



Genome Halving and Aliquoting Under the Copy Number Distance

Ron Zeira¹ ✉ 

Department of Computer Science, Princeton University, NJ, USA

Geoffrey Mon¹ ✉ 

Department of Computer Science, Princeton University, NJ, USA

Benjamin J. Raphael² ✉ 

Department of Computer Science, Princeton University, NJ, USA

Abstract

Large-scale genome rearrangements occur frequently in species evolution and cancer evolution. While the computation of evolutionary distances is tractable for balanced rearrangements, such as inversions and translocations, computing distances involving duplications and deletions is much more difficult. In the recently proposed Copy Number Distance (CND) model, a genome is represented as a Copy Number Profile (CNP), a sequence of integers, and the CND between two CNPs is the length of a shortest sequence of deletions and amplifications of contiguous segments that transforms one CNP into the other. In addition to these segmental events, genomes also undergo global events such as Whole Genome Duplication (WGD) or polyploidization that multiply the entire genome content. These global events are common and important in both species and cancer evolution. In this paper, we formulate the *genome halving problem* of finding a closest preduplication CNP that has undergone a WGD and evolved into a given CNP under the CND model. We also formulate the analogous *genome aliquoting problem* of finding the closest prepolyloidization CNP under the CND distance. We give a linear time algorithm for the halving distance and a quadratic time dynamic programming algorithm for the aliquoting distance. We implement these algorithms and show that they produce reasonable solutions on simulated CNPs.

2012 ACM Subject Classification Applied computing → Molecular evolution

Keywords and phrases Genome rearrangements, Copy number distance, Whole genome duplication, polyploidization, genome halving distance, genome aliquoting distance

Digital Object Identifier 10.4230/LIPIcs.WABI.2021.18

Supplementary Material *Software (Source Code)*: <https://github.com/raphael-group/CND-aliquoting>

Funding Benjamin J. Raphael²]Corresponding author: This work is supported by a US National Institutes of Health (NIH) grants U24CA211000 and U24CA248453.

1 Introduction

Genomes evolve over time through many types of mutations ranging from single-nucleotide mutations through large-scale alterations that affect both the order and the amount of genetic material. Such large-scale changes, termed *genome rearrangements*, are observed both in the evolution of species and of cancer cells [38, 28, 37, 10, 6, 23, 16]. Genome rearrangements can be categorized into two classes: structural rearrangements such as reversals, translocations and transpositions that change the order of DNA segments but not their quantity; and numerical rearrangements such as duplications and deletions that either create new copies of existing DNA segments or remove segments of the genome. The result of one genome evolving into another by a series of genome rearrangements is a new genome having a different structure and amount of DNA.

¹ First author

² Corresponding author



Computational models of genome rearrangements aim to calculate the minimum number of rearrangement events that transform one genome into the other, also called the *genome rearrangement distance*. Several types of rearrangement models have been proposed for structural events including the breakpoint (BP) distance [32, 31, 39], single-cut-or-join (SCJ) [13], the Hannenhalli-Pevzner model (HP) [18, 19] and the more general double-cut-and-join (DCJ) model [43, 3]. Representing the genome in these models requires the identification of homologous segments between the two genomes analyzed and determining the adjacencies between these segments in each genome. However, while these models admit polynomial time algorithms for the rearrangement distance when each genomic segment has a single copy in each genome, allowing for multiple copies often results in NP-hard problems [14].

Besides local rearrangement events, *whole genome duplication* (WGD) or polyploidization (>2 multiplication) events are viewed as a fundamental step in genome evolution as the multiplication of the genomic contents allows greater diversification of gene functions. In species evolution there has been strong evidence for WGD events reported for vertebrates [21], yeast [42] and for many plant genomes [5]. In fact, all angiosperms have undergone at least one WGD event in their evolutionary history and polyploidization is recognized as a major driving force for plant speciation [29]. In addition to species evolution, WGD is also a frequent event in cancer evolution with an estimated frequency of more than 30% in recent cancer studies [7, 45, 4, 9]. Furthermore, WGD is associated with poor prognosis across cancer types [4].

A basic question in the computational analysis of WGD is to reconstruct a closest ancestral preduplicated genome for a given extant genome. Namely, given a genome G and a rearrangement distance $d(\cdot, \cdot)$, the *genome halving problem* seeks to find an ancestral genome A such that the rearrangement distance $d(2A, G)$ between the duplicated genome $2A$ and the extant genome G is minimized [12]. The genome halving problem can be solved in linear time for the BP distance [39, 22], the SCJ distance [13], the HP distance [1] and the DCJ distance [25, 40, 2]. A generalization of the halving problem for polyploidization, i.e. finding a closest premultiplication ($p > 2$) genome is called the *genome aliquoting problem* [40]. The genome aliquoting problem can be solved in polynomial time for the BP distance [41] and the SCJ distance [13], while the complexity of the problem is not resolved for the DCJ distance, though a 2-approximation algorithm exists [41]. Apart from finding a preduplication genome, the genome halving and aliquoting problems also measure how close the extant genome is to a duplicated genome. By comparing different values of p , this allows us, for example, to distinguish if a genome has undergone duplication or triplication.

Motivated by applications in cancer evolution, alternative genome rearrangement models that focus on numerical rearrangement have recently been introduced [34, 8]. These models represent a genome as a *copy number profile* (CNP), a vector of integers indicating the number of copies of each segment from a reference genome. Thus, unlike structural rearrangement models, CNPs do not model the sequence of rearranged segments, but only the number of copies of segments of the reference genome. Therefore, genomes with different order of the segments may have the same CNP. However, the CNP representation is useful because it can be readily derived from DNA sequencing data or microarrays [7, 26, 17, 27, 35, 15]. The Copy Number Transformation (CNT) model models the evolution of CNPs by amplifications and deletions [34]. In this model an amplification (resp. deletion) increases (resp. decreases) the entries in a contiguous interval of the CNP. The Copy Number Distance (CND) between two profiles is defined as the length of a shortest sequence of amplifications and deletions that transform one profile into the other. The CND can be computed in linear time [46], and has

been used to analyze evolution of the genomes of multiple cancer types [34, 33, 36, 24, 44]. However, the CND does not adequately model WGD and polyploidization events, which are frequent in cancer.

