Sorting by cuts, joins and whole chromosome duplications

Ron Zeira and Ron Shamir

Tel-Aviv University, Tel-Aviv 69978, Israel
ronzeira@post.tau.ac.il,rshamir@tau.ac.il

Abstract. Genome rearrangement problems have been extensively studied due to their importance in biology. Most studied models assumed a single copy per gene. However, in reality duplicated genes are common, most notably in cancer. Here we make a step towards handling duplicated genes by considering a model that allows the atomic operations of cut, join and whole chromosome duplication. Given two linear genomes, Γ with one copy per gene, and Δ with two copies per gene, we give a linear time algorithm for computing a shortest sequence of operations transforming Γ into Δ such that all intermediate genomes are linear. We also show that computing an optimal sequence with fewest duplications is NP-hard.

Keywords: SCJ, genome rearrangements, computational genomics

1 Introduction

Genome organization evolves over time by undergoing rearrangement operations. Finding a shortest sequence of operations (also called a sorting scenario) between two genomes is the focus of the field of *genome rearrangements*. Such problems were studied extensively over the last two decades, due to their importance in evolution [13].

The combinatorial problems in genome rearrangements depend on the allowed operations. Hannenhalli and Pevzner showed in their seminal work that finding the minimal number of inversions that transform one signed genome into another is polynomial [15]. Many other models were studied later, allowing one or several types of operations [14, 9, 7, 8, 15, 18, 11, 17].

The *double cut and join* (*DCJ*) operation [27] models reversals, transpositions, translocations, fusions, fissions and block-interchanges as variations of one basic operation. A DCJ operation cuts the genome in two places, producing four open ends, and rejoins them in two new pairs. Finding the DCJ distance between two gene permutations can be done in linear time [4]. The *single cut or join* (*SCJ*) model [12] further simplifies the model and allows polynomial solutions to some rearrangement problems that are NP-hard under most formulations. An SCJ operation either cuts a chromosome or joins two chromosome ends. This simple model gives good results in real biological applications [5].

Models of genomes that assume a single copy of each gene are too restrictive for many real biological problems. Duplications are frequent in cancer genomes, especially in oncogenic regions [3]. Most plant genomes contain large duplicated segments [6]. A major evolutionary event is *whole genome duplication*, wherein all chromosomes are duplicated [21].

In spite of their importance, models that allow duplications as rearrangement operations have not been the subject of extensive research to date. Ozery-Flato and Shamir [19] considered a model that includes certain duplications, deletions and SCJ operations. Under some simplifying assumptions, they provided a 3-approximation algorithm that performed well on cancer genomes. Bader [1, 2] provided a heuristic for sorting by DCJs, duplications and deletions. Shao *et al.* [23] studied sorting genomes using DCJs and segmental duplications and provided an algorithm to improve an initial sorting scenario. The majority of extant models for genomes with multiple gene copies result in NP-hard problems [21, 22, 24, 25].

In this paper, we present a model that allows the operations cut, join and whole chromosome duplication. We call it the *SCJD model*. Given two linear genomes, Γ with one copy per gene, and Δ with two copies per gene, we give a linear time algorithm for computing a shortest sequence of operations transforming Γ into Δ, where all intermediate genomes must be linear too. We provide a closed form formula for that sequence length. In addition, we show that there is an optimal sequence in which all duplications are consecutive.

While cuts or joins are local events, a duplication of an entire chromosome is a more "drastic" event. We show that our algorithm actually gives an optimal scenario with a maximum number of duplications. On the other hand, we prove that finding a "conservative" optimal SCJD scenario with fewest duplications is NP-hard.

The structure of this paper is as follows. We give computational background in Section 2. In Section 3 we present the SCJD model. Section 4 gives the algorithm for the SCJD sorting problem and Section 5 shows the NP-hardness result. Finally, in Section 6, we present a brief discussion and suggest future directions. Due to lack of space, some proofs were omitted.

2 Preliminaries

Genome Representation. We use the following standard terminology in genome rearrangements [4]. The basic entities are *genes*, denoted a, b, c etc. Gene a has *extremities*: a *head* a_h and a *tail* a_t. Gene a is assumed oriented from its tail to its head and is *positively oriented* if a_t is to the left of a_h. A *negatively oriented* gene a is denoted by $-a$. A *chromosome* is a sequence of oriented genes, e.g., $C = ab{-}c{-}d$. An *adjacency* in a chromosome is a consecutive pair of extremities from distinct neighboring genes. E.g., the adjacencies in C above are: $\{a_h, b_t\}$,$\{b_h, c_h\}$,$\{c_t, d_h\}$. A *telomere* is an extremity that is not adjacent to any other gene, corresponding to the end of a chromosome, e.g., $\{a_t\}$,$\{d_t\}$ in C. Hence, a chromosome can be equivalently represented by its set of adjacen-

cies, where the telomeres are implicit. Note that the set of adjacencies defining a chromosome is identical to that of the reverse chromosome, where order and orientation of genes are inverted (the reverse of C is $-C = dc{-}b{-}a$). Hence, a chromosome and its reverse are equivalent.

A *genome* over gene set $\mathcal{G}$ is a collection of chromosomes. We assume for now that each gene appears once, e.g. $\Gamma = \{ab, c{-}d\}$. Equivalently, it can be defined by a set of adjacencies such that for each gene in $\mathcal{G}$, each extremity appears at most once. Hence $\Gamma = \big\{\{a_h, b_t\}, \{c_h, d_h\}\big\}$. The *size* of a genome Π, denoted $|\Pi|$, is the number of adjacencies in it. A chromosome is called *linear* if it starts and ends with a telomere, and *circular* if it does not contain any telomere, e.g. $D = \big\{\{a_h, b_t\}, \{b_h, a_t\}\big\}$. For a sequence of genes S, denote by S and (S) the corresponding linear and circular chromosome respectively. For example, the linear chromosome $a{-}b$ is defined by the set of adjacencies $\big\{\{a_h, b_h\}\big\}$ and the circular chromosome $(a{-}b)$ is defined by the set $\big\{\{a_h, b_h\}, \{b_t, a_t\}\big\}$. A genome is called *linear* if all its chromosomes are linear.

