could we characterize this (hypothetical) family of nucleotide sequences?

- Keep the Multiple Alignment
- Try a regular expression

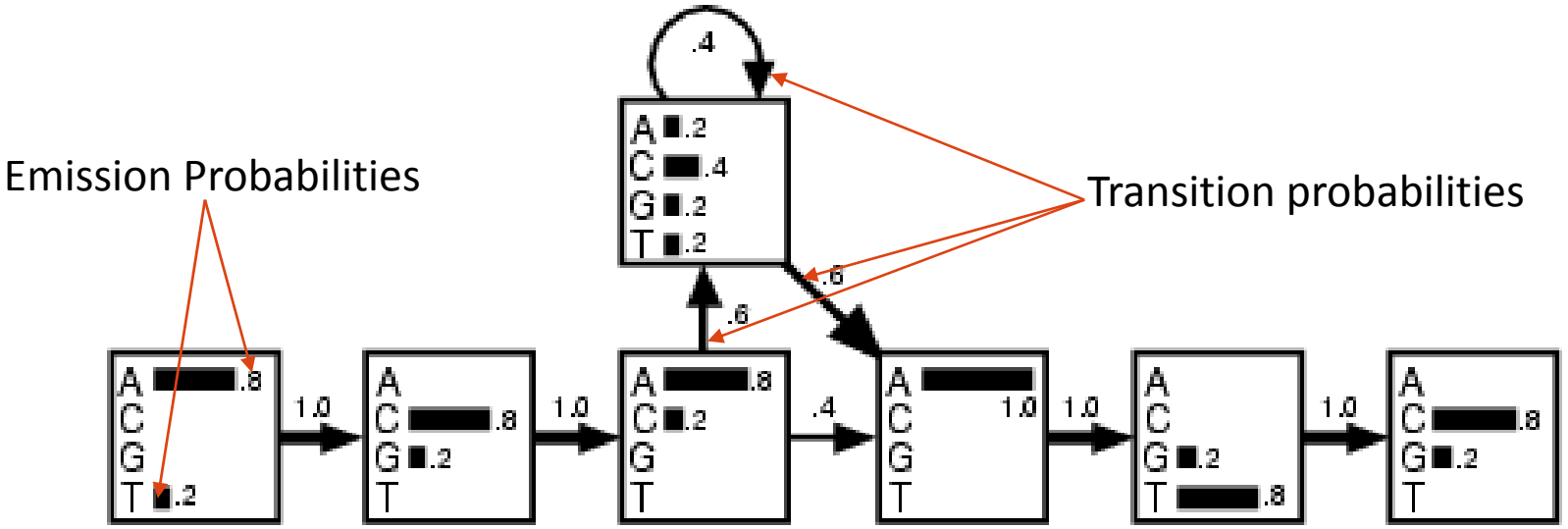
  [AT] [CG] [AC] [ACTG]* A [TG] [GC]

  - But what about?
    - T G C T - - A G G *vrs*
    - A C A C - - A T C
- Try a consensus sequence:

  A C A - - - A T C

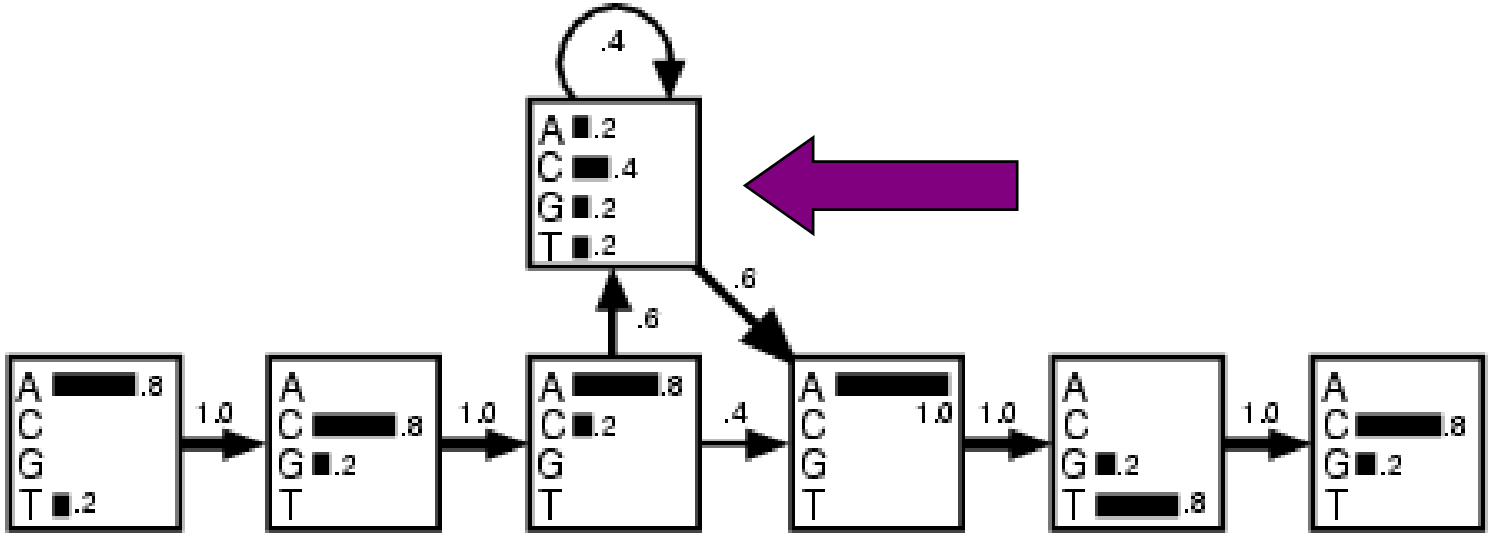
  - Depends on distance measure

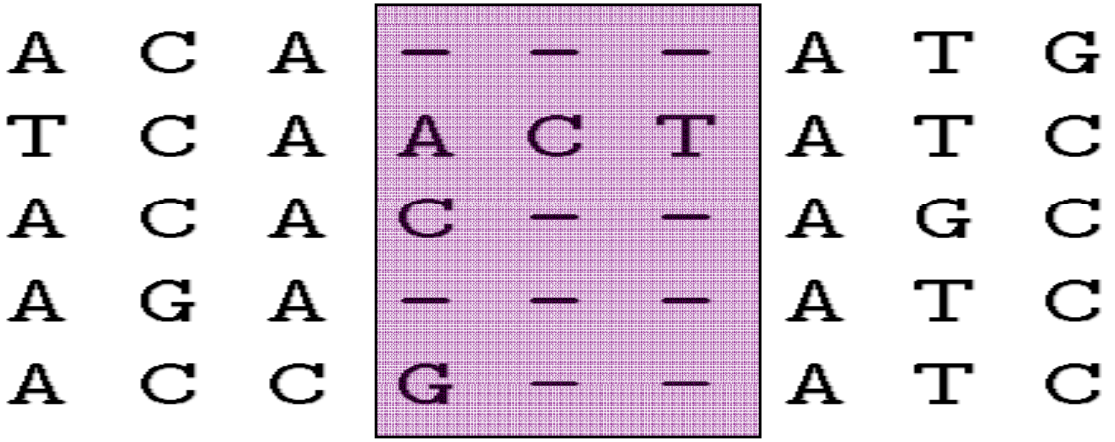
HMMs



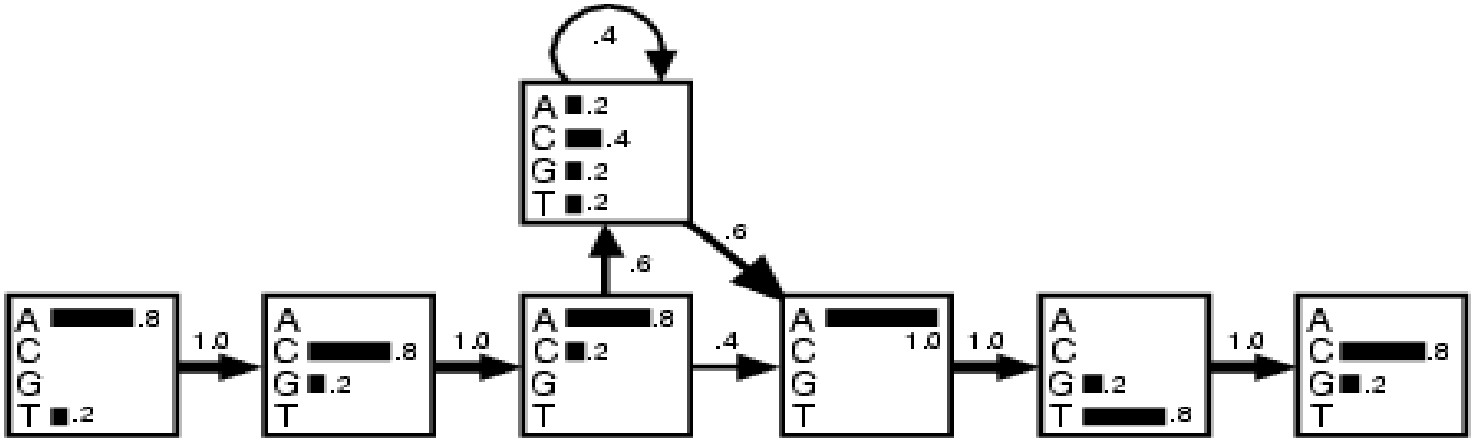
A C A — — — A T G
T C A A C T A T C
A C A C — — A G C
A G A — — — A T C
A C C G — — A T C

Insert (Loop) States





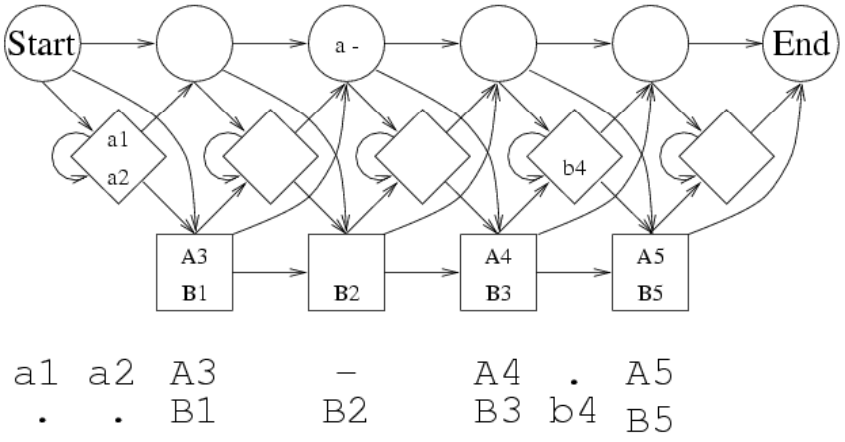
Scoring our simple HMM



- #1 - “T G C T - - A G G” *vrs:* #2 - “A C A C - - A T C”
  - Regular Expression ([AT] [CG] [AC] [ACTG]* A [TG] [GC]):
    - #1 = Member #2: Member
  - HMM:
    - #1 = Score of 0.0023% #2 Score of 4.7% (Probability)
    - #1 = Score of -0.97 #2 Score of 6.7 (Log odds)