In this paper, we formulate the genome halving and genome aliquoting problems for CNPs under the CND model and give polynomial time algorithms for both problems. Similar to other rearrangement models, we define the halving and aliquoting distances under the CND model as the minimum CND from some duplicated CNP to a given extant CNP. For the copy number halving distance problem, we give a very simple linear time algorithm for finding a closest preduplicated CNP of an extant CNP. Moreover, we show that we can find a closest preduplicated CNP such that the CND to the extant profile contains only deletions or a maximum number of amplifications. WGD followed by massive loss of genes is commonly known in evolution and thus finding a preduplication profile having such that after WGD there are only deletions is biologically reasonable [20]. For the copy number aliquoting problem we give a quadratic time dynamic programming algorithm. To this end we show that there exists a preduplicated CNP where each position in the profile is either amplified or deleted, and the number of new operations starting at each position is bounded. Furthermore, we show that each row in the dynamic programming table is a non decreasing function, thus enabling the calculation of each entry in constant time. We implement our algorithms and show on simulated data they are able to reconstruct a preduplication profile.

2 Preliminaries

In this section we present the CND model (Section 2.1) and formulate the halving and aliquoting problem under this model (Section 2.2).

2.1 Copy number profiles and distance

A *copy number profile* (CNP) $V = \langle v_1, \dots, v_n \rangle$ is a vector of non-negative integers. We refer to each coordinate in a copy number profile as a *gene* although more generally these entries correspond to genomic segments or synteny blocks. Each entry of a copy number profile gives the number of copies of the gene in the genome. For example, $V = \langle 0, 10, 15, 30 \rangle$ is a CNP with four genes, where the second gene of V has 10 copies.

For integers i, j with $i \leq j$ let $[i, j] = [i, i+1, i+2, \dots, j]$ denote the interval of integers from i to j . A *copy number operation* (CNO) is a triple (ℓ, h, w) where $\ell, h \in [1, n]$ denote two genes, and $w \in \{-1, 1\}$ denotes whether the CNO is a *deletion* or an *amplification*, respectively. We call ℓ and h the start and end genes (inclusive) of the contiguous segment $[\ell, h]$ of the CNP onto which the CNO is applied. When a CNO $c = (\ell, h, w)$ is *applied* to a CNP $V = \langle v_i \rangle_{i=1}^n$, the result is a new CNP $c(V) = U = \langle u_i \rangle_{i=1}^n$ defined as follows:

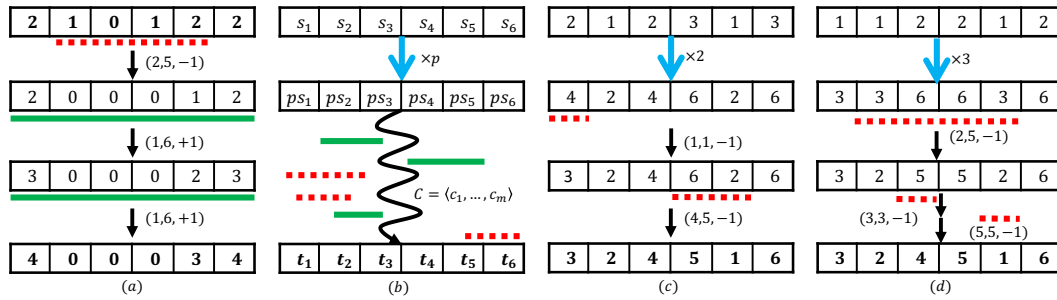
$$u_i = \begin{cases} v_i & i \notin [\ell, h] \\ 0 & v_i = 0 \text{ and } i \in [\ell, h] \\ v_i + w & v_i > 0 \text{ and } i \in [\ell, h] \end{cases}$$

We say that a gene i is *targeted* by a CNO $c = (\ell, h, w)$ if $\ell \leq i \leq h$.

A *copy number transformation* (CNT) is a vector $C = \langle c_1, c_2, \dots, c_m \rangle$ of CNOs. We *apply* a CNT C on a CNP U in order, i.e., $C(U) = c_m(c_{m-1}(\dots c_2(c_1(U)) \dots))$. The cardinality or size of a CNT is the number of CNOs, and is denoted $|C|$. We say that a CNT C has direction $U \rightarrow V$ if $C(U) = V$.

Given two CNPs U and V of n genes, the *copy number distance (CND)* $d(U, V)$ is the smallest integer t such that there exists some CNT C where $|C| = t$ and $C(U) = V$, i.e. t is the minimum number of CNOs required to transform U into V . While we use the term distance for CND, note that the CND is neither symmetric nor satisfies the triangle inequality. We call a CNT C *optimal* for the direction $U \rightarrow V$ if $|C| = d(U, V)$ and $C(U) = V$. For example, $d(\langle 2, 1, 0, 1, 2, 2 \rangle, \langle 4, 0, 0, 0, 3, 4 \rangle) = 3$ and an optimal transformation includes one deletion and two amplifications (Figure 1a). If no CNT of any size exists between U and V , then the distance $d(U, V) = \infty$. For instance, there is no transformation for the reverse direction $\langle 4, 0, 0, 0, 3, 4 \rangle \rightarrow \langle 2, 1, 0, 1, 2, 2 \rangle$ in the previous example (Figure 1a).

In addition to CNOs that increase or decrease the CNs of a CNP by 1, we introduce here a new operation for multiplying a CNP by a scalar. For a CNP S and an integer $p > 1$, we denote by $pS = \langle ps_1, ps_2, \dots, ps_n \rangle$ a *duplicated CNP* where each gene is multiplied by p . We call S the *preduplicated CNP* of duplicated CNP pS .



