SNPs Problems, Complexity, and Algorithms

Giuseppe Lancia[1,2], Vineet Bafna[1], Sorin Istrail[1],
Ross Lippert[1], and Russell Schwartz[1]

[1] Celera Genomics, Rockville MD, USA,
{Giuseppe.Lancia,Vineet.Bafna,Sorin.Istrail,Ross.Lippert,
Russell.Schwartz}@celera.com
[2] D.E.I., Università di Padova, Padova 35100, Italy

Abstract. Single nucleotide polymorphisms (SNPs) are the most frequent form of human genetic variation. They are of fundamental importance for a variety of applications including medical diagnostic and drug design. They also provide the highest–resolution genomic fingerprint for tracking disease genes. This paper is devoted to algorithmic problems related to computational SNPs validation based on genome assembly of diploid organisms. In diploid genomes, there are two copies of each chromosome. A description of the SNPs sequence information from one of the two chromosomes is called SNPs haplotype. The basic problem addressed here is the Haplotyping, i.e., given a set of SNPs prospects inferred from the assembly alignment of a genomic region of a chromosome, find the maximally consistent pair of SNPs haplotypes by removing data "errors" related to DNA sequencing errors, repeats, and paralogous recruitment. In this paper, we introduce several versions of the problem from a computational point of view. We show that the general SNPs Haplotyping Problem is NP–hard for mate–pairs assembly data, and design polynomial time algorithms for fragment assembly data. We give a network–flow based polynomial algorithm for the Minimum Fragment Removal Problem, and we show that the Minimum SNPs Removal problem amounts to finding the largest independent set in a weakly triangulated graph.

1 Introduction

1.1 Motivation

The large–scale laboratory discovery and typing of genomic sequence variation presents considerable challenges and it is not certain that the present technologies are sufficiently sensitive and scalable for the task. Computational methods that are intertwined with the experimental technologies are emerging, leading the way for this discovery process.

Single nucleotide polymorphisms (SNPs) are the most frequent form of human genetic variation and provide the highest–resolution genomic fingerprint for tracking disease genes. The SNPs discovery process has several stages: sample collection, DNA purification, amplification of loci, sequence analysis, and data management. The "large–scale" dimension of the analysis refers to either the

large number of loci or of individuals. Effective integration of these stages is important for the strategies employed in the pipeline. The choice of method, for any stage, could be influenced by the processes used in other stages.

The SNPs discovery involves the analysis of sequence differences and haplotype separation from several samples of a genomic locus from a given population. The leading two methods are based on Shotgun Genome Assembly (SGA) and on PCR amplification (PCR). The methods have complementary strengths and recognized computational bottlenecks associated with them. The SGA, in principle, generates haploid genotyping, and does not require sequence information for the loci, however, it needs good library coverage, and is computationally very challenging to distinguish paralogous repeats from polymorphism. The PCR method requires the knowledge of the genomic region of the locus, and could be done very effectively; however, it is expensive for large-scale projects.

There is need for powerful computational approaches that are sensitive enough and scalable so that they can remove noisy data and provide effective algorithmic strategies for these technologies. This paper is a first step towards such "computational SNPology". It is devoted to algorithmic problems related to computational SNPs discovery and validation based on genome assembly. The basic problem is to start from a set of SNPs prospects inferred from the assembly alignment and to find out the maximal consistent subset of SNPs by removing "errors" related to sequencing errors, repeats, and paralogous recruitment.

1.2   Preliminaries

Recent whole–genome sequencing efforts have confirmed that the genetical makeup of humans is remarkably well–conserved, and small regions of differences are responsible for our diversities. The smallest possible region consists of a single nucleotide, and is called Single Nucleotide Polymorphism, or SNP ("snip"). This is a position in our genome at which we can have one of two possible values (alleles), while in the neighborhood of this position we all have identical DNA content. Since our DNA is organized in pairs of chromosomes, for each SNP we can either be homozygous (same allele on both chromosomes) or heterozygous (different alleles). Independently of what the actual different alleles at a SNP are, in the sequel we will denote the two values that each SNP can take by the letters A and B. A chromosome content projected on a set of SNPs (or *haplotype*), is then simply a string over the alphabet {A, B}, while a *genotype* is a pair of such strings, one for each haplotype.

DNA sequencing techniques are restricted to small, overlapping fragments. Such fragments can contain errors (e.g., due to low quality reads), and can come from either one of the two chromosome copies. Further, e.g. in shotgun sequencing, some pairs of these fragments (*mate pairs*) are known to come from the same copy of a chromosome and to have a given distance between them. The basic problem is then the following: *"Given a set of fragments obtained by DNA sequencing from the two copies of a chromosome, reconstruct the two haplotypes from the SNPs values observed in the fragments."*

Fig. 1. (a) M. (b) $G_{\mathcal{F}}(M)$.   (c) $G_{\mathcal{S}}(M)$ (d) An odd cycle of fragments.

