Detecting Transcriptomic Structural Variants in Heterogeneous Contexts via the Multiple Compatible Arrangements Problem

Yutong Qiu[1]
Computational Biology Department, School of Computer Science,
Carnegie Mellon University, Pittsburgh, PA, USA
yutongq@andrew.cmu.edu

Cong Ma[1]
Computational Biology Department, School of Computer Science,
Carnegie Mellon University, Pittsburgh, PA, USA
congm1@andrew.cmu.edu

Han Xie
Computational Biology Department, School of Computer Science,
Carnegie Mellon University, Pittsburgh, PA, USA
hanx@andrew.cmu.edu

Carl Kingsford
Computational Biology Department, School of Computer Science,
Carnegie Mellon University, Pittsburgh, PA, USA
carlk@cs.cmu.edu

Abstract

Transcriptomic structural variants (TSVs) – large-scale transcriptome sequence change due to structural variation – are common, especially in cancer. Detecting TSVs is a challenging computational problem. Sample heterogeneity (including differences between alleles in diploid organisms) is a critical confounding factor when identifying TSVs. To improve TSV detection in heterogeneous RNA-seq samples, we introduce the Multiple Compatible Arrangement Problem (MCAP), which seeks k genome rearrangements to maximize the number of reads that are concordant with at least one rearrangement. This directly models the situation of a heterogeneous or diploid sample. We prove that MCAP is NP-hard and provide a $\frac{1}{4}$-approximation algorithm for $k = 1$ and a $\frac{3}{4}$-approximation algorithm for the diploid case ($k = 2$) assuming an oracle for $k = 1$. Combining these, we obtain a $\frac{3}{16}$-approximation algorithm for MCAP when $k = 2$ (without an oracle). We also present an integer linear programming formulation for general k. We characterize the graph structures that require $k > 1$ to satisfy all edges and show such structures are prevalent in cancer samples. We evaluate our algorithms on 381 TCGA samples and 2 cancer cell lines and show improved performance compared to the state-of-the-art TSV-calling tool, SQUID.

2012 ACM Subject Classification Applied computing → Computational transcriptomics

Keywords and phrases transcriptomic structural variation, integer linear programming, heterogeneity

Digital Object Identifier 10.4230/LIPIcs.WABI.2019.18

Funding This work was supported in part by the Gordon and Betty Moore Foundation's Data-Driven Discovery Initiative [GBMF4554 to C.K.]; the US National Institutes of Health [R01GM122935]; and The Shurl and Kay Curci Foundation. This project is funded, in part, by a grant (4100070287) from the Pennsylvania Department of Health. The department specifically disclaims responsibility for any analyses, interpretations, or conclusions.

[1] These authors contributed equally to this work.

Acknowledgements The results shown here are in part based upon data generated by the TCGA Research Network: `https://www.cancer.gov/tcga`. This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562. Specifically, it used the Bridges system, which is supported by NSF award number ACI-1445606, at the Pittsburgh Supercomputing Center (PSC) [17]. C. K. is co-founder of Ocean Genomics, Inc.

1 Introduction

Transcriptomic structural variations (TSVs) are transcriptome sequence alterations due to genomic structural variants (SVs). TSVs may cause the joining of parts from different genes, which are fusion-gene events. Fusion genes are known for their association with various types of cancer. For example, the joint protein products of *BCR-ABL1* genes are prevalently found in leukemia [4]. In addition to fusion genes, the joining of intergenic and genic regions, called non-fusion-gene events, are also related to cancer [22].

TSV events are best studied with RNA-seq data. Although SVs are more often studied with whole genome sequencing (WGS) [2, 12, 18, 9, 5, 20], the models built on WGS data lack the flexibility to describe alternative splicing and differences in expression levels of transcripts affected by TSVs. In addition, RNA-seq data is far more common [14] than WGS data, for example, in The Cancer Genome Atlas (TCGA, `https://cancergenome.nih.gov`).

Many methods have been proposed that identify fusion genes with RNA-seq data. Generally, these tools identify candidates of TSV events through investigation into read alignments that are discordant with the reference genome (e.g. [10, 15, 3, 16, 21, 11]). A read alignment is *concordant* with a reference sequence if the alignment to the sequence agrees with the read library preparation. For example in paired-end Illumina sequencing, the orientation of the forward read should be $5'$-to-$3'$ and the reverse for the mate read. Otherwise the alignment is *discordant* with the reference. A series of filtering or scoring functions are applied on each TSV candidate to eliminate the errors in alignment or data preparation. The performance of filters often relies heavily on a large set of method parameters and requires prior annotation [13]. Furthermore, most of the fusion-gene detection methods limit the scope to the joining of protein-coding regions and ignore the joining of intergenic regions that could also affect the transcriptome. An approach that correctly models both fusion-gene and non-fusion-gene events without a large number of ad hoc assumptions is desired.

An intuitive TSV model is the one that describes directly the rearrangement of the genome. For example, when an inversion happens, two double-strand breaks (DSB) are introduced to the genome and the segment between the DSBs is flipped. After a series of TSVs are applied to a genome, a rearranged genome is produced. In order to identify the TSVs, we can attempt to infer the rearranged genome from the original genome and keep track of the arrangements of genome segments. Since a model of the complete genome is produced, both fusion-gene and non-fusion-gene events can be detected. A recently published TSV detection tool, SQUID [14], models TSV events in this way by determining a single rearrangement of a reference genome that can explain the maximum number of observed sequencing reads. SQUID finds one arrangement of genome segments such that a maximum number of reads are able to be sequenced from it. Novel transcriptomic adjacencies appearing in the arrangement are predicted as TSVs while the ones not appearing are regarded as sequencing or alignment errors.

Despite the generally good performance of SQUID, it relies on the assumption that the sample is homogeneous, i.e. the original genome contains only one allele that can be represented by a single rearranged string. This assumption is unrealistic in diploid (or high ploidy) organisms. When TSV events occur within the same regions on different alleles, read alignments may suggest multiple conflicting ways of placing a segment. Under the homogeneous assumption, conflicting TSV candidates are regarded as errors. Therefore, this assumption leads to discarding the conflicting TSV candidates that would be compatible on separate alleles and therefore limits the discovery of true TSVs. Conflicting SV candidates are addressed in a few SV detection tools such as VariationHunter-CR [9]. However, VariationHunter-CR assumes a diploid genome, and its model is built for WGS data that lacks ability to handle RNA-seq data.

We present an improved model of TSV events in heterogeneous contexts. We address the limitation of the homogeneous assumption by extending the assumption to k alleles. We introduce the Multiple Compatible Arrangement Problem (MCAP), which seeks, assuming the number of alleles k is known, an optimal set of k arrangements of segments from GSG such that the number of concordant sequencing reads with any of the rearrangement sequences is maximized. Each arrangement is a permutation and reorientation of all segments from the reference genome, representing the altered sequence of one allele. The originally discordant edges that are concordant in any of the k arrangements are predicted as TSVs, and those edges are regarded as errors otherwise. We show that MCAP is NP-hard. To address NP-hardness, we propose a $\frac{1}{4}$-approximation algorithm for the $k = 1$ case and a $\frac{3}{4}$-approximation solution to the $k = 2$ case using an oracle for $k = 1$. Combining these, we obtain a $\frac{3}{16}$-approximation algorithm for MCAP when $k = 2$ (without an oracle). We also present an integer linear programming (ILP) formulation that gives an optimal solution for general k.

We characterize the patterns of reads that result in conflicting TSV candidates under a single-allele assumption. We show that these patterns are prevalent in both cancer cell lines and TCGA samples, thereby further motivating the importance of SV detection approaches that directly model heterogeneity.

We apply our algorithms to 381 TCGA samples from 4 cancer types and show that many more TSVs can be identified under a diploid assumption compared to a haploid assumption. We also evaluate an exact ILP formulation under a diploid assumption (D-SQUID) on previously annotated cancer cell lines HCC1395 and HCC1954, identifying several previously known and novel TSVs. We also show that, in most of the TCGA samples, the performance of the approximation algorithm is very close to optimal and the worst case of $\frac{3}{16}$-approximation is rare.

2 The Genome Segment Graph (GSG)

A Genome Segment Graph, similar to a splice graph [8], encodes relationships between genomic segments and a set of reads. A *segmentation* S of the genome is a partition of the genome into disjoint intervals according to concordant and discordant paired-end alignments with respect to the reference genome. The genome partitioning, edge construction and edge filtering is done in the same way as in [14].

