Final Exam for CAP 6515: Algorithms in Computational Biology

Distributed on 12/03/2019 4:15 pm and Due on 12/06/2019 4:15 pm at HEC-311

Note:

Please write in detail about your idea, algorithm and analysis of the correctness and computational time for each problem. Your answer will be graded on the following criteria:

1. Your algorithm must be clearly and unambiguously described.
2. You need to prove your algorithm correctly solves the problem. In some cases, correctness is easy or trivial; in this case, your correctness argument can be a short English explanation. Other times, correctness is highly non-trivial and requires a medium-sized mathematical argument. It is your job to distinguish these two cases.
3. Your time analysis must be proved correct. A time analysis is usually an upper bound on the worst-case time. At a minimum, a time analysis requires an explanation of where the calculations come from. If the analysis is easy (e.g., with a simple nested loop algorithm), these explanations can be brief (e.g., The outside loop goes from 1 to n, and each iteration, the inside loop iterates m times, so the overall time is $O(nm)$.). Other times, the time analysis is a tricky, mathematical proof is required.
4. Your algorithm must be efficient. Although we sometimes use polynomial time as a benchmark for efficient, this is not a hard and fast rule. An algorithm is efficient for a problem if there is no competing algorithm that is much faster.

Academic Honesty:

1. Do not discuss the final exam with anyone except the instructor.
2. **If you look for answers to final exam problems in other texts or on the internet, please acknowledge it**.
3. You should treat it as a 3-day in-class exam with only the notes from webcourses.

1 Sequence alignment problem (50 pts)

One disadvantage of the exon chaining formulation of the spliced alignment algorithm is that it may prefer to concatenate many short putative exons to maximize the alignment score. Modify the spliced alignment algorithm (See 3.6 Spliced Alignment) to consider the optimal alignment between a genomic sequence and an mRNA sequence with at most k exons.

2 Random shuffling of DNA sequences (50 pts)

To evaluate the statistical significance of certain properties in DNA sequences, generate many random shuffled DNA sequence instances from one or more given DNA sequences. A simple swapping-based algorithm can generate random sequences preserving the same nucleotide frequencies as the input sequences. In some applications, however, it is also expected to preserve the same k-mer ($k > 1$) frequencies in the shuffled sequences as the input sequences. For example, for $k = 2$, we expect the $4 \times 4 = 16$ dinucleotide frequencies remain the same as input sequences. Given a sequence, devise a *linear-time* sequence shuffling algorithm to generate a precisely uniform instance (radonmly) that preserve the k-mer frequencies. Note that you need discuss the randomness of your algorithm and you need use "rand()" to output a random integer between 0 and RAND_MAX in your algorithm.