A gene that has several copies in the genome is called *duplicated*. We *label* different copies of the same gene by superscripts, e.g., copies a^1 and a^2 of gene a. A *duplicated genome* has exactly two copies of each gene. A genome with a single copy of each gene is called *ordinary*. The duplication of an ordinary genome Π creates a special kind of genome [26]: Each gene and each adjacency in Π is doubled, producing the genome $\Pi \bigoplus \Pi$. Note that in $\Pi \bigoplus \Pi$ the two copies of each gene are unlabeled. The set of all possible labeled genomes corresponding to $\Pi \bigoplus \Pi$ is denoted by 2Π. A genome $\Sigma \in 2\Pi$ is called a *perfectly duplicated genome*. Hence for Γ above, $\Gamma \bigoplus \Gamma = \{ab, ab, c{-}d, c{-}d\}$ and $\Sigma = \big\{\{a_h^2, b_t^2\}, \{c_h^2, d_h^1\}, \{a_h^1, b_t^1\}, \{c_h^1, d_h^2\}\big\} \in 2\Gamma$.

SCJ distance. A *cut* operation takes an adjacency $\{x, y\}$ and breaks it into two telomeres $\{x\}$ and $\{y\}$. The reverse operation, called a *join*, combines two telomeres $\{x\}$ and $\{y\}$ into an adjacency $\{x, y\}$. A *single-cut-or-join* (*SCJ*) operation is either a cut or a join [12]. Given two ordinary genomes Π and Σ on the same gene set, a sequence of SCJ operations that transforms Π into Σ is called a *sorting scenario*. The *SCJ distance*, denoted by $d_{SCJ}(\Pi, \Sigma)$, is the length of a shortest sorting scenario between Π and Σ. Feijão and Meidanis give the following solution for the SCJ distance:

Theorem 1. *[12]* $d_{SCJ}(\Pi, \Sigma) = |\Pi \setminus \Sigma| + |\Sigma \setminus \Pi| = |\Pi| + |\Sigma| - 2|\Pi \cap \Sigma|$. $\Pi \setminus \Sigma$ *defines the set of cuts and* $\Sigma \setminus \Pi$ *defines the set of joins in an optimal sorting scenario.*

Double Distance. The *SCJ double distance* between an ordinary genome Γ and a duplicated genome Δ is defined as

$$dd_{SCJ}(\Gamma, \Delta) \equiv \min_{\Sigma \in 2\Gamma} d_{SCJ}(\Sigma, \Delta) \tag{1}$$

Hence, in the *double distance problem* one seeks a labeling of each gene copy in a perfectly duplicated genome $\Sigma \in 2\Gamma$ that minimizes the SCJ distance to Δ.

For a genome Σ and an adjacency $\alpha = \{x, y\}$, let Σ_α be the set of all adjacencies of the form $\{x^i, y^j\}$ in Σ. Hence $|\Sigma_\alpha|$ can be 0, 1 or 2 if Σ is duplicated,

and 0 or 1 if Σ is ordinary. Let $A = \{\alpha = \{x, y\} | x \neq y\}$ be the set of all possible adjacencies with extremities belonging to distinct genes. A solution to the double distance problem is given by the following theorem:

Theorem 2. *[12] The SCJ double distance between an ordinary genome Γ and a duplicated genome Δ is*

$$dd_{SCJ}(\Gamma, \Delta) = |\Delta| + 2 \sum_{\alpha \in A} |\Gamma_\alpha|(1 - |\Delta_\alpha|).$$

A perfectly duplicated genome $\Sigma \in 2\Gamma$ realizing the distance is obtained by taking, for each adjacency $\alpha = \{x, y\} \in \Gamma$: (1) the labeled adjacencies of Δ_α, and (2) adjacencies $\{x^i, y^j\}$ with arbitrary labeling that do not conflict with (1) or among themselves.

3 The SCJD Model

In this section we generalize the SCJ model to allow duplications.

A *duplication* operation on a genome Π takes a linear chromosome C in Π and produces a new genome Π' with an additional copy of the chromosome. For example, if $\Pi = \{abcd, efg\}$ then a duplication of the first chromosome will give $\Pi' = \{abcd, abcd, efg\}$. An *SCJD operation* is either an SCJ or a duplication.

Given two linear genomes on the same gene set of size n, an ordinary one Γ and a duplicated one Δ, a sequence of SCJD operations that transforms Γ into Δ is called an *SCJD sorting scenario*. The *SCJD distance*, denoted by $d_{SCJD}(\Gamma, \Delta)$, is the number of operations in a shortest SCJD sorting scenario between Γ and Δ.

Since we focus on linear genomes we will assume from now on that all chromosomes, including intermediate ones, are linear unless specified otherwise. The following simple lemma shows that this can be satisfied when using only SCJ operations:

Lemma 1. *A sequence of SCJ operations transforming one linear genome into another linear genome can be reordered, producing another sequence with the same length, such that all intermediate genomes are linear.*

The examples below demonstrate SCJ double distances and SCJD sorting scenarios. For simplicity, we drop the braces around genomes from now on.