▶ **Definition 1** (Genome Segment Graph). *A* genome segment graph *is a weighted, undirected graph $G = (V, E, \mathbf{w})$ derived from a segmentation S of the genome and a collection of reads. The vertex set, $V = \{s_h \in S\} \bigcup \{s_t \in S\}$, includes a vertex for both endpoints, head (h) and tail (t), for each segment $s \in S$. The head of a segment is the end that is closer to the $5'$ end of the genome. The tail is the end that is close to the $3'$ end. Pairs of reads that span more than one segment are represented by edges. There are four types of connections: head-head, head-tail, tail-head and tail-tail. Each edge $e = (u_i, v_j) \in E$, where $i, j \in \{h, t\}$, is undirected* *and connects endpoints of two segments. The weight ($w_e \in \mathbf{w}$) is the number of sequencing reads that support edge e.*

We also define the weight of a subset $E' \subseteq E$ of edges $w(E') = \sum_{e \in E'} w_e$. (More details on the GSG provided in Ma et al. [14].)

▶ **Definition 2** (Permutation, Orientation function and Arrangement). *A permutation is a function where $\pi(u) = i$, where i is the index of segment $u \in S$ in an ordering of a set S of segments. We also define orientation function $f(u) = 1$ if segment u should remain the original orientation, or 0 if it should be inverted. An* arrangement *is a pair of permutation and orientation functions (π, f).*

If $\pi(u) < \pi(v)$, we say that segment u is closer to the $5'$ end of the rearranged genome than segment v. Each arrangement is a concatenation of segments from different chromosomes, which retrieves the sequences affected by inter- and intra-chromosomal TSV events. The arrangement of genome segments imitates the movements of genomic sequences by SVs. One crucial difference between arrangement in GSG and sequence movements by SVs is that an arrangement in GSG only captures the movement that are relevant to transcriptome sequence alterations. Such alterations can either fuse two transcript sequences or incorporate previously non-transcribing sequences into transcripts as long as they are present in RNA-seq reads.

▶ **Definition 3** (Concordant and Discordant edges). *Let e be an edge connecting segment u on end a and segment v on end b ($a, b \in \{h, t\}$). Given arrangement (π, f), suppose $\pi(u) < \pi(v)$, edge e is* concordant *with respect to the arrangement if $f(u) = \mathbb{1}_{a=t}$ and $f(v) = \mathbb{1}_{b=h}$. Denote the concordance as $e \sim (\pi, f)$. Otherwise, e is* discordant *and denote as $e \not\sim (\pi, f)$.*

Since edges are constructed based on segment connections indicated by read alignments, the concordance and discordance of edges are extensions from read alignments. A discordant edge represents a set of discordant read alignments. Examples of discordant edges with tail-tail and head-head connections are shown in Figure 1a. Concordant edges, when connecting nodes that belong to the same chromosome, represent concordant alignments that are either continuous alignments or split-alignments due to alternative splicing. Due to alternative splicing, a node can be incident to multiple concordant edges given an arrangement. Edges that initially spanned two chromosomes but become concordant in an arrangement represent inter-chromosomal translocation events.

Segments connected by discordant edges can be arranged so that some of the discordant edges become concordant. See Figure 1b,c for examples of arrangements that make tail-tail and head-head connections concordant.

▶ **Definition 4** (Conflicts among a Set of Edges). *Given GSG $G = (V, E, \mathbf{w})$ and a subset of edges E', the edges in set E' are in conflict with each other if there is no single arrangement (π, f) such that $e \sim (\pi, f)$ ($\forall e \in E'$). Otherwise, edges in set E' are compatible with each other.*

▶ **Definition 5** (Transcriptomic Structural Variant (TSV)). *A TSV is a new adjacency in transcript sequences that cannot be explained by alternative splicing.*

In GSG, the adjacency in transcript sequences is represented by edges. New adjacencies that cannot be explained by alternative splicing belong to the set of edges that are either discordant with respect to the reference arrangement or connecting segments belonging to different chromosomes.

3 The Multiple Compatible Arrangements Problem (MCAP)

3.1 Problem Statement

Given an input GSG $G = (V, E, w)$ and a positive integer k, the Multiple Compatible Arrangements Problem seeks a set of k arrangements $A = \{(\pi_i, f_i)\}_{i=1}^{k}$ that are able to generate the maximum number of sequencing reads:

$$\max_{A} \sum_{e \in E} w(e) \cdot \mathbf{1}\left[e \sim A\right], \tag{1}$$

where $\mathbf{1}\left[e \sim A\right]$ is 1 if edge e is concordant in at least one $(\pi_i, f_i) \in A$, and 0 otherwise.

This objective function aims to find an optimal set of k arrangements of segments where the sum of concordant edge weights is maximized in the rearranged alleles, where k is the number of alleles and assumed to be known for sure. The objective seeks to maximize the agreement between rearranged allelic sequences and observed RNA-seq data. Assuming that the majority of RNA-seq reads are sequenced, the concordant edges with respect to the optimal set of arrangements represent the most confident transcriptomic adjacencies. In heterogeneous samples where $k \neq 1$, MCAP separates the conflicting edges onto k alleles as shown in an example in Figure 1.

When $k = 1$, the problem reduces to finding a single rearranged genome to maximize the number of concordant reads, which is the problem that SQUID [14] solves. We refer to the special case when $k = 1$ as Single Compatible Arrangement Problem (SCAP).

Predicted TSVs are the concordant edges with respect to any of the arrangements in a solution to MCAP that were either discordant with respect to or spanning multiple chromosomes in the reference genome.

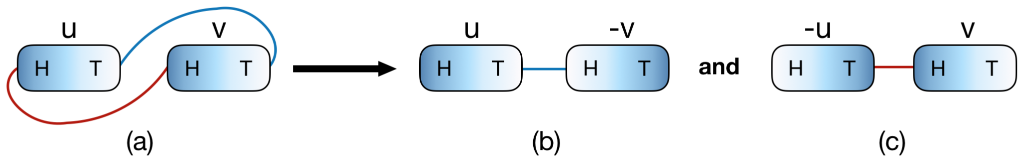

Figure 1 MCAP resolves conflicts. The darker ends of the segments represent head with respect to the original genome. The lighter ends represent tail with respect to the original genome. “H” stands for head and “T” stands for tail. (a) Two conflicting edges connecting two segments u and v. If the sample is known to be homogeneous ($k = 1$), then the conflict is due to errors. If $k = 2$, MCAP seeks to separate two edges into two compatible arrangements as in (b) and (c). (b) In the first arrangement, segment v is flipped, which makes the blue edge concordant. (c) In the second arrangement, u is flipped to make the red edge concordant.

3.2 NP-hardness of SCAP and MCAP

▶ **Theorem 6.** *SCAP is NP-hard.*

Proof Sketch. We prove the NP-hardness of SCAP by reducing from MAX-2-SAT. While 2-SAT can be solved in polynomial time, MAX-2-SAT, which asks for the maximum number of clauses that can be satisfied, is NP-hard. For boolean variables and clauses in any MAX-2-SAT instance, we create gadget segments in the GSG so that the satisfaction of each clause is determined by the edge concordance and the boolean assignment is determined by segment inversion. The gadgets force the optimal sum of concordant edge weights to directly represent the number of satisfied clauses. Correspondingly, the optimal orientations of segments represent the assignment of boolean variables. See Appendix A for a complet eproof. ◀

▶ **Corollary 7.** *MCAP is NP-hard.*

Proof. SCAP is a special case of MCAP with $k = 1$, so the NP-hardness of MCAP is immediate. ◀

4 A $\frac{1}{4}$-approximation Algorithm for SCAP

We provide a greedy algorithm for SCAP that achieves at least $\frac{1}{4}$ approximation ratio and takes $O(|V||E|)$ time. The main idea of the greedy algorithm is to place each segment into the current order one by one by choosing the current "best" position. The current "best" position is determined by the concordant edge weights between the segment to be placed and the segments already in the current order.

Algorithm 1 Greedy algorithm for SCAP.

```
Data: Segment set $S$, genome segment graph $G = (V, E, w)$
Result: An arrangement of the segments and the sum of concordant edge weights
1  $order = []$;
2  $orientation = []$;
3  for $i$ in $1 : |S|$ do
4      $s^i$ = the $i^{th}$ segment in $S$;
       // choose from 4 possible order and orientation options
5      $options$ = [($s^i$ in the beginning of $order$ in forward strand), ($s^i$ in the beginning
         of $order$ in reverse strand), ($s^i$ in the end of $order$ in forward strand), ($s^i$ in the
         end of $order$ in reverse strand)] ;
6      for $j$ in $1 : 4$ do
7          $weights[j] =$
             $w(\{e \in E : e$ connects $s^i$ with $s^k$ and concordant in $options[j], k < i\})$;
8      end
       // update the current order and orientation
9      $opt = \text{argmax}_{1 \le i \le 4, i \in \mathbb{N}} weights[i]$ ;
10     $order$ = update segment order as given by $options[opt]$ ;
11     $orientation$ = update segment orientation as given by $options[opt]$ ;
12 end
```

▶ **Theorem 8.** *Algorithm 1 approximates SCAP with at least $\frac{1}{4}$ approximation ratio.*

Proof. Denote $E' \subset E$ as the concordant edges in the arrangement of Algorithm 1. Let OPT be the optimal value of SCAP. We are to prove $w(E') \ge \frac{1}{4}w(E) \ge \frac{1}{4}OPT$.

