Genome Rearrangement Problems with Single and Multiple Gene Copies: A Review

Ron Zeira and Ron Shamir

July 25, 2018

Dedicated to Bernard Moret upon his retirement.

Abstract Problems of genome rearrangement arise in both evolution and cancer research. Basic genome rearrangement models assume that the genome contains a single copy of each gene and the only changes in the genome are structural, i.e. reordering of segments. In contrast, numerical changes such as deletions and duplications, which change the number of copies of genes, have been observed in species evolution and prominently in tumorigenesis. Here, we review various computational models of evolution by rearrangements designed for the analysis of species or cancer genomes, focusing mainly on genomes with multiple gene copies. Models differ in the assumptions taken on the genome structure and in the type of rearrangements allowed during their evolution. Most problems regarding genomes with multiple gene copies are computationally hard, and practical methods for their analysis are reviewed. As more high resolution genomes become available, especially in cancer, better models and efficient algorithms will be needed.

Prologue

The computational study of genome rearrangements is a sub-area of computational biology born about 25 years ago. Over that period, it has flourished and developed into a fascinating research area, combining beautiful combinatorial models, elegant theory and applications. Models of the first generation, motivated by species evolution, were simple (though their analysis was sometimes quite sophisticated) and

Ron Zeira
Tel Aviv Universirty, Tel Aviv, Israel, e-mail: ronzeira@post.tau.ac.il

Ron Shamir
Tel Aviv Universirty, Tel Aviv, Israel e-mail: rshamir@tau.ac.il

assumed that genomes contain only one copy of each gene. With the explosion of biological data, new analysis opportunities arose, necessitating more complex models and theory.

This manuscript describes some of the problems and results related to rearrangement models allowing multiple gene copies. This research area is motivated by both species and cancer evolution. The latter is stimulated by the recent large scale deep sequencing of thousands of tumor genomes, which has brought about a plethora of novel challenges. Our main focus is on multi-copy models, but key single-copy models are also reviewed briefly for context.

This review is by no means exhaustive. The field of modeling genome rearrangements is vast and cannot be covered in one paper. The selection of topics reflects our knowledge (or lack thereof) and taste, and we apologize to the many researchers whose work is not mentioned. For further reading see, e.g., [126, 42, 39].

1 Introduction

In this section we give biological introduction and motivation to genome rearrangements (*GR*) in both evolution and cancer. [1] Section 2 gives computational background and some fundamental results in the analysis of single copy genomes. In sections 3 and 4 we review GR models that handle genomes with multiple gene copies in the context of evolution and cancer.

1.1 Genomes and rearrangements

The *genome*[2] encodes instructions used in the development and functioning of all living organisms (bacteria, plants, animals etc.). Genomes are built of *DNA* (*DeoxyriboNucleic Acid*), a double-stranded molecule in which each *strand* is a long sequence of *nucleotides* (or *bases*). Each base can be of four types A, C, G and T. The two strands are *complementary* such that an A on one strand is coupled with a T on the other strand, and similarly C is coupled with G. Because of this complementarity, one strand completely determines the other, and DNA molecules are usually represented by the sequence of one strand.

[1] Abbreviations: aCGH: Array comparative genomic hybridization; AG: adjacency graph; BFB: breakage-fusion-bridge; BG: breakpoint graph; BP: breakpoint; CN: copy number; CNA: copy number alteration; CNO: copy number operation; CNP: copy number profile; DCJ: double cut and join; DNA: DeoxyriboNucleic Acid; FISH: fluorescence in situ hybridization; GR: genome rearrangement; ILP: integer linear programming; SCoJ: single cut or join; SNV: single nucleotide variation; SV: structural variation; WGD: whole genome duplication;

[2] Since this review concentrates mainly on the computational aspects of GR, we only give brief biological introduction. We italicize terms that actually require definitions. For concise biological definitions see, e.g., [65].

The *genome* is the total DNA material in the cell. It is partitioned into contiguous subsequences called *chromosomes*. Chromosomes can be either *linear* and contain two ends called *telomeres*, or *circular*. A *gene* is a segment along the chromosome containing information for construction of *protein*. Proteins are molecules that form the "machines" and building blocks of most cellular functions. The direction in which a gene is transcribed into a protein on a given strand determines its *orientation*. Genes are a basic unit of heredity passed from one generation to the other.

The cause of diversity of organisms is the ability of DNA to replicate itself with some inaccuracy. This inaccuracy underlies molecular evolution: DNA changes between generations open the possibilities for modified genes, new genes, and eventually new species.

Genomes can evolve in a local and global manner. *Local* alterations refer to *point mutations* in the DNA sequence, that can either *substitute* a single base with a different one, *insert* a single base into the sequence or *delete* a base from the sequence. On the other hand, a sequence can also evolve by modifying its organization on a large scale. These *global* mutations, called *genome rearrangements* or *structural variations*, relocate, duplicate, or delete large fragments of the DNA. The main rearrangement types include the following (compare Figure 1):

- *Deletion*. A segment of DNA is lost. A *chromosome deletion* is a deletion of an entire chromosome.
- *Inversion* or *reversal*. A segment is cut and reinserted in the opposite orientation. Since the insertion reverses the two strands, the result is an inverted and reverse complemented DNA sequence.
- *Transposition*. A DNA segment is moved to a different location.
- *Duplication*. A genomic segment is copied and reinserted into the genome. In a *tandem duplication* the copy is inserted right after the original one. An arbitrary (non tandem) duplication inserts the new copy at an arbitrary position (one particular type of such is *retrotransposition*). A *whole chromosome duplication* makes another copy of an entire chromosome. A *whole genome duplication* duplicates all the genome's chromosomes.
- *Translocation*. Two linear chromosomes exchange their end segments.
- *Fusion*. Two chromosomes are joined into one.
- *Fission*. A chromosome splits into two chromosomes.

The above rearrangement operations affect DNA segments rather than nucleotides and thus genomes are often represented by sequences of segments in this context. Two segments are called *homologous* if they derive from a common ancestor either by speciation (in that case the segments appear in the genomes of different species) or by duplication (where they occur on the same genome).

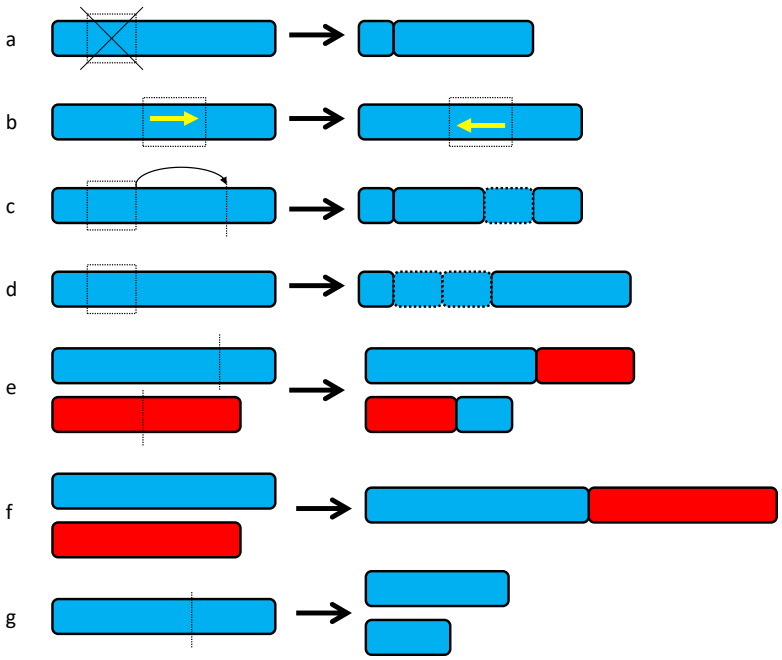

Fig. 1 *Genome rearrangements*. a. Deletion. b. Reversal. c. Transposition. d. Tandem duplication. e. Translocation. f. Fusion. g. Fission.

1.2 Genome rearrangements in Evolution

The genomes of related species are very similar. For instance, most of the mouse and human genomes can be divided into blocks in which gene content is conserved [27]. However, the order of these blocks along the human and mouse genomes is different. This difference is attributed to rearrangement events occurring after the divergence of the two lineages.

The phenomenon of GR in evolution was discovered by Sturtevant and Dobzhansky who demonstrated inversions between genomes of drosophila species [100]. Palmer and colleagues observed that mitochondrial DNA of related plant species have similar gene content but different segment ordering (Figure 2) [77, 99]. The detection of GRs in those studies was largely based on molecular cytogenetics techniques such as *chromosome banding* and *in-situ hybridization* [82]. These studies mostly focused on relatively close species and a small number of rearrangements between them [90]. With the advent of sequencing technologies, bioinformatic methods enabled locating homologous blocks in different genome sequences, thus creating finer comparative maps based on genome sequences [79]. Sankoff pioneered the computational study of GR in species evolution [87, 89]. The basic assumption of all computational models is that evolution is parsimonious and prefers a shortest sequence of events. In their seminal works, Hannenhalli and Pevzner gave the first

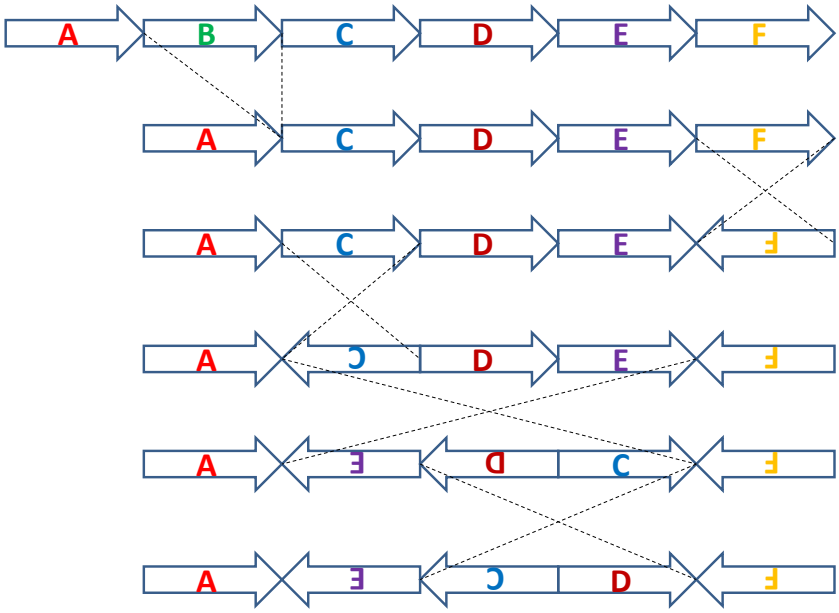

Fig. 2 A sorting scenario for the chloroplast genome evolution between two conifers. The genome at the top is transformed to the one at the bottom in five steps. The first is a deletion and the next four are inversions of genomic segments. The ends of the involved segments are indicated by the broken lines. Adapted and simplified from Strauss *et al*. [99].

polynomial algorithm for the problem of transforming one genome into the other by minimum number of reversals and of reversals and translocations, respectively [49, 50]. They used their algorithm to give a shortest event sequence between men and mice, and between cabbage and turnip.

Classical computational rearrangement models assume that each gene in the two genomes under study appears only once and that 1-1 homology between the genes of the genomes has been established. While this assumption may hold for closely related genomes, it is unwarranted for divergent species with several copies of the same genes or highly homologous genes. Duplications are an important source of new gene functions since new gene copies tend to diverge through mutations and develop new functions. For instance, evidence of whole genome duplication events have been observed in most angiosperm sequenced genomes [20].

1.3 Genome rearrangements in Cancer

Cancer is a complex disease driven by the accumulation of somatic DNA mutations over generations of cell divisions. Such mutations affect tumor growth, clinical progression, immune escape, and drug resistance [32].

Mutations in cancer cells can be local, affecting single DNA base pairs. These mutations, called *single nucleotide variants* (*SNV*), can number in thousands per cancer cell. On the other hand, large scale mutations, i.e. GRs, can relocate fragments of the DNA. Aberrations that change the amount of genomic content, called *copy number alterations* (*CNAs*) include duplications and deletions of genomic regions. The *karyotype* of a cell is its complete set of chromosomes, consisting of the number and structure of the chromosomes in it. Large-scale aberrations can have a dramatic effect on the cancer karyotype (see Figure 3).

Somatic mutations may amplify genes that promote cancer (*oncogenes*) or harm genes that inhibit cancer development (*tumor suppressor* genes). In addition, rearrangements such as translocations and inversions may change gene structure and regulation and create novel fusion genes, with or without additional changes in copy number (CN).