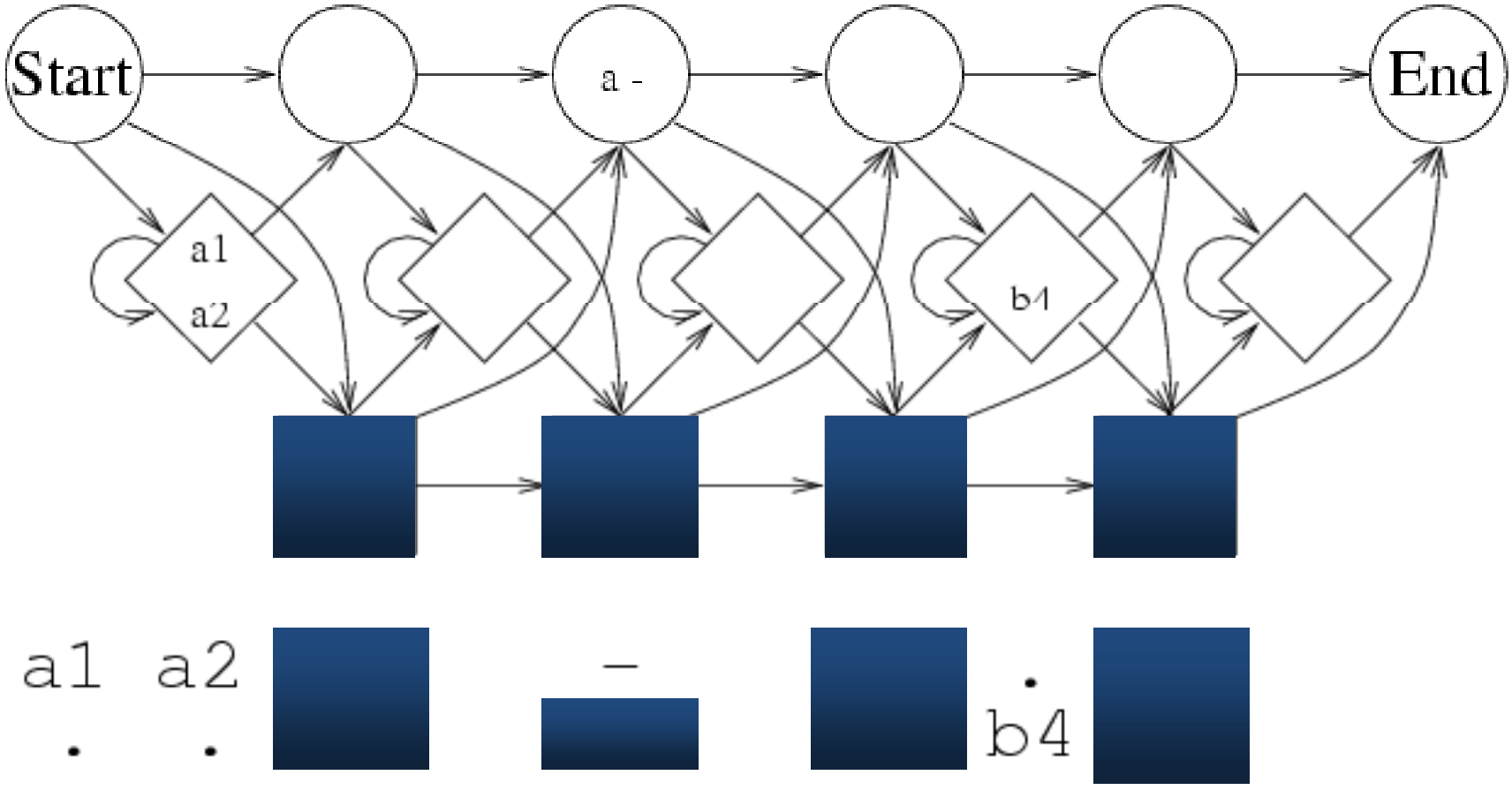
Standard Profile HMM Architecture

- Three types of states:
  - Match
  - Insert
  - Delete
- One delete and one match per position in model
- One insert per transition in model
- Start and end “dummy” states