For iteration i in the for loop, the edges $E_i = \{e \in E : e$ connects s^i with $s^j, i < j\}$ are considered when comparing the options. Each of the four options makes a subset of E_i concordant. These subsets are non-overlapping and their union is E_i. Specifically, the concordant edge subset is $\{e = (s^i_h, s^j_t)\}$ for the first option, $\{e = (s^i_h, s^j_h)\}$ for the second, $\{e = (s^i_t, s^j_h)\}$ for the third, and $\{e = (s^i_t, s^j_t)\}$ for the last.

By the selecting the option with the largest sum of concordant edge weights, the concordant edges E'_i in iteration i satisfies $w(E'_i) \geq \frac{1}{4}w(E_i)$. Therefore, the overall concordant edge weights of all iterations in the for loop satisfy

$$\sum_i w(E'_i) \geq \frac{1}{4}\sum_i w(E_i) = \frac{1}{4} w\left(\bigcup_i E_i\right) .$$

Each edge $e \in E$ must appear in one and only one of E_i, and thus $\bigcup_i E_i = E$. This implies $\sum_i w(E'_i) \geq \frac{1}{4}w(E) \geq \frac{1}{4}OPT$. ◀

Algorithm 1 can be further improved in practice by considering more order and orientation options when inserting a segment into current order. In the pseudo-code 1, only two possible insertion places are considered: the beginning and the end of the current order. However, a new segment can be inserted in between any pair of adjacent segments in the current order. We provide an extended greedy algorithm to take into account the extra possible inserting positions (Algorithm 2). Algorithm 2 has a time complexity of $O(|V|^2|E|)$, but it may achieve a higher total concordant edge weight in practice.

Algorithm 2 Extended greedy algorithm for SCAP.

```
Data: Segment set $S$, genome segment graph $G = (V, E, w)$
Result: An arrangement of the segments and the sum of concordant edge weights
1  $order = []$;
2  $orientation = []$;
3  for $i$ in $1 : |S|$ do
4      $s^i$ = the $i^{th}$ segment in $S$;
       // choose from $i+1$ possible order and orientation options
5      $options = [(s^i$ in the beginning of $order$ in forward strand), ($s^i$ in the beginning
        of $order$ in reverse strand)] ;
6      for $j$ in $1 : i-1$ do
7          Append [($s^i$ right after $order[j]$ in forward strand), ($s^i$ right after $order[j]$ in
            reverse strand)] to list of $options$ ;
8      end
9      for $j$ in $1 : 2i$ do
10         $weights[j] =$
            $w(\{e \in E : e$ connects $s^i$ with $s^k$ and concordant in $options[j], k < i\})$;
11     end
       // update the current order and orientation
12     $opt = \text{argmax}_{1 \leq j \leq i, j \in \mathbb{N}} weights[j]$ ;
13     $order$ = update segment order as given by $options[opt]$ ;
14     $orientation$ = update segment orientation as given by $options[opt]$ ;
15 end
```

5 A $\frac{3}{4}$-approximation of MCAP with $k = 2$ Using a SCAP Oracle

If an optimal SCAP solution can be computed, one way to approximate the MCAP's optimal solution is to solve a series of SCAP instances iteratively to obtain multiple arrangements. Here, we prove the iterative SCAP solution has an approximation ratio of $\frac{3}{4}$ for the special case of MCAP with $k = 2$.

Algorithm 3 $\frac{3}{4}$-approximation for MCAP with $k = 2$.

```
Data: A genome segment graph $G = (V, E, w)$
Result: a set of two arrangements, sum of weights of edges that are concordant in
        either arrangement
1 $a_1$ = optimal SCAP arrangement on $G$;
2 $E' = \{e \in E : e$ is discordant in $a_1\}$;
3 $G' = (V, E', w)$;
4 $a_2$ = optimal SCAP arrangement on $G'$;
5 $\tilde{E} = \{e \in E : e \sim A, A = \{a_1, a_2\}\}$;
6 $W = \sum_{e \in \tilde{E}} w(e)$;
7 return $(\{a_1, a_2\}, W)$;
```

▶ **Theorem 9.** *Algorithm 3 is a $\frac{3}{4}$-approximation of MCAP with $k = 2$. Denote the optimal objective sum of edge weights in MCAP with $k = 2$ as OPT, and the sum of edge weights in iterative SCAP as W, then*

$$W \geq \frac{3}{4} OPT$$

Proof. Denote MCAP with $k = 2$ as 2-MCAP. Let E_1^d and E_2^d be concordant edges in the optimal two arrangements of 2-MCAP. It is always possible to make the concordant edges of the arrangements disjoint by removing the intersection from one of the concordant edge set, that is $E_1^d \cap E_2^d = \emptyset$. Let $E^d = E_1^d \cup E_2^d$. The optimal value is $w(E^d)$.

Denote the optimal set of concordant edges in the first round of Algorithm 3 as E_1^s. The optimal value of SCAP is $w(E_1^s)$. E_1^s can have overlap with the two concordant edge sets of the 2-MCAP optimal solution. Let the intersections be $I_1 = E_1^d \cap E_1^s$ and $I_2 = E_2^d \cap E_1^s$. Let the unique concordant edges be $D_1 = E_1^d - E_1^s$, $D_2 = E_2^d - E_1^s$ and $S = E_1^s - E_1^d - E_2^d$.

After separating the concordant edges in 2-MCAP into the intersections and unique sets, the optimal value of 2-MCAP can be written as $w(E^d) = w(I_1) + w(I_2) + w(D_1) + w(D_2)$, where the four subsets are disjoint. Therefore the smallest weight among the four subsets must be no greater than $\frac{1}{4}w(E^d)$. We prove the approximation ratio under the following two cases and discuss the weight of the second round of SCAP separately:

Case (1): the weight of either D_1 or D_2 is smaller than $\frac{1}{4}w(E^d)$. Because the two arrangements in 2-MCAP are interchangeable, we only prove for the case where $w(D_1) \leq \frac{1}{4}w(E^d)$. A valid arrangement of the second round of SCAP is the second arrangement in 2-MCAP, though it may not be optimal. The maximum concordant edge weights added by the second round of SCAP must be no smaller than $w(D_2)$. Combining the optimal values of two rounds of SCAP, the concordant edge weight is

$$W \geq w(E_1^s) + w(D_2) = w(S) + w(I_1) + w(I_2) + w(D_2) \geq w(E^d) - w(D_1) \geq \frac{3}{4}w(E^d). \quad (2)$$

Case (2): both $w(D_1) \geq \frac{1}{4}w(E^d)$ and $w(D_2) \geq \frac{1}{4}w(E^d)$. The subset with smallest sum of edge weights is now either I_1 or I_2. Without loss of generality, we assume I_1 has the smallest sum of edge weights and $w(I_1) \leq \frac{1}{4}w(E^d)$. Because the first round SCAP is optimal for the SCAP problem, its objective value should be no smaller than the concordant edge weights of either arrangement in 2-MCAP. Thus

$$w(E_1^s) \geq w(E_2^d) = w(D_2) + w(I_2). \quad (3)$$

A valid arrangement for the second round of SCAP can be either of the arrangements in 2-MCAP optimal solution. Picking the first arrangement of 2-MCAP as the possible (but not necessarily optimal) arrangement for the second round of SCAP, the concordant edge weights added by the second round of SCAP must be no smaller than $w(D_1)$. Therefore, the total sum of concordant edge weights of the optimal solutions of both rounds of SCAP is

$$W \geq w(E_1^s) + w(D_1) \geq w(D_2) + w(I_2) + w(D_1) = w(E^d) - w(I_1) \geq \frac{3}{4}w(E^d). \tag{4}$$

◀

▶ **Corollary 10.** *An approximation algorithm for MCAP with $k = 2$ can be created by using Algorithm 1 as the oracle for SCAP in Algorithm 3. This approximation algorithm runs in $O(|V||E|)$ time and achieves at least $\frac{3}{16}$ approximation ratio.*

The proof of the corollary is similar to the proof of iterative SCAP approximation ratio. By adding a multiplier of $\frac{1}{4}$ to the right of inequalities (3) and (4), the $\frac{3}{16}$ approximation ratio can be derived accordingly.