Note that it is possible, even in an error–free scenario, that the above problem cannot be solved because the information is insufficient. For instance, if a set of fragments does not share any SNPs with any of the remaining fragments, we may only be able to reconstruct partial haplotypes, but then we wouldn't know how to merge them into only two (a problem known as *phasing*). It is better then to relax the requirement from "reconstruct the two haplotypes" to "reconstruct two haplotypes that would be compatible with all the fragments observed". Stated in this form, the problem becomes trivial for the error–free case (as we will see in the sequel, it is simply the problem of determining the two shores of a bipartite graph). However, experiments in molecular biology are never error–free, and, under a general parsimony principle, we are led to reformulate the problem as *"Find the smallest number of errors in the data so that there exist two haplotypes compatible with all the (corrected) fragments observed."*

Depending on the errors considered, we will define several different combinatorial problems. "Bad" fragments can be due either to contaminants (i.e. DNA coming from a different organism than the actual target) or to read errors (i.e. a false A, a false B, or a - inside a fragment, which represents a SNP whose value was not determined). A dual point of view assigns the errors to the SNPs, i.e. a "bad" SNP is a SNP for which some fragments contain read errors. Correspondingly, we may define the following optimization problems: *"Find the minimum number of fragments to ignore"*, or *"Find the minimum number of SNPs to ignore"*, so that *"the (corrected) data is consistent with the existence of two haplotypes measured by error–free DNA sequencing. Find such haplotypes."*

1.3   Notation

The basic framework for SNPs problems is as follows. There is a set $\mathcal{S} = \{1, \ldots, n\}$ of snips and a set $\mathcal{F} = \{1, \ldots, m\}$ of fragments. Each snip is covered by some of the fragments, and can take the values A or B. Hence, a snip i is defined by a pair of disjoint subsets of fragments, A_i and B_i. There is a natural (canonical) ordering of the snips, given by their physical location on the chromosome, from left to right. Then, the data can also be thought of as an $m \times n$ matrix over the alphabet $\{\texttt{A}, \texttt{B}, -\}$, which we call the *SNP matrix*, defined in the obvious way.

The Fragment– and Snip– conflict graphs. We say that two fragments i and j are *in conflict* if there exists a snip k such that $i \in A_k$, $j \in B_k$ or $i \in B_k$, $j \in A_k$. Given a SNP matrix M, the *fragment conflict graph* is the graph $G_{\mathcal{F}}(M) = (\mathcal{F}, E_{\mathcal{F}})$ with an edge for each pair of fragments in conflict. Note that if M is error–free, $G_{\mathcal{F}}(M)$ is a bipartite graph, since each haplotype defines a shore of $G_{\mathcal{F}}(M)$, made of all the fragments coming from that haplotype. Conversely, if $G_{\mathcal{F}}(M)$ is bipartite, with shores H_1 and H_2, all the fragments in H_1 can be merged into one haplotype and similarly for H_2. We call a SNP matrix M *feasible* (and *infeasible* otherwise) if $G_{\mathcal{F}}(M)$ is bipartite, and we call the haplotypes obtained by merging the fragments on each shore *derived* from M. For K a set of rows (fragments), we denote by $M[K]$ the submatrix of M containing only the rows in K. The fundamental underlying problem in SNPs haplotyping is determining an optimal set of changes to M (e.g., row– and/or column– deletion) so that M becomes feasible. We remark that $G_{\mathcal{F}}(M)$ is the union of n complete bipartite graphs, one for each column j of M, with shores A_j and B_j.

We now turn to snip conflicts. We say that two snips i and j are in conflict if A_i, B_i, A_j, B_j are all nonempty and there exist two fragments u and v such that the submatrix defined by rows u and v and columns i and j has three symbols of one type (A or B) and one of the opposite (B or A respectively). Given a SNP matrix M, the *snip conflict graph* is the graph $G_{\mathcal{S}}(M) = (\mathcal{S}, E_{\mathcal{S}})$, with an edge for each pair of snips in conflict. In section 2.2 we will state the fundamental theorem relating the two conflict graphs.

In this paper we are going to define the following optimization problems:

- MFR (*Minimum Fragment Removal*): Given a SNP matrix, remove the minimum number of fragments (rows) so that the resulting matrix is feasible.
- MSR (*Minimum Snip Removal*): Given a SNP matrix, remove the minimum number of snips (columns) so that the resulting matrix is feasible.
- LHR (*Longest Haplotype Reconstruction*): Given a SNP matrix, remove a set of fragments (rows) so that the resulting matrix is feasible, and the sum of lengths of the derived haplotypes is maximized.

