Bioinformatics exam questions

(a) Describe with one example the difference between Hamming and Edit distances [2 marks]

See Lecture 2. The Hamming distance is a simple position by position comparison. The Edit distance is formalised through insertion and delition operations. $TGCATAT \rightarrow ATCCGAT$ in 4 steps; TGCATAT (insert A at front); ATGCATAT (delete 6th T); ATGCATA (substitute G for 5th A); ATGCGTA (substitute C for 3rd G); ATCCGAT (Done).

(b) Discuss the Smith-Waterman algorithm. What is the complexity and the relationship with the problem of finding the longest common subsequences? [5 marks]

See Lecture 2. Smith-Waterman Algorithm. Originates from a modification of the Needleman-Wunsch, Idea: Ignore badly aligning regions:

Initialization: $F(0,j) = F(i,0) = 0$;

Iteration: $F(i,j) = max(F(i-1,j)\text{-d},\ F(i,j-1)\text{-d},\ F(i-1,j-1) + s(x_i,\ y_j))$;
Termination: If we want the best local alignment
FOPT $= max_{i,j}F(i,j)$;
If we want all local alignments scoring $> t$
For all i,j find $F(i,j) > t$, and trace back.

Longest Common String algorithm

LCS(v,w)
for $i \leftarrow 1$ to n
$s_{i,0} \leftarrow 0$
for $j \leftarrow 1$ to m
$s_{0,j} \leftarrow 0$
for $i \leftarrow 1$ to n
for $j \leftarrow 1$ to m
$s_{i,j} \leftarrow max(s_{i-1,j}, s_{i,j-1},\ s_{i-1,j-1} + 1, \text{ if } v_i = w_j)$
$\beta_{i,j} \leftarrow (\uparrow \text{ if } s_{i,j} = s_{i-1,j},\ \leftarrow \text{ if } s_{i,j} = s_{i,j-1},\ \nwarrow \text{ if } s_{i,j} = s_{i-1,j-1} + 1)$
$return(s_{n,m}, \beta)$

The main difference is that in the longest subsequence problem we do not allow insertions and deletions.

(c) Describe the Banded algorithm for local alignment and its complexity [5 marks]

See Lecture 2. It is a modification of the Needleman-Wunsch Algorithm; Assume

the sequences we compare, x and y, are very similar. If the optimal alignment of x and y has few gaps, then the path of the alignment will be close to diagonal. Assumption: $gaps(x, y) < k(N)$ (say $N > M$) $x_i = y_j$ implies $|i$ - $j| < k(N)$. Time, Space: $O(N \leftarrow k(N)) << O(N^2)$.

(d) Describe the four Russian speedup algorithm [8 marks]

See Lecture 2 and 3. The time complexity of the dynamic programming global alignment algorithm we've studied previously was $O(n^2)$. We examine a trick to speedup the algorithm to sub-quadratic time. Note that no non-trivial lower bound exists for global alignment and an $O(nlogn)$ is an important achievement. Let's begin by examining the block alignment problem in conjunction with the Four-Russians speedup. The next section extends the intuition here to longest common subsequence (LCS) speedup. Consider our two strings to align: v and w. Without loss of generality, assume that $n = |v| = |w|$ and are divisible by some t. We can partition v and w into chunks of size t. If we were to solve the mini-alignment of each txt sub-grid, we could then perform block alignment of the blocks defined by the partitioning. In other words we construct a path that includes going through a block (from the top left to the bottom right) or along the edges of a block. Thus we are restricting entry and exit to the corners of blocks. The block alignment problem is: Given: Two strings v and w partitioned into blocks of size t. Output: The block alignment of v and w with the maximum score Let $\beta_{i,j}$ be the alignment score for the (i, j) block. The recurrence for the block alignment algorithm is:

$$s_{ij} = max(S_{i-1,j} - \sigma_{block}, S_{i-1,j} - \sigma_{block}, S_{i-1,j} - \sigma_{block})$$

where σ_{block} is the indel block penalty. Since the indices of the recurrence vary from 0 to n , we have an $O(n^2/t)$ algorithm. But computing each block score $\beta_{i,j}$ requires solving n^2/t mini-alignments of size txt which amounts to $O(n^2/t)$ time. Therefore we have not yet achieved any speed improvement. The Four-Russians technique is to set $t = logn/4$ and precompute an exhaustive table of all $4^t x 4^t$ alignments. $4^t x 4^t = n$ total entries in the table. Computing each entry in the table requires $O(log^2 n)$ time, so to compute all n entries in the lookup table requires $O(nlog^2 n)$ time. As noted above, the block alignment recurrence requires $O(n^2/t)$ time. Looking up an element in the lookup table takes $O(t)$. Therefore, given a lookup table, the block alignment algorithm takes $O(n^2/t) = O(n^2/logn)$. We then add the time to compute the t lookup table, but see that the overall time is dominated by the n^2 term. Therefore, the overall running time is: $O(n^2/logn)$.