6 Integer Linear Programming Formulation for MCAP

MCAP, for general k, can be formulated as an integer linear programming (ILP) to obtain an optimal solution. We rewrite the i^th permutation (π_i), orientation (f_i) and decision ($\mathbf{1}[e \sim (\pi_i, f_i)]$) functions with three boolean variables y_e^i, z_e^i and x_e^i. For $i \in \{1, 2..., k\}$ and $e \in E$, we have:

- $x_e^i = 1$ if edge $e \sim (\pi_i, f_i)$ and 0 otherwise.
- $y_u^i = 1$ if $f_i(u) = 1$ for segment u and 0 if $f_i(u) = 0$.
- $z_{uv}^i = 1$ if $\pi_i(u) < \pi_i(v)$, or segment u is in front of v in arrangement i and 0 otherwise.

In order to account for the edges that are concordant in more than one arrangement in the summation in Equation 1, we define q_e such that $q_e = 1$ if edge e is concordant in one of the k arrangements and 0 otherwise. The constraints for q_e are as follows:

$$q_e \leq \sum_{i}^{k} x_e^i \tag{5}$$

$$q_e \leq 1 \tag{6}$$

The objective function becomes

$$\max_{x_e^i, y_u^i, z_{uv}^i} \sum_{e \in E} w(e) \cdot q_e \tag{7}$$

We then add ordering and orientation constraints. If an edge is a tail-head connection, i.e. concordant to the reference genome, $x_e^i = 1$ if and only if $z_{uv}^i = y_u^i = y_v^i$. If an edge is a tail-tail connection, $x_e^i = 1$ if and only if $z_{uv}^i = 1 - y_v^i = y_u^i$. If an edge is a head-tail connection, $x_e^i = 1$ if and only if $z_{uv}^i = 1 - y_u^i = 1 - y_v^i$. If an edge is a head-head connection, $x_e^i = 1$ if and only if $z_{uv}^i = 1 - y_u^i = y_v^i$. The constraints for a tail-head connection are listed below in Equation 8, which enforce the assignment of boolean variables y_e^i, z_e^i and x_e^i:

$$\begin{aligned} x_e^i &\leq y_u^i - y_v^i + 1, \\ x_e^i &\leq y_v^i - y_u^i + 1, \\ x_e^i &\leq y_u^i - z_{uv}^i + 1, \\ x_e^i &\leq z_{uv}^i - y_u^i + 1, \end{aligned} \tag{8}$$

The constraints of other types of connections are similar and detailed in Ma et al. [14]. Additionally, constraints are added so that all segments are put into a total order within each allele. For two segments u, v, segment u will be either precede or follow segment v, i.e. $z^i_{uv} + z^i_{vu} = 1$. For three segments u, v, w, if u precedes v and v precedes w, then u has to precede w: $1 \leq z^i_{uv} + z^i_{vw} + z^i_{wu} \leq 2$.

The total number of constraints as a function of k is $4k|E| + k\binom{|V|}{3} + 2|E| = O(k(|E| + V^3))$. When k increases, the number of constraints grows linearly. When $k = 1$, the ILP formulation reduces to the same formulation as SQUID.

7 Characterizing the Conflict Structures That Imply Heterogeneity

In this section, we ignore edge weights and characterize the graph structures where homogeneous assumption cannot explain all edges. We add a set of segment edges, $\hat{E}$, to the GSG. Each $\hat{e} \in \hat{E}$ connects the two endpoints of each segment, i.e. $\hat{e} = \{s_h, s_t\}$ for $s \in S$. The representation of GSG becomes $G = (E, \hat{E}, V)$.

▶ **Definition 11** (Conflict Structures and Compatible Structures)**.** *A* conflict structure*, $CS = (E', \hat{E}', V')$, is a subgraph of a GSG where there exists a set of edges E' that cannot be made concordant using any single arrangement. A* compatible structure *is a subgraph of a GSG where there exists a single arrangement such that all edges can be made concordant in it.*

▶ **Definition 12** (Simple cycle in GSG)**.** *A* simple cycle*, $C = (E', \hat{E}', \{v_0, \ldots, v_{n-1}\})$, is a subgraph of a GSG, such that $E' \subseteq E, \hat{E}' \subseteq \hat{E}$ and $v_i \in V$, with $(v_i, v_{i+1 \bmod n}) \in E' \cup \hat{E}'$ and where $v_i \neq v_j$ when $i \neq j$ except $v_{n-1} = v_0$.*

▶ **Definition 13** (Degree and special degree of a vertex in subgraphs of GSG)**.** *Given a subgraph of GSG, $G' = (E', \hat{E}', V')$, $deg_{E'}(v)$ refers to the degree of vertex $v \in V'$ that counts only the edges $e \in E'$ that connect to v. $deg(v)$ refers to the number of edges $e \in E' \cup \hat{E}'$ that connect to v.*

▶ **Theorem 14.** *Any acyclic subgraph of GSG is a compatible structure.*

▶ **Theorem 15.** *A simple cycle $C = (E', \hat{E}', V')$ is a compatible structure if and only if there are exactly two vertices, v_j and v_i such that $deg_{E'}(v_i) = deg_{E'}(v_j) = 2$ and v_i and v_j belongs to different segments.*

The details of the proofs of the above two theorems are in Appendix B.

▶ **Corollary 16.** *A necessary condition for a subgraph $(E', \hat{E}', V')$ to be a conflict structure is that it contains cycles. A sufficient condition for a subgraph $(E', \hat{E}', V')$ to be a conflict structure is that it contains a simple cycle which is not a compatible structure.*

The corollary is a direct derivation of Theorem 14 and Theorem 15 when considering general graph structures.

In practice, we determine if a discordant edge, $e = (u, v)$, is involved in a conflict structure by enumerating all simple paths using a modified depth-first search implemented in Networkx [7, 19] between u and v omitting edge e. We add e to each path and form a simple cycle. If the simple cycle satisfies Corollary 16, we stop path enumeration and label the e as discordant edge involved in conflict structure. If the running time of path enumeration exceeds 0.5 seconds, we shuffle the order of DFS and repeat enumeration. If path enumeration for e exceeds 1000 reruns, we label e as undecided.

8 Experimental Results

To produce an efficient, practical algorithm for TSV detection in diploid organisms, we use the following approach, which we denote as D-SQUID: Run the ILP (Section 6) under the diploid assumption by setting $k = 2$ on every connected component of GSG separately. If the ILP finishes or the running time of the ILP exceeds one hour, output the current arrangements.

8.1 D-SQUID Identifies More TSVs in TCGA Samples than SQUID

We calculate the fraction of discordant edges involved in conflict structures (Figure 2a) in 381 TCGA samples from four types of cancers: bladder urothelial carcinoma (BLCA), breast invasive carcinoma (BRCA), lung adenocarcinoma (LUAD) and prostate adenocarcinoma (PRAD). Among all samples, we found less than 0.5% undecided edges out of all discordant edges. The distribution of fraction of discordant edges within conflict structures are different among cancer types. The more discordant edges are involved in conflict structures, the more heterogeneous the sample is. Among four cancer types, PRAD samples exhibit the highest extent of heterogeneity and BRCA samples exhibit the lowest. On average, more than 90% of discordant edges are within conflict structures in all samples across four cancer types. This suggests that TCGA samples are usually heterogeneous and may be partially explained by the fact that TCGA samples are usually a mixture of tumor cells and normal cells [1].

We compare the number of TSVs found by D-SQUID and SQUID (Figure 2b). In all of our results, all of the TSVs found by SQUID belong to a subset of TSVs found by D-SQUID. D-SQUID identifies many more TSVs than SQUID on all four types of cancers.

A discordant edge is termed resolved if it is made concordant in one of the arrangements. Among all discordant edges in all samples, D-SQUID is able to resolve most of them (Figure 2c), while SQUID is only able to resolve fewer than 50% of them. The results demonstrate that D-SQUID is more capable of resolving conflict structures in heterogeneous contexts, such as cancer samples, than SQUID.

