

**BCB568      Spring 2010 Midterm Examination I**

**February 4, 2010, 9:30AM–10:50AM**

**Sign your name \_\_\_\_Super Solver \_\_\_\_**

**You may not use books, notes, or other media,  
except (unnecessary) calculators.**

**Explain your answers.**

**None of the answers should take more than a few lines!**

Problem 1 [8 points]

Problem 2 [4 points]

Problem 3 [6 points]

Problem 4 [12 points]

TOTAL [30 points]

1. Given two sequences  $A = a_1a_2\dots a_N$  and  $B = b_1b_2\dots b_M$  over the alphabet  $\alpha = \{A, C, G, T\}$ , a similarity scoring matrix  $\sigma(a, b)$ ,  $a, b \in \alpha$ , and gap penalties  $w(0) = 0$ ,  $w(g) < 0$ ,  $g = 1, 2, \dots$ , the optimal alignment score  $S(A, B)$  for a global alignment of  $A$  and  $B$  with end gap penalties can be obtained recursively by noting

$$S_{i,j} = \max \begin{cases} S_{i-1,j-1} + \sigma(a_i, b_j) \\ S_{i,j-k} + w(k) & k = 1, 2, \dots, j \\ S_{i-l,j} + w(l) & l = 1, 2, \dots, i \end{cases}$$

where  $S_{0,0} = 0$ ,  $S_{i,0} = w(i)$ ,  $S_{0,j} = w(j)$ , and  $S(A, B) = S_{N,M}$ , provided  $w(k) + w(l) \leq w(k+l)$  for all  $k = 1, 2, \dots$ ,  $l = 1, 2, \dots$  (Smith & Waterman, 1981).

(1) [2 points] Give an algorithm to recursively calculate  $N_{ij}$ , the number of alignments of the prefixes  $a_1a_2\dots a_i$  and  $b_1b_2\dots b_j$ . Determine the number of alignments for  $i = 3$ ,  $j = 4$ .

(2) [2 points] Determine the number of additions (+ operations) required to calculate  $S_{N,M}$ .

(3) [2 points] If  $w(g) = -\alpha - \beta g$ , then we can write

$$S_{i,j} = \max \begin{cases} S_{i-1,j-1} + \sigma(a_i, b_j) \\ H_{i,j} = \max \{S_{i,j-k} + w(k)\} & k = 1, 2, \dots, j \\ V_{i,j} = \max \{S_{i-l,j} + w(l)\} & l = 1, 2, \dots, i \end{cases}$$

where  $H_{i,j}$  and  $V_{i,j}$  can be calculated recursively as

$$H_{i,j} = \max \{H_{i,j-1} - \beta, S_{i,j-1} + w(1)\}, \quad V_{i,j} = \max \{V_{i-1,j} - \beta, S_{i-1,j} + w(1)\}$$

(Gotoh, 1982). Prove the result. [Hint: Write  $H_{i,j} = \max \{S_{i,j-k} + w(k)\}$ ,  $k = 1, 2, \dots, j = \max \{\max \{S_{i,j-k} + w(k)\}, k = 2, \dots, j; S_{i,j-1} + w(1)\}$ ]

(4) [2 points] Determine the number of additions (+ operations) required to calculate  $S_{N,M}$  in this case and compare with (1). [8 points]

**Solution:** 1) [2 points]

Graphically, any alignment ending in the  $i, j$  point of the grid representing alignments by diagonal, horizontal, and vertical line segments must have come from the  $i - 1, j - 1$  point (continuing a shorter alignment diagonally with an  $\begin{pmatrix} a_i \\ b_j \end{pmatrix}$  match), or the  $i, j - 1$  point (continuing a shorter alignment horizontally with  $\begin{pmatrix} - \\ b_j \end{pmatrix}$ ), or the  $i - 1, j$  point (continuing a shorter alignment vertically with  $\begin{pmatrix} a_i \\ - \end{pmatrix}$ ). Thus,  $N_{ij} = N_{i-1,j-1} + N_{i,j-1} + N_{i-1,j}$ . The initial conditions are set as in the following table.

		j								
		0	1	2	3	4	5	6	7	8
i	0	1	1	1	1	1				
	1	1	3	5	7	9				
	2	1	5	13	25	41				
	3	1	7	25	63	129				
	4	1								
	5									
	6									
	7									
	8									
	8									

The number of alignments for  $i = 3, j = 4$  is 129 as shown in the table above.

**Solution:** 2) [2 points]

The number of additions is seen to be  $\sum_{i=1}^M \sum_{j=1}^N [1 + j + i] = MN + M \frac{N(N+1)}{2} + N \frac{M(M+1)}{2}$ . Thus, for  $M = N$ , the algorithm is of  $O(N^3)$ .

