Computational Genomics
© Benny Chor and Igor Ulitsky
School of Computer Science
Tel-Aviv University
March 22, 2009

Computational Genomics: Assignment No. 1
due on April 5th (in Igor Ulitsky's mailbox)

General Guidelines: This assignment is part of the final grade in the course. It should be done *independently*, either individually or in pairs, without any help from others. Duplicated and copied works will be given zero grade. Using articles, books, or web sites is perfectly acceptable as long as you include the reference in your relevant answer. Exceptionally original answers will be rewarded with bonus points. If a questions requires a description of an algorithm, you must prove its correctness and analyze its time and space complexity.

Credit: Solve 6 items out of 7 items for full credit. Each itemized subquestion is an item. Solving all the items will result in bonus points.

1. **Exact Searches**: Assume the existence of a linear time algorithm that finds whether a sequence S is a subsequence of T.

   (a) Give a linear time algorithm to check whether a sequence S is a circular shift of another sequence T. For example, *ciseexer* is a circular shift of *exercise*.

   (b) Give a linear time algorithm to check whether a sequence S is a subsequence of a circular sequence T. For example, *seex* is a subsequence of *exercise*.

2. How many distinct pairwise alignments are possible for two sequences of lengths n and m? Give a closed-form formula for this number.

3. **Hirschberg's algorithm:** Show how Hirschberg's linear space technique can be used for the computation of an ends-free alignment.

4. The following problem is motivated by aligning a spliced mRNA sequence with an unspliced closely homologous RNA sequence. In such alignment we expect "jumps" over large gaps in the unspliced sequence. Given two sequences $S = s_1, ..., s_n$ and $T = t_1, ..., t_m$, with $n \leq m$, we would like to compute the global pairwise alignment using the constant gap penalty (i.e., every gap receives the same penalty, regardless of its length), such that gaps are allowed only in S (i.e., all letters of T occur in the alignment consecutively with no gaps between them). Give an $O(n(m-n))$ algorithm to find an optimal alignment. Prove the correctness and the time complexity of your algorithm.

5. **Sequence Comparison with at most l mismatches**. Many heuristics for rapid similarity searches use the following idea to find sequences with "enough" similarity. It

is based on the following lemma: Suppose that $a_1, \ldots, a_t$ and $b_1, \ldots, b_t$ can be aligned with at most l mismatches. Then: (i) For $k \leq \lfloor \frac{t}{l+1} \rfloor$, $a_1, \ldots, a_t$ and $b_1, \ldots, b_t$ share at least $t-(l+1)k+1$ k-tuples. (ii) For $k = \lfloor \frac{t}{l+1} \rfloor$, $a_1, \ldots, a_t$ and $b_1, \ldots, b_t$ share at least one k-tuple (a k-tuple is a subsequence of length k). Prove the lemma, and show how it can be used in a database search to rapidly find all sequences that can be aligned with a given query sequence with at most l mismatches. This idea is called *filtration*.

6. **Multiple sequence alignment.** Let M be an optimal multiple alignment of a set of sequences (with respect to the sum of pairs penalty and for some score matrix σ over $\{A, C, G, T, -\}$). Let $d(s,t)$ denote the distance between sequences s and t induced by M, and let $D(s,t)$ denote the pairwise optimal alignment score of s and t. Construct three short sequences x, y, and z such that
(a) $D(x,y) \neq d(x,y), D(x,z) \neq d(x,z)$ and $D(y,z) \neq d(y,z)$. Specify the score σ you use.