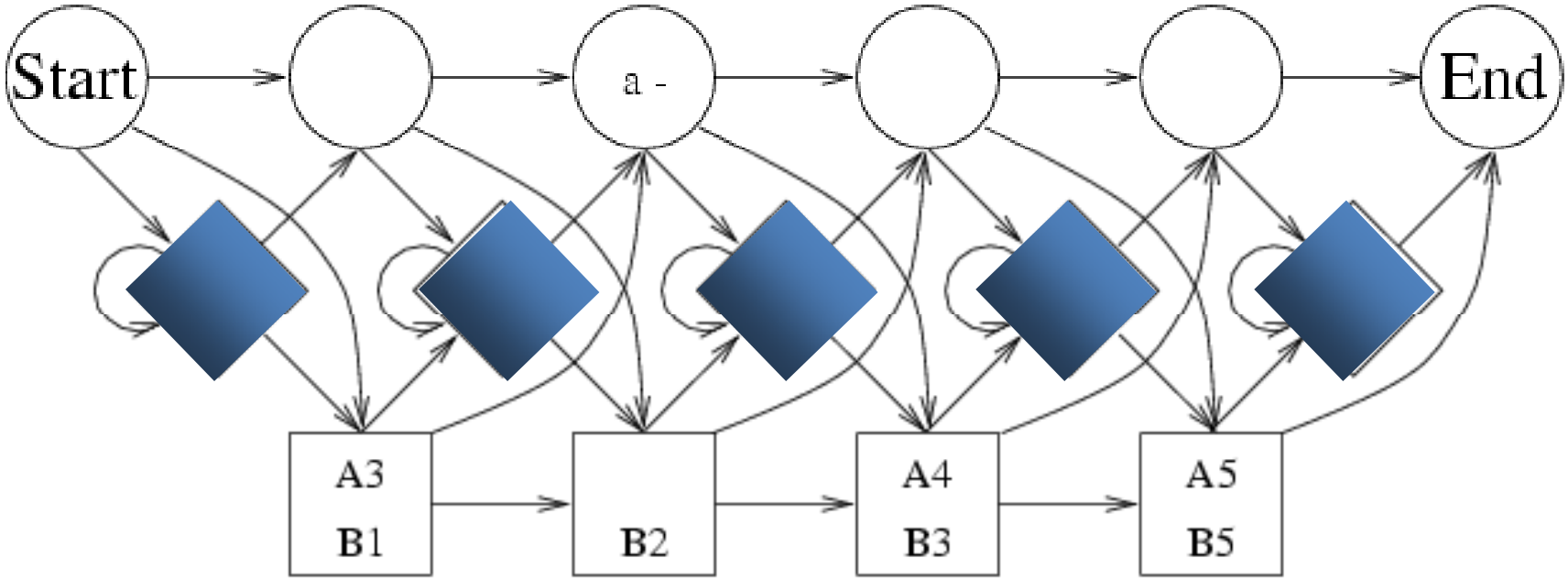
Example borrowed from Cline, 1999

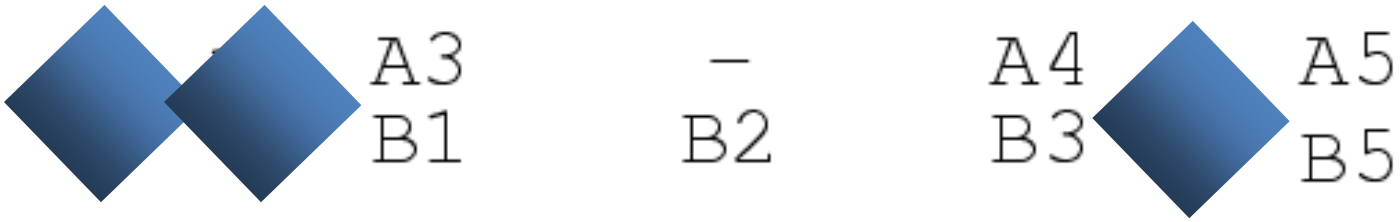
Match States



Example borrowed from Cline, 1999

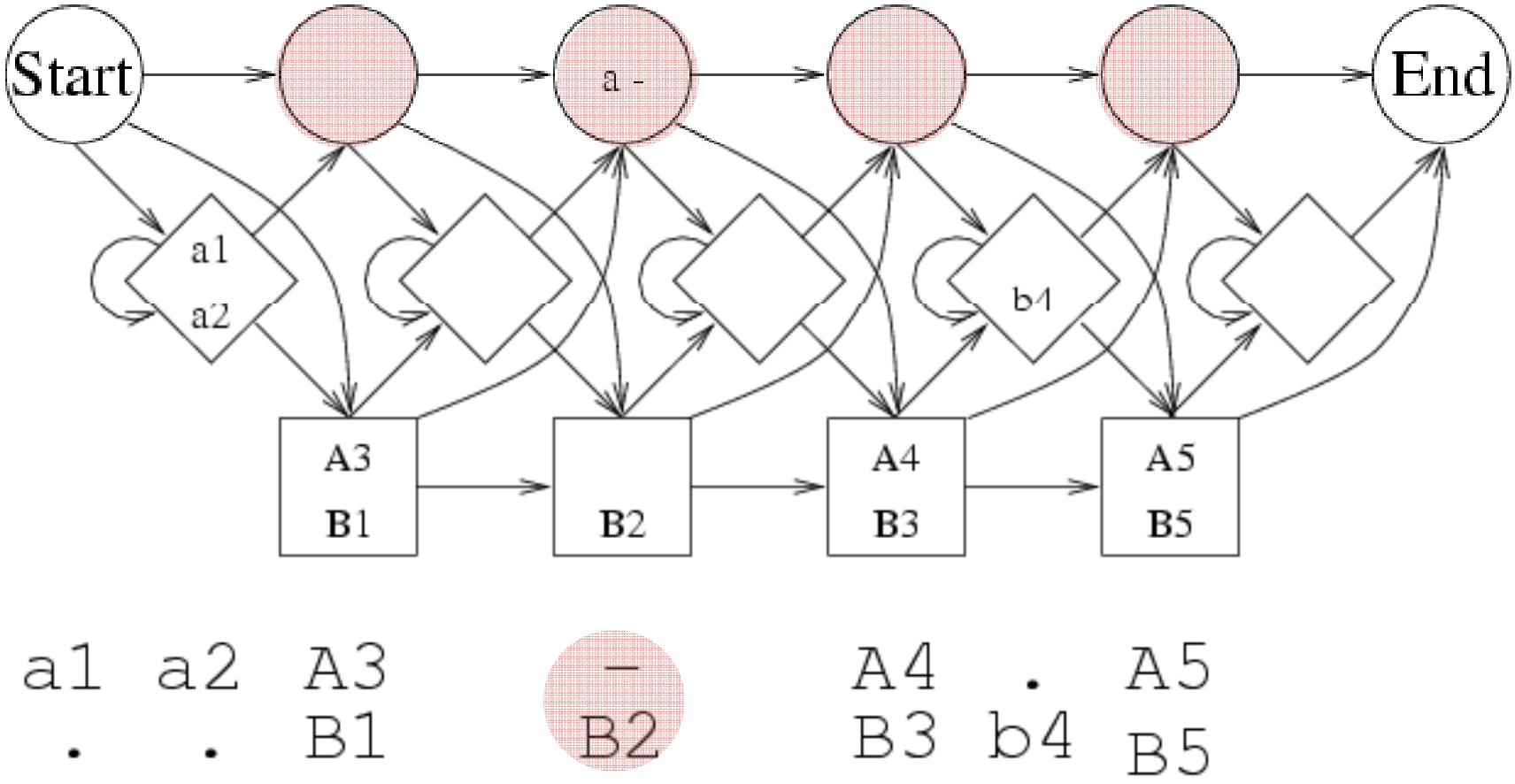
Insert States





Example borrowed from Cline, 1999

Delete States



Example borrowed from Cline, 1999

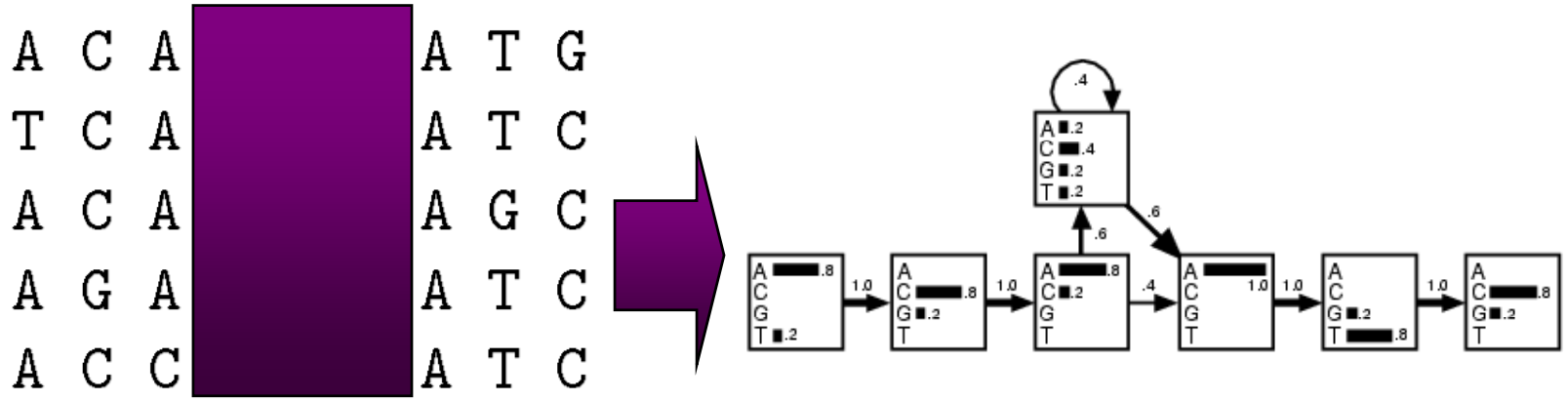
Aligning and Training HMMs

- Training from a Multiple Alignment
- Aligning a sequence to a model
  - Can be used to create an alignment
  - Can be used to score a sequence
  - Can be used to interpret a sequence
- Training from unaligned sequences

Training from an existing alignment

- This process what we've been seeing up to this point.
  - Start with a predetermined number of states in your HMM.
  - For each position in the model, assign a column in the multiple alignment that is relatively conserved.
  - Emission probabilities are set according to amino acid counts in columns.
  - Transition probabilities are set according to how many sequences make use of a given delete or insert state.

Remember the simple example

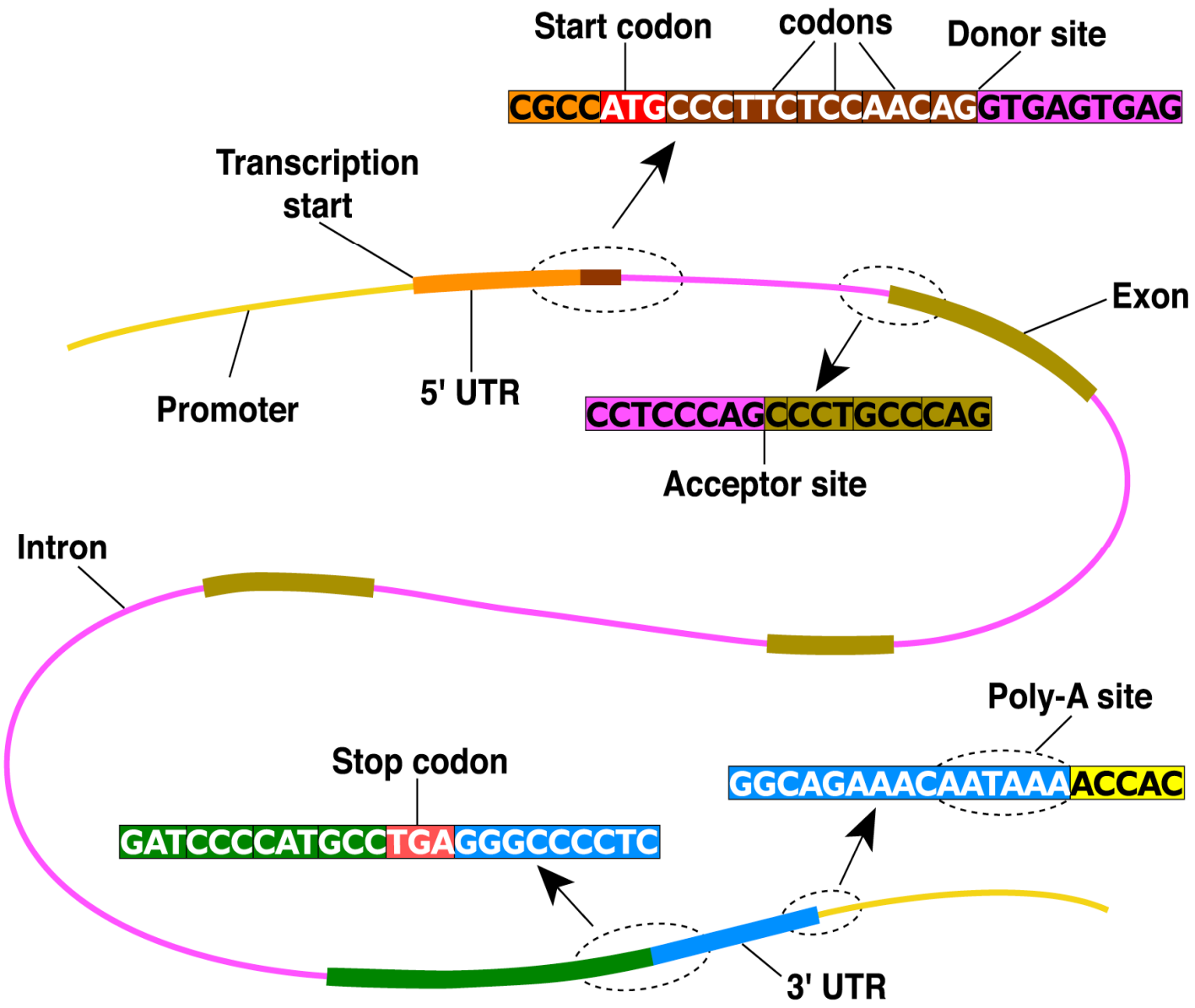


- Chose six positions in model.
- Highlighted area was selected to be modeled by an insert due to variability.
- Can also do neat tricks for picking length of model, such as model pruning.

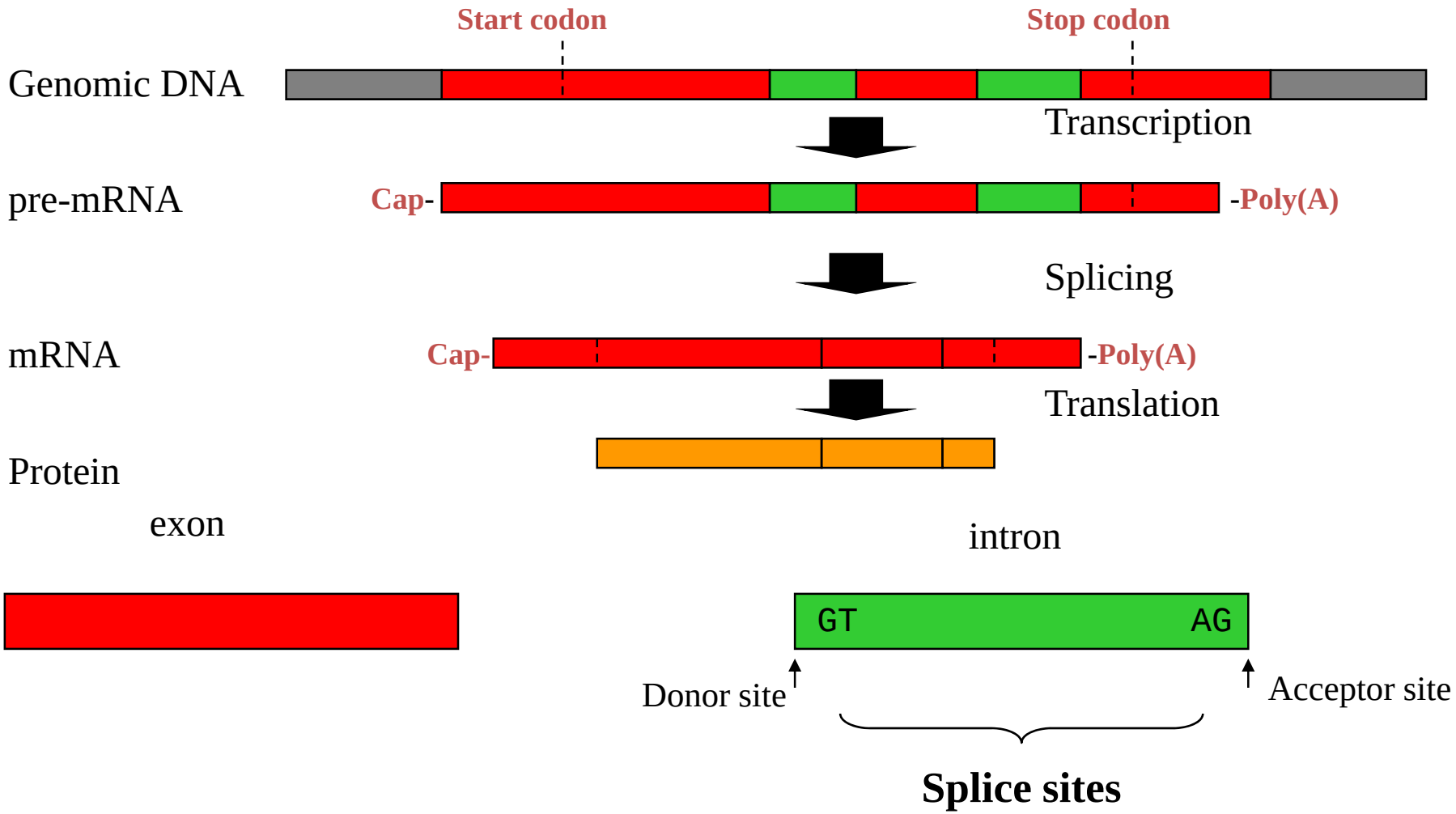
应用II：Gene Finding

- Burge, C. and Karlin, S. (1997) Prediction of complete gene structures in human genomic DNA. *J. Mol. Biol.* **268,** 78-94
- Burge, C. B. and Karlin, S. (1998) Finding the genes in genomic DNA. *Curr. Opin. Struct. Biol.* **8,** 346-354.