**Solution:** 3) [2 points] Making the substitution  $g = k - 1$  in the following, we have

$$\begin{aligned} H_{i,j} &= \max \{S_{i,j-k} + w(k)\}, k = 1, 2, \dots, j \\ &= \max \{ \max \{S_{i,j-k} + w(k)\}, k = 2, \dots, j; \quad S_{i,j-1} + w(1) \} \\ &= \max \{ \max \{S_{i,j-k} + (-\alpha - \beta k)\}, k = 2, \dots, j; \quad S_{i,j-1} + w(1) \} \\ &= \max \{ \max \{S_{i,j-k} + (-\alpha - \beta(k-1)) - \beta\}, k = 2, \dots, j; \quad S_{i,j-1} + w(1) \} \\ &= \max \{ \max \{S_{i,j-g-1} + (-\alpha - \beta g) - \beta\}, g = 1, \dots, j-1; \quad S_{i,j-1} + w(1) \} \\ &= \max \{ \max \{S_{i,(j-1)-g} + (-\alpha - \beta g)\} - \beta, g = 1, \dots, j-1; \quad S_{i,j-1} + w(1) \} \\ &= \max \{H_{i,j-1} - \beta, S_{i,j-1} + w(1)\}. \end{aligned}$$

The proof of the  $V_{i,j}$  maximization is analogous.

**Solution:** 4) [2 points]

The number of additions is seen to be  $\sum_{i=1}^M \sum_{j=1}^N [1 + 2 + 2] = MN + 2MN + 2MN$ . Thus, for  $M=N$ , the algorithm is of  $O(5N^2)$ , which is superior to the complexity of the algorithm of (1).

2. Let  $S = s_1 s_2 \dots s_N$  represent a nucleotide sequence of length  $N$  over the alphabet  $A = \{A, C, G, T\}$ , and let  $T = t_1 t_2 \dots t_N$  be a permutation of  $S$  that preserves the mononucleotide and dinucleotide composition. Prove that  $s_1 = t_1$  and  $s_N = t_N$  (in words, any such permutation has the same starting nucleotide and the same ending nucleotide as the original sequence). [4 points]

**Solution:**

If  $s_1 \neq s_N$ , then the count of dinucleotides starting with  $s_1$  is one more than the count of dinucleotides ending in  $s_1$ . Similarly, the count of dinucleotides ending with  $s_N$  is one more than the count of dinucleotides starting with  $s_N$ . All other dinucleotide counts are balanced. If  $t_1 \neq s_1$  or  $t_N \neq s_N$ , then, by the same argument, the  $T$  sequence counts would be imbalanced for different dinucleotides, i.e. such permutation would not preserve the dinucleotide counts. Thus, necessarily  $t_1 = s_1$  and  $t_N = s_N$ .

If  $s_1 = s_N$ , then a dinucleotide permutation of  $S$  not starting with  $s_1$  or ending with  $s_N$  would have a by one lower mononucleotide count for  $s_1 = s_N$ , while the mononucleotide count of  $t_1 = t_N$  would be elevated by one. Thus, to preserved the mononucleotide composition in this case, necessarily  $s_1 = s_N = t_1 = t_N$ .

3. For a random sequence over the alphabet  $\{R, Y\}$  for which  $R$  is drawn with probability  $p$  and  $Y$  is drawn with probability  $q = 1 - p$ , calculate the expected waiting time until (1) the first occurrence of the pattern  $YYRYYY$ , and (2) the first occurrence of the

pattern  $RRRYR$ . Derive the general results, then compare the results in the special case when  $p = q = 0.5$ . How would you set up equations the solution of which would allow you to calculate the probability that  $YYRYYY$  occurs before  $RRRYR$  in a random sequence? [6 points]

**Solution:** (1) For an occurrence of the pattern  $YYRYYY$  at position  $n$  occurring with probability  $u_n$ , we can decompose the probability event as follows:

$\text{—————}YYRYYY$   
 $\text{——}YYRYYYRYYY$   
 $\text{——}YYRYYYRYYY$

with probability  $pq^5 = u_{n-5}pq^4 + u_{n-4}pq^3 + u_n$

When  $n \rightarrow \infty$ ,  $u_{n-5} = u_{n-4} = u_n = \frac{1}{\mu}$ , where  $\mu$  is the waiting time.

Thus  $\mu = \frac{pq^4 + pq^3 + 1}{pq^5}$ .

If  $p = q = 0.5$ ,  $\mu = \frac{\frac{1}{2}^5 + \frac{1}{2}^4 + 1}{\frac{1}{2}^6} = 70$ .

