GENOME-SCALE ALGORITHM DESIGN
by Veli Mäkinen, Djamal Belazzougui, Fabio Cunial and Alexandru I. Tomescu
Cambridge University Press, 2015
www.genome-scale.info

Exercises for Chapter 16. Metagenomics

16.1 Show that Problem 16.1 can be reduced to a minimum-cost network flow problem, by extending the reduction constructed in Section 5.3.3 for Problem 5.7.

16.2 Show that you can reduce Problem 16.1 to one in which every read vertex in A has at least two incident edges.

16.3 Discuss how to modify the statement of Problem 16.1 if the probability of sequencing a read from a certain location inside each genome is no longer uniform for all locations, but follows a different distribution function, which is known beforehand.

16.4 Explain what changes you need to make to the reduction constructed in Exercise 16.1 if the objective function in Problem 16.1 is

$$(1-\alpha)\sum_{(x,y)\in M} c(x,y) + \alpha\sum_{i=1}^{m}\sum_{j=1}^{s_i}\left(\frac{c_i}{s_i} - d_M(y_{i,j})\right)^2. \tag{1}$$

Hint. Recall Exercise 5.16.

16.5 What can you say about the complexity of Problem 16.2 for $\alpha = 0$? What about $\alpha = 1$?

16.6 Describe an algorithm to compute the BWT of the reads that are kept and of the reads that are filtered out in Section 16.2.1, by reusing the BWT of the original file. Recall that reads in the input file R are separated by distinct binary strings on the artificial characters $\#_0$ and $\#_1$ which do not belong to $[1..\sigma]$.

16.7 Given the concatenation R of all reads in a metagenomic sample as defined in Section 16.2, consider the interval of a k_2-mer W in BWT_R. Section 16.2.2 claims that, during the creation of pre-clusters, we can discard any k_2-mer V whose interval in BWT_R coincides with the interval of aW for some $a \in [1..\sigma]$. Prove this claim, and describe an algorithm that implements it in linear time using just one scan of the bitvector `intervals`.

16.8 With reference to Section 16.2.2, describe an algorithm to create pre-clusters in which two reads are merged if they share *at least two* distinct k_2-mers. Assume that you have $|R|\log r$ bits of main memory in addition to the space required by Lemma 16.5, and assume that you have an algorithm that sorts tuples in external memory as a black box. Describe how to implement additional conditions on top of your algorithm, for example that the two shared k_2-mers are in the same order in both reads, that they do not overlap, or that they are offset by exactly one position.

16.9 With reference to Section 16.2.2, describe an algorithm to create pre-clusters in which two reads are merged if they share a k_2-mer *or its reverse complement.* Note that in this case we cannot discard k_2-mers that occur exactly once in R, since they could occur twice in $R \cdot \underline{R}$. Describe a solution that takes $K + K' \log r$ bits of space in addition to the bidirectional BWT index of R, where K is the number of distinct k_2-mers in R, and K' is the number of distinct k_2-mers that occur at least twice in $R \cdot \underline{R}$. This algorithm should perform $k_2 + 1$ operations for every position of the input file. Adapt your solution to filter out reads such that all their k_1-mers occur less than τ_1 times in $R \cdot \underline{R}$, as described in Section 16.2.1. Finally, describe another solution for creating pre-clusters that uses string $R \cdot \underline{R}$ as input and that takes approximately twice the time and the space of Lemma 16.5.

16.10 Describe a way to parallelize the algorithm in Lemma 16.5 by using samples of SA_R (see Section 9.2.3).

16.11 Adapt the algorithm for document counting described in Section 8.4.2 such that, given a reference taxonomy $\mathcal{T}$, it computes the markers of every taxon $v \in \mathcal{T}$.

16.12 Let $\mathsf{MS}_{S,T,\tau}$ be the following generalization of the matching statistics vector described in Section 11.2.3: $\mathsf{MS}_{S,T,\tau}[i]$ is the longest prefix of suffix $S[i..|S|]\#$ that occurs at least τ times in T. Similarly, let $\mathsf{SUS}_{T,\tau}[i]$ be the shortest prefix of $T[i..|T|]\#$ that occurs at most τ times in T. Describe an algorithm that computes the markers of every taxon v in a reference taxonomy $\mathcal{T}$ using the shortest unique substring vector $\mathsf{SUS}_{S,\tau}$, the matching statistics vector $\mathsf{MS}_{S,T,\tau+1}$, and a bitvector $\mathtt{flag}_{S,T,\tau}[1..|S|]$ such that $\mathtt{flag}_{S,T,\tau}[i] = 1$ if and only if $S[i..i + \mathsf{MS}_{S,T,1}[i] - 1]$ occurs at most τ times in T, for every genome $T \neq S$ in the database.

16.13 Prove that the number of k-submaximal repeats of a string S is upper-bounded by the number of distinct k-mers that occur at least twice in S.

16.14 With reference to Insight 16.1, prove that attaching the root of the smallest subtree to the root of the largest subtree guarantees $O(\log n)$ time for `find`.

16.15 With reference to Section 16.2.3, give the pseudocode of an algorithm that extracts all the maximal repeats of length at least k_4 from all pre-clusters, using a single enumeration of all the internal nodes of the suffix tree of R.

16.16 With reference to Section 16.2.3, describe a way to approximate $\mathcal{R} \otimes \mathcal{Q}$ that is more accurate than just computing $\mathcal{R} \odot \mathcal{Q}$. *Hint.* Use the operator $\odot$ iteratively on suitable subsets of $\mathcal{R}$ and $\mathcal{Q}$. Then, give the pseudocode of an algorithm that, given vector MS_{R^i} for a read R^i, finds a set of nonoverlapping substrings of length k of maximum size.