Integer programming approach to the haplotyping problem

Tam Truong, Roland Faure, Rumen Andonov

Univ. Rennes, INRIA RBA, CNRS UMR 6074, Rennes, France
{roland.faure,rumen.andonov}@irisa.fr

Mots-clés : *bioinformatics, genome assembly, bipartite graph, weighted vertex cover*

1 Introduction

Given a huge amount of small sequences, called *reads*, that are fragments of a base-paired DNA sequence (defined on the alphabet {A,C,G,T}), the task of genome assembly is to stitch all reads together to reconstitute the original DNA sequence. This task is made harder by the fact that reads have a certain error rate, typically of a few percents. In particular, when two regions of the DNA have very similar sequences, the differences between the two regions are often discarded as errors and the assembler outputs only one sequence. The situation arises often in multiploid organisms because homologous chromosomes are highly similar. In this context, distinguishing the DNA sequences of the different chromosomes is known as the *haplotyping problem.*

The existing algorithms to solve the haplotyping problem are still limited in the number of collapsed similar regions, in speed or in precision [1, 2, 3]. We propose a novel modeling of the problem using integer programming techniques and present numerical results using Gurobi solver.

2 Our approach

A set of reads aligning together come from a genomic region, which can possibly correspond to several highly similar sequences (which we will call *versions*). Our objective is to cluster these reads by version of origin.

A *SNP* (single nucleotide polymorphism) is defined as a base which differs from the majority of other bases at a given position (see for illustration figure (1)). The first step is to select positions with high number of SNPs : when many reads contain a SNP at position x, it may be an indication that there exist acutally several versions that differ at position x. However, because of the non-uniform error rate of the reads, it is possible that a high proportion of reads contain a sequencing error at a given position. The truly unlikely phenomenon is when the same set of reads simultaneously contain the same SNPs at many different positions. This indicates that the set of reads with SNPs actually comes from a different version, and that the SNPs come from the differences between the versions and not sequencing errors. We try to capture this signal using integer linear programming approach, using the model defined below.

3 Integer programming modelling

Given a matrix $A \in Z_2^{n \times m}$ with 0/1 coefficients with $n = |R|$ rows and $m = |P|$ columns (where R denotes reads, while P denotes positions), the goal is to find a partition of the rows in subsets (clusters) in a way that rows in the same cluster are “similar” in respect to their 0/1 coefficients. Towards this goal, we search for the largest sub-matrix of A containing mostly 0 coefficients (small percentage of errors is acceptable).

We introduce a bipartite graph $G = (R \cup P, E)$ where the set R(resp. P) corresponds to the rows(resp. columns) of the matrix A. The edges $e_{(uv)} \in E$ are associated with coefficients of value 1 in the matrix A. We study two models for the above problem. In the first one we solve a vertex weighted version of the minimum vertex cover problem where we minimize the number of zeros while covering all ones/edges. Here covering a matrix coefficient means deleting ether the corresponding row, or column. Hence the remaining matrix contains only cells with value zero. In the second model, the acceptable error is taken into account. However, this model requires more variables. Both models have been implemented with Python and the PuLP package and computational results will be presented.

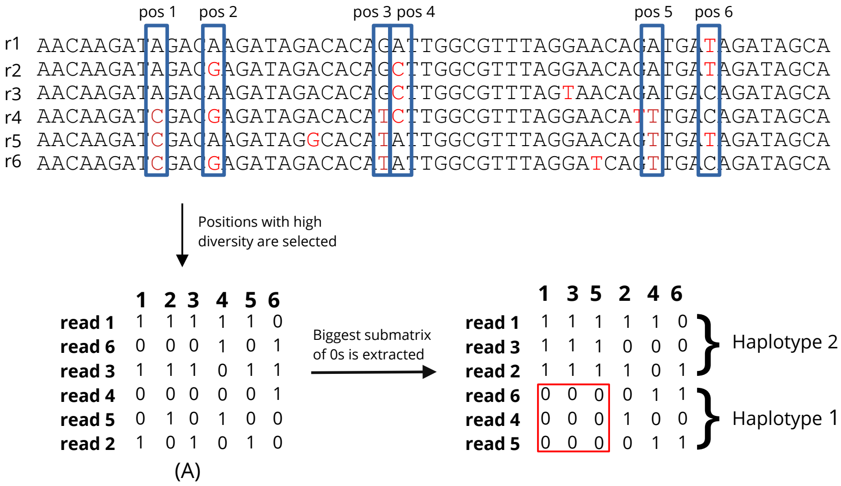
pos 1
pos 2
pos 3
pos 4
pos 5
pos 6
r1 AACAAGATAGACAAGATAGACACAGATTGGCGTTTAGGAACAGATGATAGATAGCA
r2 AACAAGATAGACGAGATAGACACAGCTTGGCGTTTAGGAACAGATGATAGATAGCA
r3 AACAAGATAGACAAGATAGACACAGCTTGGCGTTTAGTAACAGATGACAGATAGCA
r4 AACAAGATCGACGAGATAGACACATCTTGGCGTTTAGGAACATTTGACAGATAGCA
r5 AACAAGATCGACAAGATAGGCACATATTGGCGTTTAGGAACAGTTGATAGATAGCA
r6 AACAAGATCGACGAGATAGACACATATTGGCGTTTAGGATCAGTTGACAGATAGCA
Positions with high diversity are selected
1 2 3 4 5 6
read 1 1 1 1 1 1 0
read 6 0 0 0 1 0 1
read 3 1 1 1 0 1 1
read 4 0 0 0 0 0 1
read 5 0 1 0 1 0 0
read 2 1 0 1 0 1 0
(A)
Biggest submatrix of 0s is extracted
1 3 5 2 4 6
read 1 1 1 1 1 1 0
read 3 1 1 1 0 0 0
read 2 1 1 1 1 0 1
read 6 0 0 0 0 1 1
read 4 0 0 0 1 0 0
read 5 0 0 0 0 1 1
Haplotype 2
Haplotype 1

FIG. 1 – The reads sequenced from to the same (or several very similar) region(s) are aligned. Blue boxes indicate positions where there is no clear consensus. Bases that diverge from the "consensus" base are colored in red, noted by zero coefficient in the corresponding matrix A. Our approach consists in : i) extracting the largest sub-matrix containing mostly 0 coefficients (small percentage of errors is acceptable) ; ii) partitioning the reads in two groups according to the sub-matrix.

4 Results

We applied the model to solve several actual instances of the *haplotyping problem*, based on real biological data from sequencing of different strains of *E. coli*. We show that the method can successfully separate reads into a given number of haplotypes (≤ 4 in our tests). Our future line of work will be to assess how competitive this model can be to recover haplotypes in real assembly pipelines compared to other approaches.

Références

[1] Bansal Vikas, Bafna Vineet. HapCUT : an efficient and accurate algorithm for the haplotype assembly problem. *Bioinformatics*, 24(16) :i153-9, 2008.

[2] Murray Patterson, Tobias Marschall, Nadia Pisanti, Leo van Iersel, Leen Stougie, Gunnar W. Klau, Alexander Schönhuth. WhatsHap : Weighted Haplotype Assembly for Future-Generation Sequencing Reads. *Journal of Computational Biology*, 22(6)498–509, 2015.

[3] Sven D. Schrinner, Rebecca Serra Mari, Jana Ebler, Mikko Rautiainen, Lancelot Seillier, Julia J. Reimer, Björn Usadel, Tobias Marschall, Gunnar W. Klau. Haplotype threading : accurate polyploid phasing from long reads. *Genome Biology*, 21(252), 2020.