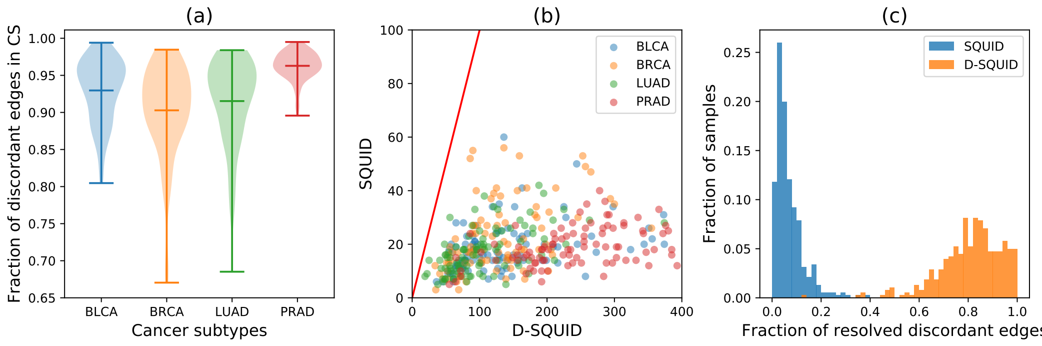

Figure 2 (a) The distribution of fractions of discordant edges that are involved in each identified conflict structure (CS) in four cancer subtypes. Minima, maxima and means of the distributions are marked by horizontal bars. (b) Number of TSVs identified by SQUID versus D-SQUID. (c) Histogram of fractions of resolved discordant edges by SQUID and D-SQUID.

8.2 D-SQUID Identifies More True TSV Events Than SQUID in Cancer Cell Lines

We compare the ability of D-SQUID and SQUID to detect fusion-gene and non-fusion-gene events on previously studied breast cancer cell lines HCC1395 and HCC1954 [6]. The annotation of true SVs is taken from Ma et al. [14]. In both cell lines, D-SQUID discovers more TSVs than SQUID. In HCC1954, D-SQUID identifies the same number of known TSVs including fusions of gene (G) regions and intergenic (IG) regions compared with SQUID. In HCC1395, D-SQUID identifies 2 more true TSV events that are fusions of genic regions. We tally the fraction of discordant edges in conflict structures (Figure 3c) and find similar fractions between HCC1395 and HCC1954, which indicates that the extent of heterogeneity in two samples are similar. Compared to Figure 2a, the fraction in HCC samples is much lower than that in TCGA samples. This matches the fact that two HCC samples contain the same cell type and are both cell line samples, which are known to be less heterogeneous than TCGA samples.

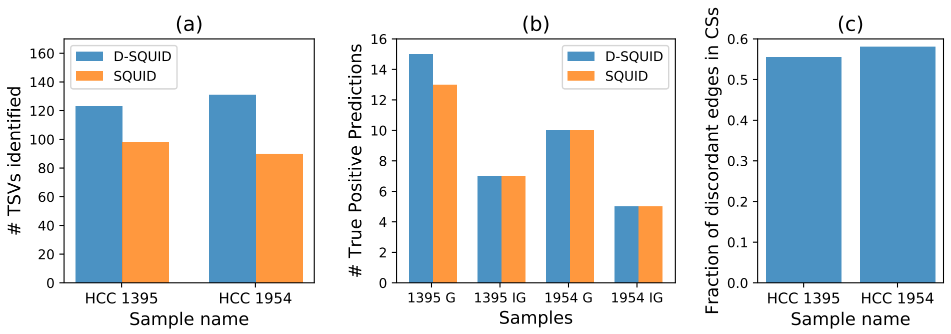

Figure 3 Performance of D-SQUID and SQUID on breast cancer cell lines with experimentally verified SV. (a) Total TSVs found. In both cell line samples, D-SQUID discovered more TSVs than SQUID. (b) Number of known fusion-gene and non-fusion-gene events recovered by D-SQUID and SQUID. G denotes TSVs that affect gene regions. IG denotes TSVs that affect intergenic regions. (c) Fraction of discordant edges in conflict structures.

8.3 Evaluation of approximation algorithms

We evaluate the approximation algorithms for diploid MCAP ($k = 2$) using two different subroutines described in Section 4. In this subsection, $A1$ refers to using Algorithm 1 with worst case runtime $O(|V||E|)$ as a subroutine and $A2$ refers to using Algorithm 2 with worst case runtime $O(|V|^2|E|)$ as a subroutine. Both $A1$ and $A2$ solve SCAP by greedily inserting segments into the best position in the current ordering. While $A1$ only looks at the beginning and ending of the ordering, $A2$ looks at all the positions.

In order to compare the performance of approximations to the exact algorithm using ILP, we run D-SQUID, $A1$ and $A2$ on TCGA samples in Section 8.1. The algorithms are evaluated on runtime and total weight of concordant edges in the rearranged genomes. "Fold difference" on the axes of Figure 4 refers to the ratio of the axis values of D-SQUID over that of $A1$ or $A2$. Both $A1$ and $A2$ output results in a much shorter period of time than D-SQUID. $A2$ achieves better approximation than $A1$, demonstrated by closer-to-one ratio of total concordant edge weight, at a cost of longer run time.

The run time of D-SQUID ILP exceeds one hour on 4.5% of all connected components in all TCGA samples. D-SQUID outputs sub-optimal arrangements in such cases. As a result, approximation algorithms, especially $A2$, appear to resolve more high-weight discordant edges than D-SQUID in some of the samples in Figure 4, which is demonstrated by data points that fall below 1 on the y axes. $A1$ resolves more high-weight edges in 10 samples and $A2$ resolves more high-weight edges in 54 samples than D-SQUID.

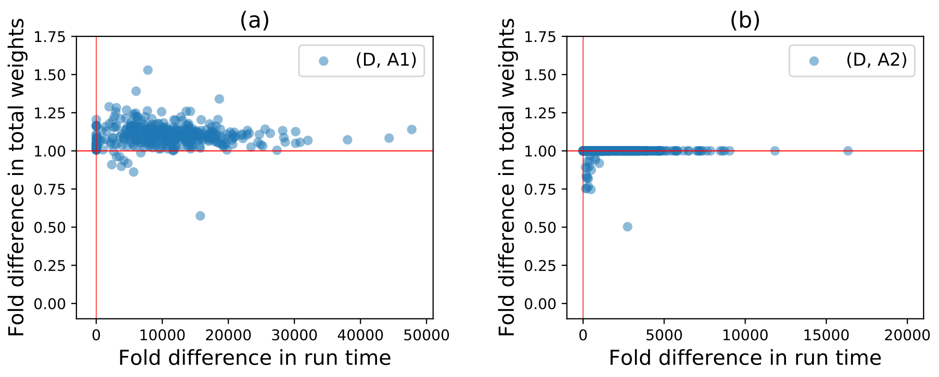

Figure 4 Fold differences (ILP/approx) in run time and total weights of concordant edges resolved by D-SQUID, $A1$ and $A2$ on TCGA samples. Horizontal and vertical red lines mark 1.0 on both axes. (a) shows fold differences between D-SQUID and $A1$. (b) shows fold differences betweeen D-SQUID and $A2$.

9 Conclusion and Discussion

We present approaches to identify TSVs in heterogeneous samples via the Multiple Compatible Arrangement Problem (MCAP). We characterize sample heterogeneity in terms of the fraction of discordant edges involved in conflict structures. In the majority of TCGA samples, the fractions of discordant edges in conflict structures are high compared to HCC samples, which indicates that TCGA samples are more heterogeneous than HCC samples. This matches the fact that bulk tumor samples often contain more heterogeneous genomes than cancer cell lines, which suggests that fraction of conflicting discordant edges is a valid measure of sample heterogeneity.

MCAP addresses this heterogeneity. In 381 TCGA samples, D-SQUID is able to resolve more conflicting discordant edges than SQUID. In HCC cell lines, D-SQUID achieves better performance than SQUID. Since D-SQUID solves MCAP by separating conflicting TSVs onto two alleles, D-SQUID's power to find TSVs generally increases as the extent of heterogeneity increases.

We show that obtaining exact solutions to MCAP is NP-hard. We derive an integer linear programming (ILP) formulation to solve MCAP exactly. We provide a $\frac{3}{16}$-approximation algorithm for MCAP when the number of arrangements is two ($k = 2$), which runs in time $O(|V||E|)$. It approximates the exact solutions well in heterogeneous TCGA samples.

Several open problems remain. MCAP relies on the number of arrangements (k) to make predictions. It is not trivial to determine the optimal k for any sample. In addition, although MCAP is solved by separating TSVs onto different alleles, there are typically many equivalent phasings. Developing techniques for handling these alternative phasings is an interesting direction for future work. Analyzing the effect of TSVs, especially non-fusion-gene ones, on cellular functions and diseases is another direction of futher work.