(e) Describe a bioinformatics application of hidden Markov models [6 marks]

One typical application is that of finding transmembrane segments (protein structure prediction). The most important pattern to identify a transmembrane segment is a long stretch (in general > 6 amino acids) of apolar amino acids. The segments outside the mebrane are enriched with polar aminoacids. So you need two classes. another problem is to identify genes in a long genomic region. Similarly for gene finding problems. A gene has start and end codons and a promoter sequence

upstream the gene. The introns and exons have also a biased composition of DNA bases. using databases of known genes would allow to build transition matrices of each element we would like to identify.

(f) Discuss the properties of the Markov clustering algorithm and the difference with respect to the k-means and hierarchical clustering algorithms [8 marks]

See Lecture 10. MCL algorithm: We take a random walk on the graph described by the similarity matrix and after each step we weaken the links between distant nodes and strengthen the links between nearby nodes.

The k-means algorithm is composed of the following steps: 1) Place K points into the space represented by the objects that are being clustered. These points represent initial group centroids. 2) Assign each object to the group that has the closest centroid. 3) When all objects have been assigned, recalculate the positions of the K centroids. 4) Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.

Hierarchical clustering: Start with each point its own cluster. At each iteration, merge the two clusters; with the smallest distance. Eventually all points will be linked into a single cluster. The sequence of mergers can be represented with a rooted tree.

(g) Describe the Gillespie algorithm and discuss its relationships with genetic or biochemical networks (make one example) [6 marks]

See Lecture 12. The Gillespie algorithm (1977) is an exact simulation algorithm. Step 1: What is the next reaction that occurs? Step 2: when does it occur? The time to the next reaction is $exp(a_0(x))$, where $a_0(x) = \sum a_j(x) and (j = 1, , \mathrm{M})$. The type of reaction will be random, picked with probabilities proportional to $a_j(x)$.

(h) Parameters of the positional independence of bacterial genome ribosomal binding site were estimated by the experimental positional nucleotide frequencies shown in the following table

$$\begin{pmatrix} & 1 & 2 & 3 & 4 & 5 & 6 \\ T & 0.16 & 0.05 & 0.01 & 0.03 & 0.12 & 0.14 \\ C & 0.08 & 0.04 & 0.01 & 0.03 & 0.05 & 0.11 \\ A & 0.68 & 0.11 & 0.02 & 0.90 & 0.16 & 0.51 \\ G & 0.08 & 0.80 & 0.96 & 0.04 & 0.67 & 0.24 \end{pmatrix}$$

Determine the parameters of the logo graph. Use the bit units for the entropy and information content values. [8 marks]

The rules for the logo graph suggest that at position j the sum of the heights of four letters is equal to the information content I_j of the position, and height H_α^j of a letter α is proportional to its probability in this position:

$H_\alpha^j = p_\alpha^j I_j$

$I_j = H_{max} - H_j$

Thus the information content in position j is the difference of the maximum entropy, corresponding to the uniform discrete distribution and the entropy for the nucleotide frequencies of the RBS model at position j. First calculate H_{max} and H_j:

$H_{max} = -\sum_\alpha \frac{1}{4} log_2 \frac{1}{4} = 2,$

$H_1 = -(p_T log_2 p_T + p_C log_2 p_C + p_A log_2 p_A + p_G log_2 p_G)$

Similarly we define the entropy for $H_j, j = 2, .., 6$. Now we can find the information content I_j and the heights of the letters $H_\alpha^j, j = 1, ..., 6$.

