

**Solutions: 1MB304: Discrete structures for bioinformatics II**

**Date: 2006.10.19**

**Time: 9-14**

**Place: B10:1, BMC**

**Contact: Torgeir R. Hvidsten (tel. 6687)**

Pocket calculator and dictionary allowed. No other aids (books, notes, etc.). Answers can be given in English or Swedish.

## Task 1

### a) (10)

1. Exhaustive algorithms (brute force): examine every possible alternative to find the solution
  - a. Partial digest problem
  - b. Motif finding problem
2. Branch-and-bound algorithms: omit searching through a large number of alternatives by branch-and-bound or pruning
  - a. Partial digest problem
  - b. Motif finding problem
3. Greedy algorithms: find the solution by always choosing the currently "best" alternative
  - a. Genome rearrangements
  - b. Motif finding
  - c. Approximation algorithms
4. Dynamic programming: use the solution of the subproblems of the original problem to construct the solution
  - a. Sequence alignment
  - b. Gene prediction: exon chaining problem
  - c. Hidden Markov models
  - d. RNA folding
5. Machine learning: induce models based on previous labeled observations (examples)
  - a. Hidden Markov models for modeling multiple alignments
6. Randomized algorithms: finds the solution based on randomized choices
  - a. Motif finding problem (Gibbs sampling)

**Correction guidelines:** 1.66 points for each method. 1 point for correct description, 0.66 for correct application. Round upwards to nearest half point.

### b) (10)

For each sequence  $i$ , the algorithm finds the best motif in that sequence given the motifs found in the previous sequences  $1, 2, \dots, i-1$ . Since this don't work for the first sequence (i.e. we would be look at only one sequence, and thus the consensus motif and the actual motif would be the same for all positions, and no motif would beat the motif starting in position one), we repeat this procedure for all starting positions in the first sequence.

**Correction guidelines: 3 points**

The algorithm is an example of a greedy algorithm. Given the starting positions for the previous sequences, it takes the best starting position in the next sequence.

**Correction guidelines: 4 points** (2 point for correct answer and 2 point for correct reason)

The running time is  $O(tm^2)$ .

**Correction guidelines: 3 points**

## Task 2 (30%)

a) (10%) H, S and C are hidden state  $Q$ . They each emit symbols from the 20 letter alphabet of amino acids  $\Sigma$ . The transition matrix  $a_{kl}$  specify the probabilities of moving from one state (secondary structure) to another (or stay in the same secondary structure) and the emission matrix  $e_k(b)$  specify the probabilities of emitting amino acid  $b$  while in the hidden state  $k$ .

**Correction guidelines: 10 points**

b) (10%)

$f_{k,i}$	i=1: C	i=2: G	i=3: C	i=4: G
k=1: I	$1/3 \cdot 1 \cdot 1/2 = 1/6$	$1/3 \cdot (1/6 \cdot 9/10 + 1/8 \cdot 1/10) = 0.0542$	$1/3 \cdot (0.0542 \cdot 9/10 + 0 + 0.0323 \cdot 1/10) = 0.0173$	$1/3 \cdot (0.0173 \cdot 9/10 + 0 + 0.00862 \cdot 1/10) = 0.00548$
k=2: L	$1/4 \cdot 1 \cdot 1/2 = 1/8$	$1/4 \cdot (1/6 \cdot 1/10 + 1/8 \cdot 9/10) = 0.0323$	$1/4 \cdot (0.0542 \cdot 1/10 + 0 + 0.0323 \cdot 9/10) = 0.00862$	$1/4 \cdot (0.0173 \cdot 1/10 + 0 + 0.00862 \cdot 9/10) = 0.00237$

$$P(\text{CGCG}) = 0.00548 + 0.00237 = 0.00785$$

**Correction guidelines: 4 points**

This is the forward algorithm. It computes the probability  $P(\mathbf{x}, \boldsymbol{\pi})$  over all paths  $\boldsymbol{\pi}$ , i.e.  $P(\mathbf{x})$ . The forward algorithm is a dynamic programming algorithm. The time complexity can be expressed as the number of edges in the edit graph:  $O(n|Q|^2)$  where  $n$  is the length of the observation  $\mathbf{x}$  and  $Q$  is the set of hidden states.

**Correction guidelines: 3 points**

The current architecture does not really detect CG-islands. In fact, the sequence GGGG or CCCC have the same probability as CGCG.

**Correction guidelines: 3 points**

b) (10%)

In line 3, the Viterbi algorithm is used. This is a dynamic program algorithm that uses the following recurrence:

$$s_{i+1} = e_i(x_{i+1}) \cdot \max_{k \in Q} \{s_{k,i} \cdot a_{kl}\}.$$

Given a sequence  $\mathbf{x}$ , this algorithm finds a path  $\boldsymbol{\pi}^*$  such that the probability  $P(\mathbf{x}, \boldsymbol{\pi}^*)$  is maximized.