References

1 Dvir Aran, Marina Sirota, and Atul J Butte. Systematic pan-cancer analysis of tumour purity. *Nature Communications*, 6:8971, 2015.

2 Ken Chen, John W Wallis, Michael D McLellan, David E Larson, Joelle M Kalicki, Craig S Pohl, Sean D McGrath, Michael C Wendl, Qunyuan Zhang, Devin P Locke, et al. BreakDancer: an algorithm for high-resolution mapping of genomic structural variation. *Nature Methods*, 6(9):677, 2009.

3 Nadia M Davidson, Ian J Majewski, and Alicia Oshlack. JAFFA: High sensitivity transcriptome-focused fusion gene detection. *Genome Medicine*, 7(1):43, 2015.

4 Michael WN Deininger, John M Goldman, and Junia V Melo. The molecular biology of chronic myeloid leukemia. *Blood*, 96(10):3343–3356, 2000.

5 Jesse R Dixon, Jie Xu, Vishnu Dileep, Ye Zhan, Fan Song, et al. Integrative detection and analysis of structural variation in cancer genomes. *Nature Genetics*, 50(10):1388, 2018.

6 Adi F Gazdar, Venkatesh Kurvari, Arvind Virmani, Lauren Gollahon, Masahiro Sakaguchi, et al. Characterization of paired tumor and non-tumor cell lines established from patients with breast cancer. *International Journal of Cancer*, 78(6):766–774, 1998.

7 Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using NetworkX. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

8 Steffen Heber, Max Alekseyev, Sing-Hoi Sze, Haixu Tang, and Pavel A Pevzner. Splicing graphs and EST assembly problem. *Bioinformatics*, 18(suppl_1):S181–S188, 2002.

9 Fereydoun Hormozdiari, Iman Hajirasouliha, Phuong Dao, Faraz Hach, Deniz Yorukoglu, Can Alkan, Evan E Eichler, and S Cenk Sahinalp. Next-generation VariationHunter: combinatorial algorithms for transposon insertion discovery. *Bioinformatics*, 26(12):i350–i357, 2010.

10 Zhiqin Huang, David TW Jones, Yonghe Wu, Peter Lichter, and Marc Zapatka. confFuse: high-confidence fusion gene detection across tumor entities. *Frontiers in Genetics*, 8:137, 2017.

11 Wenlong Jia, Kunlong Qiu, Minghui He, Pengfei Song, Quan Zhou, et al. SOAPfuse: an algorithm for identifying fusion transcripts from paired-end RNA-Seq data. *Genome Biology*, 14(2):R12, 2013.

12 Ryan M Layer, Colby Chiang, Aaron R Quinlan, and Ira M Hall. LUMPY: a probabilistic framework for structural variant discovery. *Genome Biology*, 15(6):R84, 2014.

13 Silvia Liu, Wei-Hsiang Tsai, Ying Ding, Rui Chen, Zhou Fang, et al. Comprehensive evaluation of fusion transcript detection algorithms and a meta-caller to combine top performing methods in paired-end RNA-seq data. *Nucleic Acids Research*, 44(5):e47–e47, 2015.

14 Cong Ma, Mingfu Shao, and Carl Kingsford. SQUID: transcriptomic structural variation detection from RNA-seq. *Genome Biology*, 19(1):52, 2018.

15 Andrew McPherson, Fereydoun Hormozdiari, Abdalnasser Zayed, Ryan Giuliany, Gavin Ha, et al. deFuse: an algorithm for gene fusion discovery in tumor RNA-Seq data. *PLoS Computational Biology*, 7(5):e1001138, 2011.

16 Daniel Nicorici, Mihaela Satalan, Henrik Edgren, Sara Kangaspeska, Astrid Murumagi, Olli Kallioniemi, Sami Virtanen, and Olavi Kilkku. FusionCatcher—a tool for finding somatic fusion genes in paired-end RNA-sequencing data. *BioRxiv*, page 011650, 2014.

17 Nicholas A Nystrom, Michael J Levine, Ralph Z Roskies, and J Scott. Bridges: a uniquely flexible HPC resource for new communities and data analytics. In *Proceedings of the 2015 XSEDE Conference: Scientific Advancements Enabled by Enhanced Cyberinfrastructure*, page 30. ACM, 2015.

18 Tobias Rausch, Thomas Zichner, Andreas Schlattl, Adrian M Stütz, Vladimir Benes, and Jan O Korbel. DELLY: structural variant discovery by integrated paired-end and split-read analysis. *Bioinformatics*, 28(18):i333–i339, 2012.

19 Robert Sedgewick. *Algorithms in C, Part 5: Graph Algorithms, Third Edition.* Addison-Wesley Professional, third edition, 2001.

20 Fritz J Sedlazeck, Philipp Rescheneder, Moritz Smolka, Han Fang, Maria Nattestad, Arndt von Haeseler, and Michael C Schatz. Accurate detection of complex structural variations using single-molecule sequencing. *Nature Methods*, 15(6):461–468, 2018.

21 Wandaliz Torres-García, Siyuan Zheng, Andrey Sivachenko, Rahulsimham Vegesna, Qianghu Wang, Rong Yao, Michael F Berger, John N Weinstein, Gad Getz, and Roel GW Verhaak. PRADA: pipeline for RNA sequencing data analysis. *Bioinformatics*, 30(15):2224–2226, 2014.

22 Xiaoke Wang, Renata Q Zamolyi, Hongying Zhang, Vera L Pannain, Fabiola Medeiros, Michele Erickson-Johnson, Robert B Jenkins, and Andre M Oliveira. Fusion of HMGA1 to the LPP/TPRG1 intergenic region in a lipoma identified by mapping paraffin-embedded tissues. *Cancer Genetics and Cytogenetics*, 196(1):64–67, 2010.

A Proof of NP-hardness

▶ **Theorem 1.** *SCAP is NP-hard.*

Proof. To prove the NP-hardness, we reduce from MAX-2-SAT problem. It is necessary and sufficient to show that for any MAX-2-SAT problem, a genome segment graph (GSG) can be constructed in polynomial time, and the SCAP objective directly tells the objective of the MAX-2-SAT problem. For any MAX-2-SAT instance, we are going to construct a GSG such that the satisfiability of a clause is indicated by the concordance of an edge.

Given a MAX-2-SAT problem with n booleans $\{x_1, x_2, \cdots, x_n\}$ and m clauses $\{c_1, c_2, \cdots, c_m\}$, the key gadget is the segments for boolean variables and clauses and the edges between them (Figure 5A). For each boolean variable x_i, a segment X_i is constructed and termed as a boolean segment. For each clause c_i, a segment C_i is constructed and termed as a clause segment. To ensure that the correspondence between the edge concordance and the clause satisfiability as well as the correspondence between the orientation of boolean segments and the assignment of boolean variables, we add edges between clause segments and boolean segments in the following way. For clause c_i that involves boolean x_{i_1}, an edge is added between the head of X_{i_1} and the head of C_i if clause c_i contains the negation of x_{i_1}, otherwise the edge is between the tail of X_{i_1} and the head of C_i. When the literal is x_{i_1}, setting the orientation of segment X_{i_1} to be 1 indicates assigning True to variable x_{i_1} and leads to the concordance of the edge; when the literal is $\bar{x}_{i_1}$, setting the orientation of segment X_{i_1} to be 0 indicates assigning False to variable x_{i_1} and leads to the edge concordance. The edge between clause c_i and the other involved boolean variable x_{i_2} is added in the same principle. We call the edge between boolean segments and the clause segments as Type 1 edge. Type 1 edges have weight of 1.

Two extra edges between the two boolean segments involved in each clause are added. This is the Type 2 edge with weight of 1. For each clause c_i that involves boolean x_{i_1} and x_{i_2}, two edges are added between X_{i_1} and X_{i_2} as in Table 1. When both literals in c_i are True, there are two concordant Type 1 edges; when only one literal in c_i is True, one and only one of the two Type 2 edges is guaranteed to be concordant, to compensate for the decrease of concordant Type 1 edges.

An extra $n + m + 1$ segments are added that we term blocking segments and denote as $\{B_1, B_2, \cdots, B_{n+m+1}\}$. Suppose w_1 and w_2 are large positive weights, and $w_2 \gg w_1 \gg 1$. Type 3 edges with edge weight w_2 are constructed between each adjacent pair of blocking segments, specifically between the tail of B_i and the head of B_{i+1} ($\forall i \in [1, n + m]$). Type 3 edges are used to enforce the order and orientation among blocking segments. Type 4 edges with weight w_1 are constructed between blocking segments and the other types of segments. Specifically, when $i \leq n$, an edge is added between the tail of segment B_i and both the head and the tail of X_i, as well as between the tail and the head of X_i and both the head of B_{i+1}.