■ **Figure 1** (a) A copy number transformation from $\langle 2, 1, 0, 1, 2, 2 \rangle$ to $\langle 4, 0, 0, 0, 3, 4 \rangle$ includes one deletion (red dotted line) and two amplifications (green solid lines). (b) Schematic overview of the aliquoting problem. Given a CNP $T = \langle t_1, \dots, t_n \rangle$ (bold) and integer $p \geq 2$, find a preduplication profile $S = \langle s_1 \dots s_n \rangle$ that minimizes $d(pS, T)$. (c-d) The halving and aliquoting ($p = 3$) solutions for CNP $\langle 3, 2, 4, 5, 1, 6 \rangle$.

2.2 Copy number halving and aliquoting problems

Given a CNP T , we define the *CNP halving distance* $\eta_2(T)$ as the minimum CND between a doubled profile $2S$ and T :

$$\eta_2(T) = \min_S d(2S, T)$$

Similarly, for a CNP T and an integer $p \geq 2$, we define the *CNP aliquoting distance* $\eta_p(T)$ as the minimum CND between a duplicated profile pS and T :

$$\eta_p(T) = \min_S d(pS, T)$$

We say that \hat{S} is an *optimal preduplicated CNP* of an extant CNP T for the halving (aliquoting resp.) problem if $\eta_2(T) = d(2\hat{S}, T)$ ($\eta_p(T) = d(p\hat{S}, T)$ resp.). We formulate the problems of finding an optimal preduplicated CNP as follows (Figure 1b).

► **Copy Number Profile Halving Problem.** Given a CNP T , compute the halving distance $\eta_2(T)$ and find an optimal preduplication profile \hat{S} .

► **Copy Number Aliquoting Halving Problem.** Given a CNP T and an integer p , compute the aliquoting distance $\eta_p(T)$ and find an optimal preduplication profile \hat{S} .

For example, for the CNP $T = \langle 3, 2, 4, 5, 1, 6 \rangle$ the halving distance is $\eta_2(T) = 2$ using a preduplication profile $\hat{S} = \langle 2, 1, 2, 3, 1, 1 \rangle$ (Figure 1c) whereas the aliquoting distance is $\eta_3(T) = 3$ using a preduplication profile $\hat{S} = \langle 1, 1, 2, 2, 1, 2 \rangle$ (Figure 1d). Therefore under parsimony assumption, T is more likely to have originated from whole genome duplication than triplication.

3 Algorithms

In this section we give algorithms for the CNP halving and aliquoting problems. We begin by showing several properties that enable us to reduce the problem size and limit the search space of possible preduplication profiles (Section 3.1). Then we give a simple linear time algorithm for the halving problem in Section 3.2 and a quadratic dynamic programming algorithm for the aliquoting problem in Section 3.3.

3.1 Properties of aliquoting solutions

Here, we show a few properties that will simplify the halving and aliquoting problems. We first show that we may assume without loss of generality that the input CNP T has no zeroes and that T has no two adjacent genes that are congruent mod p . Therefore we can preprocess an T to remove such positions and reduce the profile size.

We start by showing we can remove genes with zero copy number without changing the halving/aliquoting distance. For brevity, we refer the reader to Appendix Section S1.1.1 for the full proof.

► **Proposition 1.** *Let $T = \langle t_1, t_2, \dots, t_{i-1}, 0, t_{i+1}, \dots, t_n \rangle$ be a profile with $t_i = 0$ and let $T' = \langle t_1, t_2, \dots, t_{i-1}, t_{i+1}, \dots, t_n \rangle$ be the profile with gene i removed. Then, $\eta_p(T) = \eta_p(T')$.*

Next, we show that we can assume that no two consecutive genes in T have the same value modulo p . We rely on the following observation proved in [46]. The full proof of Proposition 2 is omitted and given in Appendix Section S1.1.2.

► **Observation 1.** *Let S and T be profiles whose entries are strictly positive and let C be a CNT from $S \rightarrow T$. Let a_i be the number of amplification events in C that target gene i , and let d_i be the number of deletion events that target gene i . Then, $t_i = s_i + a_i - d_i$.*

► **Proposition 2.** *Let $T = \langle t_1, t_2, \dots, t_n \rangle$ be a CNP with $t_i \equiv t_{i+1} \pmod{p}$ and $t_i, t_{i+1} \neq 0$. Without loss of generality, assume that $t_i \leq t_{i+1}$. Let $T' = \langle t_1, t_2, \dots, t_{i-1}, t_{i+1}, \dots, t_n \rangle$ be the CNP obtained by removing gene i from T . Then, $\eta_p(T) = \eta_p(T')$.*

By applying Propositions 1 and 2 repeatedly on T we obtain a shorter CNP T' such that $\eta_p(T) = \eta_p(T')$. Moreover, the proofs of Propositions 1 and 2 are also constructive, enabling us to obtain a preduplication profile for T given a preduplication profile for T' . Therefore, we can now assume without loss of generality that for the input CNP T , $t_i > 0$ and $t_i \not\equiv t_{i+1} \pmod{p}$ for all i .

We now turn to showing properties of optimal preduplication profiles and transformations that will help us analyze the problems and reduce the search space. We say the CNT C for $U \rightarrow V$ is *disjoint* if no gene is both amplified and deleted. We first show that there exists an optimal disjoint transformation for $\eta_p(T)$. This reduces the number of solutions we need to consider.

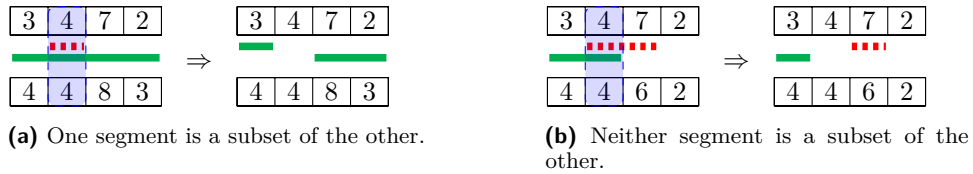
► **Lemma 1.** *For any optimal preduplication profile \hat{S} there exists an optimal transformation C for $p\hat{S} \rightarrow T$ such that C is disjoint.*

Proof. As noted above, we assume that \hat{S}, T have strictly positive values. We prove the claim by induction on the number of pairs of amplifications and deletions that overlap. In the base case, we show that one such pair can be replaced by two operations that do not overlap. Let $(i, j, 1)$ be an amplification event, and let $(k, \ell, -1)$ be a deletion event, such that $a = [i, j]$ and $d = [k, \ell]$ intersect.