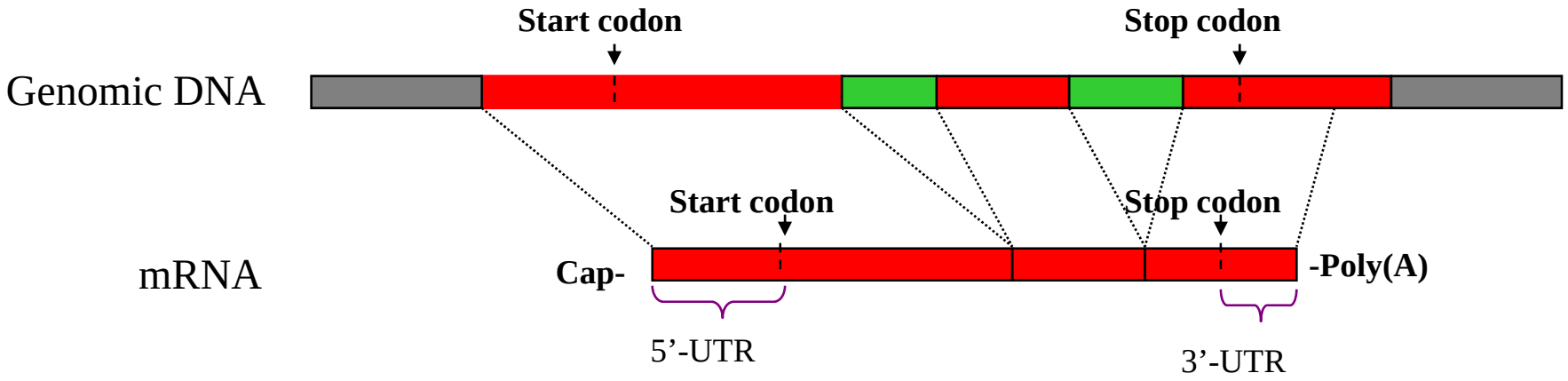
*This part is modified from slides download from www.cs.ubc.ca/~rogic/**GeneFinding.ppt***


Start codon
codons
Donor site
CGCCATGCCCTTCTCCAACAGGTGAGTGAG
Transcription start
Exon
Promoter
5' UTR
CCTCCCAGCCCTGCCCAG
Acceptor site
Intron
Poly-A site
Stop codon
GGCAGAAACAATAAAACCAC
GATCCCCATGCCTGAGGGCCCCTC
3' UTR

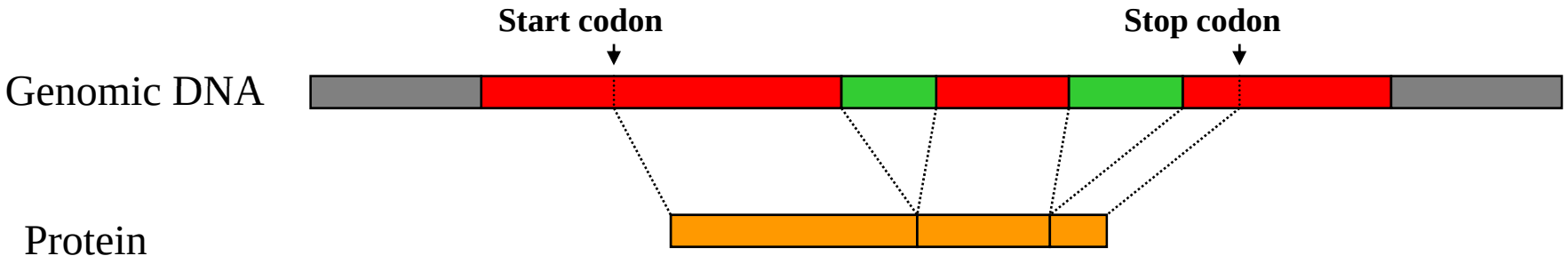
Signals: Pre-mRNA Splicing



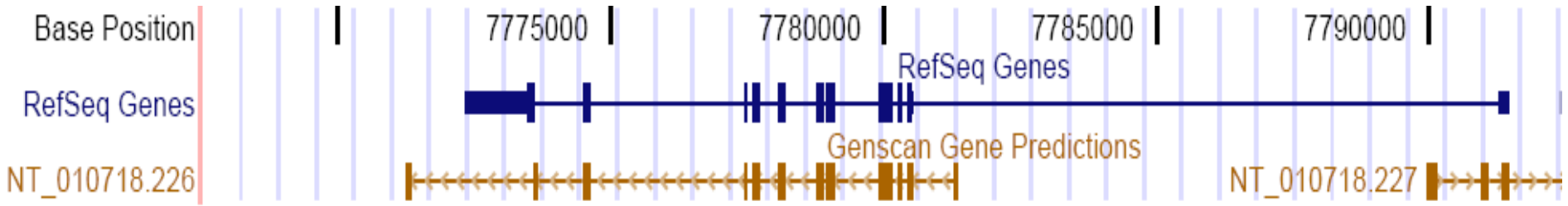
Spliced Alignment
Compare with cDNA or EST probes



Spliced Alignment
Compare with protein probes

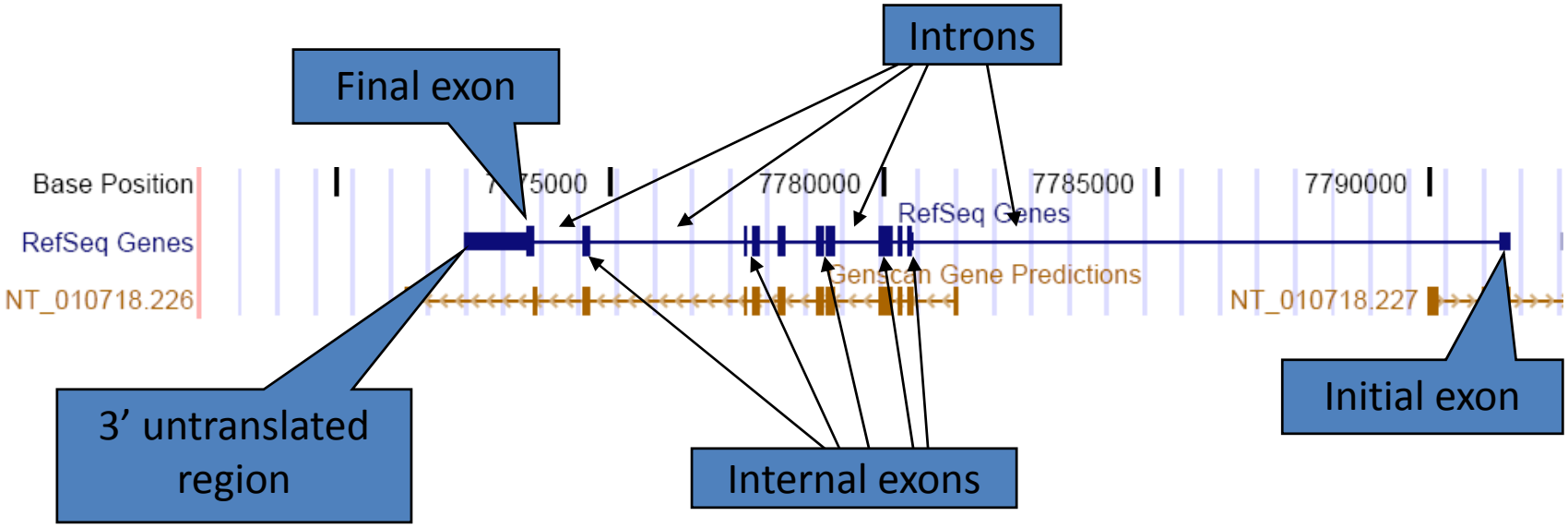


A eukaryotic gene



- 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

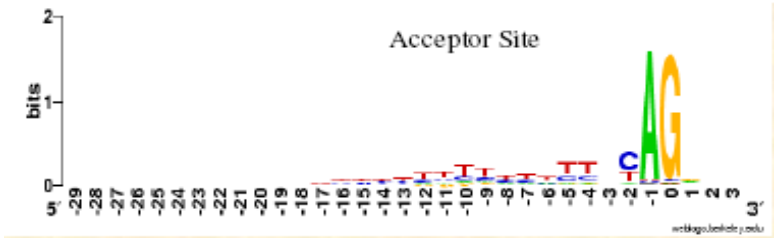
revcomp(CT)=AG

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

- 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.

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.