Example 1. $\Gamma = a$, $\Delta = a{-}a$; $dd_{SCJ}(\Gamma, \Delta) = 1$; $d_{SCJD}(\Gamma, \Delta) = 2$:

$$\Gamma \underset{dup}{\longrightarrow} a, a \underset{join}{\longrightarrow} \Delta$$

Example 2. $\Gamma = ab$, $\Delta = ab, ab$; $dd_{SCJ}(\Gamma, \Delta) = 0$; $d_{SCJD}(\Gamma, \Delta) = 1$:

$$\Gamma \underset{dup}{\longrightarrow} \Delta$$

Example 3. $\Gamma = a, bc$, $\Delta = ab, abcc$; $dd_{SCJ}(\Gamma, \Delta) = 4$; $d_{SCJD}(\Gamma, \Delta) \leq 4$:

$$\Gamma \underset{join}{\longrightarrow} abc \underset{dup}{\longrightarrow} abc, abc \underset{cut}{\longrightarrow} abc, ab, c \underset{join}{\longrightarrow} \Delta$$

Example 4. $\Gamma = acb$, $\Delta = abab, cc$; $dd_{SCJ}(\Gamma, \Delta) = 8$; $d_{SCJD}(\Gamma, \Delta) \leq 7$:

$$\Gamma \underset{cut}{\longrightarrow} a, cb \underset{cut}{\longrightarrow} a, b, c \underset{join}{\longrightarrow} ab, c \underset{dup}{\longrightarrow} ab, ab, c \underset{dup}{\longrightarrow} ab, ab, c, c \underset{join}{\longrightarrow} abab, c, c \underset{join}{\longrightarrow} \Delta$$

Let $\#_c \Pi$ be the number of linear chromosomes in genome Π. Let Γ be an ordinary linear genome and let Δ be a duplicated linear genome on the same gene set. A trivial upper bound for the SCJD distance between Γ and Δ is given by solving the double distance between Δ and Γ. This corresponds to first duplicating each chromosome in Γ and then computing the SCJ distance between Δ and $\Gamma \bigoplus \Gamma$. We get $d_{SCJD}(\Gamma, \Delta) \leq dd_{SCJ}(\Gamma, \Delta) + \#_c \Gamma$. However, Example 3 shows that this bound is not tight. It is tempting to guess that $dd_{SCJ}(\Gamma, \Delta) \leq d_{SCJD}(\Gamma, \Delta)$. Alas, Example 4 shows this conjecture is incorrect.

4 Computing the SCJD distance

In this section we will solve the SCJD distance problem. The key idea is to show that there is an optimal scenario in which all the duplication operations are performed in sequence, one after the other. Having shown that, the sorting scenario between Γ and Δ can be presented as follows:

1. Transform Γ into another ordinary linear genome Γ' using only SCJ operations.
2. Duplicate all the chromosomes of Γ' resulting in a duplicated genome $\Gamma' \bigoplus \Gamma'$.
3. Solve the SCJ double distance problem between Γ' and Δ.

Let $O^* = o_1, \ldots, o_d$ be an optimal SCJD sorting scenario. Let $\Gamma_0 \equiv \Gamma$ and and for every $1 \leq i \leq d$ let $\Gamma_i = o_i(\Gamma_{i-1})$ be the genome resulting from performing o_i on Γ_{i-1}. By definition, $\Gamma_d \equiv \Delta$. Let D_i be the set of duplicated genes in Γ_i. We have $D_0 = \emptyset$ and $D_d = \mathcal{G}$. Given a gene set $\mathcal{H}$, denote its extremity set by $\mathcal{E}_\mathcal{H} = \{a_t | a \in \mathcal{H}\} \cup \{a_h | a \in \mathcal{H}\}$.

Proposition 1. *In an optimal sorting scenario O^*, if o_i is a join operation acting on the telomeres x and y, then either $x, y \in \mathcal{E}_{D_i}$ or $x, y \notin \mathcal{E}_{D_i}$.*

Proof. Since o_i is not a duplication, we have $D_{i-1} = D_i$. Suppose by contradiction that $x \in \mathcal{E}_{D_i}$ but $y \notin \mathcal{E}_{D_i}$. Let o_j $(i < j)$ be the first duplication such that $y \in \mathcal{E}_{D_j}$. The duplication operation must act on a chromosome in which all genes are not yet duplicated. Therefore, there is a cut operation o_k $(i < k < j)$ that breaks the adjacency $\{x, y\}$ created by o_i.

Let $O' = o'_1, \ldots, o'_{d-2} = o_1, \ldots, o_{i-1}, o_{i+1}, \ldots, o_{k-1}, o_{k+1}, \ldots, o_d$ be an alternative sorting sequence that results from removing o_i and o_k from O^*. Let $\Gamma'_0 \equiv \Gamma$, and denote $\Gamma'_l = o'_l(\Gamma'_{l-1})$. For every l with $1 \leq l \leq i-1$, by definition, $o'_l = o_l$ and therefore $\Gamma'_l = \Gamma_l$.

We first show that for every l with $i \leq l \leq k-2$, $\Gamma'_l = \Gamma_{l+1} \setminus \big\{\{x,y\}\big\}$. Since o_i creates the adjacency $\{x,y\}$ we have that $\Gamma_i = \Gamma_{i-1} \cup \big\{\{x,y\}\big\}$. For every such l, $o'_l = o_{l+1}$ and since none of these operations creates a new copy of y we have that $\Gamma'_l = \Gamma_{l+1} \setminus \big\{\{x,y\}\big\}$.

Next, we show that for every l with $k-1 \leq l \leq d-2$, $\Gamma'_l = \Gamma_{l+2}$. From the previous result, and the fact that $\Gamma_k = \Gamma_{k-1} \setminus \big\{\{x,y\}\big\}$, we have $\Gamma'_{k-2} = \Gamma_k$. Now, for every such l, $o'_l = o_{l+2}$ and therefore $\Gamma'_l = \Gamma_{l+2}$.

We have established that O' is an SCJD sorting sequence of length $d-2$, contradicting the optimality of O^*. □

Proposition 2. *In an optimal sorting scenario O^*, if o_i is a cut operation acting on the adjacency $\{x,y\}$, then either $x,y \in \mathcal{E}_{D_i}$ or $x,y \notin \mathcal{E}_{D_i}$.*

Corollary 1. *In an optimal sequence of SCJD operations, at the time of a cut or a join operation on the two extremities x and y, either the genes corresponding to both x and y have both already been duplicated or none of them have.* □

Observe that a join operation in a sorting scenario is valid only if the two extremities it joins are not already part of any other adjacency. Similarly, a cut operation is valid only if the adjacency it breaks exists. A duplication operation is valid only if it duplicates a linear chromosome such that all its genes were not previously duplicated. A sorting scenario is valid if all its operations are valid.