(j) Find the optimal pairwise global alignment of the sequences $TACGA\mathbf{G}TACGA$ and $ACTGAC\mathbf{G}ACTGAC$ with the condition that G nucleotides shown in bold font must be aligned together. The scoring parameter are defined as $+2$ for match, -1 for mismatch, and $d = -2$ for a linear gap penalty. [7 marks]

it is easy to see that the middle nucleotides G divide each sequence into two identical subsequences. Hence we find the optimal alignment satisfying the defined above condition will be the concatenation of the two subalignments with a pair of aligned G's between them. We compute See Lecture Notes 2

(i) In modeling a metabolic process describe the advantages and disadvantages of using a stochastic approach (for example agents) with respect to using a set of differential equations [5 marks]

With differential equation we are able to compute the average behavior of a process but we know nothing or little of the variance. The stochastic methods, for example the Gillespie, allow to run simulations of the systems. Only after many simulations we may have a good estimate of average and variance. From a practical purpose we use models that leave out many details of the state of a system (such as the position, orientation and momentum of every single molecule under consideration), in favour of a higher level view. Viewed at this higher level, the dynamics of the system are not deterministic, but intrinsically stochastic. Considering a population X described by an ordinary differential equation

$\frac{dX(t)}{dt} = (\lambda - \mu)X(t)$

the analytical solution $X(t) = x_0 exp(\lambda - \mu)t$ tells us the average behavior of the population size. If the population is composed by bacteria, the problem is that they do not vary in number continuously and deterministically. They vary discretely and stochastically. Moreover, $\lambda - \mu$ controls the essential shape of the process, while $\lambda + \mu$ tells us the degree of noise which will greatly affect the extinction time and

other quantities.

(l) Hidden Markov models (HMM) are widely used in Bioinformatics

(ii) In a HMM when would you use EM (Baum Welsh) and when Viterbi training methods and why. Give biologically motivated examples. [7 marks]

Once the architecture of an HMM has been decided, an HMM must be trained to closely fit the process it models. The most common and straightforward algorithm for HMM training is expectation maximization (EM)1 which adapts the transition and output parameters by continually re-estimating these parameters until $P(O|M)$ has been locally maximized. Expectation maximization (EM) allows maximizing the HMM parameters when the paths through the model for each training sequence are unknown. The standard EM algorithm is the Baum-Welch algorithm. This iterative algorithm has two steps, the expectation step and the maximization step. First, it calculates the expected number of times each transition and emission is used for the training set. Then, the transition and emission parameters are updated using reestimation formulas.

HMM decoding involves the prediction of hidden states given an observed sequence. The problem is to discover the best sequence of states $Q = q_1 q_2 ... q_T$ visited that accounts for an emitted sequence $O = O_1, O_2, ..., O_T$ and a model l. There may be several different ways to define a best sequence of states. A common decoding algorithm is the Viterbi algorithm. The Viterbi algorithm uses a dynamic programming approach to find the most likely sequence of states Q given an observed sequence O and model M. The Viterbi algorithm computes the most likely path through the model. The algorithm employs a matrix; the columns of the matrix are indexed by the states in the model, and the rows are indexed by the sequence. Once the most probable path through the model is known, the probability of a sequence given the model can be computed by multiplying all probabilities along the path. The main computational biology problems with HMM-based solutions are protein family profiling, protein binding site recognition and gene finding in DNA.

See Lecture Notes 6,7 and An Introduction To Bioinformatics Algorithms Neil C. Jones and Pavel A. Pevzner; The MIT Press, 2004, pag. 393-8, par 11.3-4.

(ii) Any machine learning model (such as HMM) for protein secondary structure determination relays on discovering characteristic statistical properties of protein sequences. Name a property that helps to localize (and distinguish) transmembrane segments and coil in a protein sequence or a gene in a genomic region or exon/intron boundaries. [3 marks]

The most important pattern to identify a transmembrane segment is a long stretch (> 6) of apolar amino acids. A gene has start and end codons and a promoter sequence upstream the gene.

(m) Discuss the complexity of an algorithm to reconstruct a genetic network from microarray perturbation data [7 marks]

Reconstruction: O(nka) where k is the average number of entries in the accession list; a is the average number of entries in adjacency list. Large scale experimental gene perturbations in the yeast Saccharomyces cerevisiae (n=6300) suggests that $k < 50$, $a < 1$, and thus that $nka << n^2$. Cycle Reduction: O(nk) See Lecture Notes 12

(n) Give the alignment matrix of the sequences $AATCGCGCGGT$ and $ATGCGCCGT$ assuming the following costs: $Cost(a, a) = 0$; $Cost(a, b) = 3$ when $a \neq b$, $Cost(a, -) = Cost(-, a) = 2$. [10 marks]. How would you set the function Cost in order to compute the longest subsequence common to x and y?

$Cost(a, a) = 1$; $Cost(a, b) = 0$ when $a \neq b$, $Cost(a, -) = Cost(-, a) = 0$.