Prokaryotic Vs. Eukaryotic Gene Finding

Prokaryotes:

- small genomes 0.5 – $10 \cdot 10^6$ bp
- high coding density (>90%)
- no introns

⬇

– Gene identification relatively easy, with success rate ~ 99%

Problems:

- overlapping ORFs
- short genes
- finding TSS and promoters

Eukaryotes:

- large genomes 10^7 – 10^{10} bp
- low coding density (<50%)
- intron/exon structure

⬇

– Gene identification a complex problem, gene level accuracy ~50%

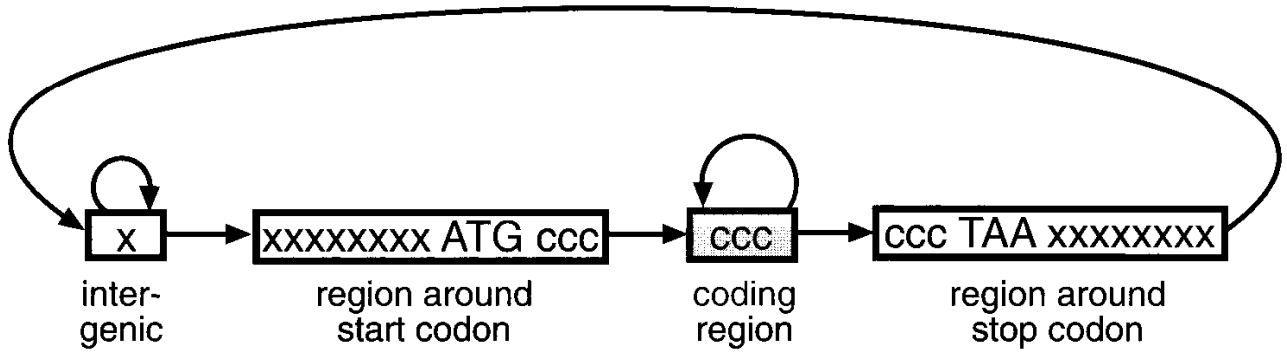
Problems:

- many

HMMs and Gene Structure

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

  A simple HMM for unspliced genes:



  AAAGC ATG CAT TTA ACG AGA GCA CAA GGG CTC TAA TGCCG

- The sequence of states is an annotation of the generated string – each nucleotide is generated in intergenic, start/stop, coding state

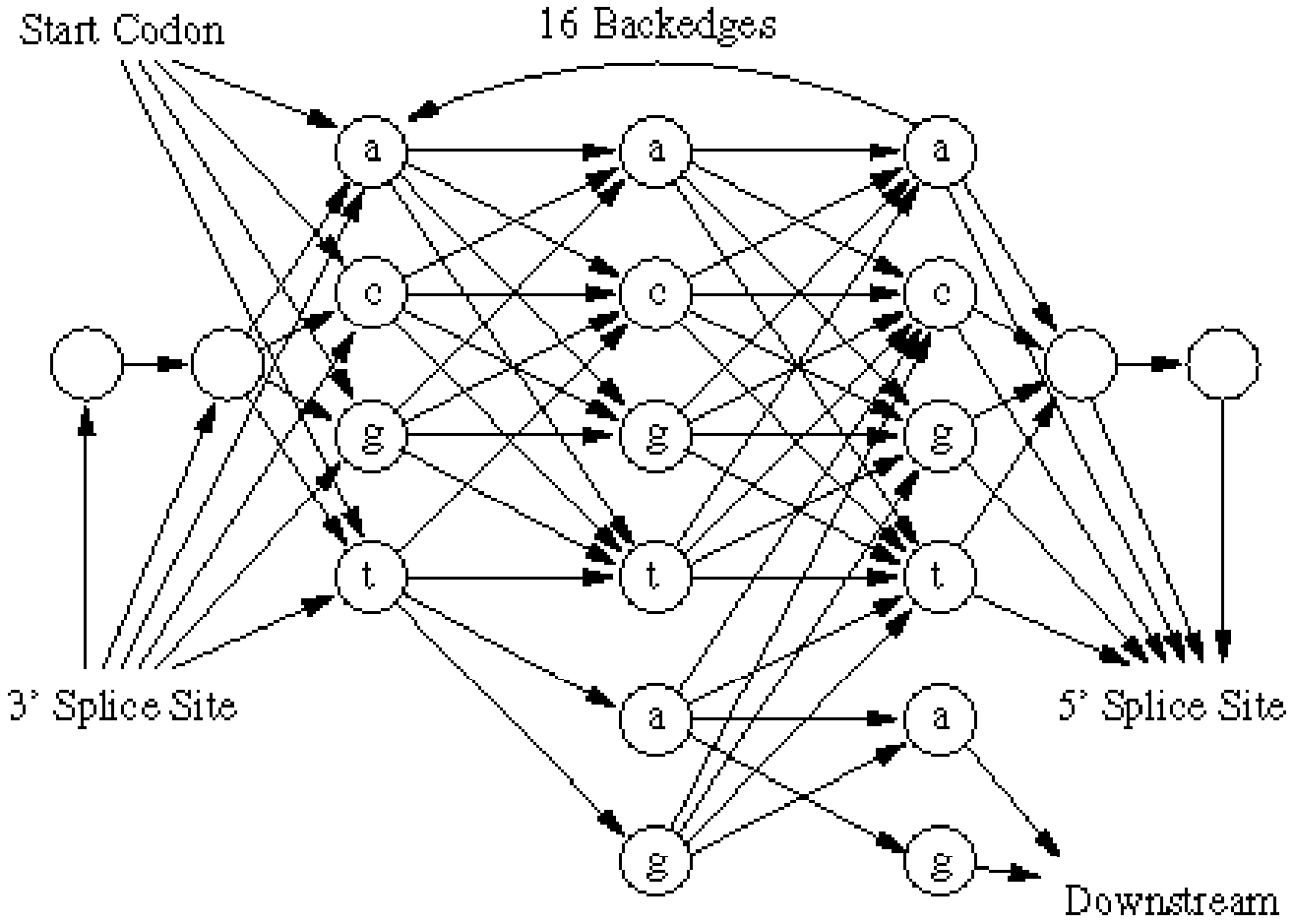
Examples of Gene Finders Using HMM

- GeneMark – HMMs enhanced with ribosomal binding site recognition
- Genie – neural networks for splicing, HMMs for coding sensors, overall structure modeled by HMM
- Genscan – WM, WA and decision trees as signal sensors, HMMs for content sensors, overall HMM
- HMMgene – HMM trained using conditional maximum likelihood
- Morgan – decision trees for exon classification, also Markov Models
- VEIL – sub-HMMs each to describe a different bit of the sequence, overall HMM

EXAMPLE: finding genes with VEIL

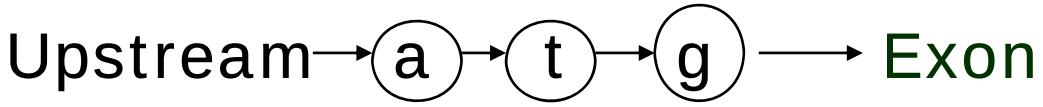
- The **Viterbi Exon-Intron Locator** (**VEIL**) was developed by John Henderson, Steven Salzberg, and Ken Fasman at Johns Hopkins University.
- Gene finder with a modular structure:
- Uses a HMM which is made up of sub-HMMs each to describe a different bit of the sequence: upstream noncoding DNA, exon, intron, …
- Assumes test data starts and ends with noncoding DNA and contains exactly one gene.
- Uses biological knowledge to “hardwire” part of HMM, eg. start + stop codons, splice sites.

The exon sub-model

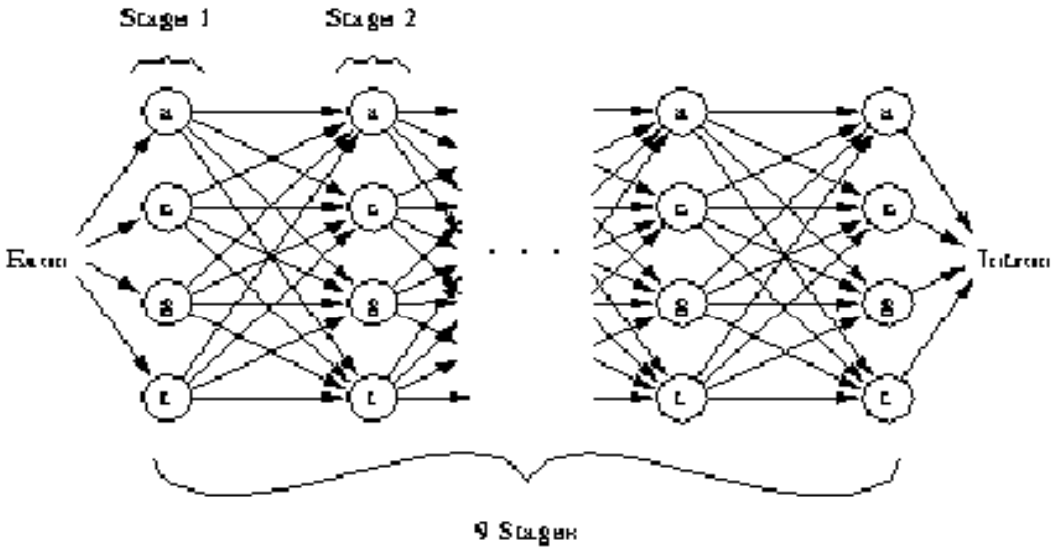


Other submodels

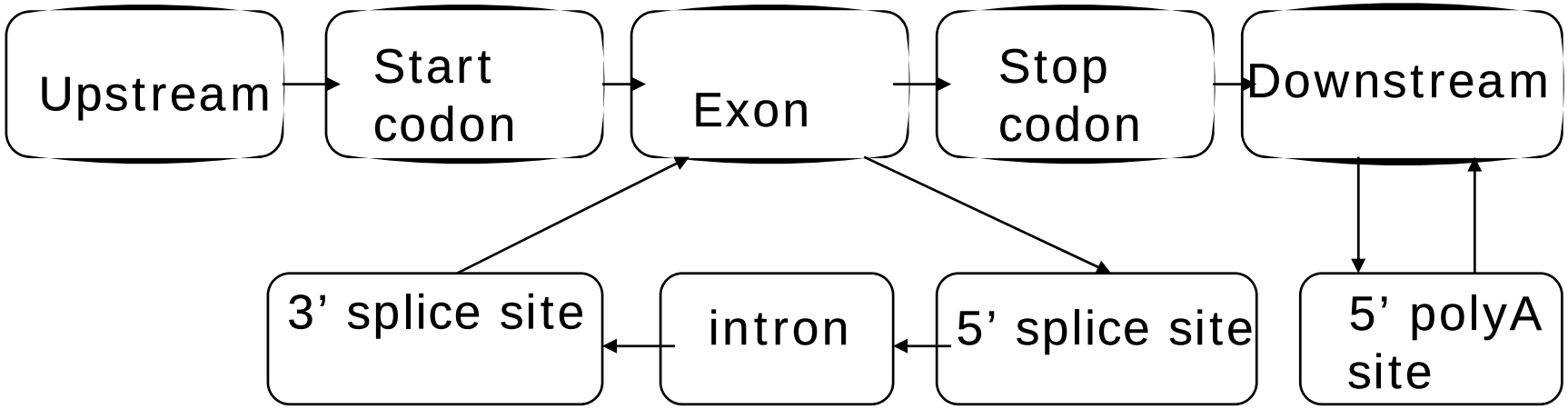
- The start codon model is very simple:



- The splice junctions are also quite simple and can be hardwired (here is the 5’ splice site):