Cancer can be viewed as an evolutionary process in which a normal genome accumulates mutations that eventually transform it into a cancerous one [11]. The gain of advantageous mutations leads to a *clonal expansion*, forming a larger population of the mutated cells. Subsequent clonal expansions occur as additional advantageous mutations accumulate in descendant cells. A single tumor biopsy will often contain a mixture of several competing tumor clones. These tumor clones frequently differ in their genomic content and structure. When sequencing the tumor, one actually obtains a conflation of several tumor clones and of normal cells. Recent research suggests that this heterogeneity has profound clinical implications [32].

The classical ways to detect chromosomal abnormalities in cytogenetics are *G-banding* and *fluorescence in situ hybridization* (*FISH*), which allow viewing the chromosome in metaphase at low resolution [81]. FISH measures the CN of tens to hundreds of targeted genes [29]. *Array comparative genomic hybridization* (*array CGH*) gives a higher resolution of CN estimation for a cell population [107]. Today, next generation sequencing techniques are the main data source for cancer mutation analyses [33]. Whole genome sequencing provides tens to hundreds of millions of DNA reads that enable the detection of SNVs. In addition, paired-end read technologies give evidence of structural rearrangement operations when the two ends of the same read pair are mapped to two discordant locations in the genome [73]. Sequencing read depth data can also be used to assess CNAs [73].

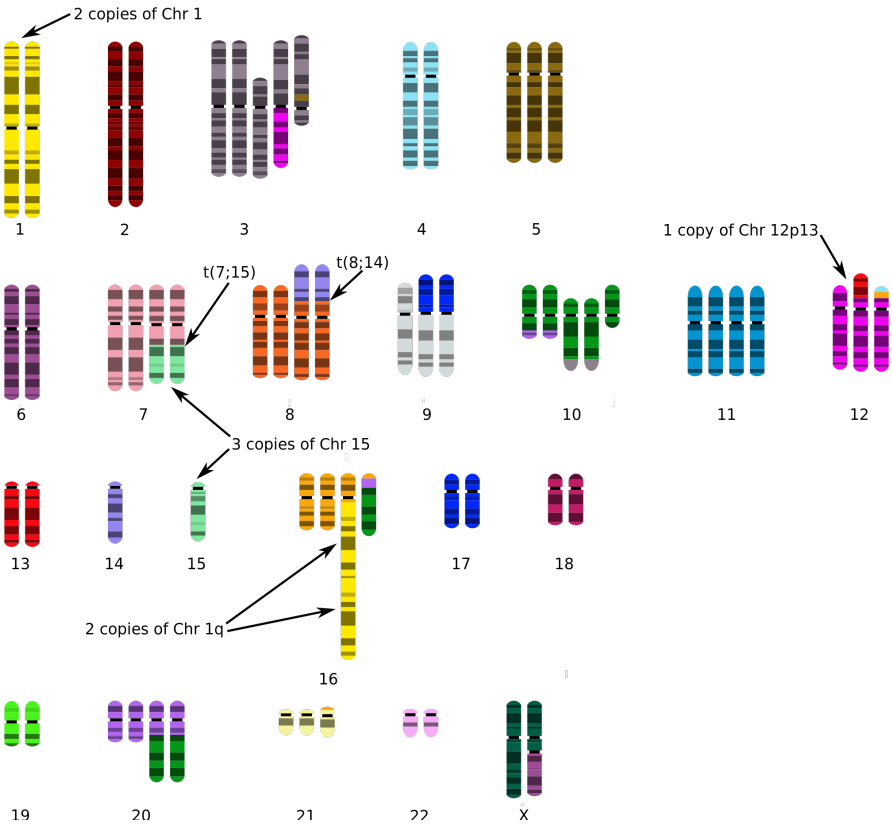

Fig. 3 A schematic of the karyotype of the T47D breast cancer cell line. The chromosome numbers in the normal diploid are indicated below each subfigure. In a normal karyotype, each chromosome has two copies, as for Chr. 4, 13, 17 and 18. Among the GRs in this cancer genome we see chromosomal duplications (e.g., four copies of Chr. 11), translocations (between Chr. 8 and Chr. 14), and more complex events (e.g., tandem duplication of one arm of Chr. 1 and fusion with an extra arm of Chr. 16). Image source: [10] and Wikimedia Commons [52].

2 Single gene models, operation types and distance measures

In this sections we give a brief introduction to GR models. We start by giving the definitions and terminology used in computational GR analysis. We then review several classical single gene models.

2.1 Genome representation

Consider a set $\mathscr{G}$ of n segments in the genome. For convenience, we call the segments in $\mathscr{G}$ *genes*, though they do not necessarily represent biological gene enti-

ties. A gene g is an oriented sequence of DNA that starts with a *tail* and ends with a *head*, denoted as g_t and g_h respectively. The set of *extremities* of the genes is $\mathscr{E} = \{g_t | g \in \mathscr{G}\} \cup \{g_h | g \in \mathscr{G}\}$.

An *adjacency* between two consecutive genes in a genome is an unordered pair of extremities. Thus, an adjacency between two genes $a, b \in \mathscr{G}$ can take one of four forms, depending on their orientation: $\{a_h, b_t\}, \{a_t, b_h\}, \{a_t, b_t\}, \{a_h, b_h\}$. An extremity that is not adjacent to another extremity is called a *telomere*, and is represented by a singleton set, e.g. $\{a_h\}$.

In some formulations, a gene may have multiple copies corresponding, for example, to homologous yet distinguishable genes. The copies of such a gene $g \in \mathscr{G}$ are identified by a superscript. For example, g^1, g^2, g^3 are three distinct copies of gene g. Such a gene with multiple copies is called a *labeled* gene. A gene that has a single copy or has multiple indistinguishable copies is called *unlabeled*. For a gene g, we call the number of copies it has its *copy number* and denote it by $cn(g)$. A gene set $\mathscr{G}$ with one copy for each gene is called an *ordinary gene set*. A *labeled gene set* is a set $\mathscr{G}^L = \{g^i | g \in \mathscr{G}, 1 \leq i \leq cn(g)\}$ and an *unlabeled gene set* $\mathscr{G}^U$ is a multiset $\mathscr{G}^U = \cup_{g \in \mathscr{G}} \cup_{1 \leq i \leq cn(g)} \{g\}$. For instance, $\mathscr{G}^L = \{a^1, a^2, b^1, c^1, c^2, c^3\}$ and $\mathscr{G}^U = \{a, a, b, c, c, c\}$ are labeled and unlabeled gene sets, respectively, that have two copies of gene a, one of b and three of c.

A *labeled genome* Π over a labeled gene set $\mathscr{G}^L$ is a set of adjacencies and telomeres such that every labeled extremity $e^i \in \mathscr{E}^L$ appears exactly once in an adjacency or telomere of Π. Similarly, an *unlabeled genome* Π over an unlabeled gene set $\mathscr{G}^U$ is a multiset of adjacencies and telomeres such that every unlabeled extremity $e \in \mathscr{E}^U$ of gene g appears exactly $cn(g)$ times in adjacencies or telomeres of Π. $\Pi = \{\{a_t^1\}, \{a_h^1, b_h^1\}, \{b_t^1, c_t^1\}, \{c_h^1\}, \{a_t^2\}, \{a_h^2, b_t^2\}, \{b_h^2\}, \{b_t^3\}, \{b_h^3, c_h^2\}, \{c_t^2\}\}$ and $\Gamma = \{\{a_t\}, \{a_h, b_h\}, \{b_t, c_t\}, \{c_h\}, \{a_t\}, \{a_h, b_t\}, \{b_h\}, \{b_t\}, \{b_h, c_h\}, \{c_t\}\}$ are examples of labeled and unlabeled genomes. If the gene set of a genome is ordinary, we call it an *ordinary genome* (or a *single copy genome*).

The *graph representation* of a genome Π is an undirected graph $G_\Pi = (\mathscr{E}, E)$. Its nodes are the extremities of Π (either labeled or unlabeled) and E consists of interval edges and adjacency edges. An *interval edge* connects the head and tail of a gene. For unlabeled genomes there are $cn(g)$ parallel interval edges of the edge (g_h, g_t) for every gene g. For labeled genomes, each labeled gene copy g^i has a single interval edge (g_h^i, g_t^i). *Adjacency edges* connect the extremities x and y where $\{x, y\}$ is an adjacency of Π. We call G_Π the *genome graph* of Π. The representations Π and G_Π are equivalent and thus we use them interchangeably. Figures 4, 5 and 6 show genome graphs for ordinary, labeled and unlabeled genomes, respectively.

An *alternating route* in G_Π is either a path or a cycle in which no two consecutive edges are of the same type (interval/adjacency). A *chromosome decomposition* D_Π of the genome Π is a decomposition of G_Π into a set of alternating cycles and alternating paths that start and end with telomeres. Note that a chromosome decomposition is always possible since the interval-degree is equal to the adjacency-degree for every node that is not in a telomere, and that paths must start and end with telomeres. Labeled and ordinary genomes have a unique chromosome decomposition, since the interval-degree and the adjacency-degree of every non-telomere node

is 1 (see Figures 4 and 5). There may be several decompositions for a multi-copy unlabeled genome (see Figure 6). Each route in a decomposition is called a *chromosome*. A chromosome is called *circular* if the corresponding route is a cycle, and *linear* otherwise. A decomposition is called *linear* if all its chromosomes are linear, *circular* if all its chromosomes are circular, and otherwise *mixed*. Figure 4 shows an ordinary genome with one linear and one circular chromosome. An ordinary genome composed of a single linear chromosome is called a *signed permutation*.

For a chromosome $C \in D_\Pi$, we define the *chromosome string* of C as follows. Start at one of the ends of a linear chromosome with the string '('. Traverse the route until all edges along the route are covered. For each traversal of an interval edge from a tail g_t to a head g_h append g to the string. For traversal from g_h to g_t append $-g$ to the string. After finishing the traversal, append the string with ')'. For a chromosome string C, let $-C$ be the chromosome string in which the order and orientation of all gene are in inverted, e.g. if $C = (1\ 2\ 3)$ then $-C = (-3\ -2\ -1)$. C and $-C$ are *equivalent* as they correspond to the same set of adjacencies. For a circular chromosome, do the same starting from an arbitrary extremity interval edge without appending brackets. The resulting sequence is cyclic and all shifts and inversions of it are equivalent. We use $<>$ to denote circular genomes. (Figures 4, 5 and 6).

A *string representation of a genome decomposition* D_Π is the multi-set of chromosome strings for each chromosome in the decomposition (Figures 4, 5 and 6). Two string representations are equivalent if there is a bijective mapping between equivalent chromosome strings in them. For labeled and ordinary genomes, the string representation is unique (up to equivalence) and therefore we sometime use this representation.

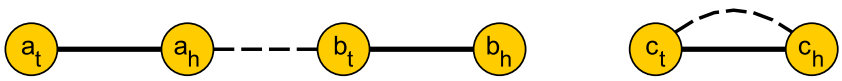

Fig. 4 A genome graph G_Π of an ordinary genome $\Pi = \{\{a_t\},\{a_h,b_t\},\{b_h\},\{c_t,c_h\}\}$. Bold edges correspond to interval edges; dashed edges correspond to adjacencies. Since Π is an ordinary genome, it has a unique decomposition D_Π whose string representation is $\{(a\,b),<c>\}$

Given an unlabeled genome Π over the gene set $\mathscr{G}^U$, a *labeling* of Π produces a labeled genome Γ over the gene set $\mathscr{G}^L$ such that distinct gene copies of a gene g are mapped to distinct labeled genes $g^1,\ldots,g^{cn(g)}$ in $\mathscr{G}^L$. For example, the labeled genomes in Figure 5 and 6B are two possible labelings of the unlabeled genome in Figure 6A. We denote $L(\Pi)$ to be the set of all possible labelings of Π.

Given a genome Π_1 over the gene set $\mathscr{G}_1$, an *operation* creates a new genome $\Pi_2 \neq \Pi_1$ over a new gene set $\mathscr{G}_2$. An operation is said to be *structural* if $\mathscr{G}_1 = \mathscr{G}_2$. An operation is said to be a *numerical* if the CN of some gene is different under $\mathscr{G}_1$ and $\mathscr{G}_2$.