Let $S = s_1, \ldots, s_m$ be a valid SCJD sorting scenario between Γ and Δ. We say the operation s_{i+1} can *preempt* the operation s_i if the sequence $S' = s_1, \ldots, s_{i+1}, s_i, \ldots, s_m$ is also a valid SCJD sorting scenario between Γ and Δ.

Proposition 3. *In a valid SCJD scenario S transforming Γ into Δ, if s_{i+1} is an SCJ operation acting on two extremities x,y that were not duplicated and s_i is a duplication, then s_{i+1} can preempt s_i.*

Proof. Suppose s_i duplicates the linear chromosome C and produces another copy of it C'. Since s_{i+1} operates on genes that are not duplicated yet, none of those genes belong to C or C'. Therefore, the sequence $s_1, \ldots, s_{i-1}, s_{i+1}$ is valid. Any operation that creates an adjacency or a telomere of C must precede s_i. Hence, $s_1, \ldots, s_{i-1}, s_{i+1}, s_i$ is valid. Finally, any s_j for $j > i+1$ that requires the results of s_i or s_{i+1} is still valid. Thus, $S' = s_1, \ldots, s_{i-1}, s_{i+1}, s_i, s_{i+2}, \ldots, s_m$ is a valid sequence.

To conclude the proof, we need to show that $\Gamma_{i+1} \equiv \Gamma'_{i+1}$. Indeed, s_{i+1} does not alter any of the adjacencies or telomeres of C or C', and therefore, $\Gamma_{i+1} = s_{i+1}(\Gamma_{i-1} \cup C') \equiv s_{i+1}(\Gamma_{i-1}) \cup C' = \Gamma'_{i+1}$. □

Proposition 4. *In a valid SCJD scenario S transforming Γ into Δ, if s_{i+1} is a duplication and s_i is a cut or join acting on two duplicated extremities, then s_{i+1} can preempt s_i.*

Proposition 5. *In a valid SCJD scenario S transforming Γ into Δ, if s_{i+1} is an SCJ acting on two extremities that were not duplicated yet and s_i is an SCJ acting on two duplicated extremities, then s_{i+1} can preempt s_i.*

For a sequence of SCJ operations S, let S^D ($\overline{S}^D$, respectively) be the subsequence of operations that act on two extremities of genes that have (have not, respectively) already been duplicated at the time of the operation. By Corollary 1, for optimal S, $\overline{S}^D$ is indeed the complement of S^D.

Proposition 6. *There exists an optimal sorting scenario in which all duplication events are consecutive.*

Proof. Let $o_{i_1}, \ldots, o_{i_p}$ be the duplication events in an optimal sorting scenario. Denote by S_{i_j} the sequence of SCJ operations occurring between the duplications o_{i_j} and $o_{i_{j+1}}$. Also, denote by S_{i_0} and S_{i_p} the sequence of SCJ operations before the first duplication and after the last duplication, respectively.

Given an optimal scenario $O^* = S_{i_0}, o_{i_1}, S_{i_1}, o_{i_2}, S_{i_2}, \ldots, S_{i_{p-1}}, o_{i_p}, S_{i_p}$ we modify it into a new sorting scenario O' as follows: Using Propositions 3 and 5, preempt SCJ operations acting on un-duplicated genes. Using Proposition 4, preempt duplication events. These steps are iterated until no preemption is possible. We get that $O' = S_{i_0}, \overline{S}^D_{i_1}, \ldots, \overline{S}^D_{i_p}, o_{i_1}, \ldots, o_{i_p}, S^D_{i_1}, \ldots, S^D_{i_{p-1}}, S_{i_p}$ is a valid SCJD optimal sequence in which all duplications are consecutive. □

Corollary 2. *There exists an optimal SCJD sorting scenario, consisting, in this order, of (1) SCJ operations on single-copy genes, (2) duplications, (3) SCJ operations acting on duplicated genes.* □

Denote by Γ' the intermediate (ordinary) genome after step (1). Then we can conclude:

Theorem 3. $d_{SCJD}(\Gamma, \Delta) = \min_{\Gamma'} \Big(d_{SCJ}(\Gamma, \Gamma') + \#_c\Gamma' + dd_{SCJ}(\Gamma', \Delta) \Big)$ □

Recall that n is the number of genes in Γ. Using Theorems 1 and 2 and the fact that $\#_c\Pi = n - |\Pi|$, the distance formula can be simplified:

$$d_{SCJD} = \min_{\Gamma'} \Big(|\Gamma| + |\Gamma'| - 2|\Gamma \cap \Gamma'| + n - |\Gamma'| + |\Delta| + 2 \sum_{\alpha \in A} |\Gamma'_\alpha|(1 - |\Delta_\alpha|) \Big)$$

$$= n + |\Delta| + |\Gamma| - 2 \max_{\Gamma'} \Big(|\Gamma \cap \Gamma'| + \sum_{\alpha \in A} |\Gamma'_\alpha|(|\Delta_\alpha| - 1) \Big)$$

$$= n + |\Delta| + |\Gamma| - 2 \max_{\Gamma'} \sum_{\alpha \in \Gamma'} (|\Gamma_\alpha| + |\Delta_\alpha| - 1)$$

$$= n + |\Delta| + |\Gamma| - 2 \max_{\Gamma'} \sum_{\alpha \in \Gamma'} \eta(\alpha) = n + |\Delta| + |\Gamma| - 2 \max_{\Gamma'} H(\Gamma') \qquad (2)$$

where $\eta(\alpha) = \eta(\alpha, \Gamma, \Delta) = |\Gamma_\alpha| + |\Delta_\alpha| - 1$ and $H(\Gamma') = \sum_{\alpha \in \Gamma'} \eta(\alpha)$. Since we want to maximize $H(\Gamma')$, we will focus on adjacencies with positive contribution in equation 2.