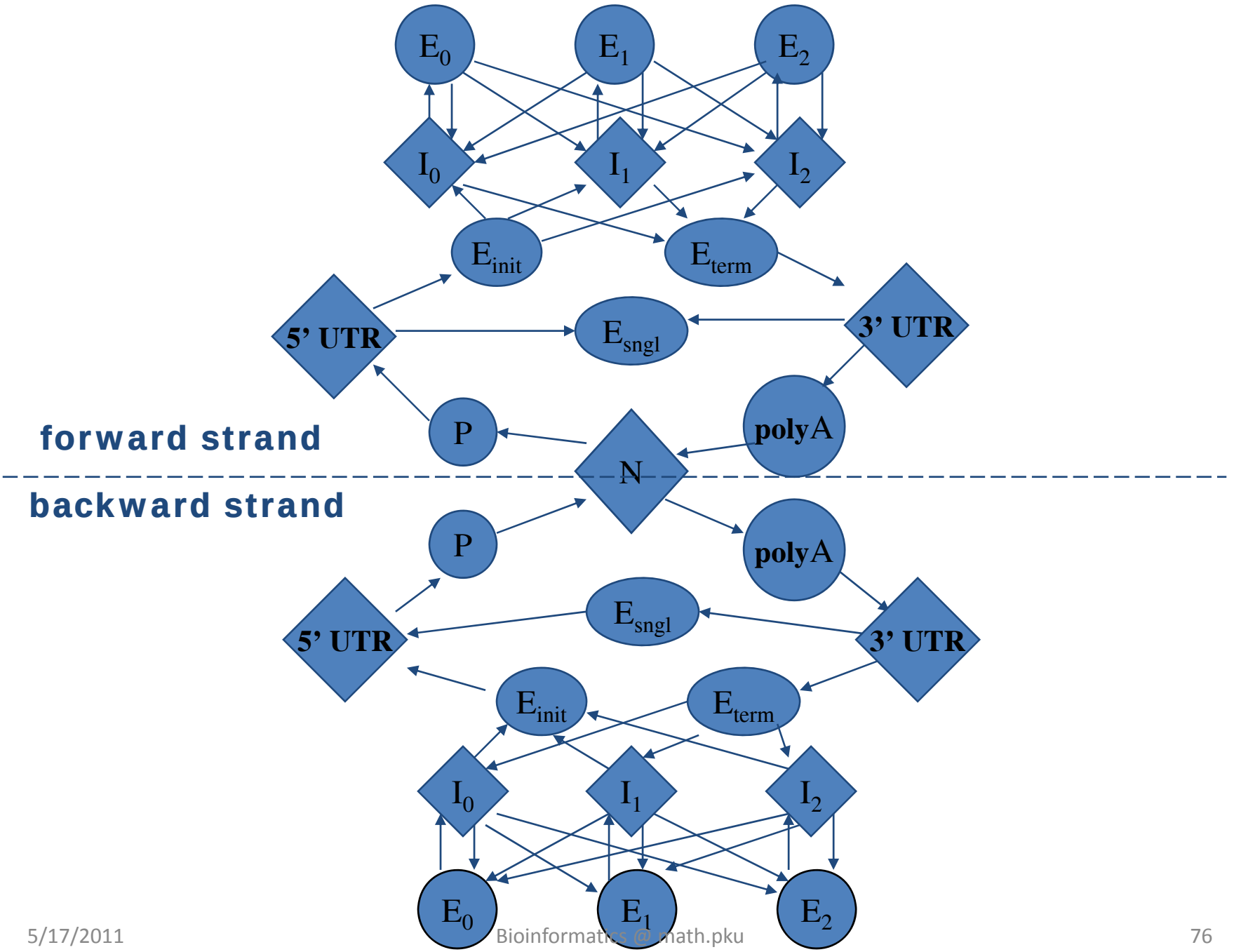
The overall model



For more details, see J. Henderson, S.L. Salzberg, and K. Fasman (1997) Journal of Computational Biology 4:2, 127-141.

Genscan Example

- Developed by Chris Burge 1997
- One of the most accurate *ab initio* programs
- Uses explicit state duration HMM to model gene structure (different length distributions for exons)
- Different model parameters for regions with different GC content

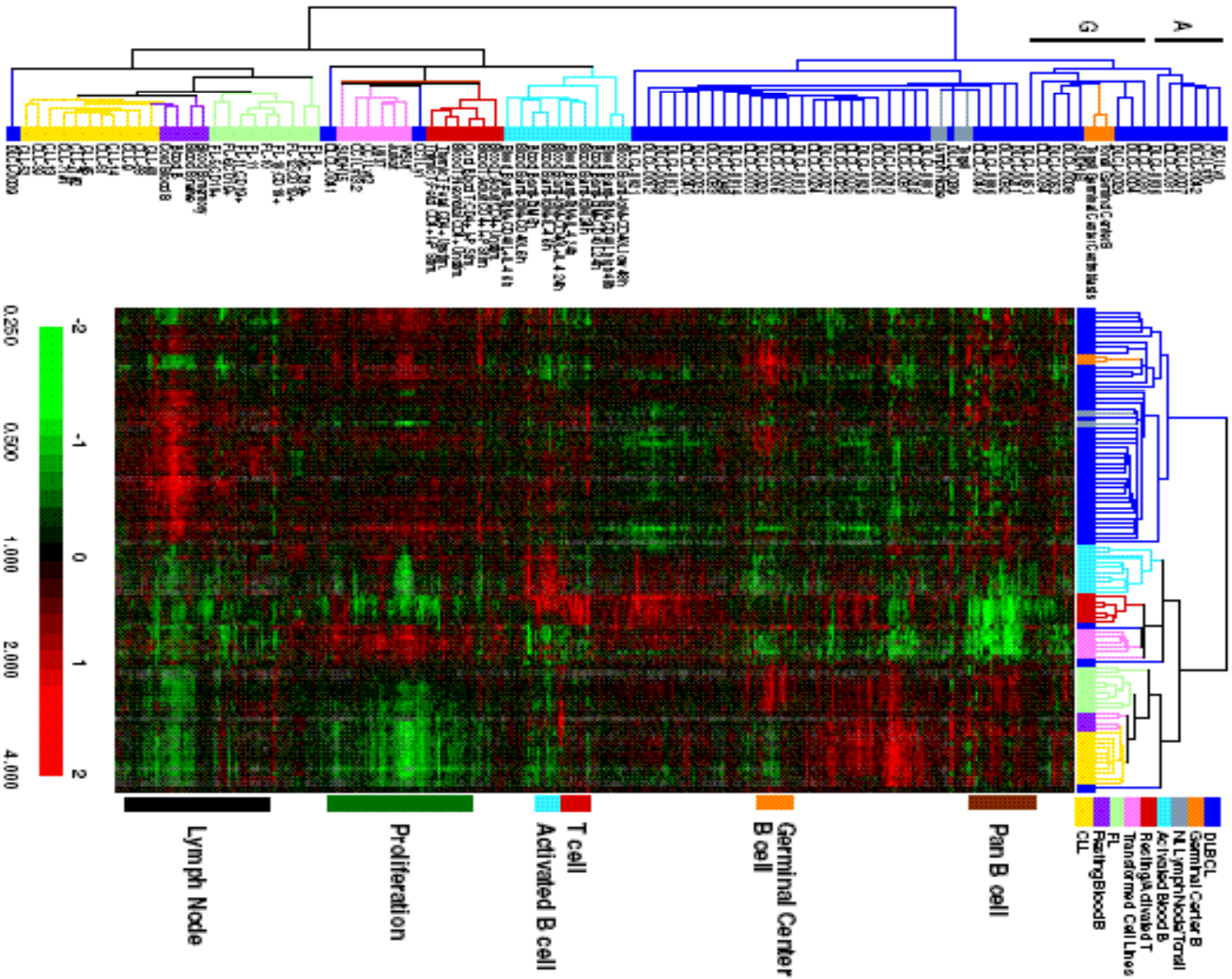

E0
E1
E2
I0
I1
I2
Einit
Eterm
5' UTR
Esngl
3' UTR
P
N
polyA
forward strand
backward strand
P
polyA
5' UTR
Esngl
3' UTR
Einit
Eterm
I0
I1
I2
E0
E1
E2

Genscan's Architecture

- HMM's states for exons and introns in three different phases, single exon, 5' and 3' UTRs, promoter region and polyA site and intergenic region
- Explicit length modeling
- HMMs for exons, introns and intergenic regions
- WM and WA for acceptor site, branch point, polyA site and promoter region
- Decision tree (maximal dependence decomposition) for donor sites

应用III: Gene Expression Clustering

- A. Schliep et al. Using hidden Markov model to analyze gene expression time course data. Bioinformatics 19(Suppl. 1) i255-i263. 2003.
- M.Yuan and C.Kendziorski. Hidden Markov Models for Microarray Time Course Data in Multiple Biological Conditions (with Discussion). *Journal of the American Statistical Association* 101(476): 1323-1332; Discussion 1332-1340, 2006.(http://www.ima.umn.edu/talks/workshops/9-29-10-3.2003/kendziorski/kendziorski_ima03.ppt)



Alizadeh et al., Nature 403:503-11, 2000

连续观测HMM

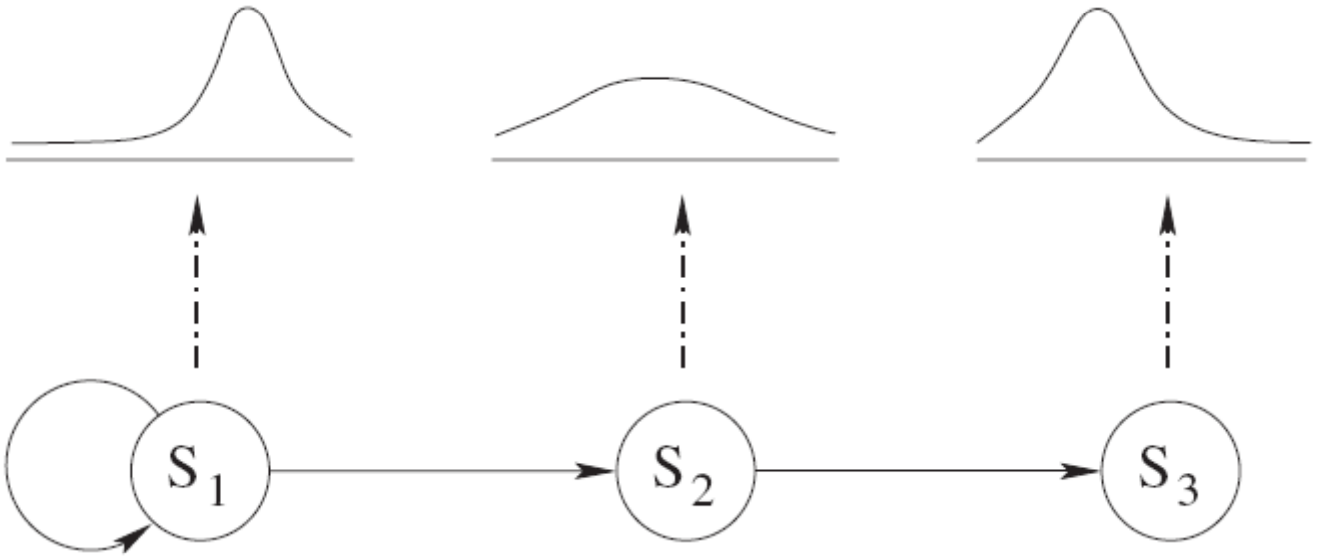


Fig. 1. A Hidden Markov Model visualized as directed graph, the emission pdfs are attached to the nodes. The model depicted is a prototype for down-regulation.