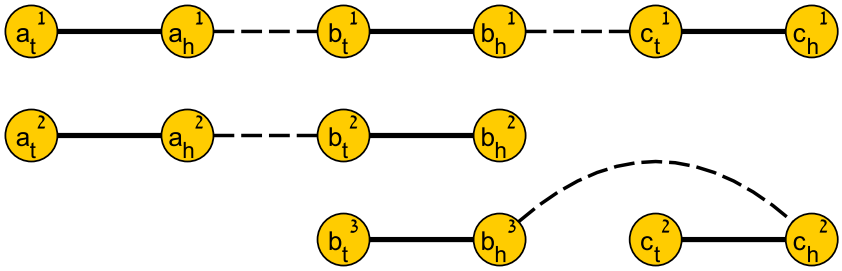

Fig. 5 A genome graph G_Δ of a labeled genome $\Delta = \{\{a_t^1\},\{a_h^1,b_t^1\},\{b_h^1,c_t^1\},\{c_h^1\},\{a_t^2\},\{a_h^2,b_t^2\},\{b_h^2\},\{b_t^3\},\{b_h^3,c_h^2\},\{c_t^2\}\}$. Bold edges correspond to interval edges; dashed edges correspond to adjacencies. Since Δ is an ordinary genome, it has a unique decomposition D_Δ whose string representation is $\{(a^1\ b^1\ c^1),(a^2\ b^2),(b^3\ -c^2)\}$

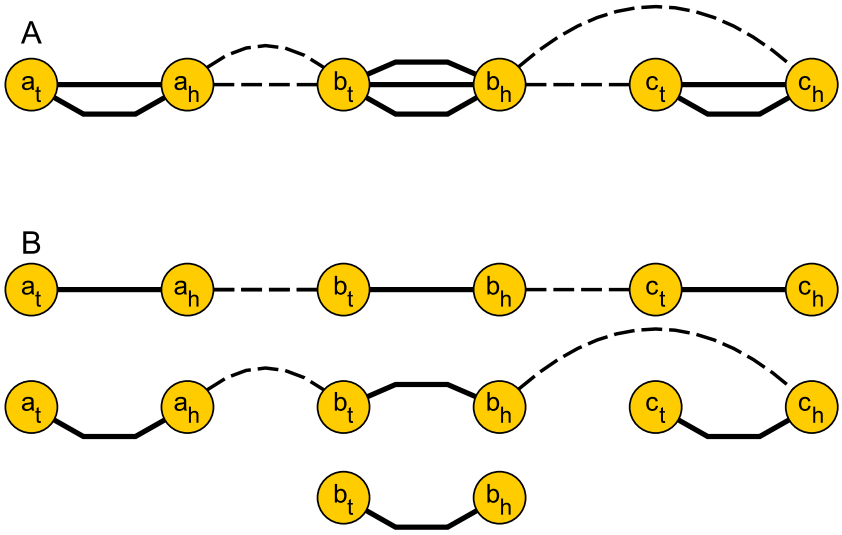

Fig. 6 **A**: A genome graph G_Γ of an unlabeled genome $\Gamma = \{\{a_t\},\{a_h,b_t\},\{b_h,c_t\},\{c_h\},\{a_t\},\{a_h,b_t\},\{b_h\},\{b_t\},\{b_h,c_h\},\{c_t\}\}$. Bold edges correspond to interval edges; dashed edges correspond to adjacencies. **B**: One possible decomposition D_Γ^1 of Γ whose string representation is $\{(a\ b\ c),(a\ b\ -c),(b)\}$. A different decomposition D_Γ^2 corresponding to $\{(a\,b\,c),(a\,b),(b\,-c)\}$ can be seen in Figure 5 by suppressing the superscripts.

A *genome rearrangement model* is composed of a set of allowed operations $\mathscr{O}$ and additional constraints on genomes. A *sorting scenario* of length d from Π into Γ is a series of genomes $\Pi_0,\ldots,\Pi_d$ such that $\Pi_0=\Pi,\Pi_d=\Gamma$ and for each i, Π_{i+1} is a legal genome (under the model constraints) that is a result of an allowed operation on Π_i. The *sorting distance* is the length of a shortest sorting scenario from Π into Γ. We call Π the *source* genome and Γ the *target* genome. The *sorting problem* under model $\mathscr{O}$ receives as input Π and Γ, and looks for a sorting scenario of minimum length from Π to Γ. Figure 2 shows a sorting scenario of length 5 comprising one deletion followed by four inversions.

2.1.1 Operations

Reversal. An *inversion* of a string reverses the string and multiplies all elements by -1. Hence the inversion of $(2\ {-3}\ 5\ {-1})$ denoted as $-(2\ {-3}\ 5\ {-1})$, is $(1\ {-5}\ 3\ {-2})$. For a string $S=s_1\ldots s_n$, $S[i,j]$ is the substring $s_i\ldots s_j$. Let C be a chromosome string. A *reversal* $\rho(i,j)$ inverts $C[i,j]$, resulting in a new chromosome $C'=C[1,\ldots,i-1]\cdot -C[i,\ldots,j]\cdot C[j+1,\ldots,m]$, where $\cdot$ is the concatenation operator. For example, $\rho(3,5)$ of $C=(1\ 3\ 2\ 4\ 5\ 6)$ is $C'=(1\ 3\ {-5}\ {-4}\ {-2}\ 6)$. Reversals can be similarly defined on a single chromosome in the genome graph, by cutting two adjacencies and reconnecting the loose extremities such that the result is a linear chromosome. A reversal on a labeled or ordinary genome is a reversal on one of its chromosomes. Reversals for general (not ordinary) unlabeled genomes are not defined as they may have several chromosome decompositions. See Figure 1B and Figure 7A,B.

Translocation. Let $C=c_1\ldots c_m$ and $D=d_1\ldots d_n$ be two linear chromosomes in string representation of an ordinary or labeled genome. A *translocation* $tr(C,D,i,j)$ transforms C and D into two new chromosomes, either $C[1,\ldots,i]\cdot D[j+1,\ldots,n]$ and $D[1,\ldots,j]\cdot C[i+1,\ldots,m]$, or $C[1,\ldots,i]\cdot -D[1,\ldots,j]$ and $-C[i+1,\ldots,m]\cdot D[j+1,\ldots,n]$. That is, the adjacencies C_i,C_{i+1} and D_j,D_{j+1} are cut, and the four loose ends are reconnected in a new way. An equivalent definition can be made on chromosome graphs, i.e., breaking an adjacency on each chromosome and reconnecting the nodes (see Figure 1e and Figure 8). Again notice that translocations are not uniquely defined for general unlabeled genomes.

DCJ. A *double-cut-and-join* (*DCJ*) is an operation that cuts two adjacencies and reconnects the four loose ends in a new way into two adjacencies. It can be applied on labeled and unlabeled genomes. A DCJ can take one of the following forms:

1. If adjacencies $\{p,q\},\{r,s\}\in\Pi$ are cut, replace them with either $\{p,r\},\{q,s\}$ or $\{p,s\},\{r,q\}$ (Figures 7, 8).
2. If adjacency $\{p,q\}\in\Pi$ is cut and telomere $\{r\}\in\Pi$ is involved, replace them with either $\{p,r\},\{q\}$ or $\{r,q\},\{p\}$ (Figure 9).
3. If telomeres $\{q\},\{r\}\in\Pi$ are involved, replace them with an adjacency $\{r,q\}$ thereby joining the two chromosomes (Figure 10). This operation is referred as a *fusion* or a *join*.
4. If adjacency $\{p,q\}\in\Pi$ is cut and an empty adjacency is involved, replace them with two telomeres $\{p\},\{q\}$ (Figure 10). Hence, a linear chromosome containing

the adjacency is cut into two chromosomes, or becomes linear if it was circular. This operation is referred as a *fission* or a *cut*.

Note that a DCJ realizes both reversals (when the two adjacencies come from the same chromosome) and translocations (when they are from different chromosomes). When the adjacencies that are cut are from the same chromosome the result of a DCJ can also be splicing out of a segment between the cuts into a separate cyclic chromosome.

SCoJ. A *single-cut-or-join* (*SCoJ*) operation either cuts an adjacency or joins two telomeres, respectively (Figure 10).

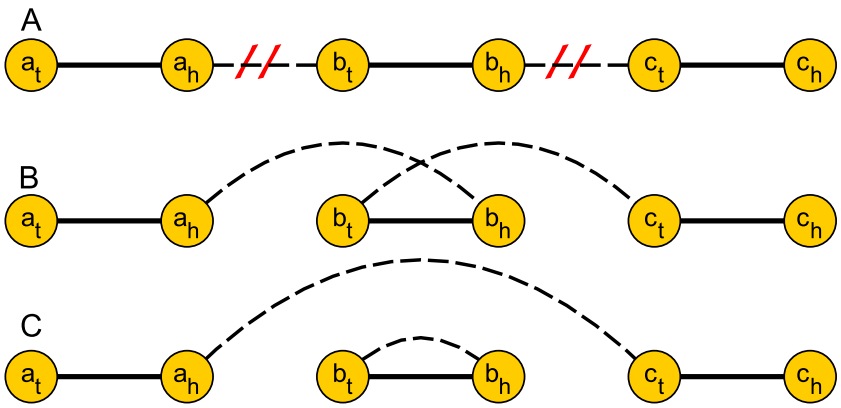

Fig. 7 Reversal and DCJ. **A**: The genome graph of $(a\ b\ c)$; the two diagonal stripes correspond to the cut adjacencies. **B**: The genome $(a\ -b\ c)$ is a result of a reversal or a DCJ. **C**. The genome $\{(a\ c), < b >\}$ corresponds to the other DCJ option.

In the next subsection we briefly review basic results on ordinary genome models. As our focus is primarily on multiple-copy problems, we only skim selected results. For much more on these problems see [126] and [42].

2.2 Breakpoint distance

The *breakpoint* (*BP*) *distance* is a simple measure of dissimilarity between two genomes that is not related to a specific type of operation. Generally speaking, the breakpoint distance measures the number of adjacencies and telomeres that are in one genome but not in the other. The breakpoint distance has several definitions depending on the different weights of common adjacencies and telomeres [80, 103].

For two ordinary genomes Π and Γ over the same n genes, Tannier *et al*. [103] give the following formula for the breakpoint distance:

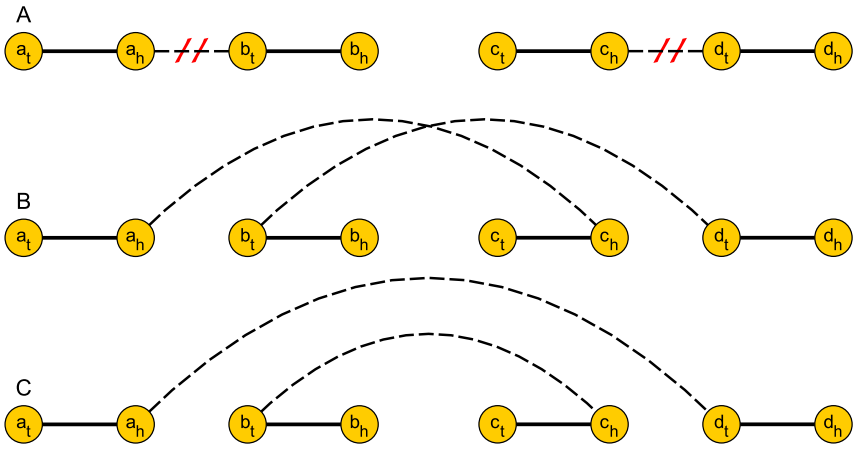

Fig. 8 Translocation. **A**: Two chromosomes $\{(a\,b),(c\,d)\}$; the two diagonal stripes correspond to the cut adjacencies. **B**,**C**: Two possible translocations (or DCJs) corresponding to $\{(a\,{-c}),({-b}\,d)\}$ (B) and $\{(a\,d),(c\,b)\}$ (C).

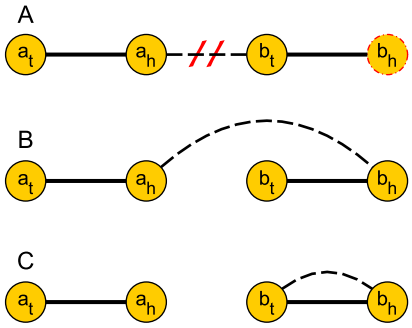

Fig. 9 DCJs on telomeres. **A**: Chromosome $(a\,b)$; the diagonal stripes and the dotted circle show the cut adjacency and telomere involved. **B**,**C**: Two possible DCJs corresponding to $(a\,{-b})$ (B), i.e. reversal, and $\{(a),<b>\}$ (C).

$$d_{BP} = n - (A + E/2) \tag{1}$$

where A is the number of common adjacencies and E the number of common telomeres of Π and Γ. Clearly, the distance is computable in linear time.