**Solution:** (2) For the pattern  $RRRYR$  we decompose the probability event as follows:

$\text{—————}RRRYR$   
 $\text{——}RRRYR$

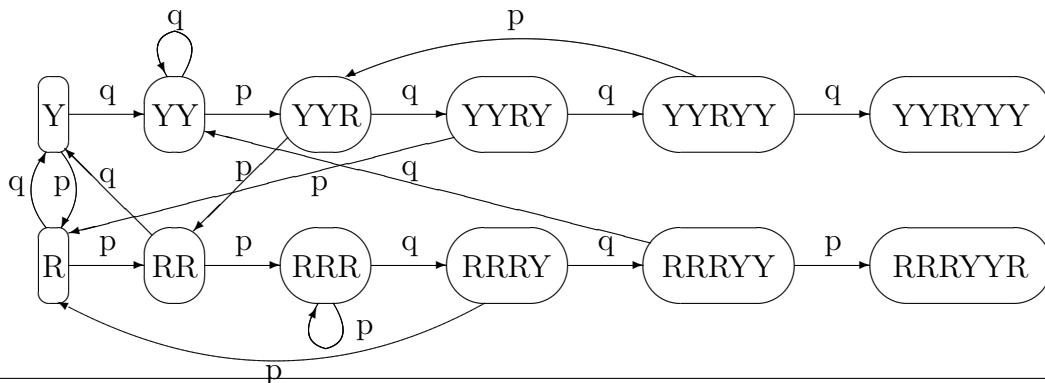
$p^4q^2 = u_{n-5}p^3q^2 + u_n$

When  $n \rightarrow \infty$ ,  $u_{n-5} = u_n = \frac{1}{\mu}$ .

Thus  $\mu = \frac{p^3q^2 + 1}{p^4q^2}$ .

If  $p = q = 0.5$ ,  $\mu = \frac{\frac{1}{2}^3 + 1}{\frac{1}{2}^6} = 66$

**Solution:** (3) The probability that  $YYRYYY$  occurs before  $RRRYR$  in a random sequence could be calculated by first-step analysis on  $\rho_X$ , the probability that  $YYRYYY$  occurs first when in state  $X$  of the Markov chain defined by the following diagram:



$$\begin{aligned}
\rho_Y &= q * \rho_{YY} + p * \rho_R \\
\rho_{YY} &= q * \rho_{YY} + p * \rho_{YYR} \\
\rho_{YYR} &= q * \rho_{YYRY} + p * \rho_{RR} \\
\rho_{YYRY} &= q * \rho_{YYRY} + p * \rho_R \\
\rho_{YYRY} &= q * \rho_{YYRY} + p * \rho_{YYR} \\
\rho_{YYRY} &= 1 \\
\rho_R &= q * \rho_Y + p * \rho_{RR} \\
\rho_{RR} &= q * \rho_Y + p * \rho_{RRR} \\
\rho_{RRR} &= q * \rho_{RRRY} + p * \rho_{RRR} \\
\rho_{RRRY} &= q * \rho_{RRRY} + p * \rho_R \\
\rho_{RRRY} &= q * \rho_{YY} + p * \rho_{RRRY} \\
\rho_{RRRY} &= 0
\end{aligned}$$

4. For a Hidden Markov Model with three states  $A$ ,  $B$ , and  $C$ , initial state probabilities  $\pi_A, \pi_B$ , and  $\pi_C$ , state transition probabilities  $\tau_{XY}$ , and output probabilities  $P(O|S)$ , show how to calculate  $P(O_1O_2 \dots O_N)$  in  $O(18N)$  operations. [12 points]

**Solution:**

Define  $X_i = P(O_1O_2 \dots O_i, Q_i = X)$ , where  $X$  is  $A$ ,  $B$ , or  $C$ , and  $Q_i$  is the state at position  $i$ . Then  $P(O_1O_2 \dots O_N) = A_N + B_N + C_N$ , and all the  $X_i$  can be calculated recursively as follows:

$$\begin{aligned}
A_1 &= \pi_A \times P(O_1|A) \\
B_1 &= \pi_B \times P(O_1|B) \\
C_1 &= \pi_C \times P(O_1|C)
\end{aligned}$$

$$\begin{aligned}
A_i &= [A_{i-1} \times \tau_{AA} + B_{i-1} \times \tau_{BA} + C_{i-1} \times \tau_{CA}] \times P(O_i|A) \\
B_i &= [A_{i-1} \times \tau_{AB} + B_{i-1} \times \tau_{BB} + C_{i-1} \times \tau_{CB}] \times P(O_i|B) \\
C_i &= [A_{i-1} \times \tau_{AC} + B_{i-1} \times \tau_{BC} + C_{i-1} \times \tau_{CC}] \times P(O_i|C)
\end{aligned}$$

Let's calculate the number of operations for the general case of  $S$  states (here  $S = 3$ ). The number of multiplications required is  $S + (N - 1)S(S + 1)$ , and the number of additions requires is  $(N - 1)S(S - 1) + (S - 1)$ , accounting for the initial, recursive, and final calculations. Adding up the terms involving  $N$  gives  $N2S^2$ , equal to  $18N$  in the case  $S = 3$ .