Model-based Clustering

- Given n sequences O^i, clustering is a partition

$$C = \{C_1, \cdots, C_K\}.$$

- K clusters, each cluster is a hidden Markov model λ_k
- Joint likelihood

$$f(C) = \prod_{k=1}^{K} \prod_{i \in C_k} L(O^i | \lambda_k)$$

Examples

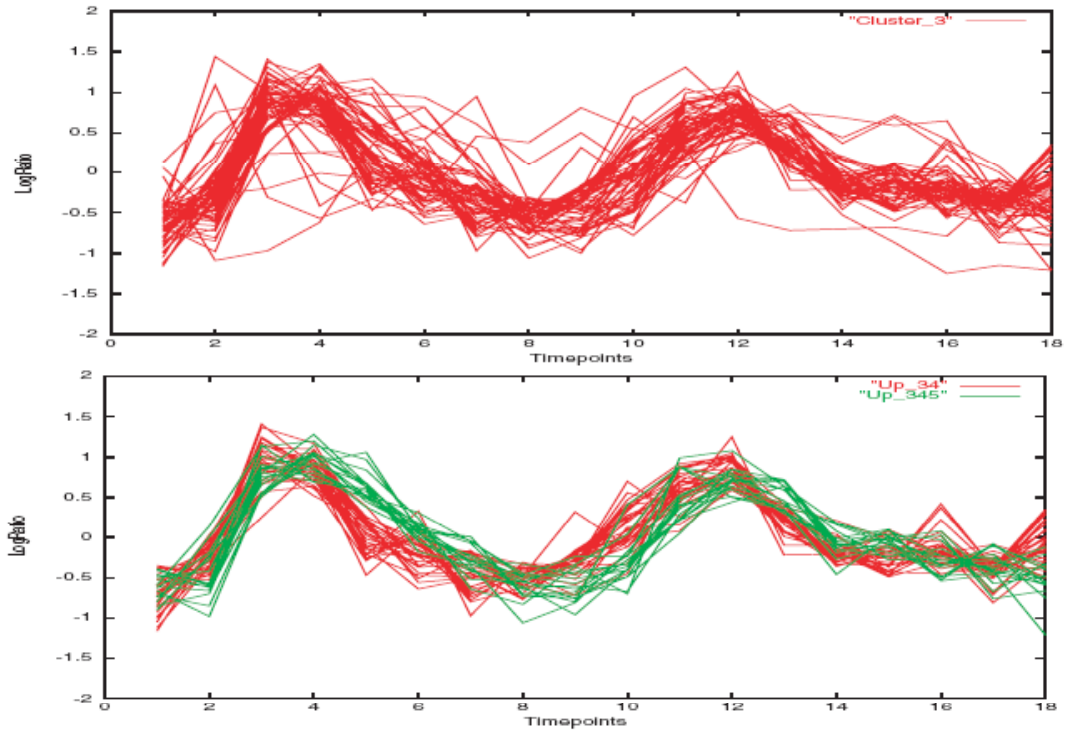


Fig. 4. Yeast: A cluster containing cell-cycle regulated gene expression time courses and a partial decomposition according to the first over-expression state.

Iterative Algorithm

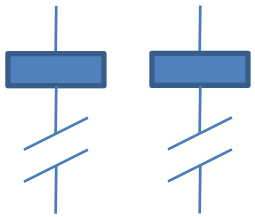
- Iteration (t = {1, 2, . . .})
  - Generate a new partitioning of the sequences by assigning each sequence O^i to the model k for which the likelihood $L(O^i|\lambda_k^{(t-1)})$ is maximal.
  - Calculate new parameters $\lambda_1^{(t)}, \cdots, \lambda_K^{(t)}$ using the re-estimation algorithm for each model with their start parameters $\lambda_1^{(t-1)}, \cdots, \lambda_K^{(t-1)}$ and assigned sequence.
- Stop: if the improvement of the objective function is below a given threshold, *ε, the grouping of the* sequences does not change or a given iteration number is reached

应用IV: CNV Detection

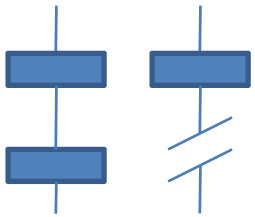
- K. Wang et al. PennCNV: an integrated hidden Markov model designed for high-resolution copy number variation detection in whole-genome SNP genotyping data. Genome Research 17: 1665-1674, 2007.

This part is modified from Yoon Soo Pyon's slides download from *https://wiki.case.edu/images/6/6f/**Cnv**_snparray.**ppt.***

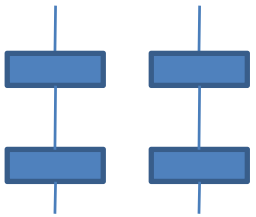
What is CNV and LOH



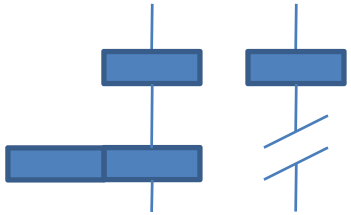
Homozygous deletion
Copy number 0



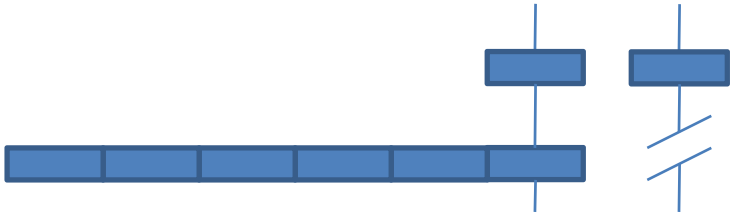
Hemizygous deletion
Copy number 1



Normal
Copy number 2



Copy neutral LOH
Copy number 2



amplication
Copy number 6

Two methods of CNV identification

- Clone-based comparative genomic hybridization (Array CGH)
  - Test and reference DNA are differentially fluorescent labeled and hybridized to the array.
  - cons: low resolution (Cannot find small CNV region)
- SNP genotyping array
  - pros: Higher resolution
  - Cons: poor signal-to-noise ratio of hybridization

Generation of SNP genotyping array

- Ilummina Bead Array
  - Human-1 Beadchip (100,000)
  - 240,000 BeadArray
  - 300,000
  - 550,000
  - 650,000
  - 1 Million just released. (human1M)
- Affymetrix SNP array
  - 10,000 (Mapping 10K array, 2003)
  - 100,000 (Mapping 100K array)
  - 500,000 (Mapping 500K array)
  - 1 Million just released (Genome-wide Human SNP 6.0)

Genotyping Array

SNP probe

- Target (250-2000 bp)
- …CAGACAGAAGTCTTG[A/C]AATCTATTTCTCATA…

PM_A : TGTCTTCAGAACTTTAGATAAAGAG

MM_A : TGTCTTCAGAACATTAGATAAAGAG

PM_B : TGTCTTCAGAACGTTAGATAAAGAG

MM_B : TGTCTTCAGAACCTTAGATAAAGAG

PM_A^o: TCTTCAGAACTTTAGATAAAGAGTA

MM_A^o : TCTTCAGAACTTAAGATAAAGAGTA

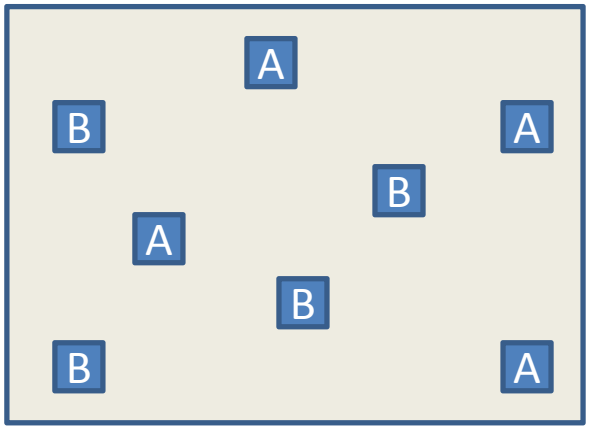
PM_B^o : TCTTCAGAACGTTAGATAAAGAGTA

MM_B^o : TCTTCAGAACGTAAGATAAAGAGTA

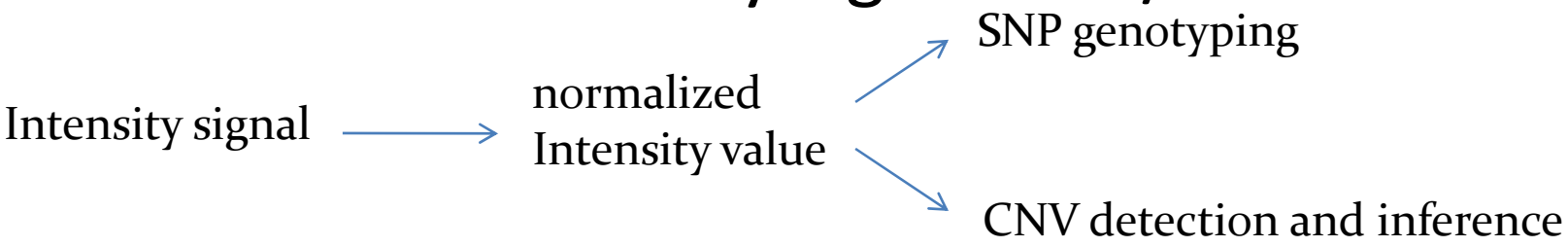
SNP probe, CNV probe (Affymetrix)

- Mapping 100K:
  - 1 probe set: 40 probes (20 PM, 20 MM), 25 bp/each
- SNP6.0:
  - 906,600 SNP probes, 946,000 CNV probes
  - 1 SNP probe set: 6~8 probes (all PM), 25 bp/each
  - CNV probe (1 probe/probe set) : 202,000 probes targeting 5,677 known regions of copy number variation, 3,182 distinct, nonoverlapping segments, each interrogated with an average of 61 probes. In addition, more than 744,000 probes were chosen evenly spaced along the genome to find novel CNVs.