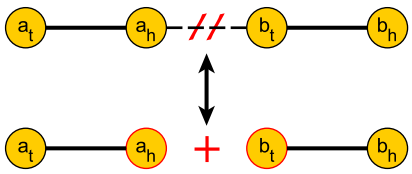

Fig. 10 Single cut or join. A cut breaks an adjacency into two telomeres corresponding to the transition from the top to the bottom genome. A join is the reverse operation corresponding to the transition from the bottom to the top genome.

2.3 Reversal and Translocation distances

Given signed permutations Π and Γ over the same n genes, we seek a shortest sequence of reversals from Π into Γ. We can assume w.l.o.g. that Γ is the identity permutation $(1 \dots n)$.

Sorting signed permutations by reversals is undoubtedly the most famous GR problem [9]. In their seminal work, Hannenhalli and Pevzner gave the first polynomial time algorithm for the problem [51]. Since then, the theory was greatly simplified [17, 59, 6, 13]. Bader, Moret and Yan have shown that finding the reversal distance can be done in linear time [6], whereas computing a shortest sorting scenario can be done in $O(n^{3/2})$ [46, 102]. Interestingly, sorting *unsigned* permutations (i.e., without gene orientations) by reversals is NP-hard [25].

The problem of sorting multi-chromosomal genomes by translocations was first introduced by Kececioglu and Ravi [61]. Hannenhalli [48] gave the first polynomial time algorithm for the problem and an improved, linear time algorithm was introduced by Bergeron *et al*. [15].

Sorting by reversals and translocations was proved to be polynomial by Hannenhalli and Pevzner [50], who reduced the problem to sorting by reversals. The theory and algorithm were later slightly corrected and revised [105, 74, 75, 55, 16]. The algorithm was used to compute for the first time a sorting scenario and distance between the mouse and human genome [50]. Interestingly, the distance achieved closely matched a prediction by Nadeau and Taylor from the 1980s [70]. Efficient implementations of the algorithms for sorting by reversals and translocations are available as part of the *GRAPPA* [6] and *GRIMM* [106] tools. Those tools also use the ability to compute exact pairwise distances efficiently in order to compute a tree of evolution by reversals and translocations among multiple species, albeit heuristically.

The main representation used for the analysis of this problem (and other rearrangement models) is the Breakpoint Graph (*BG*). Given two genomes Π and Γ, the *breakpoint graph* $BG(\Pi,\Gamma)$ is an undirected graph whose nodes are the extremities of both genomes, and whose edges are the adjacencies of both genomes

distinguished by color. Edges corresponding to Π (Γ) adjacencies are called red or Π-edges (blue or Γ-edges, respectively). See an example in Figure 11.

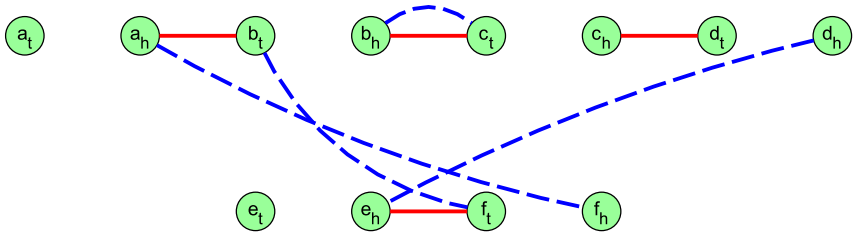

Fig. 11 A breakpoint graph for $\Pi = \{(a\ b\ c\ d), (e\ f)\}$ and $\Gamma = \{(a\ -f\ b\ c), (d\ -e)\}$. Π-edges are solid; Γ-edges are dashed;

Hannenhalli and Pevzner [51] gave a formula for the reversal distance between signed permutations based on the number of cycles in the BG and certain structures in it called "hurdles" and "fortresses". The distance formula for sorting by reversals and translocations has been devised over the years and depends on more complex structures in the BG [50, 105, 74, 55, 16]. The definitions of these structures are beyond the scope of this review, so the exact distance formulas are omitted. Bergeron [13], and Jean and Nikolski [55], give fairly elementary presentations for sorting by reversals and sorting by reversals and translocations, respectively, including good expositions of the distance formulas.

2.4 DCJ distance

The input for this model are two ordinary genomes Π and Γ over the same set of n genes. The operations allowed in this model are DCJs.

The DCJ operation, introduced by Yancopoulos *et al.* [115], has gained much attention in GR models in the last decade. The reason is that DCJs capture both reversals and translocations (but also splicing out a circular sub-chromosome) while allowing much simpler algorithms. Both the distance and an optimal sorting scenario can be computed in linear time [14].

In the analysis of this problem a new graph representation was introduced. The *adjacency graph* $AG(\Pi,\Gamma)$ of genomes Π and Γ is a bipartite undirected multigraph whose set of nodes are the adjacencies and telomeres of Π and Γ. Nodes belonging to Π (Γ) are called red- or Π-nodes (blue- or Γ-nodes, respectively). For every Π-node u and Γ-node v, there are $|u \cap v|$ edges between u and v, i.e., there is an edge for each common extremity between the two nodes. Note that $BG(\Pi,\Gamma)$ is the line graph of $AG(\Pi,\Gamma)$ and vice versa. See Figure 12.

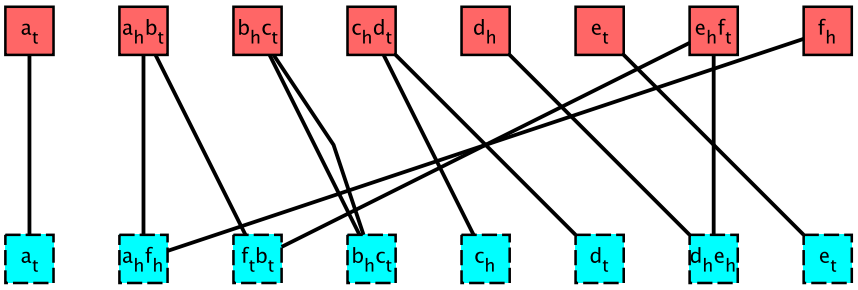

Fig. 12 An adjacency graph for $\Pi = \{(a\ b\ c\ d), (e\ f)\}$ and $\Gamma = \{(a\ -f\ b\ c), (d\ -e)\}$. Π-nodes are solid; Γ-nodes are dashed;

Bergeron *et al.* [14] prove that for ordinary genomes Π and Γ defined over the same set of n genes:

$$d_{DCJ} = n - (C + I/2) \tag{2}$$

where C is the number of cycles and I the number of odd length paths starting and ending in telomeres in $AG(\Pi, \Gamma)$. For example, the AG in Figure 12 has one cycle and two odd paths. Thus, since there are 6 genes, the DCJ distance between the two genomes in this case is 4. Notice that there are two additional even paths in the graph but they do not affect the distance formula.

2.5 SCoJ distance

The input for this model are two ordinary genomes Π and Γ over the same set of n genes. The operations allowed in this model are SCoJs.

Similar to DCJ, the SCoJ distance and scenario can be found in linear time [41]. Some rearrangements problems for which no polynomial solution is known for DCJ and other operations, are known to be tractable for SCoJ distance. We give examples of such problems in section 3.1.

For two ordinary genomes Π and Γ over the same n genes, let A_Π (A_Γ) be the set of adjacencies of Π (Γ, respectively). The SCoJ distance is given by [41]:

$$d_{SCoJ} = |A_\Pi| + |A_\Gamma| - 2|A_\Pi \cap A_\Gamma| \tag{3}$$

3 Multi copy models in evolution

This sections discusses multi-copy GR models inspired by species evolution. In subsection 3.1 we present models allowing whole genome duplication events, but no other copy number changes. The models in subsection 3.2 allow for the insertion

and deletion of new genomic segments but do not account for multiple copies of segments. Models in subsection 3.3 handle genomes with multiple copies of each gene but do not allow numeric operations. Subsection 3.4 describes a few models that can handle genomes with multiple gene copies and allow numerical operations such as deletions or duplications.

We limit our discussion here to distance problems between two genomes. We refer the reader to the review by El-Mabrouk and Sankoff on the analysis of gene order evolution beyond single-copy genes [39], which gives more focus on the phylogenetic aspects of GR models in the context of evolution.

3.1 Polyploidy

We discuss here problems motivated by *whole genome duplication* (*WGD*) events in evolution. WGD is viewed as a fundamental step in evolution, as doubling of the gene contents allows great diversification of gene functions. For example, strong evidence for WGD events was reported for yeast [114] and for plant genomes [20]. The basic question tackled by these formulations is the sorting scenario between the ancestral genome (before or right after WGD) and the extant genome.

A *duplicated genome* (either labeled or unlabeled) is a genome in which every gene has CN=2. For an ordinary genome Π over $\mathscr{G}$, a *doubled genome* $2\Pi = \Pi \cup \Pi$ is an unlabeled duplicated genome over $2\mathscr{G} = \mathscr{G} \cup \mathscr{G}$ in which every gene, adjacency and telomere has two copies. For example, if $\Pi = \{\{a_t\},\{a_h,b_h\},\{b_h\}\}$ then $2\Pi = \{\{a_t\},\{a_h,b_h\},\{b_h\},\{a_t\},\{a_h,b_h\},\{b_h\}\}$.

The *double distance problem* [2] is defined as follows. Given an ordinary genome Π over $\mathscr{G}$, a labeled duplicated genome Θ and an operation distance measure d, find the minimum distance of Θ to some labeling of 2Π. Formally, the *double distance* between Π and Θ is:

$$dd(\Pi,\Theta) = \min_{\Gamma \in L(2\Pi)} d(\Gamma,\Theta) \tag{4}$$

where $L(2\Pi)$ is the set of all possible labelings of 2Π.

The double distance problem can be solved in linear time for the BP [62] and the SCoJ measures [41]. However, it is NP-hard under the DCJ distance [103].

Given a labeled duplicated genome Θ and an operation distance measure d, the *genome halving problem* seeks to find an ordinary genome Π that minimizes the double distance to Θ [38]. Formally, the *halving distance* of Θ is defined as:

$$hd(\Theta) = \min_{\Pi} dd(\Pi,\Theta) \tag{5}$$

The halving distance can be solved in linear time for the BP measure, but if we restrict the genome Π to be linear or unichromsomal it becomes NP-hard [62]. For the SCoJ distance, the problem is solvable in linear time even when Π is restricted to be a linear or circular genome [41]. Under the DCJ distance the halving problem

can be solved in linear time [68, 111] even with Π restricted to a unichromsomal genome [1].

A generalization of the halving problem for finding an ordinary pre-WGD genome given an extant genome with exactly $m > 2$ copies is called *genome aliquoting* [111]. Aliquoting is polynomially solvable for the BP [112] and SCoJ [41] distances, while a 2-approximation algorithm is known for the problem under the DCJ distance [112]. Recently, efficient ILP formulations were suggested for genome halving and aliquoting under the DCJ distance [5].

The *guided genome halving problem* tries to combine both the genome halving and double distance [125]. Given an ordinary genome Δ, a labeled duplicated genome Θ, and an operation distance measure d, find an ordinary genome Π that minimizes the sum of the double distance between Π and Θ, plus the distance between Π and Δ. Formally, the guided halving distance is:

$$ghd(\Delta,\Theta) = \min_{\Pi}\left[dd(\Pi,\Theta) + d(\Pi,\Delta)\right] \tag{6}$$

The problem can be solved in $O(n^{1.5})$ time for the BP distance, but it becomes NP-hard with additional restrictions [62]. For the SCoJ distance, the problem has linear solutions even with restrictions to linear or circular genomes [41]. It is NP-hard for the DCJ distance [103].

3.2 Single copy models with indels

Models presented in this subsection allow new numerical operations while maintaining the assumptions of ordinary genomes. The input is two ordinary genomes Π and Γ over potentially different gene sets $\mathscr{G}_1$ and $\mathscr{G}_2$. The goal is to transform Π into Γ with structural operations and additional operations that introduce new genes or remove genes. All genomes in the sorting scenario must be ordinary.

Given a chromosome string $C = c_1 \dots c_n$, a *deletion* $del(i,j)$ produces a new chromosome $C[1,i-1]\cdot C[j+1,n]$. An *insertion* $ins(S,i)$ of a sequence $S = s_1 \dots s_m$ into a chromosome C at position i results in $C[1,i]\cdot S \cdot C[i+1,n]$ (see Figure 1). Insertions and deletions are commonly referred to as *indel* operations [23]. Since these models assume that all genomes are ordinary, insertions cannot introduce new copies of genes. Instead, indels are used to add and remove genes that appear in one genome but not in the other.