- (a) If $a \subseteq d$, then we can replace the pair with two deletions, where each targets one of the two (contiguous) segments of $d \setminus a$ (Figure 2a). If $d \subseteq a$, vice versa. One or both of the subsequent segments may be empty; we can delete those operations because they do not affect any gene.
- (b) If $a \not\subseteq d$, then replace the pair with one amplification targeting $a \setminus d$ and one deletion targeting $d \setminus a$ (Figure 2b).

Since we assume \hat{S}, T are strictly positive, by Observation 1 we only require that the number of amplifications minus the number of deletions stays the same at each gene. Hence, validity is preserved for all genes, and we have eliminated a pair of overlapping amplification and deletion events.

For the inductive case, we can apply the same modification to eliminate a pair of overlapping amplification and deletion events. To complete this case, it only remains to show that we decreased the number of such overlapping pairs. This is easy to see because each new operation targets a segment which is a subset of the segment of some operation of the same type that it replaced, so any overlapping pairs after the modification would have been overlapping pairs before the modification. ◀



■ **Figure 2** Modifying pairs of overlapping operations to obtain a disjoint transformation.

In conjunction with Observation 1, Lemma 1 implies that given t_i , the value of \hat{s}_i determines how many events (either all amplifications or all deletions) target gene i in some optimal transformation. Conversely, the number of amplifications or deletions that affect each i uniquely determines the preduplication profile s_i . So, it suffices to find an optimal preduplication profile for $\eta_p(T)$, which induces an optimal disjoint transformation.

Finally, we bound the number of events in affecting each gene in any optimal disjoint transformation for $\eta_p(T)$, which also bounds how many preduplication profiles we need to consider to find an optimal one.

► **Lemma 2.** For all CNPs T and $p \geq 2$, $\eta_p(T) \leq np$.

Proof. Pick $S = \lceil T/p \rceil$, which may not be optimal in general. Then, for each of the n genes, we will need $p \lceil t_i/p \rceil - t_i \leq p$ deletion events. Assuming for an upper bound that each event targets exactly gene i , we can build a transformation for $pS \rightarrow T$ with $\leq np$ events. ◀

► **Corollary 1.** For any CNP T and integer $p \geq 2$, there is an optimal preduplication profile with an optimal disjoint transformation in which every gene is affected by at most np deletions or at most np amplifications.

3.2 CNP halving

In this section, we derive a simple algorithm for the CNP halving problem. We note that some cases of CNP halving are easy. For instance, if every value of T is even, then the CNP halving distance is zero because $\hat{S} = T/2$ is an optimal preduplication profile, and we need no CNOs at all since $2\hat{S} = T$. On the other hand if every value of T is odd, then the CNP halving distance is always 1 by setting $\hat{S} = \lceil T/2 \rceil = (T+1)/2$ as a preduplication profile and applying one CNO $(1, n, -1)$, which decrements by one. We will show here how to generalize this result to CNPs that contain both even and odd numbers.

To derive an algorithm for CNP halving, we make a few observations. We define an *odd run* in a CNP as a maximal-length contiguous segment of genes such that all of the gene values are odd. Similarly, an *even run* of a CNP is a maximal-length contiguous segment of genes such that all of the gene values are even. For example, in the CNP $\langle 1, 2, 4, 3, 5 \rangle$ there are two odd runs, $\langle 1 \rangle$ and $\langle 3, 5 \rangle$, and one even run $\langle 2, 4 \rangle$. We denote by $\text{odd}(V)$ ($\text{even}(V)$) the number of odd (even resp.) runs in a CNP V . We first show in the following proposition how each CNO affects the number of odd runs in a profile.

► **Proposition 3.** *Let V be a CNP and let $c = (\ell, h, w)$ be a CNO such that $\forall i, c(V)_i \geq 1$. Then, $\text{odd}(c(V)) - \text{odd}(V) \leq 1$.*

Proof. Denote by $\Delta_o = \text{odd}(c(V)) - \text{odd}(V)$ and $\Delta_e = \text{even}(c(V)) - \text{even}(V)$, the difference in odd and even runs respectively between $c(V)$ and V . Notice that amplifications and deletions affect the parity of each value in the CNP in the same way and therefore our proof is invariant to the operation type. We first observe that for any run $[i, j]$ fully contained within the target segment $[\ell, h]$, the parity of the run in $c(V)$ is toggled. Thus a fully contained odd run becomes an even run and vice versa. This enables us to separate our analysis into two cases: (a) operations that start and end in runs with the same parity, and (b) operations that start and end in runs with opposite parities (Figure 3ab). We divide each such case into five sub-cases: (I) the start and end of the operation are strictly within a run, (II) one side of the operation is bordering the next run and the other not, (III) one side of the operation is bordering the next run and the other is bordering the end of the profile, (IV) both sides of the operation are bordering the ends of the profile, and (V) both sides of the operation are bordering adjacent runs (Figure 3I-V).

