Heuristic primer selection using alignment and folding

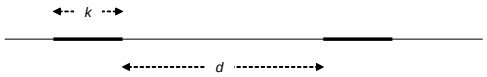
Saad Mneimneh
Computer Science
Hunter College of CUNY

General description

In biology, the primer selection problem can be (over) simplified as the following problem: Given a DNA strand, i.e. a string over the alphabet $\{A, G, C, T\}$, find two substrings of length k and at distance d from each other, such that:

- each of the two substrings does not align well to itself
- the two substrings don't align well to each other
- each of the two substrings does not fold well on itself

The two substrings are called primers, and are used for DNA replication.



The goal is to determine, for each such pair of primers, a set of alignment and folding scores, and set up a threshold for choosing those pairs that achieve a score below the threshold. The project will feature the following:

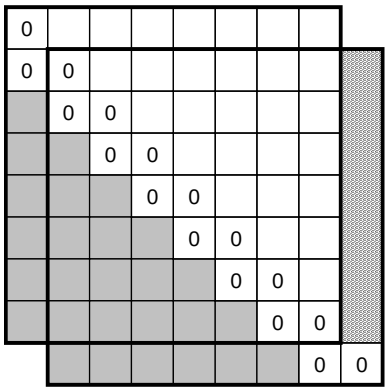
- DNA strands are read from a file or through redirection and pipes.
- For each DNA strand, the program will output the "good" primers.
- The program will have an option to output the scores for all pairs of primers in a format that can be browsed.
- A separate browser using the curses library can be used to view the primers. The browser will mainly show a DNA window of size $d - 1 + 2k$, and will respond to the left and right arrows to scroll that window to the left or to the right, respectively. The browser will also display the two primers at the extremities of the window with their scores.

Alignment

Each substring of size k will be aligned to itself to determine an alignment score. Each substring of size k will also be aligned to the other substring of size k at a distance of d from it, to determine another alignment score. The dynamic programming algorithm for alignment will be used. This requires $O(k^2)$ time per substring for a total of $O(k^2 l)$ time, where l is the length of the entire DNA strand.

Folding

Each substring of size k will be folded to determine a folding score. The Nussinov dynamic programming algorithm will be used. This, however, will require $O(k^3)$ time per substring, for a total of $O(k^3 l)$ time, where l is the length of the entire DNA strand. Therefore, a "trick" will be used to reduce this total time to $O(k^2 l)$. Consider a substring of size k, $s_i \dots s_{i+k-1}$. Nussinov algorithm can be used to fold this substring is $O(k^3)$ time. The upper $k \times k$ matrix shown below represents the dynamic programming algorithm for the $s_i \dots s_{i+k-1}$.



Now consider the substring $s_{i+1} \dots s_{i+k}$. Observe that by shifting the upper matrix above one row to the bottom and one column to the right, we obtain the matrix corresponding to the dynamic programming algorithm for $s_{i+1} \dots s_{i+k}$ (the lower matrix in the figure above). This matrix is missing only the last column. The last column can be computed in $O(k^2)$ time since it involves $k-1$ entries, each requiring $O(k)$ time to compute using the Nussinov algorithm.

Therefore, by folding $s_1 \dots s_k$ using $O(k^3)$ time, and folding all subsequent substrings of size k using this shifting technique, the total running time for folding all size k substrings will be $O(k^2 l)$.

Scoring

Each pair of primers will have five scores: two self alignment scores, two folding scores, and one alignment score. These scores, together with a threshold, will be used to decide the "good" primers.

The scoring mechanism used will be based on the following heuristic (which is a good approximation of the real energy values at 37 degrees): $score(G,T) = 1$, $score(A,T) = 2$, $score(G,C) = 3$. All other scores, including gaps for alignments, will be 0.

Further detail

Further detail on the project and implementation can be obtained through discussion between the student and the professor.