El-Mabrouk was the first to address sorting permutations by reversals and indels and gave exact an algorithm and a heuristic for specific cases [37]. Improved bound for this problem were later devised [113]. Yancopoulos and Friedberg [116] analyzed the problem of sorting ordinary genomes with DCJs and indels. Their model allowed to insert and delete any genes that are common to both the source and target genomes, and thus a possible sorting scenario can delete all the chromosomes of the source genome and insert the chromosomes of the target genome. Braga *et al.* [23] gave a linear time algorithm for finding a minimum sorting scenario with DCJs and

indels, restricting indels to affect genes that are not common to the source and target genomes. The problem is solvable in linear time even when DCJs and indels have different weights [30].

Braga *et al.* [22] introduced a new operation that generalizes both insertions and deletions. A *substitution* is an operation that replaces a sequence of consecutive genes with a another sequence. This operation can be thought of as a deletion of the sequence to be replaced followed by an insertion of the new sequence in the same place. Notice that this operation can implement both deletions and insertions by taking an empty sequence as the new or old sequence, respectively. Sorting ordinary genomes with DCJs and substitutions can be solved in linear time [22], even when substitutions have different weight than DCJs [31].

3.3 Multi-copy models without duplications/deletions

In this subsection, we focus on comparing genomes with multiple gene copies but without explicit deletion or duplication operations. The comparison can be used to assign orthology relationship between gene copies in the source and target genomes [26]. The general approach used is matching gene copies in the source and target genomes in order to minimize some structural operation distance. Gene copies that are not matched are ignored, so they are implicitly deleted and do not incur the cost of a true deletion operation. Most formulations result in NP-hard problems.

There are three main formulation strategies depending on the cardinality of the matching of multi-copy genes:

- *Exemplar* strategy [88], in which in each genome, exactly one copy of each gene is selected and all other copies are ignored.
- *Intermediate* strategy [4], in which the same number of copies (at least one) for each gene are selected and matched between genomes, and all other copies are ignored.
- *Maximum matching* strategy [19], in which for each gene, the maximum possible gene copies (the smaller of the gene's CNs in the two genomes) are selected and matched between genomes, and the remaining copies are ignored.

Although most formulations are NP-hard, several exhaustive and heuristic algorithms have been suggested. In recent years, Integer Linear Programming (ILP) formulations were used to solve such problems, and have shown good results and scalability [94, 95, 96]. Table 1 summarizes selected results for different operations and different formulations.

The majority of hardness results, as well as exact and heuristic algorithms for these problems, originate from the *breakpoint graph decomposition* problem [60, 25]. The goal in this problem is to find a decomposition of a breakpoint graph into a maximum number of edge-disjoint alternating red/blue cycles. A similar maximum cycle decomposition can also be defined for the adjacency graph [93, 94]. Such

Table 1 Multi-copy model results

Operations	Exemplar	Intermediate	Matching
BP	NP-hard [24] Branch and bound [88, 71] ILP [3, 95]	NP-hard [18] ILP [4, 96] Heuristics [4]	NP-hard [18] Branch and bound [19] ILP [4, 96] Heuristics [4]
Reversals and translocations	NP-hard [24]	NP-hard [26] ILP [101] Heuristics [26, 43]	NP-hard [26] ILP [101] Heuristics [26, 43]
DCJ	Branch and bound [117]	NP-hard [94]	NP-hard [94] ILP [94] Branch and bound [117] Approximation [93, 85]

decomposition induces a matching between genes and the maximum number of cycles minimizes an operation distance measure [93, 94].

3.4 Models with duplications or deletions

We now describe several models that include deletions or duplications as explicit numerical operations. Unlike the classical structural operations, numerical operations such as deletions and duplications have no standard definitions.

Chen *et al*. [26] analyzed a model for sorting unlabeled genomes with multiple gene copies using only reversals. Their heuristic, called *SOAR*, was the first method to assign orthology relationship between genes based on not only sequence similarity but also GRs. In a follow-up paper [43], the authors studied a model that allows reversals and single gene duplications. The latter can insert new gene copies at arbitrary positions in the genome. They developed a heuristic called *MSOAR* for matching gene copies between the two input genomes such that the number of reversals plus gene duplications would be minimal. While SOAR requires every gene to have equal number of copies in the two input genomes, MSOAR alleviates this assumption. In *MSOAR 2.0* [98], only tandem single gene duplications are allowed, and again, an efficient heuristic for this sorting problem is given.

Kahn and Raphael [58] introduced a measure called the *string duplication distance* that models building a target string by repeatedly copying substrings of a fixed source string. The *string duplication* operation, $\delta_{s,t,p}(X)$, copies a substring $x_s \dots x_t$ of string X and pastes it into another string Z at position p. Given a source string X without duplicate genes and a target string Y the goal is to find a minimum length sequence of string duplications needed to build the string Y. The authors described a polynomial dynamic programming algorithm for computing the distance [58]. In a follow-up work, they enhanced the model to allow substring deletions and inversions. A polynomial dynamic programming algorithm is given for computing the

sorting problem [57]. The string duplication model was used for the analysis of duplication blocks in the human genome [56].

A model introduced by Bader [7] allows tandem duplications, segmental deletions and DCJs. Given a labeled chromosome C in string representation, a *tandem duplication* $td(i,j)$ inserts a new copy of the segment $C[i,\ldots,j]$ after the j'th position, i.e., the new chromosome is $C' = C[1,\ldots,i-1]\cdot C[i,\ldots,j]\cdot C[i,\ldots,j]\cdot C[j+1,\ldots,n]$ (Figure 1). A *deletion* $del(i,j)$ removes the segment C[i,...,j] and produces $C' = C[1,\ldots,i-1]\cdot C[j+1,\ldots,n]$ (Figure 1). The goal in the model is to find a minimum sorting scenario of the identity chromosome into the input multi copy labeled chromosome. The author gave a lower bound and heuristic for the problem based on the structure of the breakpoint graph.

In a model presented by Shao and Moret [97], labeled genomes are sorted using DCJs and segmental duplications. A *segmental duplication* copies a segment of labeled genes $g_1,\ldots,g_m$ of a genome Σ and inserts the new labeled copy in Σ in a spot outside the original segment. The model allows different costs for different (preidentified) duplications and unit cost for DCJs. However, the optimization problem implicitly assumes that all segmental duplications either precede or follow all DCJ event. Given two labeled genomes Π,Γ, the goal is to find segmental duplications in Π and Γ, remove them, and then find a bijection between the remaining genes such that the cost of segmental duplications plus the DCJ distance is minimized. The authors analyzed this problem and gave an ILP formulation. It is based on the adjacency graph cycle decomposition formulation proposed in [94], applied to a problem instance simplified by detection of optimal substructures.

Paten *et al*. [78] presented a model for genome evolution that does not fit entirely into the standard GR terminology. This model can represent both single base substitutions and structural/numerical rearrangements such as DCJs, deletions and duplications. They defined a data structure called *history graph*, which holds partial order information on the sequence of events. The goal is to find a full sequence of events consistent with the input history graph that minimizes the cost of substitutions and DCJs, while gene deletions and WGDs are free. The authors analyzed this problem and gave polynomially tractable bounds for the cost. In a follow-up paper, Zerbino *et al*. [124] further analyzed the history graph model and showed that the space of possible evolutionary histories can be sampled ergodically.

4 Multi-copy models in cancer

Cancer genomes are known to undergo structural and numerical changes [47]. These include inversions, chromosomal translocations, tandem duplications, segmental deletions, whole chromosome amplifications or losses and more [109]. Figure 3 shows an example of a real cancer karyotype and Figure 13 shows a hypothetical sorting scenario for cancer evolution. A large research effort has focused on detecting signatures of these events in tumor genomic data. Currently the effort uses mainly deep sequencing data [33], though traditional methods such as FISH and

aCGH are still used to assess the CN of genomic regions. Accurate reconstruction of the numerical and structural variations remains a challenge, and a myriad of computational methods has been devised for this task [33, 104]. Some evolutionary GR models such as those presented in Section 3 could also be applied to cancer genomes. Nevertheless, the complexity of tumor karyotypes and their unique characteristics necessitate development of dedicated cancer GR models.

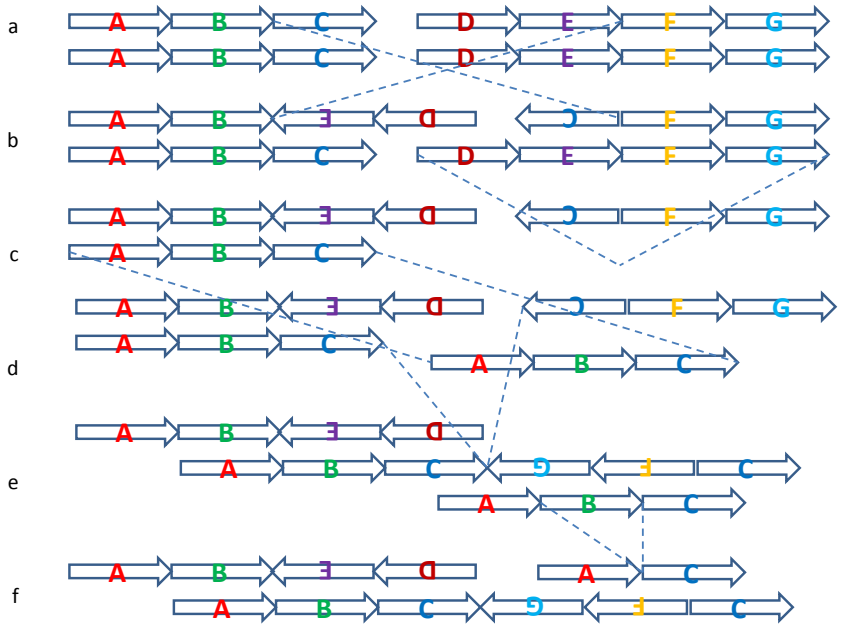

Fig. 13 A hypothetical sorting scenario for cancer evolution. a. Normal diploid karyotype with two chromosomes. a-b. Translocation. b-c. Chromosome deletion c-d. Chromosome duplication d-e. Fusion e-f. Internal deletion. f. The cancer karyotype. The breakpoints and telomeres involved in each operation are indicated by the broken lines.

In subsection 4.1 we discuss several classical GR models that were applied to cancer data. Subsection 4.2 describes CN edit distance problems in cancer. Subsection 4.3 presents a few other cancer models involving GRs.

4.1 Models with duplications/deletions

Ozery-Flato and Shamir [76] proposed a GR model designed specifically to analyze chromosomal aberrations in cancer. The input for the model are a normal unlabeled source genome with two identical copies of each chromosome and a tumor (tar-

get) genome. Both genomes are described as sets of chromosomes, each consisting of a sequence of segments. The goal is to sort the normal genome into the tumor with fewest cuts, joins, chromosome duplications and chromosome deletions. The authors proved a lower bound for the distance, and presented a polynomial-time 3-approximation algorithm for the problem. They applied the algorithm to over 50,000 low-resolution karyotypes from the Mitelman database [67], which records cancer karyotypes reported in the scientific literature. Interestingly, the approximation algorithm gave an optimal solution in all but 30 karyotypes.

Bader [8] extended his previous model [7] in order to cope with cancer alterations. The revised model accepts multi-chromosomal genomes and allows chromosome deletions and duplications, tandem duplications, segmental deletions and DCJs. A lower bound and a heuristic algorithm were devised, and applied on the Mitelman database [67]. The average calculated distance was 4.08 while the average lower bound was 2.72.

Zeira and Shamir [121] analyzed a model for genome sorting using cuts, joins and whole chromosome duplications. In this model, an ordinary linear genome is to be transformed into a duplicated linear genome such that all intermediate genomes are linear. The authors gave a linear time algorithm for the sorting problem and showed that finding such a sequence with fewest duplications is NP-hard.

A more comprehensive model presented by Zeira and Shamir [122] accounted for the evolution of unlabeled genomes via DCJs, tandem duplications, segmental deletions, and chromosomal amplifications and deletions. They showed that the sorting problem is NP-hard and gave an ILP formulation that solves the problem exactly under some mild assumptions. The algorithm was applied to sort complex ovarian cancer genomes taken from TCGA sequencing data [12]. Figure 14 gives an example of a sorting scenario inferred by the ILP.