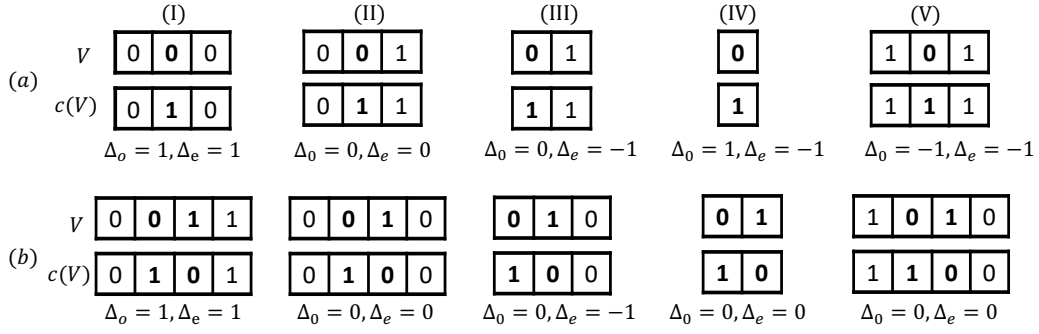
We see that in all cases analyzed, both $\Delta_o \leq 1$ and $\Delta_e \leq 1$ (Figure 3). Notice that although all operations in case (a) affect only even runs, the complement operations that affect only odd runs are symmetrical by replacing each odd run with an even run. We conclude that each operation can increase the number of odd runs in a profile by at most 1. ◀

We now use this property to solve the CNP halving problem in linear time.

► **Theorem 1.** $\eta_2(T) = \text{odd}(T)$ and $\hat{S} = \lceil T/2 \rceil$ is an optimal preduplicated CNP for a profile T .

Proof. First, we show that $\eta_2(T) \leq d(2\hat{S}, T) \leq \text{odd}(T)$. Note that $2\hat{s}_i = t_i$ if and only if t_i is even, so we need CNOs to correct the odd genes. We can do this with $\text{odd}(T)$ deletions, one for each odd run. Each deletion decrements the genes in one odd run, and the deletions target pairwise disjoint segments because each odd run is a maximal segment. Hence, $\eta_2(T) \leq d(2\hat{S}, T) \leq \text{odd}(T)$.

Next, we show that $\text{odd}(T) \leq \eta_2(T)$. Any duplicated profile $2S$ has no odd runs, while T has $\text{odd}(T)$ odd runs. On the hand, by Proposition 3, each CNO can increase the number of odd runs in a profile by at most 1. Therefore, it takes at least $\text{odd}(T)$ CNOs to transform a doubled profile $2S$ into T showing that $d(2S, T) \geq \text{odd}(T)$ for any CNP S . Specifically we have $\eta_2(T) = \min_S d(2S, T) \geq \text{odd}(T)$. ◀



■ **Figure 3** The affect of a CNO c on a CNP V in terms of the number of odd ($\Delta_o = \text{odd}(c(V)) - \text{odd}(V)$) and even ($\Delta_e = \text{even}(c(V)) - \text{even}(V)$) runs. Values in each profile represent the parity of the CN and the affected segment is marked in bold. Horizontal partition – operations that start and end in runs with (a) the same parity, (b) opposite parities. Vertical partition – the start and end of the operation are (I) strictly within a run, (II) one side bordering the next run and the other not, (III) one side bordering the next run and the other bordering the end of the profile, (IV) both sides bordering the ends of the profile, (V) both sides bordering adjacent runs (Figure 3(I-V)).

Notice that an optimal preduplication profile for the halving problem is not unique. For example, $\hat{S} = \langle 1 \rangle$ and $\hat{S} = \langle 2 \rangle$ are both optimal preduplication profiles for $T = \langle 3 \rangle$. One way to distinguish between optimal preduplication profiles is to look at the transformation they induce to the extant profile. For instance, Theorem 1 gives us the following corollary:

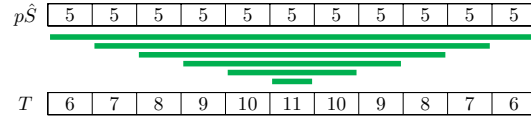
► **Corollary 2.** $\hat{S} = \lceil T/2 \rceil$ is an optimal preduplicated CNP for a profile T and the transformation $\hat{S} \rightarrow T$ uses only deletions.

On the other hand, in the following proposition we show how to select an optimal preduplication profile such that the transformation will use a maximum number of amplifications.

► **Proposition 4.** The maximum number of amplifications in a transformation from an optimal duplicated genome $2\hat{S}$ to T is $\text{odd}(T)$ if there is no $t_i = 1$ and $\text{odd}(T) - 1$ if there is some $t_i = 1$.

Proof. First, suppose there is no i such that $t_i = 1$. In this case we use $\hat{S} = \lceil T/2 \rceil$. We define a transformation $\hat{S} \rightarrow T$ that uses $\text{odd}(T)$ amplifications by applying one amplification on every odd run. For every even t_i value we have that $2\hat{s}_i = t_i$ and therefore this genes do not need to be modified. On the other hand, for odd t_i values we have that $2\hat{s}_i = t_i - 1$ and therefore one amplification on every odd runs transform $2\hat{S}$ into T .

Conversely, suppose there is a gene i having $t_i = 1$. For any CNP S , a duplicated CNP $2S$ has only values greater or equal to 2. Hence any transformation from a duplicated profile $2S$ to an extant profile T containing a value one must use at least one deletion. We define $\hat{S} = \lceil T/2 \rceil$ and show how to construct a transformation $\hat{S} \rightarrow T$ that uses one deletion and $\text{odd}(T) - 1$ amplifications. Let \hat{i} be the leftmost gene of the leftmost odd run and let \hat{j} be the rightmost gene of the rightmost odd run. We apply one deletion $(\hat{i}, \hat{j}, -1)$ which adjusts every odd value to its value in T . However, now we need to adjust the even runs in $[\hat{i}, \hat{j}]$. We do so by applying one amplification on each even run of T in $[\hat{i}, \hat{j}]$. Since $[\hat{i}, \hat{j}]$ covers all odd runs, there are $\text{odd}(T) - 1$ even runs in that segment. ◀



■ **Figure 4** A non-trivial example of the aliquoting distance $\eta_5(T)$ with an optimal preduplication profile $\hat{S} = \langle 1, 1, \dots, 1 \rangle$. For genes 5, 6 and 7 (with copy numbers 10, 11, and 10 in T , respectively), $\hat{s}_i \notin \{\lceil t_i/p \rceil, \lfloor t_i/p \rfloor\}$.