A *gapless* fragment is one covering a set of consecuitive SNPs. We say that a fragment has k gaps if it covers $k + 1$ blocks of consecutive SNPs. Such a fragment is equivalent to $k + 1$ gapless fragments with the constraint that they must all be put in the same haplotype or all discarded. Particularly important is the case $k = 1$, which is equivalent to 2 gapless fragments coming from the same chromosome. This is the case of *mate pairs*, used for shotgun sequencing [7]. In the remainder of the paper we show that the above problems are NP–hard in general. Furthermore, we show that MFR is NP–hard if even a single gap per fragment is allowed and MSR is NP–hard for fragments with two gaps. On the positive side, we study the special case of gapless fragments, and show that in this case the problems can be solved effectively. We provide polynomial algorithms for MFR, MSR and LHR. Note that the gapless case arises often in practical applications. For space limitations, some of the proofs are omitted in the sequel.

2   Getting Started: The Gapless Case

The simplest scenario for the SNPs haplotype reconstruction problem is when
the fragments are consecutive, gapless genomic regions. This is not an unreal-
istic situation, since it arises, for example, in EST (Expressed Sequence Tags)
mapping. This is in fact the case without mate pairs and with no missed SNPs
inside a fragment because of thresholding or base–skipping read errors.

2.1   The Minimum Fragment Removal

In this section we show that in the gapless case, the minimum fragment removal
(MFR) problem can be solved in polynomial time. For this section, we assume
that there are no fragment inclusions, i.e., denoting by f_i and l_i the first and
last snip of a fragment i, $f_i \leq f_j$ implies $l_i \leq l_j$. We define a directed graph
$D = (\mathcal{F}, A)$ as follows. Given two fragments i and j, with $f_i \leq f_j$, there is an
arc $(i, j) \in A$ if i and j can be aligned without any mismatch, i.e., they agree in
all their common snips (possibly none). Note that the common snips are a suffix
of i and a prefix of j.

Lemma 1. *Let M be a SNP matrix, and P_1, P_2 be node–disjoint directed paths
in D such that $|V(P_1)| + |V(P_2)|$ is maximum. Let $R = \mathcal{F} - (V(P_1) \cup V(P_2))$.
Then R is a minimum set of fragments to remove such that $M[\mathcal{F} - R]$ is feasible.*

Theorem 1. *There is a polynomial time algorithm for finding P_1 and P_2 in D
such that $|V(P_1)| + |V(P_2)|$ is maximum.*

Proof. We will use a reduction to a maximum cost flow problem. We turn D
into a network as follows. First, we introduce a dummy source s, a dummy sink
t, and an arc (t, s) of capacity 2 and cost 0. s is connected to each node i with an
arc (s, i) of cost 0, and each node i gets connected to t, at cost 0 and capacity 1.
Then, we replace each node $i \in D$, with two nodes i' and i'' connected by an arc
(i', i'') of cost 1 and capacity 1. All original arcs (u, v) of D are then replaced by
arcs of type (u'', v'). A maximum cost circulation can be computed in polynomial
time, by, e.g., Linear Programming. Since D is acyclic, the solution is one cycle,
which uses the arc (t, s) and then splits into two s–t dipaths, saturating as many
arcs of type (i', i'') as possible, i.e. going through as many nodes as possible of
D. Since the capacity of arcs (i', i'') is 1, the paths are node–disjoint.

With a similar reduction, we can show that the problem LHP can also be
solved in polynomial time. The problem consists in finding two haplotypes of
maximum total length (where the length of an haplotype is the number of SNPs
it covers). We use a similar reduction as before, with the same capacities, but
different costs for the arcs. Now the arcs of type (i', i'') have cost 0, while an
arc (i'', j') has cost equal to the number of SNPs in j that are not also in i
(e.g., an arc $(-\texttt{ABB}, - - \texttt{BBABA})$ has cost 3). Arcs (s, i') have cost equal to the
number of SNPs in i. An s–t unit flow in this network describes a path that goes
through some fragments, such that the total length of the SNPs spanned (i.e. of
the haplotype) is equal to the cost of the path. Hence, the max cost circulation
individues two haplotypes of max total length. This proves the following

Theorem 2. *The LHP for gapless fragments is polynomial.*

2.2   The Minimum Snip Removal

Theorem 3. *Let M be a gapless SNP matrix. Then $G_{\mathcal{F}}(M)$ is a bipartite graph if and only if $G_{\mathcal{S}}(M)$ is an independent set.*

Proof. (If) Consider a cycle of fragments in a SNP matrix (see Fig. 1d). Wlog, assume the cycle involves fragments $0, 1, \ldots, k$. For each pair of consecutive fragments i, $i + 1 \pmod{(k + 1)}$ there is a position u_i at which one has an A and the other a B. We associate to a fragment cycle a directed cycle between entries in the matrix, made of horizontal arcs from u_{i-1} to u_i in fragment i, and vertical arcs from u_i in fragment i to u_i in fragment $i + 1$. We call a *vertical line* a maximal run of vertical arcs in such a cycle. In a vertical line, the letters A and B alternate. By definition, an infeasible SNP matrix contains an odd cycle of fragments. Let us call *weight* of an infeasible SNP matrix the minimum number of vertical lines of any odd cycles of fragments it contains.

