Introduction to Computational Biology
Homework 2
09/30/09

Due 10/14/09

Problem 1: Number of alignments
We discussed in class that the number of alignments of two sequences x and y is exponential in their length. This problem is designed to prove this fact.

We will count only the number of distinct alignments. Of course we need to define what distinct really means. For this, define an equivalence on alignments as follows: Two alignments are equivalent if they align the same indices of x to the same indices of y.

For instance, the following two alignments are equivalent:

```
A-GCTTT-GA
-AG-T--CG-

-AGCT-TTGA
A-G-TC--G-
```

because they align x_2 with y_2, x_4 with y_3, and x_7 with y_5.

It follows that two alignments are distinct if they are not equivalent.

Let $|x| = m$ and $|y| = n$.

(a) Show that the number of distinct alignments is $\begin{pmatrix} m+n \\ n \end{pmatrix}$.

(b) For $m = n$, use Stirling formula $n! \sim \sqrt{2\pi n}(\frac{n}{e})^n$ to prove that the number of distinct alignments is approximately $\frac{2^n}{\sqrt{\pi n}}$

(c) Can two sequences x and y have an exponential number of distinct *optimal* alignments? If yes, construct two such sequences. If no, prove it.

(d) Suggest an algorithm that produces all optimal alignments in $O(mn + L(m+n))$ where L is the number of optimal alignments.

Problem 2: Linear space alignments
Let x and y be two sequences with $|x| = m$ and $|y| = n$.

(a) Describe a linear space $O(mn)$ time algorithm to compute the optimal semi-global alignment between x and y.

(b) Describe a linear space $O(mn)$ time algorithm to compute the optimal local alignment between x and y. *Hint*: Find the two highest scoring substring in linear space and $O(mn)$ time, then compute their optimal global alignment in linear space and $O(mn)$ time.

(c) Describe a linear space $O(mn)$ time algorithm to compute the optimal global alignment between x and y under an affine gap penalty function. (this is not hard, but can be messy, so it is optional).

Problem 3: Concave gap penalty function (optional for a better understanding of gap functions)
Let γ be a gap penalty function defined over non-negative integers. The function γ is called sub-additive iff it satisfies the following: $\gamma(k_1 + k_2 + ... + k_n) \leq \gamma(k_1) + \gamma(k_2) + ... + \gamma(k_n)$.

(a) Show that a concave γ, i.e. one that satisfies $\gamma(x+1) - \gamma(x) \leq \gamma(x) - \gamma(x-1)$, is sub-additive if $\gamma(0) \geq 0$.

The next set of questions are *intended* to help you understand why the DP algorithm we saw in class requires γ to be concave. Here's the algorithm again:

$$A(i,j) = \max \begin{cases} A(i-1, j-1) + s(i,j) & (1) \\ A(i, j-k) - \gamma(k),\ k = 1...j & (2) \\ A(i-k, j) - \gamma(k),\ k = 1...i & (3) \end{cases}$$

Without loss of generality, we restrict our attention to alignments that end with a gap in x. Call such an alignment of $x_1...x_i$ and $y_1...y_j$ "good" if it ends with a gap of length k in x for some $k > 0$ and **optimally** aligns $x_1...x_i$ to $y_1...y_{j-k}$.

(b) Show that a "good" alignment of $x_1...x_i$ and $y_1...y_j$ is not necessarily optimal.

(c) Show that if γ is concave, then an optimal alignment (that ends with a gap in x) of $x_1...x_i$ and $y_1...y_j$ is a "good" alignment.

(d) Show that if γ is concave, then for any given alignment of $x_1...x_i$ and $y_1...y_j$ with score S, if we split the alignment in two parts with scores S_1 and S_2, then $S_1 + S_2 \leq S$.

(e) Based on parts (b) and (c) and (d), explain why step (2) in the algorithm above is both **necessary**, i.e. it must check all $k = 1...j$, and **correct**, i.e. it computes the optimal alignment of $x_1...x_i$ and $y_1...y_j$ that ends with a gap in x.

(f) Construct an instance of an optimal alignment (that ends with a gap in x) that is not "good". (you cannot use a concave gap function according to part (c)). Argue that the algorithm above will not work properly if such an instance can be constructed.

Problem 4: Edit distance and beyond
The edit distance of two strings x and y is defined as the minimum number of operations needed to change x into y, where each operation can be an insertion, a deletion, or a substitution.

(a) Devise a scoring scheme such that the score of the optimal alignment can be easily related to the edit distance.

(b) Devise a scoring scheme such that the score of the optimal alignment can be easily related to the length of the longest common subsequence.

(c) Is the length of the longest common subsequence related to the edit distance in any way?

(d) We define a scatter of a string u of length n as a any string

$$V_0 u_1 V_1 u_2 V_3 ... V_{n-1} V_{n-1} u_n V_n$$

where V_i are arbitrary strings of any lengths. Design an $O(mn)$ algorithm to find the shortest scatter of both x and y (*Hint*: relate this to the longest

common subsequence of x and y).

Problem 5: Circular DNA alignment
Consider two circular DNAs x and y of length m and n respectively.

We are after the optimal global alignment of x and y. This can be obtained as follows: Consider a circular shift of x, $x_i...x_m x_1...x_{i-1}$ for some $1 \leq i \leq m$. Consider a circular shift of y, $y_j...y_n y_1...y_{j-1}$ for some $1 \leq j \leq n$. Find their optimal global alignment, and repeat for every possible pair of circular shifts of x and y. Finally pick the highest scoring alignment.

Since there are m circular shifts of x and n circular shifts of y, the above algorithm will take $O(m^2n^2)$ time.

(a) Design an $O(mn^2)$ time algorithm that will find the optimal global alignment of two circular DNAs x and y.

(b) Can you obtain the optimal global alignments for all pairs of circular shifts (i.e. mn) of x and y in $O(mn^2 + nm^2)$?