3.3 CNP aliquoting

In this section we derive an algorithm for the CNP aliquoting problem. While CNP halving is equivalent to CNP aliquoting with $p = 2$, generalizing the solution to CNP halving for $p > 2$ by rounding T/p up or down does not work with CNP aliquoting. Namely, there are instances T, p for the aliquoting problem where the genes of an optimal preduplication profile are not necessarily $\lceil t_i/p \rceil$ nor $\lfloor t_i/p \rfloor$. Moreover, even for genes where $t_i \equiv 0 \pmod p$, the optimal preduplication may not contain t_i/p in the i 'th gene. For example, let $p = 5$ and consider the following “triangle” CNP $T = \langle 6, 7, 8, 9, 10, 11, 10, 9, 8, 7, 6 \rangle$ (Figure 4). Using a preduplication profile $\lceil T/p \rceil = \langle 2, 2, 2, 2, 2, 3, 2, 2, 2, 2, 2 \rangle$, the CN distance $d(p\lceil T/p \rceil, T)$ is 12. Similarly, using a preduplication profile $\lfloor T/p \rfloor = \langle 1, 1, 1, 1, 2, 2, 2, 1, 1, 1, 1 \rangle$ the CN distance $d(p\lfloor T/p \rfloor, T)$ is 9. On the other hand, if we use the preduplication profile $\hat{S} = \langle 1, 1, \dots, 1 \rangle$, we have that $d(p\hat{S}, T) = 6$ using a “triangle” of amplification CNOs (Figure 4):

$$(1, 11, +1), (2, 10, +1), (3, 9, +1), \dots, (6, 6, +1)$$

Each successive amplification targets two fewer genes than the previous amplification. For this example $\eta_p(T) \leq 6$, which is also the best possible distance (proof omitted) and therefore \hat{S} is an optimal preduplication profile.

We begin by proposing a simple dynamic programming algorithm (Section 3.3.1). Then, we refine it with additional observations that enables us to improve the run time (Section 3.3.2).

3.3.1 An $O(n^3)$ dynamic programming algorithm

As we showed in Corollary 1, we can limit our search to finding an optimal preduplication profile \hat{S} where we have $\leq 2n$ choices for the value of each s_i . We explicitly enumerate these choices:

- We use $\leq np$ deletions at gene i to reach t_i from \hat{s}_i . In this case, $p\hat{s}_i \geq t_i$ and $p\hat{s}_i - t_i \leq np$. Let b_i^- be the “base” number of deletions, i.e., the minimum number of deletions needed to reach t_i from any choice of $p\hat{s}_i$.

$$b_i^- = p - t_i \pmod p = -t_i \pmod p$$

This equation holds because $p\hat{s}_i$ must be a multiple of p , and so we calculate the minimum number of deletions we need to reach t_i from any multiple of p . Now, gene i could be subject to either b_i^- deletions, $b_i^- + p$ deletions, etc. until we reach our bound of $\leq np$ deletions. Each of these choices for number of deletions corresponds to a unique value for \hat{s}_i . We denote the set of possible number of deletion of gene i as:

$$\mathcal{D}_i = \{b_i^- + kp \mid k \geq 0 \wedge b_i^- + kp \leq np\} = \{b_i^-, b_i^- + p, \dots, b_i^- + (n-1)p\}$$

Notice that $|\mathcal{D}_i| = n$.

18:10 CND Halving and Aliquoting

- Alternatively, we use $\leq np$ amplifications at gene i to reach t_i . However, unlike the deletion case where we could always increase \hat{s}_i in order to accommodate more deletions at a gene, for amplifications, we must decrease \hat{s}_i to add more amplifications. At the same time, we must have $\hat{s}_i \geq 1$ since $t_i \neq 0$. This bounds the maximum number of amplifications we can have at each gene. For example, if $t_i < p$ then we cannot reach t_i using amplifications.

Similar to deletions, we can also define a base number of amplification, corresponding to the minimum number of amplifications required to reach t_i for some choice of \hat{s}_i :

$$b_i^+ = t_i \bmod p$$

In addition, we define the set of choices for the number of amplifications of gene i as follows:

$$\mathcal{A}_i = \{b_i^+ + kp \mid k \geq 0 \wedge b_i^+ + kp \leq np \wedge t_i - b_i^+ - kp \geq p\}$$

Note that $p\hat{s}_i \geq p$ since $\hat{s}_i \geq 1$, which is where we derive the additional condition in the set. Therefore, $|\mathcal{A}_i| = \min\{n, \lfloor t_i/p \rfloor\}$ and in the case $t_i < p$ we have $\mathcal{A}_i = \emptyset$.

We define the following dynamic programming tables. For every $1 \leq i \leq n$ and every $x \in \mathcal{D}_i$, $D[i, x]$ will hold $\min_S d(pS, \langle t_1, \dots, t_i \rangle)$ such that $ps_i - x = t_i$, i.e. there is a disjoint transformation that applies exactly x deletions on the i 'th gene. Similarly, for every $1 \leq i \leq n$ and every $x \in \mathcal{A}_i$, $A[i, x]$ will hold $\min_S d(pS, \langle t_1, \dots, t_i \rangle)$ such that $ps_i + x = t_i$, i.e. there is a disjoint transformation that applies exactly x amplifications on the i 'th gene. There are at most $2n^2$ values in the arrays A and D put together, because there are $\leq n$ choices for x in each table and n for each coordinate i , each corresponds to a value for s_i . We now show how to calculate each value in the tables in $O(n)$ time, using the values for the previous gene. For completeness, we initialize the tables using $D[0, 0] = A[0, 0] = 0$.