Assume there exists an infeasible gapless SNP matrix M such that $G_{\mathcal{S}}(M)$ is an independent set, and pick M to have minimum weight among all such M. Consider an odd cycle in M achieving this weight. Since an infeasible matrix cannot have weight 1 there are at least two vertical lines, and hence a "rightmost" vertical line, say between fragments f and g. Since the line is rightmost, $u_{f-1}, u_g \leq u_f$. Assume $u_g \geq u_{f-1}$ (same argument if $u_{f-1} \geq u_g$). Since M is gapless, there exists a symbol at row f, column u_g. The symbols M_{f,u_f} and M_{g,u_f} are the same if and only if M_{f,u_g} and M_{g,u_g} are the same (otherwise, the rows f and g individue a snip conflict of columns u_g, u_f). Now, consider the SNP matrix M' obtained by M by first deleting (i.e., replacing with gaps) all the symbols between rows f and g (excluded), and then inserting an alternating chain of As and Bs, starting with M'_{g,u_g}, between rows f and g in column u_g. M' is an infeasible gapless SNP matrix of weight at least one smaller than M. Further, there are no snip conflicts in M' that were not already in M, so $G_{\mathcal{S}}(M')$ is an independent set. Hence, M was not minimal.

(Only if) We omit the simple proof.

Note that, in the presence of gaps, only the "only if" part of the theorem holds. We now show that the Minimum SNP removal problem can be solved in polynomial time on gapless SNP matrices. In particular, we prove that $G_{\mathcal{S}}(M)$ is a perfect graph. The basic tool to do this is the following: if I are the nodes of a hole or a antihole in $G_{\mathcal{S}}(M)$, and $\{i, j\}$ is a conflict, with $i, j \in I$, then for any $k \in I$ such that column k is between columns i and j in M, some relations of k with either i or j are forced. This will allow us to forbid long holes and antiholes.

Lemma 2. *Let M be a gapless SNP matrix and c_1, c_2, c_3 be snips (columns of M) with $c_1 \leq c_2 \leq c_3$. If $\{c_1, c_3\}$ is a snip conflict, then at least one of $\{c_1, c_2\}$ and $\{c_2, c_3\}$ is also a snip conflict.*

Proof. There are two rows r_1 and r_2 such that the 2×2 submatrix induced by rows r_1, r_2 and columns c_1, c_3 has three symbols of one type and one of the opposite. In this submatrix call a 2×1 column of identical symbols type I and one of different symbols type D. Since M is gapless, c_2 must be either I or D. But one of c_1 and c_3 is I and the other D.
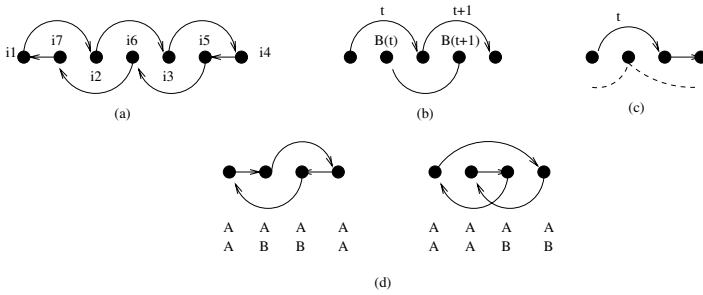
Fig. 2. (a) A cycle without long jumps. (b) two jumps in a row. (c) a jump and a shift. (d) The only possible holes.

Lemma 3. *Let M be a gapless SNP matrix and c_1, c_2, c_3, c_4 be snips with $c_1 \leq c_2 \leq c_3 \leq c_4$. Assume $\{c_1, c_4\}$ is a snip conflict. Then, if $\{c_2, c_3\}$ is not a conflict, one of c_1 and c_4 conflicts with both c_2 and c_3. If $\{c_2, c_3\}$ is a conflict, then the conflicts in $\{c_1, c_2, c_3, c_4\}$ contain a length–4 cycle.*

Lemma 4. *If M is a gapless SNP matrix, $G_{\mathcal{S}}(M)$ does not have a chordless cycle of length > 4.*