Lemma 2. *Let $\alpha = \{x, y\}$ be an adjacency such that $\eta(\alpha) > 0$. Then, for every extremity $z \neq y$, the conflicting adjacency $\alpha' = \{x, z\}$ has $\eta(\alpha') \leq 0$.*

Combining Lemma 2 and Theorem 3 we get a closed formula for the SCJD distance:

Theorem 4. *The genome* $\Gamma' = \{\alpha = \{x, y\} | \eta(\alpha) > 0\}$ *minimizes Equation 2. If* Γ' *is a linear genome, then the SCJD distance is given by* $d_{SCJD}(\Gamma, \Delta) = n + |\Delta| + |\Gamma| - 2H(\Gamma')$. □

Let us return to the examples in Section 3:

- Example 1: $n = 1$, $|\Delta| = 1$, $|\Gamma| = 0$, $\Gamma' = \emptyset$, $H(\Gamma') = 0 \rightarrow d = 1 + 1 + 0 - 2 * 0 = 2$
- Example 2: $n = 2$, $|\Delta| = 2$, $|\Gamma| = 1$, $\Gamma' = \{\{a_h, b_t\}\}$, $H(\Gamma') = 2 \rightarrow d = 2 + 2 + 1 - 2 * 2 = 1$
- Example 3: $n = 3$, $|\Delta| = 4$, $|\Gamma| = 1$, $\Gamma' = \{\{a_h, b_t\}, \{b_h, c_t\}\}$, $H(\Gamma') = 1 + 1 \rightarrow d = 3 + 4 + 1 - 2 * 2 = 4$
- Example 4: $n = 3$, $|\Delta| = 4$, $|\Gamma| = 2$, $\Gamma' = \{\{a_h, b_t\}\}$, $H(\Gamma') = 1 \rightarrow d = 3 + 4 + 2 - 2 * 1 = 7$

Example 5. $\Gamma = abc$ and $\Delta = cab, bca$. According to Theorem 4, we get $\Gamma' = (abc)$ because $\eta(\{a_h, b_t\}) = \eta(\{b_h, c_t\}) = \eta(\{c_h, a_t\}) = 1$. The corresponding distance is $d = 3$, providing the following invalid sorting scenario:

$$\Gamma \underset{join}{\longrightarrow} (abc) \underset{dup^*}{\longrightarrow} (abc), (abc) \underset{cut}{\longrightarrow} cab, (abc) \underset{cut}{\longrightarrow} \Delta$$

dup^* indicates a duplication of a circular chromosome, an operation that is not allowed in the SCJD model (and has no cost). It is not difficult to verify that there is no valid sorting scenario with $d \leq 3$.

The reason for the discrepancy in Example 5 is that $\#_c(\Gamma') = n - |\Gamma'| = 0$ is not equal to the number of duplications if there are circular chromosomes. Therefore, in order to minimize the SCJD distance given by Equation 2, we need to maximize $H(\Gamma')$ under the constraint that Γ' is a linear genome, i.e., $H(\Gamma') \geq H(\tilde{\Gamma})$ for every linear genome $\tilde{\Gamma}$. Lemma 3 shows that we can do so simply by removing one adjacency with $\eta = 1$ from each circular chromosome in Γ' and that such adjacency must exist.

Lemma 3. *Let* $\Gamma' = \{\alpha = \{x, y\} | \eta(\alpha) > 0\}$ *and let* Γ'' *be a genome obtained by removing one adjacency* α *with* $\eta(\alpha) = 1$ *from each circular chromosome in* Γ'. *Then,* Γ'' *is a linear genome that maximizes* $H(\cdot)$ *and the SCJD distance is given by* $d_{SCJD}(\Delta, \Gamma) = n + |\Delta| + |\Gamma| - 2H(\Gamma'')$.

Applying Lemma 3 to Example 5 we get $\Gamma'' = abc$ and $d = 5$:

$$\Gamma \underset{dup}{\longrightarrow} abc, abc \underset{cut}{\longrightarrow} a, bc, abc \underset{join}{\longrightarrow} bca, abc \underset{cut}{\longrightarrow} bca, ab, c \underset{join}{\longrightarrow} \Delta$$

We can choose instead $\Gamma'' = cab$, which gives a different optimal sorting scenario:

$$\Gamma \underset{cut}{\longrightarrow} ab, c \underset{join}{\longrightarrow} cab \underset{dup}{\longrightarrow} cab, cab \underset{cut}{\longrightarrow} cab, a, bc \underset{join}{\longrightarrow} \Delta$$

Algorithm 1 gives the full procedure for solving the SCJD distance and sorting problems. Each step of the algorithm takes $O(|\Gamma| + |\Delta|)$ time. In conclusion:

Theorem 5. *Algorithm 1 computes the SCJD distance in linear time.* □

Algorithm 1 SCJD distance.
Input: An ordinary genome Γ and a duplicated genome Δ (both linear) on the same gene set.
Output: The SCJD distance $d_{SCJD}(\Gamma, \Delta)$ and an optimal sorting scenario $o_1, \ldots, o_d$ in which all intermediate genomes are linear.

1: $\Gamma' \leftarrow \{\alpha = \{x, y\} | \eta(\alpha) > 0\}$ (Theorem 4)
2: Create a linear genome Γ'' by removing one adjacency α with $\eta(\alpha) = 1$ from each circular chromosome in Γ' (Lemma 3)
3: $d_{SCJD}(\Gamma, \Delta) \leftarrow n + |\Delta| + |\Gamma| - 2H(\Gamma'')$ (Theorem 4, Lemma 3)
4: $o_1, \ldots, o_i \leftarrow$ Sort Γ into Γ'' (Theorem 1, Lemma 1)
5: $o_{i+1}, \ldots, o_j \leftarrow$ Duplicate all chromosomes in Γ''.
6: $o_{j+1}, \ldots, o_d \leftarrow$ Sort $2\Gamma''$ into Δ (Theorem 2, Lemma 1).
7: **return** $d, \overrightarrow{o}$

5 Controlling the number of duplications

In this section we discuss how to control the number of duplications in an optimal SCJD sequence. Since the number of duplications is $n - |\Gamma''|$, selecting different intermediate genomes Γ'' that preserve the SCJD distance can produce scenarios with different number of duplications.