► **Theorem 2.** *The following hold:*

$$D[i, x] = \min \left\{ \min_{y \in \mathcal{D}_{i-1}} \{D[i-1, y] + \max\{0, x - y\}\}, \min_{y \in \mathcal{A}_{i-1}} \{A[i-1, y] + x\} \right\} \quad (1)$$

$$A[i, x] = \min \left\{ \min_{y \in \mathcal{A}_{i-1}} \{A[i-1, y] + \max\{0, x - y\}\}, \min_{y \in \mathcal{D}_{i-1}} \{D[i-1, y] + x\} \right\} \quad (2)$$

Proof. We prove the result for $D[i, x]$; an analogous proof works for $A[i, x]$. We show our result by induction on i . For an empty CNP the property holds by our initialization $D[0, 0] = A[0, 0] = 0$. Assume now that the theorem holds up to $i - 1$.

First, we show that $D[i, x] \leq \text{RHS}(1)$, the right hand side of Equation 1:

$$\min_{y \in \mathcal{D}_{i-1}} \{D[i-1, y] + \max\{0, x - y\}\}, \min_{y \in \mathcal{A}_{i-1}} \{A[i-1, y] + x\}.$$

This is because we can assemble a solution for $D[i, x]$ using the following three cases (Figure 5):

- (a) We select $y \in \mathcal{D}_{i-1}$ such that $y \geq x$ and look at the transformation corresponding to $D[i-1, y]$. In this case, we do not need to add any new operations, because we can pick x arbitrary deletions that target coordinate $i - 1$ and extend them to also target coordinate i (Figure 5a). Hence,

$$D[i, x] \leq \min_{\substack{y \in \mathcal{D}_{i-1} \\ y \geq x}} \{D[i-1, y]\}$$

- (b) We select $y \in \mathcal{D}_{i-1}$ such that $y \leq x$ and look at the transformation corresponding to $D[i-1, y]$. Then, we extend all of the deletions at $i-1$ to also affect coordinate i , and add $x-y$ new deletions at coordinate i (Figure 5b). We have,

$$D[i, x] \leq \min_{\substack{y \in \mathcal{D}_{i-1} \\ y \leq x}} \{D[i-1, y] + (x-y)\}$$

- (c) Finally, we select $y \in \mathcal{A}_{i-1}$ and look at the transformation corresponding to $A[i-1, y]$. We always need to introduce x new deletions because we cannot make use of any of the existing operations at $i-1$ since they are all amplifications and the transformation we are looking for is disjoint (Figure 5c). Therefore,

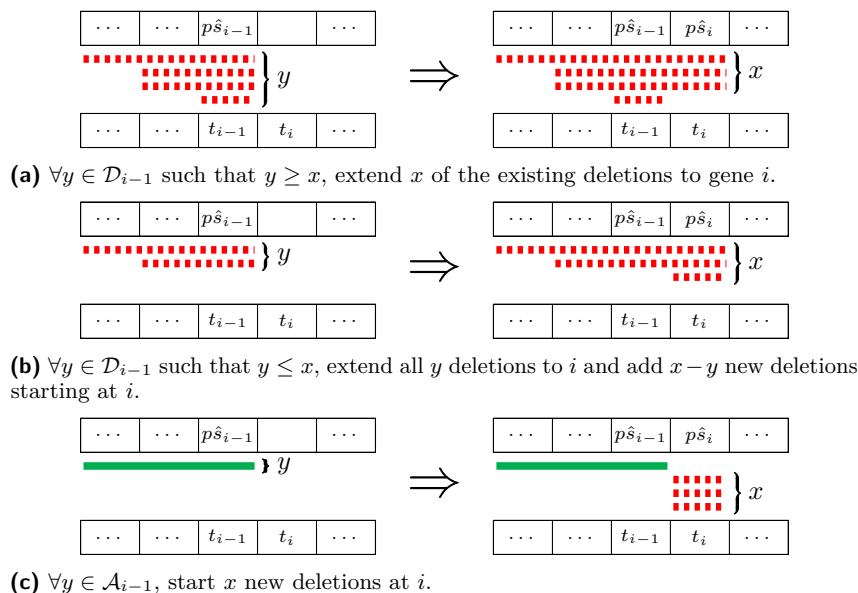
$$D[i, x] \leq \min_{y \in \mathcal{A}_{i-1}} \{A[i-1, y] + x\}$$

Taking the min of all three cases gives us RHS(1).

To complete the proof, suppose for contradiction that $D[i, x] < \text{RHS}(1)$. Then, there exists some optimal transformation for the i 'th prefix where coordinate i is affected by x deletions and coordinate $i-1$ is affected by y^* operations (either amplifications or deletions).

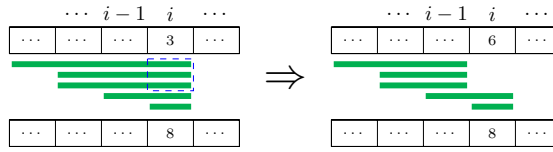
- If $i-1$ is affected by y^* amplifications, then removing the x deletions at i gives a copy number transformation for the $(i-1)$ th prefix with y^* amplifications at coordinate $i-1$. But, $D[i, x] - x < \min_y \{A[i-1, y] + x\} - x = \min_y \{A[i-1, y]\}$ which is a contradiction.
- Similarly, if $i-1$ is affected by y^* deletions, we can remove either zero or $x-y^*$ deletions to get a copy number transformation for the $(i-1)$ th prefix with y^* deletions at coordinate $i-1$ that is better than $D[i-1, y^*]$, which is a contradiction.

In all cases we get a contradiction, which completes the proof. ◀



■ **Figure 5** Illustration of different cases in the dynamic programming algorithm to compute $D[i, x]$ (Theorem 2).

Tables D and A can be populated in time $O(n^3)$, by trying each of the $O(n)$ values for $D[i-1, y]$ and $A[i-1, y]$ in order to calculate $D[i, x]$ or $A[i, x]$. Once all the values are calculated, we can find the aliquoting distance $\eta_p(T) = \min \{ \min_x \{D[n, x]\}, \min_x \{A[n, x]\} \}$,



■ **Figure 6** Trimming $p = 3$ events to show that $A[i, x]$ is non-decreasing in x .

which takes time $O(n)$. We also record the argmin that we use to populate each entry of D and A , which allows us to backtrack in order to compute the optimal pre-duplication profile. Hence, the entire algorithm works in time $O(n^3)$ and $O(n^2)$ space. Notice however that due to Propositions 1 and 2, the length of the profile n that we solve the problem for can be smaller than the original input profile.