Proof. Assume $C = (i_1, \ldots, i_k, i_{k+1} = i_1)$ is a chordless cycle, $k \geq 4$, in $G_{\mathcal{S}}(M)$ (i.e. the columns of M, listed in the order defined by the cycle). For $t = 1, \ldots, k$, let $B(t)$ be the set of nodes in C which lie between i_t and i_{t+1} as columns of M. We say that t *hits* i_t and i_{t+1}. We call t a *long jump* if $|B(t)| > 1$, a *jump* if $|B(t)| = 1$ and a *shift* if $|B(t)| = 0$. For a jump, we denote by b_t the node such that $B(t) = \{b_t\}$. The following facts are true: (*i*) If C has a long jump, it must be $k = 4$. (*ii*) If C has no long jump, each jump t must be followed by a shift, pointing to the node b_t.

To prove (*i*), consider a long jump, and let $B(t) = \{a_1, a_2, \ldots\}$. By lemma 3, if $\{a_1, a_2\}$ is not a conflict, that either i_t or i_{t+1} would have degree ≥ 3 in C, impossible. Hence it must be a conflict. So, by lemma 3, $\{i_1, a_1, a_2, i_k\}$ must contain a cycle, and hence $k = 4$, since C cannot contain other cycles.

To prove (*ii*), note that if t is a shift for $t = 1, \ldots, k - 1$, then k must be a long jump. Hence, if there are no long jumps, there must be jumps (see Fig. 2a for an example of a generic such cycle). Now, assume that a jump t is followed by another jump. Then (see Fig. 2b), since neither b_t nor b_{t+1} can be hit by a long jump, there must be a jump hitting b_t and b_{t+1}. But, then lemma 2 applies to b_t, i_{t+1}, b_{t+1}, so that i_{t+1} would have degree 3 in C. Hence, the jump t is followed by a shift. If the shift points away from b_t, then b_t should be hit by a long jump (see Fig. 2c), impossible. But then, the only possible hole is C_4.

Note that C_4 is actually achieved by some matrices (see Fig. 2d).

The following lemma generalizes lemma 3.

Lemma 5. *Let M be a gapless SNP matrix and c_1, c_2 be snips, with $c_1 < c_2$ and $\{c_1, c_2\}$ a snip conflict. Let $(a_1, a_2, \ldots, a_t)$ be a path in $G_{\mathcal{S}}^c(M)$, such that*

$c_1 < a_i < c_2$ for all $i = 1, \ldots, t$. If $\{c_1, a_1\}$ is not a snip conflict, then $\{a_t, c_2\}$ is a snip conflict. If $\{c_2, a_1\}$ is not a snip conflict, then $\{a_t, c_1\}$ is a snip conflict.

Proof. Since M is gapless, in the $2 \times n$ submatrix of M for which (wlog) c_1 is type I and c_2 is type D, $a_1, \ldots, a_t$ must be I or D. We prove the first part of the lemma, the other being the same argument. Let $a_0 := c_1$. Then a_i is type I for all $i = 1, \ldots, t$ to avoid a conflict of a_{i-1} and a_i. This forces the conflict $\{a_t, c_2\}$.

Lemma 6. *If M is a gapless SNP matrix, $G_S^c(M)$ does not have a chordless cycle of length > 4.*

Proof. Let us call a path in $G_S^c(M)$ an *antipath* and a cycle in $G_S^c(M)$ an *anticycle*. Assume $(i_1, i_2, \ldots, i_k, i_{k+1} = 1)$ is a chordless anticycle of length $k \geq 5$. Let $i_x = \min\{i_1, \ldots, i_k\}$ and $i_y = \max\{i_1, \ldots, i_k\}$. $\{i_x, i_y\}$ cannot be a snip conflict, or otherwise the part of the anticycle from i_x to either i_{y-1} or i_{y+1}, can be used in lemma 5 to derive a contradiction. So, after possibly changing origin and orientation of the antihole, we can assume that $x = 1$ and $y = 2$. We will argue that the only possible ordering of these columns in M is $i_1 < i_3 < i_5 < \ldots < i_6 < i_4 < i_2$. In fact, we can apply the same argument from left to right and from right to left. Assume i_3 is not successive to i_1, but i_p, $p > 3$, is. Then, the antipath $(i_2, i_3, \ldots, i_{p-1})$ would be contained within i_p and i_2, and, by lemma 5, $\{i_{p-1}, i_p\}$ would be a conflict. Similarly, now assume that i_4 is not the second–to–last, but some i_p, $p > 4$ is. Then the antipath $(i_3, \ldots, i_{p-1})$ is contained within i_3 and i_p and, by lemma 5, $\{i_{p-1}, i_p\}$ would be a conflict. We can continue this way to prove the only ordering possible, but the important part is that the ordering looks like $i_1 < \ldots < i_4 < i_2$. Then, the antipath $(i_1, i_k, i_{k-1}, \ldots, i_5)$ is all contained within i_1 and i_4. By lemma 5, $\{i_5, i_4\}$ must be a conflict, contradiction.