Similarly when $n < i \leq n + m$, two edges are added between the tail of B_i and C_{i-n}, and two other edges are added between the head and tail of C_{i-n} and B_{i+1}. Type 4 edges are used to enforce the relative order between blocking segments and the boolean and clause segments. But the orientation of the boolean and clause segments can be changed freely.

Table 1 Construction of Type 4 edges based on the clause.

clause c_i	edge 1	edge 2
$x_{i_1} \vee x_{i_2}$	head of X_{i_1} to head of X_{i_2}	tail of X_{i_1} to tail of X_{i_2}
$\bar{x}_{i_1} \vee x_{i_2}$	tail of X_{i_1} to head of X_{i_2}	head of X_{i_1} to tail of X_{i_2}
$x_{i_1} \vee \bar{x}_{i_2}$	tail of X_{i_1} to head of X_{i_2}	head of X_{i_1} to tail of X_{i_2}
$\bar{x}_{i_1} \vee \bar{x}_{i_2}$	head of X_{i_1} to head of X_{i_2}	tail of X_{i_1} to tail of X_{i_2}

We first prove that the order of the blocking segments in the optimal arrangement is $B_1 < B_2 < \cdots < B_{n+m+1}$ and the orientations of them are all in forward strand, where $<$ denotes the ordering between segments. Under the arrangement that uses the forward strand of all $\{B_i\}$ and have an order of $B_1 < B_2 < \cdots < B_{n+m+1}$, the sum of concordant edge weights is at least $(n + m)w_2$. If the optimal arrangement contains any violations of the adjacencies between B_i and B_{i+1}, there will at least one Type 3 edge that does not connect blocking segments in a tail-to-head manner and become a discordant edge in the arrangement. Therefore, the optimal arrangement can at most have an objective value of $(n + m - 1)w_2 + 4(n + m)w_1 + 4m$. Since $w_2 \gg w_1 \gg 1$, the objective value is smaller than $(n+m)w_2$, and the arrangement is not optimal, which contradicts the assumption. Therefore assuming the whole chain of segments is not reverse complemented, the orientations of blocking segments are all in forward strand, and order is $B_1 < B_2 < \cdots < B_{n+m+1}$ in the optimal arrangement.

We then prove that the Type 2 edges restrict the order of all segments but not the orientation of boolean and clause segments. The order between blocking segments and boolean segments must be $B_i < X_i < B_{i+1}$, the order between blocking and clause segments must be $B_i < C_{i-n} < B_{i+1}$, and all boolean segments must be before clause segments. When the order is $B_i < X_i < B_{i+1}$ among the three segments, and the orientations of B_i and B_{i+1} are both in forward strand, the concordant edge weights of Type two edge sum to $2w_1$ no matter whether X_i is in forward strand or inverted. The same weight can be achieved for order $B_i < C_{i-n} < B_{i+1}$. The arrangement with order $B_1 < X_1 < B_2 < \cdots < B_n < X_n < B_{n+1} < C_1 < B_{n+2} < \cdots < C_m < B_{n+m+1}$ and with all blocking segments in their forward strand will achieve a sum of concordant edge weight $(n + m)w_2 + 2(n + m)w_1$ at least. This concordant weight is summed over Type 3 and Type 4 edges. However, if the optimal arrangement violates any $B_i < X_i < B_{i+1}$ or $B_i < C_{i-n} < B_{i+1}$ order, the violated triplet can achieve at most w_1 of concordant edge weights, and thus the maximum sum of concordant edge weights is $(n + m)w_2 + 2(n + m - 1)w_1 + w_1 + 4m$. Since $w_1 \gg 1$, the "optimal" arrangement objective is smaller than $(n + m)w_2 + 2(n + m)w_1$, which contradicts the optimality. Therefore, the order of all segments in the optimal arrangement must be

$$B_1 < X_1 < B_2 < \cdots < B_n < X_n < B_{n+1} < C_1 < B_{n+2} < \cdots < C_m < B_{n+m+1}.$$

Third, we prove that under the above segment order there are always two concordant edges of weight 1 when clause segment C_i has any concordant Type 1 edge. Suppose there is a clause c_i involving boolean variables x_{i_1} and x_{i_2}, segment C_i has one Type 1 edge between X_{i_1} and one Type 1 edge between X_{i_2}. When both Type 1 edges are concordant, both

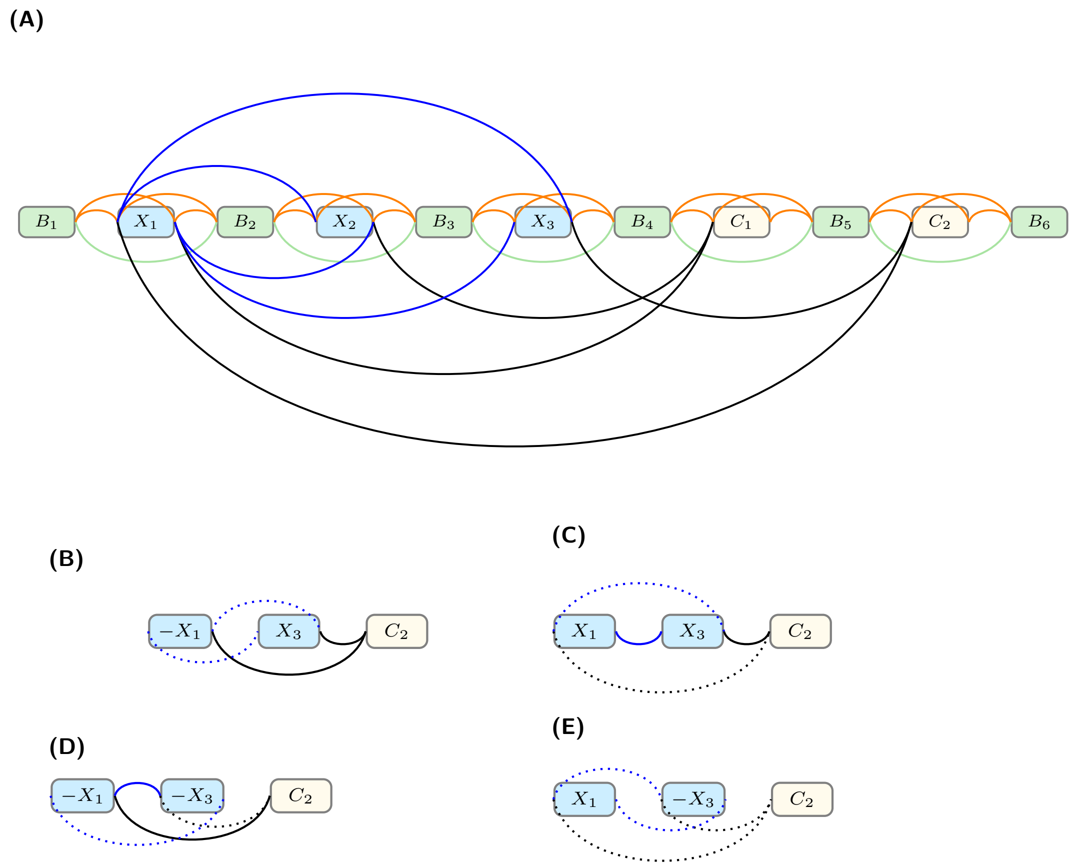

Figure 5 (A) Constructed GSG for boolean expression $(x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_3)$. There is a segment for each boolean variable x_i (blue) and clause c_i (white), and 6 blocking segments (green) to separate between boolean segments and clause segments. Type 1 edges, black edges, are connecting between boolean segments and clause segments. Type 2 edges, blue edges, are connecting between a pair of boolean segments that appear in the same clause. Type 3 edges, green edges in the figure, are chaining the blocking segments. Type 4 edges, orange edges, are connecting between blocking and boolean / clause segments. (B-E) The subgraph corresponding clause $\bar{x_1} \vee x_3$. $-X_1$ and $-X_3$ means the segment is inverted. Solid lines indicate the concordant edges in the arrangement, and dotted lines indicate the discordant edges. (B) The clause is satisfied with both literals satisfied. (C) The clause is satisfied with x_3 satisfied. (D) The clause is satisfied with $\bar{x}_1$ satisfied. (E) The clause is not satisfied.