4.2 Copy number profile distances

In this section we discuss several models for edit distance between CN profiles. These profiles give the number of copies of each segment (gene) but do not hold information about their order along the genome. A *copy number profile* (*CNP*) of a chromosome is a vector mapping each gene to a non-negative integer corresponding to the number of copies of the gene in the chromosome. As the order of the genes in a CNP is unknown, it is assumed to be some predefined order (typically the normal genome order). A *genome CNP* is a collection of its chromosome CNPs.

Let $V = (v_1, \ldots, v_n)$ where $v_i \in \mathbb{N} \cup \{0\}$ be a CNP of a chromosome with n genes. A *copy number operation* (*CNO*) is a triple $c = (\ell, h, w)$ where $1 \leq \ell \leq h \leq n$ and $w \in \{1, -1\}$. We say that the operation is a *deletion* if $w = -1$ and an *amplification* if $w = 1$. Applying an operation c on a CNP V results in a new CNP $c(V) = (c(v_1), \ldots, c(v_n))$ such that for every $\ell \leq i \leq h$, $v_i > 0$ we have $c(v_i) = v_i + w$, and otherwise $c(v_i) = v_i$. In other words, the operation increases or decreases the CN

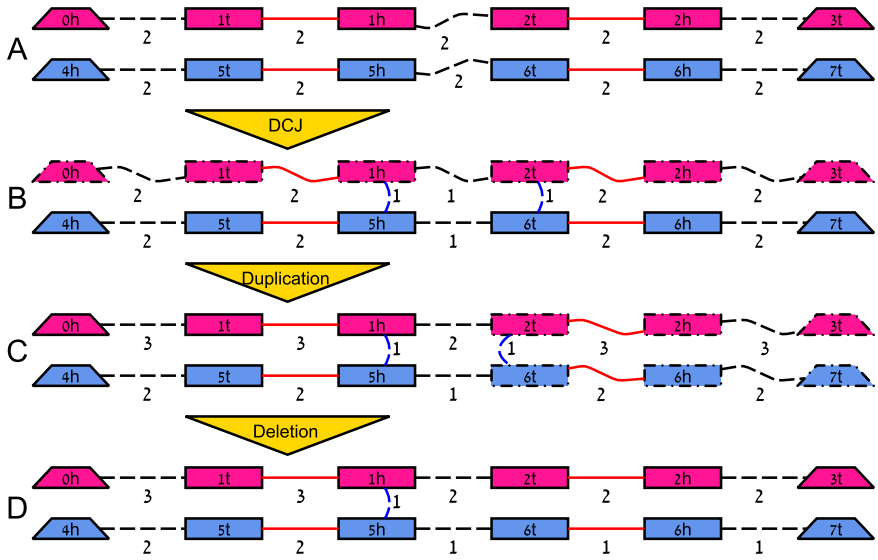

Fig. 14 Inferred GR scenario in ovarian cancer sample (TCGA-13-1411). A sequence of operations transforming a genome graph of chromosomes 1 (upper) and 3 (lower) from a diploid genome (A) to tumor genome (D). Square nodes represent segment extremities and trapezoid nodes represent telomeres. Dashed edges are adjacency edges, full straight (red) edges are interval edges and dashed arcs (purple) are novel adjacencies caused by the tumor process. The number next to each edge is its CN. One operation transforms each genome graph into the one below. The operation type is listed in the triangle and the affected genes or adjacencies appear as dashed nodes and wavy edges, respectively, in the predecessor genome. The scenario was inferred using the ILP formulation of [122].

of the genes in the interval $[\ell, h]$ if they have a positive CN, while the values of genes outside the interval and zero values are unchanged (see Figure 15).

Chowdhury *et al*. [29] defined edit distance between CNPs obtained from FISH, where the edit operations are amplification or deletion of single genes, single chromosomes, or the whole genome, and presented an algorithm for calculating the distance. The algorithm was exponential in the number of genes and therefore is limited to low-resolution FISH data. An algorithm based on the pairwise distance matrix was used to heuristically infer tumor phylogenies from FISH single cell data. A follow up paper [28] accounted for different weights for different types of operations, again providing an exponential time algorithm.

Schwartz *et al*. [92] introduced a model that admits amplifications and deletions of general contiguous segments in a chromosome CNP. The edit distance between two CNPs is the minimum number of CNOs over all possible separations of the profiles into two alleles. The authors developed an algorithm called *MEDICC* for computing the edit distance, which uses finite-state transducers [69] and is exponential in the maximum CN. MEDICC was used to infer tumor phylogenies from CGH arrays of high grade serous ovarian cancer samples [91].

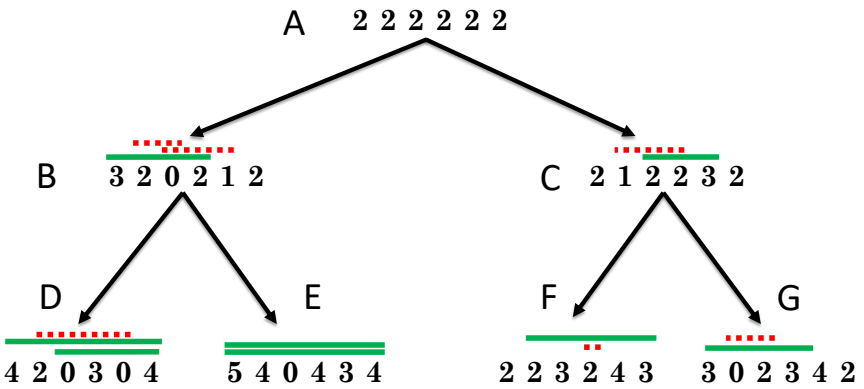

Fig. 15 Copy number profile evolution. A diploid CNP (A) evolves via CNOs into four extant CNPs (D, E, F, G). Dotted lines represent deletions and bold lines represent amplifications. The order of operations is from top to bottom. For instance, CNP A evolves into CNP B by a deletion of positions 2-3, a deletion of positions 3-5 and an amplification of positions 1-4 (in this order). The corresponding sequence of profiles is $2\,2\,2\,2\,2\,2 \rightarrow 2\,1\,1\,2\,2\,2 \rightarrow 2\,1\,0\,1\,1\,2 \rightarrow 3\,2\,0\,2\,1\,2$. The entire tree has six deletions and eight amplifications.

Zeira *et al*. [123] analyzed the problem of sorting one CNP into another using a minimum number of CNOs. They showed that this problem is solvable in linear time and constant space. Notice that this edit distance is not symmetric and in fact there may not be any sequence of CNOs from one given CNP to another since genes with zero copies cannot reappear later in the sequence. To cope with this drawback, El-Kebir *et al*. [36] analyzed a symmetric version that given two CNPs aims to find a common ancestor profile that minimizes the sum of distances to these CNPs. They gave a pseudo-polynomial dynamic programming algorithm that is linear in the profile length, and an ILP formulation.

In the more general cancer context, El-Kebir *et al*. [36] showed that it is NP-hard to build a phylogenetic tree whose leaves are the input CNPs that minimizes the total number of CNOs along edges in the tree (see Figure 15), and gave a practical ILP formulation for this problem. Extending the CNP tree model, Zaccaria *et al*. [118] considered a model in which a fractional (non integer) CNP is allowed, due to the superposition of several CNPs of different subclones. The goal in this case is to deconvolve the fractional CNPs into a weighted sum of integer CNPs such that the phylogenetic tree built over them has minimum CNOs. A heuristic algorithm was given for the problem.

4.3 Other cancer models

Reconstruction of the exact cancer chromosomes based on short paired-end deep sequencing read data remains a hard challenge. There is a plethora of methods for

detection of local rearrangement events and breakpoints [33], but only few methods try to reconstruct the entire genome.

Oesper *et al*. [73] expanded the genome graph into a structure called the *interval adjacency graph*, which represents breakpoints, discordant reads and CN information. Their method, called *PREGO*, uses the number of reads supporting each edge to resolve the CN of genomic segments and identify discordant adjacencies in the tumor genome, and maps this information to the graph. PREGO was shown to efficiently identify complex rearrangements in ovarian cancer data. Eitan and Shamir [35] expanded this model and tested it in extensive simulations and on real cancer data. Their analysis shows that perfect reconstruction of a complete karyotype based on short read data is very hard, but that by several measures, reasonably good reconstructions are obtainable.

Weaver, developed by Li *et al*. [64], is a different probabilistic graph model proposed in order to estimate both the CNs and inter-connectivity of SVs. Weaver detects and quantifies CNs and SVs specific for each allele, and was also used for predicting partial timing of SVs relative to chromosome amplifications. A recent expansion of Weaver based on ILP formulation enabled improved prediction of SV phasing and interconnectivity [83].

A probabilistic framework based on breakpoint graphs was presented by Greenman *et al*. [45] for the analysis of mutations and karyotypes from sequencing data. This work tries to reconstruct both the temporal sequence of rearrangements and assemble genomic segments into karyotypes. It uses allelic integer CNs for each segment, the adjacencies between segments and the multiplicity distribution of somatic SNVs. Taking into consideration SNVs can disambiguate some sorting scenarios, since duplicated segments carry the SNVs of the original one. The method can derive partial order of accumulating numerical and single nucleotide mutations. The framework, called *GRAFT*, was demonstrated to work well with a breast cancer sample and cancer cell lines, albeit with limitations imposed by the data quality and the genome complexities.

Epilogue

GR models and theory have developed significantly in the last couple of decades. Earlier models focused on evolution and accounted for simple genomes with a single copy of each gene. These models concentrated on different operations transforming one genome into the other. The elegant theory and algorithms underlying the elementary models served as a basis to more complex models to come. Despite its over-simplification of biology, the research of genomic sorting has been fruitful, both computationally and biologically.

Later studies started addressing more complex genome models where each segment may have two copies. These studies were motivated by whole genome duplication events, which double the gene content of a genome. Most formulated problems

were shown to be NP-hard, but heuristics based on the theory developed were utilized to derive ancestral genomes of several species.

More complex evolutionary models allow for arbitrary number of copies and numerical operations such as insertions, deletions and duplications. Models vary in their assumptions and in the operations they allow. Most of the problems are NP-hard with several heuristic and exact algorithms proposed.

Family-free genome comparison was recently proposed as an alternative multiple gene copy model [21, 66, 86, 34]. In this setting, each gene is unique but we are additionally provided with a pairwise similarity score between every pair of genes, for instance based on their sequence similarity. This generalizes the multi-copy models as one can assign similarity of 1 to copies of the same gene and 0 between all others. In one formulation of the family-free DCJ distance, the goal is to find a matching between genes such that the DCJ distance minus the weight of the matching is minimal [66]. Studies showed that this problem is NP-hard and even hard to approximate, and gave heuristics and ILP formulations [66, 86].

Cancer now provides a key motivation for development of GR models handling multiple copies. During tumor progression, the genome accumulates both structural and numerical changes, thus resulting in a complex genome with varying number of gene copies. The various models trying to represent tumor evolution differ in the type of data they rely on, types of events they allow, and other assumptions. Even determining a tumor's genome and identifying structural and numerical variations (i.e. reconstructing the tumor karyotype) remains a tough problem due to the data and genome complexity as well as tumor heterogeneity. Therefore, sorting cancer genomes remains a challenging task and better models and algorithms are needed.

Some cancer genomes were explained by complex structural and numerical events that are beyond the models discussed here. For example, a *breakage-fusion-bridge* (*BFB*) is an event in which a loss of a chromosome's end is followed by "doubling-up" and fusion of the surviving part (i.e., a chromosome (a,b) is replaced by $(a,-a)$). In a BFB cycle, this process is repeated several times. Detection of BFB cycles can be done using sequencing and CN data [120, 119]. Dramatic rearrangement events also include *chromothripsis* and *chromoplexy*, in which one or more chromosomes are shattered into many pieces and some of the pieces are assembled in random order. Identifying these events in cancer genomes from sequencing data is still a hard challenge [72]. Computational models are in need to account for such events in the analysis of cancer evolution.

Advanced sequencing technologies could help in tackling GR problems in cancer. Long read sequencing techniques such as those of Pacific Biosciences and Oxford Nanopore can link distant DNA segments providing additional information on the relative location of different copies and simplify breakpoint identification [84, 54]. The linked short reads sequencing technology of 10X Genomics was recently shown to help in identifying structural variations in cancer genomes [40]. We expect these technologies and others to play a prominent role in GR analysis in cancer in the years to come.

