GENOME-SCALE ALGORITHM DESIGN
by Veli Mäkinen, Djamal Belazzougui, Fabio Cunial and Alexandru I. Tomescu
Cambridge University Press, 2nd edition, 2023
www.genome-scale.info

Exercises for Chapter 14. Haplotype analysis

14.1 Complete the proof of Theorem 14.1, by proving that, if A and B are two haplotypes obtained with $|V||E| - 2t$ bit flips from M, then the set $C = \{i \mid A[2i-1] = A[2i] = 0\} \subseteq V$ has the property that there are t edges between C and $V \setminus C$. Obtain this proof by showing the following intermediary properties:

- $A[2i-1] = A[2i]$, for all $i \in \{1, \ldots, |V|\}$;
- $B[2i-1] = B[2i]$, for all $i \in \{1, \ldots, |V|\}$;
- A is the bit-wise complement of B.

14.2 Show that the algorithm given for the haplotype assembly in the framework of the minimum error correction problem can be optimized to run in $O((2^K + r)s)$ time. How could you get the time complexity down to $O(2^K s)$?

14.3 Show how the algorithm given for the haplotype assembly in the minimum error correction problem can be implemented to use $O(2^K \sqrt{s})$ memory, while maintaining the same asymptotic time complexity. *Hint:* Use the same idea as in Exercise 7.2.

14.4 Let T denote the maximum number of bit flips needed in any column of the matrix M in an optimal solution to Problem prob:haplotypeassembly. Show that the algorithm given for this problem can be modified to run in time $O((K^T + r)s)$, and even in time $O(K^T s)$ (see also Exercise 14.2 above).

14.5 Consider the problem of haplotype assembly under minimum error correction in which each cell of M has also a cost of flipping, and one is looking for a partition of the rows of M that is compatible with two haplotypes and minimizes the sum of the costs of all bit flips. What changes to the algorithm we presented are needed in order to solve this more general problem with the same time complexity?

14.6 Recall the definition of matching statistics from Section 11.3.3. We are interested in the related notion of *positional matching statistics*: given an $m \times n$ haplotype matrix P and a query haplotype sequence S of length n, compute the array $\mathsf{pMS}_{S,P}[1..n]$ that stores at position j the length of the longest suffix of $S[1..j]$ that equals $P[i..j]$ for some $i \leq j$. Describe how to compute $\mathsf{pMS}_{S,P}$ using the prefix and divergence arrays from Section 14.2.2. What is the time complexity of your algorithm?