An optimal SCJD scenario with fewer duplications can be viewed as more conservative. The assumption behind this is that duplications are more "radical" events than breakage (cut) or fusion (join), which are local events.

Lemma 4. *Algorithm 1 gives an optimal sorting scenario with a maximum number of duplications.*

Proof. Observe first that for any sorting scenario (optimal or suboptimal) transforming Γ into Δ, we can assume w.l.o.g. that all duplications are consecutive without affecting the number of operations (Corollary 2). Call the genome right before the duplications the *last ordinary genome*. Denote by $d(\Gamma, \Pi, \Delta)$ the shortest scenario transforming Γ into Δ given that the last ordinary genome is Π. The proof of Theorem 3 implies that $d(\Gamma, \Pi, \Delta) = n + |\Delta| + |\Gamma| - 2H(\Pi)$.

Let Γ' be the last ordinary genome produced by the algorithm. Consider an optimal scenario O with a maximum number of duplications and let $\tilde{\Gamma}$ be the last ordinary linear genome in O. Since O is optimal, $H(\tilde{\Gamma})$ must be maximal. Hence, $\tilde{\Gamma}$ cannot contain adjacencies with $\eta < 0$. Moreover, it cannot contain adjacencies with $\eta = 0$, as such adjacencies increase $|\tilde{\Gamma}|$ and thus decrease the number of duplications in O. Therefore, $\tilde{\Gamma} \subseteq \Gamma'$.

We now show that $\forall \alpha \in \Gamma' \setminus \tilde{\Gamma}$, $\eta(\alpha) = 1$. Suppose by contradiction that there is an adjacency $\alpha \in \Gamma' \setminus \tilde{\Gamma}$ with $\eta(\alpha) > 1$ and let $\Pi = \tilde{\Gamma} \cup \{\alpha\}$. If Π is a linear genome, $d(\Gamma, \Pi, \Delta) < d(\Gamma, \tilde{\Gamma}, \Delta)$ contradicting the optimality of O. Otherwise, Π contains a circular chromosome and by Lemma 3, there is an adjacency $\beta \in \tilde{\Gamma}$ with $\eta(\beta) = 1$ such that $\Pi \setminus \{\beta\}$ is a linear genome with $H(\Pi \setminus \{\beta\}) > H(\tilde{\Gamma})$, again contradicting the optimality of O. Thus, $|\Gamma' \setminus \tilde{\Gamma}| = |\Gamma'| - |\tilde{\Gamma}| = H(\Gamma') - H(\tilde{\Gamma})$.

Γ' may contain circular chromosomes. By Lemma 3, Γ'' is produced by removing one adjacency with $\eta = 1$ from each circular chromosome in Γ'. Hence $|\Gamma' \setminus \Gamma''| = |\Gamma'| - |\Gamma''| = H(\Gamma') - H(\Gamma'')$.

Since both $\tilde{\Gamma}$ and Γ'' are last ordinary genomes in optimal SCJD scenarios, $H(\tilde{\Gamma}) = H(\Gamma'')$. Thus, $|\Gamma'|-|\tilde{\Gamma}| = H(\Gamma')-H(\tilde{\Gamma}) = H(\Gamma')-H(\Gamma'') = |\Gamma'|-|\Gamma''|$, which implies that $|\tilde{\Gamma}| = |\Gamma''|$. □

One can decrease the number of duplications in an optimal SCJD scenario by adding adjacencies with $\eta(\alpha) = 0$ to Γ''. However, we need to make sure that the resulting genome is still linear. Consider the following example:

Example 6. $\Gamma = a, b, c$, $\Delta = abccba$. From Theorem 4 we have that $\Gamma' = \Gamma$ and so the SCJD distance is 8. The scenario produced by Algorithm 1 will first duplicate the three chromosomes of Γ and then perform five joins to create Δ. An alternative optimal sorting scenario is:

$$\Gamma \underset{JJ}{\longrightarrow} abc \underset{D}{\longrightarrow} abc, abc \underset{CC}{\longrightarrow} abc, a, b, c \underset{JJJ}{\longrightarrow} \Delta$$

Here, since each adjacency $\alpha \in \Delta$ has $\eta(\alpha) = 0$, we chose $\Gamma'' = abc$ and obtained an optimal scenario with a single duplication. In contrast, if we add to Γ'' the adjacencies $\{b_h, c_t\}$ and $\{c_h, b_t\}$ (which also have $\eta = 0$) we create a circular chromosome and an invalid SCJD sorting scenario.

In order to minimize the number of duplications we must add to Γ'' a maximum set of adjacencies with $\eta = 0$ such that the resulting genome is still linear. Here we show that this problem is NP-hard using a reduction similar to [16].

Theorem 6. *Given an ordinary linear genome Γ, a duplicated linear genome Δ on the same gene set, and an integer k, the problem of finding an optimal SCJD scenario with at most k duplications is NP-hard.*

Proof. Call a directed graph in which all in- and out-degrees are 2 a *2-digraph*. Deciding if a 2-digraph contains a Hamiltonian cycle is NP-hard [20, 16]. This implies that the following variant is also NP-hard: Given a 2-digraph G with an edge (x, y), decide if there is a Hamiltonian path from y to x in G.

Let $G = (V, E)$ be a 2-digraph with an edge (x, y) as above. We may assume w.l.o.g. that G is strongly connected, since otherwise it would not contain a Hamiltonian path from y to x. Notice that $G \setminus (x, y)$ contains an Eulerian path from y to x [10]. Denote it by $P = e_1, e_2, \ldots, e_m$.