Single-cell sequencing technologies open new opportunities and challenges in computational cancer analysis [110]. Specifically, variations between individually

sequenced cells taken from a tumor have been used to identify its evolutionary history [53, 63]. Detection of SVs and CNAs in single-cell sequencing is still a tough challenge due to the noise and biases in the data [108, 44]. The use of single-cell SVs or CNAs for clonal reconstruction have not been addressed yet, to the best of our knowledge. Additionally, one might use the heterogeneity among cells and their abundance in order to guide the rearrangement scenario. Alternatively, given a rearrangements scenario, we can try to map cells to stages along this sequence.

Acknowledgements We thank Nimrod Rappoport for helpful comments. Study supported in part by the Naomi Prawer Kadar Foundation, the Bella Walter Memorial Fund of the Israel Cancer Association and by Len Blavatnik and the Blavatnik Family foundation. R.Z. was supported in part by a fellowship from the Edmond J. Safra Center for Bioinformatics at Tel-Aviv University.

References

1. M. A. Alekseyev and P. A. Pevzner. Colored de Bruijn graphs and the genome halving problem. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 4(1):98–107, 2007.
2. M. A. Alekseyev and P. A. Pevzner. Whole genome duplications and contracted breakpoint graphs. *SIAM Journal on Computing*, 36(6):1748–1763, 2007.
3. S. Angibaud, G. Fertin, I. Rusu, A. Thévenin, and S. Vialette. A pseudo-boolean programming approach for computing the breakpoint distance between two genomes with duplicate genes. In *Proc. RECOMB Comparative Genomics*, pages 16–29. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.
4. S. Angibaud, G. Fertin, I. Rusu, A. Thévenin, and S. Vialette. Efficient tools for computing the number of breakpoints and the number of adjacencies between two genomes with duplicate genes. *Journal of Computational Biology*, 15(8):1093–115, 2008.
5. P. Avdeyev, N. Alexeev, Y. Rong, and M. A. Alekseyev. A unified ILP framework for genome median, halving, and aliquoting problems under DCJ. In J. Meidanis and L. Nakhleh, editors, *Proc. RECOMB Comparative Genomics*, pages 156–178, Cham, 2017. Springer International Publishing.
6. D. A. Bader, B. M. Moret, and M. Yan. A linear-time algorithm for computing inversion distance between signed permutations with an experimental study. *Journal of Computational Biology*, 8(5):483–491, 2001.
7. M. Bader. Sorting by reversals, block interchanges, tandem duplications, and deletions. *BMC Bioinformatics*, 10 Suppl 1:S9, 2009.
8. M. Bader. Genome rearrangements with duplications. *BMC Bioinformatics*, 11 Suppl 1:S27, 2010.
9. V. Bafna and P. Pevzner. Genome rearrangements and sorting by reversals. *SIAM Journal on Computing*, 25(2):272–289, 1996.
10. E. Barillot, L. Calzone, P. Hupé, J.-P. Vert, and A. Zinovyev. *Computational systems biology of cancer*. CRC Press, 2012.
11. N. Beerenwinkel, C. D. Greenman, and J. Lagergren. Computational cancer biology: An evolutionary perspective. *PLOS Computational Biology*, 12(2):e1004717, 2016.
12. D. Bell, A. Berchuck, M. Birrer, et al. Integrated genomic analyses of ovarian carcinoma. *Nature*, 474(7353):609–15, 2011.
13. A. Bergeron. A very elementary presentation of the Hannenhalli-Pevzner theory. *Discrete Applied Mathematics*, 146(2):134–145, 2005.

14. A. Bergeron, J. Mixtacki, and J. Stoye. A unifying view of genome rearrangements. In P. Bücher and B. M. Moret, editors, *Proc. Workshop on Algorithms in Bioinformatics*, volume 4175 of *LNCS*, pages 163–173. Springer, 2006.
15. A. Bergeron, J. Mixtacki, and J. Stoye. On sorting by translocations. *Journal of Computational Biology*, 13(2):567–578, 2006.
16. A. Bergeron, J. Mixtacki, and J. Stoye. A new linear time algorithm to compute the genomic distance via the double cut and join distance. *Theoretical Computer Science*, 410(51):5300–5316, 2009.
17. P. Berman and S. Hannenhalli. Fast sorting by reversal. In *Proc. Combinatorial Pattern Matching*, pages 168–185. Springer, Berlin, Heidelberg, 1996.
18. G. Blin, C. Chauve, G. Fertin, R. Rizzi, and S. Vialette. Comparing genomes with duplications: A computational complexity point of view. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 4(4):523–534, 2007.
19. G. Blin, G. Fertin, and C. Chauve. The breakpoint distance for signed sequences. In *Proc. 1st Conference on Algorithms and Computational Methods for biochemical and Evolutionary Networks*, volume 3, pages 3–16. King's College London publications, 2004.
20. J. E. Bowers, B. A. Chapman, J. Rong, and A. H. Paterson. Unravelling angiosperm genome evolution by phylogenetic analysis of chromosomal duplication events. *Nature*, 422(6930):433, 2003.
21. M. D. V. Braga, C. Chauve, D. Doerr, et al. The potential of family-free genome comparison. In *Models and Algorithms for Genome Evolution*, pages 287–307. Springer, 2013.
22. M. D. V. Braga, R. Machado, L. C. Ribeiro, and J. Stoye. Genomic distance under gene substitutions. *BMC Bioinformatics*, 12 Suppl 9(Suppl 9):S8, 2011.
23. M. D. V. Braga, E. Willing, and J. Stoye. Double cut and join with insertions and deletions. *Journal of Computational Biology*, 18(9):1167–84, 2011.
24. D. Bryant. The complexity of calculating exemplar distances. In D. Sankoff and J. H. Nadeau, editors, *Comparative Genomics: Empirical and Analytical Approaches to Gene Order Dynamics, Map Alignment and the Evolution of Gene Families*, pages 207–211. Springer Netherlands, Dordrecht, 2000.
25. A. Caprara. Sorting by reversals is difficult. In *Proc. First annual international conference on Research in Computational Molecular Biology*, pages 75–83, New York, New York, USA, 1997.
26. X. Chen, J. Zheng, Z. Fu, et al. Assignment of orthologous genes via genome rearrangement. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2(4):302–15, 2005.
27. A. T. Chinwalla, L. L. Cook, K. D. Delehaunty, et al. Initial sequencing and comparative analysis of the mouse genome. *Nature*, 420(6915):520–562, 2002.
28. S. A. Chowdhury, E. M. Gertz, D. Wangsa, et al. Inferring models of multiscale copy number evolution for single-tumor phylogenetics. *Bioinformatics*, 31(12):i258–67, 2015.
29. S. A. Chowdhury, S. E. Shackney, K. Heselmeyer-Haddad, et al. Algorithms to model single gene, single chromosome, and whole genome copy number changes jointly in tumor phylogenetics. *PLoS Computational Biology*, 10(7):e1003740, 2014.
30. P. H. da Silva, R. Machado, S. Dantas, and M. D. V. Braga. Restricted DCJ-indel model: sorting linear genomes with DCJ and indels. *BMC Bioinformatics*, 13 Suppl 1(Suppl 19):S14, 2012.
31. P. H. da Silva, R. Machado, S. Dantas, and M. D. V. Braga. DCJ-indel and DCJ-substitution distances with distinct operation costs. *Algorithms for Molecular Biology*, 8(1):21, 2013.
32. L. Ding, T. J. Ley, D. E. Larson, et al. Clonal evolution in relapsed acute myeloid leukaemia revealed by whole-genome sequencing. *Nature*, 481(7382):506–10, 2012.
33. L. Ding, M. C. Wendl, J. F. McMichael, and B. J. Raphael. Expanding the computational toolbox for mining cancer genomes. *Nature Reviews Genetics*, 15(8):556–570, 2014.
34. D. Doerr, P. Feijão, and J. Stoye. Family-free genome comparison. In J. C. Setubal, J. Stoye, and P. F. Stadler, editors, *Comparative Genomics: Methods and Protocols*, pages 331–342. Springer New York, New York, NY, 2018.
35. R. Eitan and R. Shamir. Reconstructing cancer karyotypes from short read data: the half empty and half full glass. *BMC Bioinformatics*, 18(1):488, 2017.

36. M. El-Kebir, B. J. Raphael, R. Shamir, et al. Complexity and algorithms for copy-number evolution problems. *Algorithms for Molecular Biology*, 12(1):13, 2017.
37. N. El-mabrouk. Sorting signed permutations by reversals and insertions/deletions of contiguous segments. *Journal of Discrete Algorithms*, 1:105–122, 2001.
38. N. El-Mabrouk, J. H. Nadeau, and D. Sankoff. Genome halving. In M. Farach-Colton, editor, *Proc. Combinatorial Pattern Matching*, pages 235–250, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.
39. N. El-Mabrouk and D. Sankoff. Analysis of gene order evolution beyond single-copy genes. *Methods in Molecular Biology*, 855:397–429, 2012.
40. R. Elyanow, H.-T. Wu, and B. J. Raphael. Identifying structural variants using linked-read sequencing data. *Bioinformatics*, 34(2):353–360, 2018.
41. P. Feijão and J. Meidanis. SCJ: a breakpoint-like distance that simplifies several rearrangement problems. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8(5):1318–29, 2011.
42. G. Fertin, A. Labarre, I. Rusu, E. Tannier, and S. Vialette. *Combinatorics of Genome Rearrangements*. MIT Press, 2009.
43. Z. Fu, X. Chen, V. Vacic, et al. MSOAR: A high-throughput ortholog assignment system based on genome rearrangement. *Journal of Computational Biology*, 14(9):1160–1175, 2007.
44. T. Garvin, R. Aboukhalil, J. Kendall, et al. Interactive analysis and assessment of single-cell copy-number variations. *Nature Methods*, 12:1058 EP –, 2015.
45. C. D. Greenman, E. D. Pleasance, S. Newman, et al. Estimation of rearrangement phylogeny for cancer genomes. *Genome Research*, 22(2):346–61, 2012.
46. Y. Han. Improving the efficiency of sorting by reversals. In *Proc. 2006 International Conference on Bioinformatics and Computational Biology*, volume 6, pages 406–409. Citeseer, 2006.
47. D. Hanahan and R. A. Weinberg. Hallmarks of cancer: the next generation. *Cell*, 144(5):646–74, 2011.
48. S. Hannenhalli. Polynomial-time algorithm for computing translocation distance between genomes. *Discrete Applied Mathematics*, 71(1):137 – 151, 1996.
49. S. Hannenhalli and P. A. Pevzner. Transforming cabbage into turnip. In *Proc. Annual ACM Symposium on the Theory of Computing*, volume 46, pages 178–189, New York, New York, USA, 1995.
50. S. Hannenhalli and P. A. Pevzner. Transforming men into mice (polynomial algorithm for genomic distance problem). In *Proc. IEEE Symposium on Foundations of Computer Science*, volume 36, pages 581–592, 1995.
51. S. Hannenhalli and P. A. Pevzner. Transforming cabbage into turnip: polynomial algorithm for sorting signed permutations by reversals. *Journal of the ACM*, 46(1):1–27, 1999.
52. P. Hupé. Karyotype of the t47d breast cancer cell line. Wikimedia Commons file.
53. K. Jahn, J. Kuipers, and N. Beerenwinkel. Tree inference for single-cell data. *Genome Biology*, 17(1):86, 2016.
54. M. Jain, S. Koren, K. H. Miga, et al. Nanopore sequencing and assembly of a human genome with ultra-long reads. *Nature Biotechnology*, 36(4):338–345, 2018.
55. G. Jean and M. Nikolski. Genome rearrangements: a correct algorithm for optimal capping. *Information Processing Letters*, 104(1):14 – 20, 2007.
56. C. L. Kahn, B. H. Hristov, and B. J. Raphael. Parsimony and likelihood reconstruction of human segmental duplications. *Bioinformatics (Oxford, England)*, 26(18):i446–52, 2010.
57. C. L. Kahn, S. Mozes, and B. J. Raphael. Efficient algorithms for analyzing segmental duplications with deletions and inversions in genomes. *Algorithms for Molecular Biology*, 5(1):11, 2010.
58. C. L. Kahn and B. J. Raphael. Analysis of segmental duplications via duplication distance. *Bioinformatics (Oxford, England)*, 24(16):i133–8, 2008.
59. H. Kaplan, R. Shamir, and R. E. Tarjan. Faster and simpler algorithm for sorting signed permutations by reversals. *SIAM Journal on Computing*, 29(3):880, 1997.