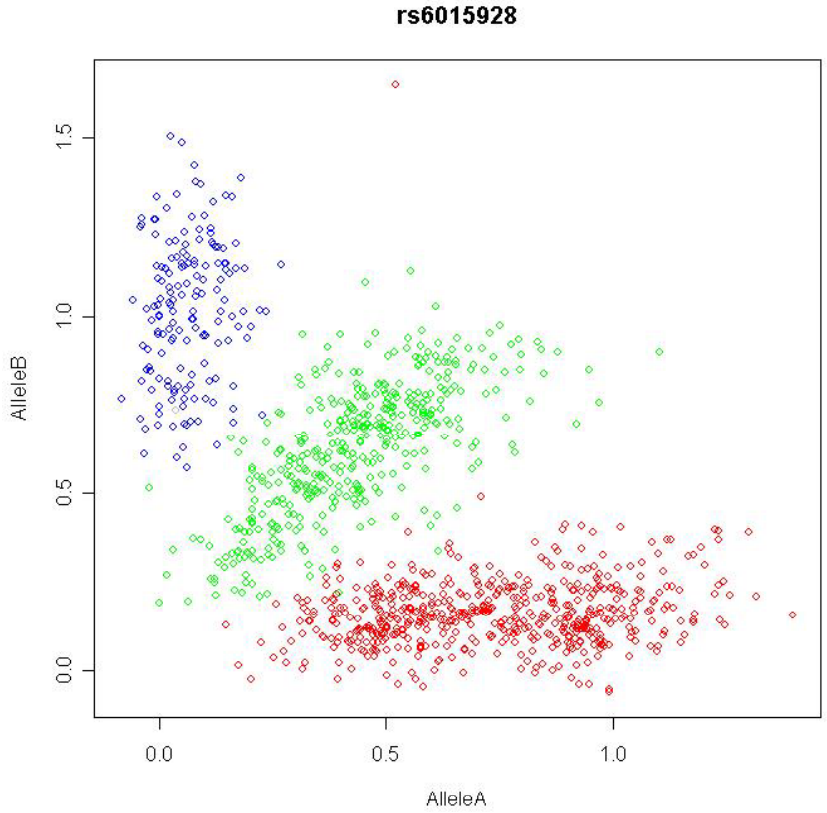
SNP Genotyping



- Fluorescent intensity signal of A/B allele



SNP genotyping



PennCNV

- Hidden Markov Model designed for high resolution CNV detection in whole genome SNP genotyping data

Table 1. Hidden states, copy numbers, and their descriptions

Copy no. state	Total copy no.	Description (for autosome)	CNV genotypes
1	0	Deletion of two copies	Null
2	1	Deletion of one copy	A, B
3	2	Normal state	AA, AB, BB
4	2	Copy-neutral with LOH	AA, BB
5	3	Single copy duplication	AAA, AAB, ABB, BBB
6	4	Double copy duplication	AAAA, AAAB, AABB, ABBB, BBBB

PennCNV (cont'd.)

- Log R ratio (LRR): total fluorescent intensity signals from both sets of probe/allele at each SNP
- B Allelle Frequence (BAF) : relative ratio of the intensity signals between two probes/allele at each SNP
- Accurate model for log R ratio and B Allele Frequency
- + Population allele frequency + distance between adjacent SNPs + family information

PennCNV (cont'd.)

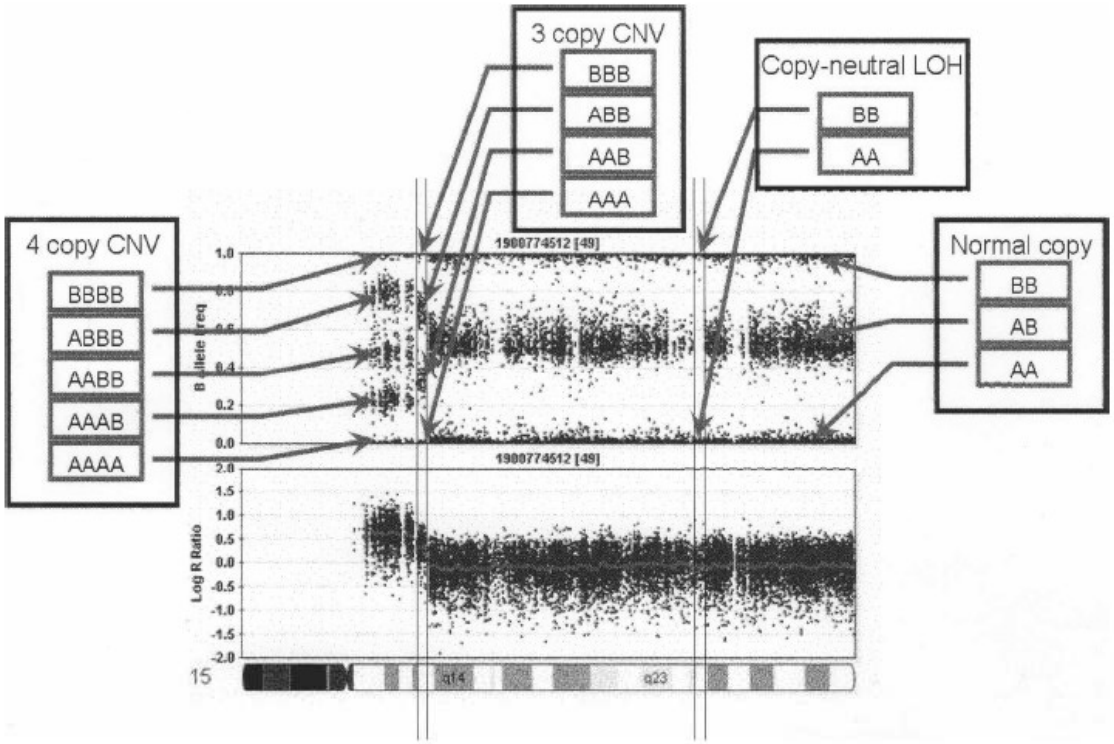
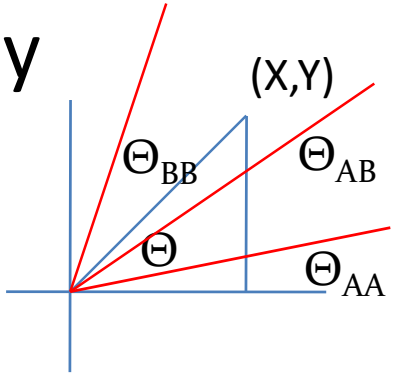


Figure 1. An illustration of log R Ratio (LRR) and B Allele Freq (BAF) values for the chromosome 15 q-arm of an individual. A normal chromosome region has three BAF genotype clusters, as represented as AA, AB, and BB genotypes in boxes, and with LRR values centered around zero. The copy-neutral LOH region has normal LRR values, but without the AB genotype cluster. The increased copy number for a CNV region can be detected based on an increased number of peaks in the BAF distribution, as well as increased LRR values. The patterns of LRR and BAF for different CNV regions, normal regions, and copy-neutral LOH regions are distinct from each other, thus the combination of LRR and BAF can be used to generate CNV calls.

PennCNV – inference of LRR and BAF

- X, Y : normalized signal intensity
- R = X+Y : total signal intensity
- Θ = arctan(Y/X)/(π/2)



$$\mathrm{LRR} = \log_2(\mathrm{R}_{\mathrm{observed}}/\mathrm{R}_{\mathrm{expected}})$$

$$\mathrm{BAF} = \begin{cases} 0, if\ \theta < \theta_{AA} \\ 0.5(\theta - \theta_{AB})/(\theta_{AB} - \theta_{AA}), if\ \theta_{AA} \le \theta < \theta_{AB} \\ 0.5 + 0.5(\theta - \theta_{AB})/(\theta_{BB} - \theta_{AB}), if\ \theta_{AB} \le \theta < \theta_{BB} \\ 1, if\ \theta \ge \theta_{BB} \end{cases}$$

PennCNV (cont'd.)

- First order HMM assumes that the hidden copy number state at each SNP depends only the copy number state of the most preceding SNP.
- $\{r_i, b_i, z_i\}$: log R ratio, B allele Frequency, Copy number state at SNP i ($1 \le i < M$)

$$P(r_1,\ldots,r_M,b_1,\ldots,b_M) = \sum_{z_1}\cdots\sum_{z_M} P(r_1,\ldots,r_M,b_1,\ldots,b_M \mid z_1,\ldots,z_M)P(z_1,\ldots,z_M)$$

$$= \sum_{z_1}\cdots\sum_{z_M}\left\{\left(\prod_{i=1}^{M} P(r_i \mid z_i)P(b_i \mid z_i)\right)\left(P(z_1)\prod_{i=2}^{M} P(z_i \mid z_{i-1})\right)\right\}$$

PennCNV, emission probability

- Emission probability of log R ratio

$$P(r \mid z) = \pi_r + (1-\pi_r)\phi(r; \mu_{r,z}, s_{r,z})$$

- Emission probability of B allele Frequency

$$P(b \mid z) = \pi_b + (1-\pi_b)\sum_{g=2}^{K(z)-1} BN[g-1; K(z)-1, p_B]\phi(b; \mu_{b,g}, s_{b,g})$$
$$+ (1-\pi_b)BN[0; K(z)-1, p_B][I_{\{b=0\}}M_0 + I_{\{0<b<1\}}\phi(b; \mu_{b,1}, s_{b,1})]$$
$$+ (1-\pi_b)BN[K(z)-1; K(z)-1, p_B][I_{\{b=1\}}M_1 + I_{\{0<b<1\}}\phi(b; \mu_{b,K(z)}, s_{b,K(z)})]$$

$$where\ BN[g-1; K(z)-1, p_B] = \binom{K(z)-1}{g-1} {p_B}^{g-1}(1-p_B)^{K(z)-g}$$

PennCNV Transition probability of hidden states

- Probability of having a copy number state change between two adjacent SNPs.
- Intuition: The copy number state is unlikely to change for SNPs that are nearby but is more likely to change for SNPs that are far apart.

$$P(z_i = l \mid z_{i-1} = j) = \begin{cases} 1 - \sum_{k=2}^{6} P_{j,k-1}(1 - e^{-d_i / D}), \; if \; l = j \\ P_{j,l-1}(1 - e^{-d_i / D}), \; if \; l \neq j \end{cases}$$

- D is constant number. 100MB for state4 and 100KB for others
- Value p are treated as unknown parameter and estimated in the Baum-Welch algorithm

PennCNV –parameter estimation and CNV calling

- Baum-Welch algorithm for training model to maximize the likelihood of the observed data of each individual
- Viterbi algorithm to infer most likely path.
- CNV is called most likely state sequence whenever a stretch of states that is different from normal state is observed.