Theorem 4. *If M is a gapless SNP matrix, then $G_S(M)$ is a perfect graph.*

Proof. Because of lemma 4 and lemma 6, $G_S(M)$ is weakly triangulated, i.e. neither $G_S(M)$ or its complement have a chordless cycle of length > 4. Since weakly triangulated graphs are perfect (Hayward, [5]), the result follows.

The next corollary follows from Theorem 3, Theorem 4, and the fact that the max independent set can be found in polynomial time on perfect graphs ([3,4]).

Corollary 1. *The Minimum SNP Removal problem can be solved in polynomial time on gapless SNP matrices.*

3   Dealing with Gaps

If gaps in the fragments are allowed, SNP problems become considerably more complex. Typically, a gap corresponds to a SNP whose value at a fragment which in fact covers it was not determined (e.g., because of thresholding of low quality reads, or for sequencing errors which missed some bases). Also, an important case of gaps occurs when fragments are paired up in the so called *mate pairs*. These, used in shotgun sequencing, are fragments taken from the same chromosome,

with some fixed distance between them. A mate pair can be thought of as a single fragment, with a large gap in the middle and SNPs reads at both ends.

A class of inputs in which the SNP matrix has gaps but that can still be solved in polynomial time is the following. A 0-1 matrix is said to have the *consecutive ones property* (C1P) if the columns can be rearranged so that in each row the 1s appear consecutively. Analogously, we say that a SNP matrix is C1P if there exists a permutation π of the SNPs such that each fragment covers a consecutive (in π) subset of SNPs. Since finding such π is polynomial ([1]), it follows from Theorem 1, 2 and Corollary 1 the

Corollary 2. *The problems MFR, LSH and MSR can be solved in polynomial time on SNP matrices that are C1P.*

For those matrices that are not C1P, it is easy to show NP–hardness for many problems, by using the following lemma, that shows how to code a graph into a SNP matrix.

Lemma 7. *Let $G = (V, E)$ be a graph. Then there exists a SNP matrix M such that $G_{\mathcal{F}}(M) = G$.*

Proof. G can be made into a $|V| \times |E|$ SNP matrix by having a fragment for each node in V and a SNP for each edge $e = \{i, j\}$, with value A in i and B in j.

We can now give a simple proof that MFR is NP–hard.

Theorem 5. *MFR is NP-hard.*

Proof. We use a reduction from the following NP-hard problem [6,8]: Given a graph $G = (V, E)$, remove the fewest number of nodes to make it bipartite. This is exactly MFR when G is encoded into a SNP matrix as described in lemma 7.

In the following theorem, we show that it is the very presence of gaps in fragments, and not their quantity, that makes the problem difficult.

Theorem 6. *MFR is NP-hard for SNP matrices in which each fragment has at most 1 gap.*

The proof is through a polynomial time reduction from Max2SAT [2]. Consider an instance $\Phi(k, n)$ of Max2SAT with k clauses over n boolean variables. Denote the clauses of Φ as $C_1, C_2, \ldots, C_k$, and the variables as $x_1, x_2, \ldots x_n$. By definition, each clause contains at most 2 literals. Without loss of generality, we assume that a variable appears at most once in a clause.

We transform the instance $\Phi(n, k)$ into a SNP matrix M_{Φ} with a set $\mathcal{F}$ of $n(nk+3k+1)+3k$ fragments (rows), and $\mathcal{S}$ of $2n+5k$ SNPs (columns). See Fig. 3. Each variable x contributes $nk + 3k + 1$ fragments, which can be partitioned into 3 sets with $k, k, (nk + k + 1)$ fragments respectively. The fragments with labels $f_{x,1}, \ldots, f_{x,k}$ form the set $T(x)$(true). Similarly, the set $F(x)$(False) is the set containing fragments $f_{\bar{x},1}, \ldots, f_{\bar{x},k}$, and the set $S(x)$ (support) contains the remaining $nk + k + 1$ fragments. No two fragments from different sets can be in the same haplotype, and any number of fragments from the same set can be in the same haplotype. Denote a literal of the variable x as $x_l \in \{x, \bar{x}\}$. If $x_l = x$,
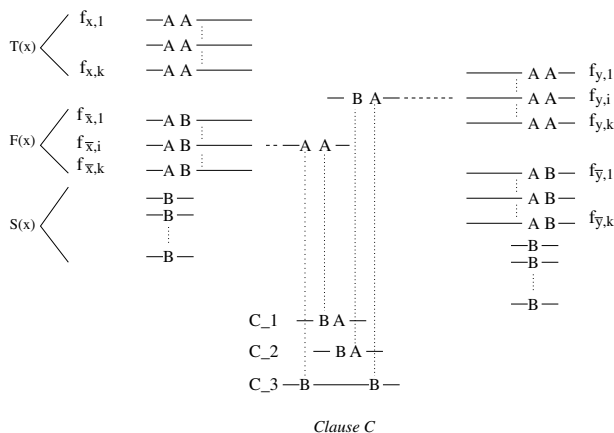
Clause C