We construct a duplicated genome Σ as follows: for each $e_q = (u, v) \in P$ add the adjacency $\{u_h^i, v_t^j\}$ where $i = 2$ if there is an edge $e_l = (u, v')$ with $l < q$, and $i = 1$ otherwise. Similarly, $j = 2$ if there is an edge $e_m = (u', v)$ with $m < q$ and $j = 1$ otherwise. The result is a linear chromosome created by traversing P and numbering the first occurrence of each vertex v in P as the gene copy v^1 and the second occurrence as v^2. Denote by $\overset{P}{\rightsquigarrow}$ the sequence of genes along the path P. In addition, we add two new genes w, z and the adjacencies $\{w_h^1, y_t^1\}, \{x_h^2, z_t^1\}$. Thus, Σ has three linear chromosomes: $w^1 y^1 \overset{P}{\rightsquigarrow} x^2 z^1, w^2$ and

z^2. Let $\Pi = \{\{w_h, y_t\}, \{x_h, z_t\}\}$ be an ordinary genome with n chromosomes over the same set of genes. (Note that every vertex in $V \setminus \{x, y\}$ corresponds to a separate chromosome in Π.)

Let $\Sigma_{(i)}$ and $\Pi_{(i)}$ be genomes in which every gene $v \in V$ is renamed $v_{(i)}$. We define $\Delta = \bigcup_{i=1}^{k} \Sigma_{(i)}$ and $\Gamma = \bigcup_{i=1}^{k} \Pi_{(i)}$ to be the disjoint union of k different copies of Σ and Π respectively. This completes the reduction, which is clearly polynomial. We will show that there is an optimal SCJD scenario between Γ and Δ with at most k duplications iff G admits a Hamiltonian path from y to x.

For each edge $e = (u, v) \in E$ and every i, the corresponding adjacency $\alpha = \{(u_{(i)})_h^j, (v_{(i)})_t^l\}$ has $\eta(\alpha) = 1$ if there are two parallel edges from u to v, and $\eta(\alpha) = 0$ otherwise. In addition, for every i, $\eta(\{(w_{(i)})_h, (y_{(i)})_t\}) = \eta(\{(x_{(i)})_h, (z_{(i)})_t\}) = 1$, and every other adjacencies of $w_{(i)}, z_{(i)}$ have $\eta < 0$.

Suppose G contains a Hamiltonian path S from y to x. Let Γ' be the genome formed by the set of adjacencies $\big\{\{(w_{(i)})_h, (y_{(i)})_t\}, \{(x_{(i)})_h, (z_{(i)})_t\} | i = 1 \dots k\big\}$ $\cup \big\{\{(u_{(i)})_h, (v_{(i)})_t\} | (u, v) \in S, i = 1 \dots k\big\}$. Since S is a Hamiltonian path, Γ' is a valid ordinary linear genome with k chromosomes of the form $w_{(i)} y_{(i)} \overset{S}{\rightsquigarrow} x_{(i)} z_{(i)}$. To prove that Γ' maximizes $H(\cdot)$ we need to show it contains every adjacency with $\eta = 1$ and no adjacency with $\eta < 0$. Indeed, (suppressing the copy index i for clarity) the only adjacencies α with $\eta(\alpha) = 1$ are $\{w_h, y_t\}, \{x_h, z_t\}$ ($|\Delta_\alpha| = |\Gamma_\alpha| = 1$) and parallel edges in G ($|\Delta_\alpha| = 2, |\Gamma_\alpha| = 0$), one copy of which must be included in S. All other adjacencies in Γ' have $|\Delta_\alpha| = 1, |\Gamma_\alpha| = 0$ and $\eta(\alpha) = 0$. We conclude that Γ' is part of an optimal scenario with k duplications.

Conversely, suppose there is an optimal scenario O^* with at most k duplications and let $\tilde{\Gamma}$ be the last ordinary genome in O^*. Let $\Gamma' = \{\alpha | \eta(\alpha) > 0\}$ be a genome that minimizes the SCJD distance according to Theorem 4. First, notice that Γ' is indeed a linear genome. Otherwise, a circular chromosome of adjacencies with $\eta(\alpha) = 1$ would imply a strongly connected component without the vertices x, y, contradicting the strong connectivity of G. It follows that $\Gamma' \subseteq \tilde{\Gamma}$, $H(\Gamma') = H(\tilde{\Gamma})$ and $\#_c \tilde{\Gamma} \leq k$.

Since $\Sigma_{(i)}$ and $\Sigma_{(j)}$ for $i \neq j$ contain different genes, an adjacency between a gene in $\Sigma_{(i)}$ and a gene $\Sigma_{(j)}$ has negative η. Therefore, $\tilde{\Gamma}$ contains no such adjacencies. Since $\tilde{\Gamma}$ has at most k linear chromosomes, it must contain exactly k linear chromosomes, each containing all the genes of $\Sigma_{(i)}$ for one i.

Let $C = w_{(1)} y_{(1)} \dots x_{(1)} z_{(1)}$ be the linear chromosome in $\tilde{\Gamma}$ that contains all the genes of $\Sigma_{(1)}$. Define an edge set S in G by taking for each adjacency $\{(u_{(1)})_h, (v_{(1)})_t\} \in C \setminus \big\{\{(w_{(1)})_h, (y_{(1)})_t\}, \{(x_{(1)})_h, (z_{(1)})_t\}\big\}$ the edge (u, v). Since C is an ordinary linear chromosome containing all the genes of $\Sigma_{(1)}$, S is a Hamiltonian path in G from y to x. □

6 Discussion

In this paper, we presented the SCJD rearrangement model, which allows the operations cut, join and whole chromosome duplication. We analyzed the problem

of finding the minimum number of SCJD operations that transform an ordinary linear genome into a duplicated linear genome and provided a linear time algorithm for it. Furthermore, we showed that this algorithm gives an optimal scenario with a maximum number of duplications and that finding one with fewest duplications is NP-hard.

In the analysis, we focused on the SCJD sorting problem, which restricts the target genome to have exactly two copies of each gene. However, it is not difficult to generalize our algorithm to address the more general situation where each gene in the target genome has *at most* two copies. One can show that in this case too, an optimal solution in which all duplications are consecutive exists. In addition, each adjacency in the original genome between a gene that has two copies and a gene that has one copy in the target genome, must first be cut. This is true because duplications are defined over linear chromosomes in which every gene is unduplicated.