60. J. Kececioglu and D. Sankoff. Exact and approximation algorithms for sorting by reversals, with application to genome rearrangement. *Algorithmica*, 13(1-2):180–210, 1995.
61. J. D. Kececioglu and R. Ravi. Of mice and men: Algorithms for evolutionary distances between genomes with translocation. In *Proc. Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 604–613, Philadelphia, PA, USA, 1995. Society for Industrial and Applied Mathematics.
62. J. Kováč. On the complexity of rearrangement problems under the breakpoint distance. *Journal of Computational Biology*, 21(1):1–15, 2014.
63. J. Kuipers, K. Jahn, B. J. Raphael, and N. Beerenwinkel. Single-cell sequencing data reveal widespread recurrence and loss of mutational hits in the life histories of tumors. *Genome Research*, 2017.
64. Y. Li, S. Zhou, D. C. Schwartz, and J. Ma. Allele-specific quantification of structural variations in cancer genomes. *Cell Systems*, 3(1):21 – 34, 2016.
65. H. Lodish, A. Berk, J. E. Darnell, et al. *Molecular Cell Biology*. Macmillan, 2008.
66. F. V. Martinez, P. Feijão, M. D. Braga, and J. Stoye. On the family-free DCJ distance and similarity. *Algorithms for Molecular Biology*, 10(1):13, 2015.
67. F. Mitelman, B. Johansson, and F. Mertens. Mitelman database of chromosome aberrations and gene fusions in cancer, 2018. http://cgap.nci.nih.gov/Chromosomes/Mitelman.
68. J. Mixtacki. Genome halving under DCJ revisited. In X. Hu and J. Wang, editors, *Proc. Computing and Combinatorics*, volume 5092 of *Lecture Notes in Computer Science*, pages 276–286. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.
69. M. Mohri. Edit-distance of weighted automata: General definitions and algorithms. *International Journal of Foundations of Computer Science*, 14(06):957–982, 2003.
70. J. H. Nadeau and B. A. Taylor. Lengths of chromosomal segments conserved since divergence of man and mouse. *Proceedings of the National Academy of Sciences*, 81(3):814–818, 1984.
71. C. T. Nguyen, Y. C. Tay, and L. Zhang. Divide-and-conquer approach for the exemplar breakpoint distance. *Bioinformatics*, 21(10):2171–2176, 2005.
72. L. Oesper, S. Dantas, and B. J. Raphael. Identifying simultaneous rearrangements in cancer genomes. *Bioinformatics*, 34(2):346–352, 2018.
73. L. Oesper, A. Ritz, S. J. Aerni, R. Drebin, and B. J. Raphael. Reconstructing cancer genomes from paired-end sequencing data. *BMC Bioinformatics*, 13 Suppl 6(Suppl 6):S10, 2012.
74. M. Ozery-Flato and R. Shamir. Two notes on genome rearrangement. *Journal of Bioinformatics and Computational Biology*, 1(1):71–94, 2003.
75. M. Ozery-Flato and R. Shamir. Sorting by translocations via reversals theory. In G. Bourque and N. El-Mabrouk, editors, *Proc. RECOMB Comparative Genomics*, pages 87–98, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.
76. M. Ozery-Flato and R. Shamir. Sorting cancer karyotypes by elementary operations. *Journal of Computational Biology*, 16(10):1445–60, 2009.
77. J. D. Palmer and L. A. Herbon. Plant mitochondrial DNA evolves rapidly in structure, but slowly in sequence. *Journal of Molecular Evolution*, 28(1-2):87–97, 1988.
78. B. Paten, D. R. Zerbino, G. Hickey, and D. Haussler. A unifying model of genome evolution under parsimony. *BMC Bioinformatics*, 15(1):206, 2014.
79. P. Pevzner and G. Tesler. Genome rearrangements in mammalian evolution: Lessons from human and mouse genomes. *Genome Research*, 13(1):37–45, 2003.
80. P. Pevzner and G. Tesler. Transforming men into mice. In *Proc. Seventh annual international conference on Research in Computational Molecular Biology*, pages 247–256, New York, New York, USA, 2003. ACM Press.
81. D. Pinkel, T. Straume, and J. W. Gray. Cytogenetic analysis using quantitative, high-sensitivity, fluorescence hybridization. *Proceedings of the National Academy of Sciences*, 83(9):2934–2938, 1986.
82. P. Popescu and H. Hayes. *Techniques in Animal Cytogenetics*. Springer Science & Business Media, 2000.
83. A. Rajaraman and J. Ma. Toward Recovering Allele-specific Cancer Genome Graphs. *Journal of Computational Biology*, 25(7):624–636, 2018.

84. A. Rhoads and K. F. Au. PacBio sequencing and its applications. *Genomics, Proteomics & Bioinformatics*, 13(5):278–289, 2015.
85. D. P. Rubert, P. Feijão, M. D. V. Braga, J. Stoye, and F. H. V. Martinez. Approximating the DCJ distance of balanced genomes in linear time. *Algorithms for Molecular Biology*, 12(1):3, 2017.
86. D. P. Rubert, E. A. Hoshino, M. D. V. Braga, J. Stoye, and F. V. Martinez. Computing the family-free DCJ similarity. *BMC Bioinformatics*, 19(S6):152, 2018.
87. D. Sankoff. Edit distance for genome comparison based on non-local operations. In A. Apostolico, M. Crochemore, Z. Galil, and U. Manber, editors, *Proc. Combinatorial Pattern Matching*, pages 121–135, Berlin, Heidelberg, 1992. Springer Berlin Heidelberg.
88. D. Sankoff. Genome rearrangement with gene families. *Bioinformatics*, 15(11):909–917, 1999.
89. D. Sankoff, G. Leduc, N. Antoine, et al. Gene order comparisons for phylogenetic inference: Evolution of the mitochondrial genome. *Proceedings of the National Academy of Sciences*, 89(14):6575–6579, 1992.
90. H. Scherthan, T. Cremer, U. Arnason, et al. Comparative chromosome painting discloses homologous segments in distantly related mammals. *Nature Genetics*, 6(4):342, 1994.
91. R. F. Schwarz, C. K. Y. Ng, S. L. Cooke, et al. Spatial and temporal heterogeneity in high-grade serous ovarian cancer: A phylogenetic analysis. *PLOS Medicine*, 12(2):e1001789, 2015.
92. R. F. Schwarz, A. Trinh, B. Sipos, et al. Phylogenetic quantification of intra-tumour heterogeneity. *PLoS Computational Biology*, 10(4):e1003535, 2014.
93. M. Shao and Y. Lin. Approximating the edit distance for genomes with duplicate genes under DCJ, insertion and deletion. *BMC Bioinformatics*, 13(Suppl 19):S13, 2012.
94. M. Shao, Y. Lin, and B. M. Moret. An exact algorithm to compute the double-cut-and-join distance for genomes with duplicate genes. *Journal of Computational Biology*, 22(5):425–435, 2015.
95. M. Shao and B. M. Moret. A fast and exact algorithm for the exemplar breakpoint distance. *Journal of Computational Biology*, 23(5):337–346, 2016.
96. M. Shao and B. M. Moret. On computing breakpoint distances for genomes with duplicate genes. *Journal of Computational Biology*, 24(6):571–580, 2017.
97. M. Shao and B. M. E. Moret. Comparing genomes with rearrangements and segmental duplications. *Bioinformatics*, 31(12):i329–i338, 2015.
98. G. Shi, L. Zhang, and T. Jiang. MSOAR 2.0: Incorporating tandem duplications into ortholog assignment based on genome rearrangement. *BMC Bioinformatics*, 11(1):10, 2010.
99. S. H. Strauss, J. D. Palmer, G. T. Howe, and A. H. Doerksen. Chloroplast genomes of two conifers lack a large inverted repeat and are extensively rearranged. *Proceedings of the National Academy of Sciences*, 85(11):3898–3902, 1988.
100. A. H. Sturtevant and T. Dobzhansky. Inversions in the third chromosome of wild races of drosophila pseudoobscura, and their use in the study of the history of the species. *Proceedings of the National Academy of Sciences*, 22(7):448–450, 1936.
101. J. Suksawatchon, C. Lursinsap, and M. Bodén. Computing the reversal distance between genomes in the presence of multi-gene families via binary integer programming. *Journal of Bioinformatics and Computational Biology*, 5(1):117–33, 2007.
102. E. Tannier, A. Bergeron, and M.-F. Sagot. Advances on sorting by reversals. *Discrete Applied Mathematics*, 155(6-7):881–888, 2007.
103. E. Tannier, C. Zheng, and D. Sankoff. Multichromosomal median and halving problems under different genomic distances. *BMC Bioinformatics*, 10(1):120, 2009.
104. L. Tattini, R. D'Aurizio, and A. Magi. Detection of Genomic Structural Variants from Next-Generation Sequencing Data. *Frontiers in Bioengineering and Biotechnology*, 3:92, 2015.
105. G. Tesler. Efficient algorithms for multichromosomal genome rearrangements. *J. Comput. Syst. Sci.*, 65(3):587–609, 2002.
106. G. Tesler. GRIMM: genome rearrangements web server. *Bioinformatics*, 18(3):492–493, 2002.

107. A. E. Urban, J. O. Korbel, R. Selzer, et al. High-resolution mapping of DNA copy alterations in human chromosome 22 using high-density tiling oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, 103(12):4534–9, 2006.
108. T. Voet, P. Kumar, P. Van Loo, et al. Single-cell paired-end genome sequencing reveals structural variation per cell cycle. *Nucleic Acids Research*, 41(12):6119–6138, 2013.
109. B. Vogelstein, N. Papadopoulos, V. E. Velculescu, et al. Cancer genome landscapes. *Science*, 339(6127):1546–58, 2013.
110. Y. Wang, J. Waters, M. L. Leung, et al. Clonal evolution in breast cancer revealed by single nucleus genome sequencing. *Nature*, 512:155 EP –, 2014. Article.
111. R. Warren and D. Sankoff. Genome halving with double cut and join. *Journal of Computational Biology*, 7(2):357–371, 2009.
112. R. Warren and D. Sankoff. Genome aliquoting revisited. *Journal of Computational Biology*, 18(9):1065–1075, 2011.
113. E. Willing, S. Zaccaria, M. D. Braga, and J. Stoye. On the inversion-indel distance. *BMC Bioinformatics*, 14 Suppl 15:S3, 2013.
114. K. H. Wolfe and D. C. Shields. Molecular evidence for an ancient duplication of the entire yeast genome. *Nature*, 387:708 EP –, 1997.
115. S. Yancopoulos, O. Attie, and R. Friedberg. Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics*, 21(16):3340–3346, 2005.
116. S. Yancopoulos and R. Friedberg. DCJ path formulation for genome transformations which include insertions, deletions, and duplications. *Journal of Computational Biology*, 16(10):1311–38, 2009.
117. Z. Yin, J. Tang, S. W. Schaeffer, and D. A. Bader. Exemplar or matching: modeling DCJ problems with unequal content genome data. *Journal of Combinatorial Optimization*, 2015.
118. S. Zaccaria, M. El-Kebir, G. W. Klau, and B. J. Raphael. Phylogenetic Copy-Number Factorization of Multiple Tumor Samples. *Journal of Computational Biology*, page cmb.2017.0253, 2018.
119. S. Zakov and V. Bafna. Reconstructing Breakage Fusion Bridge Architectures Using Noisy Copy Numbers. *Journal of Computational Biology*, 22(6):577–594, 2015.
120. S. Zakov, M. Kinsella, and V. Bafna. An algorithmic approach for breakage-fusion-bridge detection in tumor genomes. *Proceedings of the National Academy of Sciences*, 110(14):5546–51, 2013.
121. R. Zeira and R. Shamir. Sorting by cuts, joins, and whole chromosome duplications. *Journal of Computational Biology*, 24(2):127–137, 2017.
122. R. Zeira and R. Shamir. Sorting cancer karyotypes using double-cut-and-joins, duplications and deletions. *Bioinformatics*, page bty381, 2018.
123. R. Zeira, M. Zehavi, and R. Shamir. A linear-time algorithm for the copy number transformation problem. *Journal of Computational Biology*, 24(12):1179–1194, 2017.
124. D. R. Zerbino, T. Ballinger, B. Paten, G. Hickey, and D. Haussler. Representing and decomposing genomic structural variants as balanced integer flows on sequence graphs. *BMC Bioinformatics*, 17(1):400, 2016.
125. C. Zheng, Q. Zhu, and D. Sankoff. Genome halving with an outgroup. *Evolutionary Bioinformatics Online*, 2:295–302, 2007.
126. Zimao Li, Lusheng Wang, and Kaizhong Zhang. Algorithmic approaches for genome rearrangement: a review. *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, 36(5):636–648, 2006.