Fig. 3. Gadget for the reduction.

then the fragment $f_{x_l,i}$ corresponds to $f_{x,i}$, and $f_{\bar{x}_l,i} = f_{\bar{x},i}$. Similarly, if $x_l = \bar{x}$, $f_{x_l,i} = f_{\bar{x},i}$, and $f_{\bar{x}_l,i} = f_{x,i}$.

Next, consider a clause $C_i = (x + \bar{y})$. There are three other fragments $C_{i,1}, C_{i,2}, C_{i,3}$ for each clause. The clause fragments are located in between the fragments for x and y, $C_{i,1}$ and $C_{i,2}$ conflict with each other. Extend fragment $f_{\bar{x},i}$ with a mate-pair, so that it shares a SNP (and a conflict) with the clause fragment $C_{i,1}$. Likewise, extend $f_{y,i}$ to conflict with $C_{i,2}$. Finally the fragment $C_{i,3}$ is a mate-pair which conflict with both $f_{\bar{x},i}$ and $f_{y,i}$. Denote the fragment conflict graph on M_Φ as $G(M_\Phi)$ (for simplicity, we drop the subscript $\mathcal{F}$).

Lemma 8. *Given a clause $C_i = (x_l + y_l)$ of a 2SAT instance Φ, the fragments $(C_{i,1}, C_{i,2}, f_{y_l,i}, C_{i,3}, f_{\bar{x}_l,i})$ form a chordless cycle in $G(M_\Phi)$.*

Lemma 9. *Each fragment in M_Φ has at most one gap.*

Lemma 10. *Let K be a set of fragments (rows) in M_Φ. The following are sufficient conditions for $M_\Phi[K]$ to be feasible.*

1. *For every variable x, $K \cap F(x) = \{\}$, or $K \cap T(x) = \{\}$.*
2. *For every clause $C_i = (x_l + y_l)$, K does not contain all the four fragments $f_{\bar{x}_l,i}, f_{\bar{y}_l,i}, C_{i,1}$, and $C_{i,2}$.*

Proof. The proof is constructive. If $G(M_\Phi[K])$ is bipartite, its nodes can be partitioned into two independent sets (shores) K_1, and K_2. We employ the following construction.

(1) For all x, add $S(x)$ to K_1, and $T(x)$ (or, $F(x)$) to K_2. (2) For all clauses $C_i = (x_l + y_l)$, add $C_{i,3}$ to K_1. (3) For all clauses $C_i = (x_l + y_l)$: (a) if $(f_{\bar{x}_l,i} \notin F)$, add $C_{i,1}$ to K_2, and $C_{i,2}$ to K_1. (b) else if $(f_{\bar{y}_l,i} \notin F)$, add $C_{i,2}$ to K_2, and $C_{i,1}$ to K_1. (c) else add $C_{i,2}$ (or, $C_{i,1}$) to K_1.

We need to show that the graphs induced by K_1 and K_2 are both independent sets. Note that $S(x)$ in K_1 only has edges to $T(x)$ and $F(x)$ which are in K_2. Likewise for all i, $C_{i,3}$ in K_1 only has edges to nodes in K_2. If both $C_{i,1}$ and $C_{i,2}$ are present, then the condition ensures that both $f_{\bar{x}_l,i}$ and $f_{\bar{y}_l,i}$ are not in K. Therefore, the construction in 3a, and 3b ensures that $C_{i,1}$ and $C_{i,2}$ are put on different shores.

Next, consider the fragments in $T(x)$, and $F(x)$ for all x. Condition 1 ensures that they can be all placed in K_2 without conflicting edges between different literals of x. Next, from 3a, $C_{i,1}$ is placed in K_2 only if $f_{\bar{x}_l,i}$ is not in K. From 3b, $C_{i,2}$ is placed in K_2 only if $f_{\bar{y}_l,i}$ is not in K. Thus, K_1 and K_2 induce independent sets, and $M_\Phi[K]$ is feasible.

Lemma 11. *An optimal solution to the MFR problem on M_Φ has at most $nk+k$ fragments.*

Proof. Consider a set of fragments R with $F(x)$ for all x, and $C_{i,1}$, for all clauses C_i. Removing R satisfies the conditions of lemma 10, implying that R is a solution to the MFR problem on M_Φ with $nk + k$ fragments.