Our algorithm relies on the property that all duplications in the optimal solution can be clustered (Corollary 2). In this sense, the problem we study is similar to the SCJ Guided Genome Halving problem [12]. In that model the whole genome is duplicated at once, while in ours there is one duplication per chromosome, and accounting for these duplications is part of the optimization challenge.

Many aspects in the analysis of the SCJD mode require further research: How can we address the problem if there are more than two copies of each gene? Can we find the SCJD distance between two arbitrary genomes - each containing single copy and multiple copy genes? How does removing the requirement of linearity affect various SCJD problems? Moreover, duplications may be defined differently, e.g. tandem duplications [1] and segmental duplications [23]. Finally, developing a rigorous model that will allow both duplications and deletions is needed in order to analyze the full complexity of real biological data such as cancer samples.

Acknowledgments. We thank our referees for many helpful and insightful comments. This study was supported by the Israeli Science Foundation (grant 317/13) and the Dotan Hemato-Oncology Research Center at Tel Aviv University. RZ was supported in part by fellowships from the Edmond J. Safra Center for Bioinformatics at Tel Aviv University and from the Israeli Center of Research Excellence (I-CORE) Gene Regulation in Complex Human Disease (Center No 41/11).

References

1. M. Bader. Sorting by reversals, block interchanges, tandem duplications, and deletions. *BMC Bioinformatics*, 10 Suppl 1:S9, 2009.
2. M. Bader. Genome rearrangements with duplications. *BMC Bioinformatics*, 11 Suppl 1:S27, 2010.

3. J. Bayani, S. Selvarajah, G. Maire, B. Vukovic, K. Al-Romaih, M. Zielenska, and J. A. Squire. Genomic mechanisms and measurement of structural and numerical instability in cancer cells. *Seminars in Cancer Biology*, 17(1):5–18, 2007.
4. A. Bergeron, J. Mixtacki, and J. Stoye. A unifying view of genome rearrangements. In P. Bücher and B. M. Moret, editors, *Algorithms in Bioinformatics*, volume 4175 of *Lecture Notes in Computer Science*, pages 163–173. Springer, 2006.
5. P. Biller, P. Feijão, and J. Meidanis. Rearrangement-based phylogeny using the Single-Cut-or-Join operation. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 10(1):122–34, 2013.
6. G. Blanc, A. Barakat, R. Guyot, R. Cooke, and M. Delseny. Extensive duplication and reshuffling in the Arabidopsis genome. *The Plant cell*, 12(7):1093–101, 2000.
7. L. Bulteau, G. Fertin, and I. Rusu. Sorting by transpositions is difficult. *SIAM Journal on Discrete Mathematics*, 26(3):1148–1180, 2012.
8. A. Caprara. Sorting by reversals is difficult. In *Proceedings of the first annual international conference on Computational molecular biology (RECOMB)*, pages 75–83, New York, New York, USA, 1997.
9. D. A. Christie. Sorting permutations by block-interchanges. *Information Processing Letters*, 60(4):165–169, 1996.
10. T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, and Others. *Introduction to algorithms*, volume 2. MIT press Cambridge, 2001.
11. Z. Dias and J. Meidanis. Genome rearrangements distance by fusion, fission, and transposition is easy. In *International Symposium on String Processing and Information Retrieval*, page 250. IEEE Computer Society, 2001.
12. 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.
13. G. Fertin, A. Labarre, I. Rusu, E. Tannier, and S. Vialette. *Combinatorics of Genome Rearrangements*. MIT Press, 2009.
14. S. Hannenhalli. Polynomial-time algorithm for computing translocation distance between genomes. *Discrete Applied Mathematics*, 71(1-3):137–151, 1996.
15. S. Hannenhalli and P. A. Pevzner. Transforming cabbage into turnip. In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing (STOC)*, volume 46, pages 178–189, New York, New York, USA, 1995.
16. J. Kováč. On the complexity of rearrangement problems under the breakpoint distance. *Journal of Computational Biology*, 21(1):1–15, 2014.
17. C. L. Lu, Y. L. Huang, T. C. Wang, and H.-T. Chiu. Analysis of circular genome rearrangement by fusions, fissions and block-interchanges. *BMC Bioinformatics*, 7(1):295, 2006.
18. C. V. G. Mira and J. Meidanis. Sorting by block-interchanges and signed reversals. *ITNG*, 7:670–676, 2007.
19. M. Ozery-Flato and R. Shamir. Sorting cancer karyotypes by elementary operations. *Journal of Computational Biology*, 16(10):1445–60, 2009.
20. J. Plesnik. The NP-completeness of the Hamiltonian cycle problem in planar digraphs with degree bound two. *Information Processing Letters*, 8(4):199–201, 1979.
21. O. T. Savard, Y. Gagnon, D. Bertrand, and N. El-Mabrouk. Genome halving and double distance with losses. *Journal of Computational Biology*, 18(9):1185–99, 2011.
22. 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.

23. M. Shao, Y. Lin, and B. Moret. Sorting genomes with rearrangements and segmental duplications through trajectory graphs. *BMC Bioinformatics*, 14(Suppl 15):S9, 2013.
24. M. Shao, Y. Lin, and B. Moret. An exact algorithm to compute the DCJ distance for genomes with duplicate genes. In R. Sharan, editor, *Research in Computational Molecular Biology*, volume 8394 of *Lecture Notes in Computer Science*, pages 280–292. Springer, 2014.
25. E. Tannier, C. Zheng, and D. Sankoff. Multichromosomal median and halving problems under different genomic distances. *BMC Bioinformatics*, 10(1):120, 2009.
26. R. Warren and D. Sankoff. Genome aliquoting revisited. *Journal of Computational Biology*, 18(9):1065–1075, 2011.
27. S. Yancopoulos, O. Attie, and R. Friedberg. Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics*, 21(16):3340–3346, 2005.