**Correction guidelines: 5 points**

Given the paths in line 3 and the training sequences,  $a_{kl}$  and  $e_k(b)$  can be calculated as follows:

- $a_{kl}$  is the number of transitions from state  $k$  to state  $l$  in the paths divided by the number of transitions from state  $k$  to any state.
- $e_k(b)$  is the number of times the symbol  $b$  is emitted from state  $k$  in the paths divided by the number of times any symbol is emitted from state  $k$ .

**Correction guidelines: 5 points**

### Task 3 (15%)

→ means that we come from  $i, j-1$ , and ↑ means that we come from  $i+1, j-1$

$N_{i,j}$	A, j=1	G, j=2	U, j=3	G, j=4	U, j=5	C, j=6	G, j=7	U, j=8
A, i=1	0	0	0	0	1	→ 1	→ 1	↑ 2
G, i=2		→ 0	0	0	0	↑ 1	→ 1	↑ 2
U, i=3			0	0	0	0	1	→ 1
G, i=4				→ 0	→ 0	→ 0	0	1
U, i=5					0	0	0	0
C, i=6						0	0	0
G, i=7							0	0
U, i=8								0

**Correction guidelines: 8 points** (5 point for the basic scores and 3 for correctly using the “max”-part of the recurrence (i.e. long arrows))



**Correction guidelines: 7 points** (4 points for the first structure, 3 for the two using the “max”-part of the recurrence (1 point for drawing them correctly))

### Task 4 (15%)

a) (5%)

- Homology/comparative modeling: detect a sequence of known structure (template) with high sequence similarity to your sequence of interest (target). Align the target and the template and transfer the structure. Application: When a template of high sequence similarity exists in the database of known structures.
- Fold recognition (threading): Classify the target to a class of protein structures (i.e. fold), and thread the target sequence through the structures in that class. Application: When the fold of the target exists, but where there is very little detectable sequence similarity.
- Ab initio (de novo, new folds) methods: Assemble the structure of the target from a library of building blocks/fragments. Application: Where no sequence similarity can be detected to any sequence of known structure or where the structure of the target does not exist in the structure database.

**Correction guidelines:** 1.66 points for each method. 1 point for correct description, 0.66 for correct application. Round upwards to nearest half point.

b) (10%)

Profile: Construct a multiple alignment of the family and represent it with a profile. The profile can then be used to search the sequence database using the normal pairwise alignment approach.

Profile HMM: Construct a multiple alignment of the family and represent it as a profile hidden Markov model. The HMM can be used to search the sequence database using the forward algorithm and the Viterbi algorithm can be used for alignment.

Note: Since the structures of the proteins in the family are known, the sequence multiple alignment should be constructed from the best structure alignment rather than by aligning sequences.

**Correction guidelines: 5 points** (2 for each method and 1 for realizing that structures should be used to create multiple alignments)

The modeling approach is more sensitive than the pairwise approach, i.e. it can find more distant family member, because a faint sequence similarity is typically not statistically significant in a pairwise alignment, but is significant when aligning to a whole family of sequences. For example, the model will tell us which positions are important (i.e. the conserved positions in the multiple alignment) and which positions that are less important (i.e. positions in the multiple alignment that are not conserved).

**Correction guidelines: 5 points**

## Task 5 (20%)

### a) (15%)

Approximation algorithms are not correct algorithms, that is, they do not produce the correct or optimal output for all inputs.

**Correction guidelines: 5 points**

ExhaustiveReversalSort ( $\pi$ )

```
1   if  $\pi$  is the identity permutation
2       return
3   for each reversal  $\rho(i,j)$   $1 < i < j < |\pi|$ 
4        $\pi^* \leftarrow \pi \cdot \rho(i,j)$ 
5       ExhaustiveReversalSort ( $\pi^*$ )
```

The time complexity is  $O(|\pi|^{2x})$  where  $x$  is the number of reversals in the worst solution.

**Correction guidelines: 5 points** (3+2)

This is a minimization algorithm, so the performance guarantee is the maximum ratio between the output of the algorithm and the optimal output. Since the exhaustive algorithm provides the optimal output, the maximum ratio in the data is for  $\pi_2$ , i.e.  $8/2 = 4$ . So the true performance guarantee is at least 4.

**Correction guidelines: 5 points**

### b) (5%)

Line 3: “Among all possible reversals that reduce  $b(\pi)$ , choose one by random”

**Correction guidelines: 2 points**

The disadvantage of the greedy approach (as always) is that taking the best reversal now might prevent us from finding the optimal solution in the end because the optimal solution requires choosing a worse reversal first in order to be able to select several good reversals later. The randomized algorithm does not suffer from this greediness. By running it several times, the randomized algorithm could improve on the greedy one.

**Correction guidelines: 2 points**

The randomized algorithm is not correct, thus it is a Monte Carlo algorithm.

**Correction guidelines: 1 point**