Lemma 12. *Let R be an optimal solution to the MFR problem on M_Φ. Then, $R \cap S(x) = \phi$ for all x.*

Proof. Consider an optimal solution R that contains a fragment f from $S(x)$, for an arbitrary variable x. Let $K = \mathcal{F} - R$. As R is optimal, adding f to $G(M_\Phi[K])$ must create an odd-cycle C. Consider any other fragment $f' \in S(x)$. By construction, $C - \{f\} + \{f'\}$ is also an odd-cycle. This implies that all fragments in $S(x)$ are in R. Therefore, $|R| \geq |S(x)| = nk + k + 1 > nk + k$, a contradiction to lemma 11!

Lemma 13. *Let R be an optimal solution to the MFR problem for M_Φ. Then, for all x, either $T(x) \subseteq R$, or $F(x) \subseteq R$.*

Proof. Consider an optimal solution R with a variable x, and fragments $f_1 \in T(x) - R$, and $f_2 \in F(x) - R$. By lemma 12, there is a fragment $f \in S(x) - R$. By construction, f, f_1, and f_2 form an odd cycle, a contradiction!

Theorem 7. *Consider a Max2SAT instance Φ with n variables and k clauses, and the associated SNP matrix M_Φ. $k' \leq k$ clauses of Φ are satisfiable if and only if there exists a solution to the MFR problem for M_Φ with $nk+k-k'$ fragments.*

Proof. Consider an assignment of variables satisfying k' clauses. For each variable x set to TRUE, add all the fragments in $F(x)$ to R, and for every fragment set to FALSE, add all the fragments in $T(x)$ to R. Next, consider all the clauses that are satisfied. If $C_i = (x_l + y_l)$ is satisfied, at least one of $f_{\bar{x}_l,i}$, and $f_{\bar{y}_l,i}$ is in R, breaking the odd cycle, and we do nothing. If C_i is not satisfied, we add $C_{i,1}$ to R. The number of fragments in R due to variables is nk, and the number of fragments in R due to clauses is $k - k'$. By lemma 10, $M_\Phi[\mathcal{F} - R]$ is feasible.

Next, consider an optimal solution R of the MFR problem on M_Φ with $nk + k - k'$ fragments. For every x, by lemma 13, either $F(x) \subseteq R$ or $T(x) \subseteq R$.

If $F(x) \subseteq R$ set x to TRUE. Otherwise set x to FALSE. We need to show that exactly k' clauses are satisfied. Note that a set D of nk nodes must be in any optimal solution R to the MFR problem (lemma 13). Further, each clause is associated with 5 fragments that induce an odd cycle in the conflict graph (lemma 8). At least one of these fragments must be in R. If a clause $C_i = (x_l + y_l)$ is satisfied, then this fragment can be attributed to the set D. If however, C_i is not satisfied, the number of fragments in R increases by at least one. Thus if the total number of clauses satified is k'', then $|R| \geq nk + k - k''$. If $k'' < k'$, then $|R| > nk + k - k'$, a contradiction. On the other hand, if $k'' > k$, then by earlier argument, there is a solution to the MFR problem with $nk + k - k'' < nk + k - k'$ fragments, which is a contradiction to optimality.

We close this section with a complexity result for the snip removal problem, which follows using lemma 7 and the fact that MAXCUT is NP–hard for 3–regular graphs.

Theorem 8. *The MSR problem is NP-hard for SNP matrices with at most 2 gaps per fragment.*

References

1. K. S. Booth and S. G. Lueker. Testing for consecutive ones property, interval graphs and planarity using PQ–tree algorithms, *J. Comput. Syst. Sci.* 13, 335–379, 1976.
2. M. R. Garey and D. S. Johnson. Computers and Intractability: A Guide to the Theory of NP-completeness. W. Freeman and Co, SF, 1979.
3. M. C. Golumbic. *Algorithmic Graph Theory and Perfect Graphs.* Academic press, NY, 1980.
4. M. Groetschel, L. Lovasz and A. Schrijver. A polynomial algorithm for perfect graphs, *Annals of Discr. Math.* 21, 325–356, 1984.
5. R. B. Hayward. Weakly triangulated graphs, *J. Comb. Th. (B)* 39, 200–209, 1985.
6. J. M. Lewis, On the complexity of the maximum subgraph problem, *Xth ACM Symposium on Theory of Computing*, 265–274, 1978
7. J. C. Venter, M. D. Adams, E. W. Myers *et al.*, The Sequence of the Human Genome, *Science*, 291, 1304–1351, 2001.
8. M. Yannakakis, Node– and Edge– deletion NP–complete Problems, *Xth ACM Symposium on Theory of Computing*, 253–264, 1978.