Type 2 edges between X_{i_1} and X_{i_2} are discordant (Figure 5B. When only one of the Type 1 edges is concordant, there is also one Type 2 edge between X_{i_1} and X_{i_2} that is concordant (Figure 5C,D). When neither of the Type 1 edges is concordant, both of the two Type 2 edges between X_{i_1} and X_{i_2} are discordant (Figure 5E). In this case, there is zero concordant edges of weight 1 incident to C_i. Any arrangement solution of objective value W that satisfies the above segment order has $W - (n+m)w_2 - 2(n+m)w_1$ concordant edges of weight 1. Therefore, the arrangement solution will have $\frac{1}{2}(W - (n+m)w_2 - 2(n+m)w_1)$ clause segments with non-zero concordant Type 1 edges.

When multiple clauses involve the same pair of segment, multi-edges between X_{i_1} and X_{i_2} are constructed to make sure that two edges of weight 1 are contributed by any clause segment when it has non-zero concordant Type 1 edges.

Suppose the optimal number of satisfied clauses of the MAX-2-SAT instance is OPT_m and the optimal sum of concordant edge weights of the constructed SCAP instance is OPT_s, the following inequality holds: $\frac{1}{2}(OPT_s - (n+m)w_2 - 2(n+m)w_1) \geq OPT_m$. Given the optimal solution of the MAX-2-SAT instance, a SCAP solution can be constructed by reversing segment X_i if x_i is assigned to False while keeping the order of $B_1 < X_1 < B_2 < \cdots < B_n < X_n < B_{n+1} < C_1 < B_{n+2} < \cdots < C_m < B_{n+m+1}$. By the construction of the Type 1 edges, a clause segment will have at least one concordant Type 1 edge if and only if it corresponds to a satisfied clause in the MAX-2-SAT solution. Denoting the objective value of the constructed solution of arrangement problem as W and applying the third proof, we have the following equality $OPT_m = \frac{1}{2}(W - (n+m)w_2 - 2(n+m)w_1)$. Since the optimal objective value of the arrangement problem is as least W,

$$OPT_m = \frac{1}{2}(W - (n+m)w_2 - 2(n+m)w_1) \leq \frac{1}{2}(OPT_s - (n+m)w_2 - 2(n+m)w_1). \quad (9)$$

Meanwhile $\frac{1}{2}(OPT_s - (n+m)w_2 - 2(n+m)w_1) \leq OPT_s$. Given the optimal solution of arrangement problem, there are $\frac{1}{2}(OPT_s - (n+m)w_2 - 2(n+m)w_1)$ clause segments with non-zero concordant Type 1 edges. Construct a MAX-2-SAT solution by assigning False to boolean variables if the corresponding boolean segment is reversed otherwise assigning True. The concordance of Type 1 edges guarantees that the corresponding literals in the MAX-2-SAT clauses are True. Thus the constructed MAX-2-SAT solution will have $\frac{1}{2}(OPT_r - (n+m)w_2 - 2(n+m)w_1)$ satisfied clauses, which is smaller than or equal to the optimal number of satisfied clauses. Therefore

$$\frac{1}{2}(OPT_s - (n+m)w_2 - 2(n+m)w_1) \leq OPT_m. \quad (10)$$

Combining inequality (9) and inequality (10), the maximum number of satisfied clauses in MAX-2-SAT instance can be directly calculated from the optimal concordant edge weights in the arrangement problem, that is, $OPT_m = \frac{1}{2}(OPT_s - (n+m)w_2 - 2(n+m)w_1)$. ◀

B Proof of Characterization of Conflict Structures

▶ **Theorem 4.** *Any acyclic subgraph of GSG is a compatible structure.*

Proof. We show that any acyclic subgraph with N edges ($|E'|+|\hat{E}'| = N$), $G'_N = (E', \hat{E}', V')$, of GSG is a compatible structure by induction.
When $|E'| + |\hat{E}'| = 1$, G'_1 is a compatible structure because no other edge in G' is in conflict with the only edge $e \in E'$.
Assume the theorem hold for any acyclic subgraph that contains n edges. Let $G'_{n+1} = (E', \hat{E}, V')$ be an acyclic subgraph with $n+1$ edges. Since G'_{n+1} is acyclic, there must be a leaf edge that is incident to a leaf node. Denote the leaf node as v_b and the leaf edge $e = (u_a, v_b) \in E' \cup \hat{E}'$ $(a, b \in \{h, t\})$. By removing edge e and leaf node v_b, the subgraph $G'_n = (E' - \{e\}, \hat{E}' - \{e\}, V' - \{v_b\})$ is also acyclic and contains n edges. According to the assumption, G'_n is a compatible structure and there is an arrangement of the segments in which all edges in $E' \cup \hat{e}' - \{e\}$ is concordant. Because no other edge in $E' \cup \hat{E}'$ except e connects to v_b, it is always possible to place segment v back to the arrangement such that e is concordant. Specifically, one of the four placing options will satisfy edge e: the beginning of the arrangement with orientation 1, the beginning with orientation 0, the end with orientation 1 and the end with orientation 0. Therefore, G'_{n+1} is a compatible structure. By induction, acyclic subgraph G'_N of GSG with any $|E'|$ is a compatible structure. ◀

▶ **Theorem 5.** *A simple cycle $C = (E', \hat{E}', V')$ is a compatible structure if and only if there are exactly two vertices, v_j and v_i such that $deg_{E'}(v_i) = deg_{E'}(v_j) = 2$ and v_i and v_j belongs to different segments.*

Proof. We prove sufficiency and necessity separately in Lemma 6 and Lemma 7. ◀

▶ **Lemma 6.** *If C is a compatible structure, there are exactly two vertices, v_i, v_j that belong to different segments, such that $deg_{E'}(v_i) = deg_{E'}(v_j) = 2$*

Proof. We discuss compatiblity in two cases:

Case (1): All edges are concordant in C. Sort the vertices by genomic locations in ascending order and label the first vertex v_1 and the last v_n, assuming $|V'| = n$. Similarly, sort the set of segments S' in C by the values of their permutation function π and label the first segment s^1 and the last s^m, assuming $|S'| = m$. Since concordant connections can only be tail-head connections (e.g. Figure 1 b,c), $v_1 = s^1_t$ and $v_n = s^m_h$. Since C is a simple cycle, all vertices $v \in V'$ have $deg(v) = 2$. Because v_1 and v_n are the first and last vertices in this arrangement, the edges incident to v_1 or v_n must be in E'. It follows that the two edges incident to v_1 connects to s^2_h and s^m_h. Similarly, edges incident to v_n connects to s^1_t and s^{n-1}_t. Therefore, we have $deg_{E'}(v_1) = deg_{E'}(v_n) = 2$. Any other vertex v_i $(1 < i < n)$ is connected by one $e \in E'$ and one $\hat{e} \in \hat{E}'$ and thus has $deg_{E'}(v_i) = 1$.

Case (2): Some edges are discordant in C. If discordant edges exist in cycle C, according to the definition of compatible structure, segments in C can be arranged such that all edges are concordant. This reduces to case (1). ◀

▶ **Lemma 7.** *If there are exactly two vertices in V' that belong to different segments, v_i and v_j, such that $deg_{E'}(v_i) = deg_{E'}(v_j) = 2$, then C is a compatible structure.*

Proof. Let v_i and v_j be the one of the end points of segments s^i and $s^j (i \neq j)$, respectively. We can arrange s^i and s^j such that $\pi(s^i) = \min_{s \in S'} \pi(s)$, $\pi(s^j) = \max_{s \in S'} \pi(s)$ and that $v_i = s^i_t$, $v_j = s^j_h$. Rename v_i to v_1 and v_j to v_n. Since C is a simple cycle, we can find two simple paths, P_1 and P_2, between v_1 and v_n and there is no edge between P_1 and P_2. Let P'_1 and P'_2 denote P_1 and P_2 that exclude v_1 and v_n and the edges incident to v_1 and v_n. Since P'_1 and P'_2 as acyclic subgraphs of GSG, according to Theorem 14, P'_1 and P'_2 are compatible structures and therefore segments in P'_1 and P'_2 can be arranged so that all edges are concordant. Denote the first and last vertices in the arranged P'_1 as v_2 and v_3, and the first and last vertices in the arranged P'_2 as v_4 and v_5. Because all the edges are concordant in P'_1, v_2 and v_3 are the head and tail of the first and last segments in P'_1. Because only v_1 and v_n have $deg_{E'} = 2$ in C, v_2 must be connected to v_1 or v_n and v_3 must be connected to v_n or v_1. A similar argument applies to v_4 and v_5. To ensure concordance of edges connected to v_1 and v_n, if v_n is connected to v_2 and v_1 is connected to v_3, we flip all the segments in P'_1. The similar operation is applied to v_4, v_5 and P'_2. Now we have a compatible structure. ◀