3.3.2 An improved $O(n^2)$ dynamic programming algorithm

Here, we show that we only need to check $O(1)$ possibilities to calculate each entry of D and A , which reduces the run time of the algorithm to $O(n^2)$, the number of entries in both tables. First, we note that $D[i, x]$ and $A[i, x]$ are non-decreasing as we increase the number of operations x .

► **Lemma 3.** *If $\{x, x + p\} \subseteq \mathcal{D}_i$, then $D[i, x + p] \geq D[i, x]$ (similarly, if $\{x, x + p\} \subseteq \mathcal{A}_i$, $A[i, x + p] \geq A[i, x]$).*

Proof. We prove the statement for A (a similar proof works for D). Suppose that $\{x, x + p\} \subseteq \mathcal{A}_i$. Then, we show we can take a disjoint transformation for the i 'th prefix that has $A[i, x + p]$ events, including $x + p$ amplifications at gene i , and modify it to get a transformation that has $\leq A[i, x + p]$ events but has x amplifications at gene i . This implies that $A[i, x] \leq A[i, x + p]$.

To do so, we can pick p arbitrary amplifications that target gene i , and shrink each of them by decrementing the end index of their segments, so that they no longer target gene i , but that they still target all of the genes other than i that it targeted before (Figure 6). Note that this implies that we increment the preduplication gene \hat{s}_i , because we have shown that the number of events at a gene implies the value of the preduplication gene there, and vice versa. In addition, since we are only considering the i 'th prefix, this modification preserves the contiguity of every amplification. ◀

Moreover, we now show we can bound the increase in aliquoting distance when we increase the number of deletions/amplifications on a gene.

► **Lemma 4.** *If $\{x, x + p\} \subseteq \mathcal{A}_i$, then $A[i, x + p] - A[i, x] \leq p$ (and similarly, $D[i, x + p] - D[i, x] \leq p$).*

Proof. We prove the statement for A (a similar proof works for D). We can always add p new operations to $A[i, x]$ to get a solution for $A[i, x + p]$ if $x + p \in \mathcal{A}_i$, so $A[i, x + p] \leq A[i, x] + p$. ◀

Using these results, we improve the performance of the dynamic programming algorithm. First, at the end of the algorithm we return either $D[n, b_n^-]$ or $A[n, b_n^+]$ (base number of deletions or amplifications) instead of checking each entry in $D[n, \dots]$ and $A[n, \dots]$, since larger number of operations at coordinate n will have at least as large aliquoting distances. However, this does not improve the overall asymptotic time complexity of the algorithm. To achieve our improved time complexity, we show that we only need to try $O(1)$ possibilities

to calculate each entry in D and A . To accomplish this, we prove that the minimum of $O(n)$ y values in RHS(1) and RHS(2) in Theorem 2 can be expressed as the minimum of $O(1)$ values. To that end we define the following functions for every i and x :

$$\begin{aligned} y_1(x) &:= \min\{y \in \mathcal{D}_{i-1} \mid y \geq x\}; & y_2(x) &:= \max\{y \in \mathcal{D}_{i-1} \mid y \leq x\}; \\ y'_1(x) &:= \min\{y \in \mathcal{A}_{i-1} \mid y \geq x\}; & y'_2(x) &:= \max\{y \in \mathcal{A}_{i-1} \mid y \leq x\}; \end{aligned}$$

► **Theorem 3.** *The following hold:*

$$\begin{aligned} D[i, x] &= \min\{D[i-1, y_1(x)], D[i-1, y_2(x)] + (x - y_2(x)), A[i-1, b_{i-1}^+] + x\} \\ A[i, x] &= \min\{A[i-1, y'_1(x)], A[i-1, y'_2(x)] + (x - y'_2(x)), D[i-1, b_{i-1}^-] + x\} \end{aligned}$$

Proof. We prove the result for $D[i, x]$ with $y_1(x)$ and $y_2(x)$; an analogous proof works for $A[i, x]$ together with $y'_1(x)$ and $y'_2(x)$. We rearrange equation (1) in Theorem 2 as follows:

$$\begin{aligned} D[i, x] &= \min \left\{ \min_{y \in \mathcal{D}_{i-1}} \{D[i-1, y] + \max\{0, x - y\}\}, \min_{y \in \mathcal{A}_{i-1}} \{A[i-1, y] + x\} \right\} \\ &= \min \left\{ \min_{\substack{y \in \mathcal{D}_{i-1} \\ y \geq x}} \{D[i-1, y]\}, \min_{\substack{y \in \mathcal{D}_{i-1} \\ y \leq x}} \{D[i-1, y] + (x - y)\}, \min_{y \in \mathcal{A}_{i-1}} \{A[i-1, y] + x\} \right\} \end{aligned} \quad (3)$$

Now, we show that each of the three inner min expressions in equation (3) can be replaced with a single term (Figure 7).

(a) If there exists at least one $y \in \mathcal{D}_{i-1}$ such that $y \geq x$ (Figure 7a), then

$$\min_{\substack{y \in \mathcal{D}_{i-1} \\ y \geq x}} \{D[i-1, y]\} = D[i-1, y_1(x)]$$

This is because the non-decreasing property from Lemma 3 implies that we can check the entry for the minimum y to get the smallest value. So, we can replace the first min terms in equation (3) with $D[i-1, y_1(x)]$.

(b) If there exists $y \in \mathcal{D}_{i-1}$ such that $y \leq x$ (Figure 7b), then

$$\min_{\substack{y \in \mathcal{D}_{i-1} \\ y \leq x}} \{D[i-1, y] + (x - y)\} = D[i-1, y_2] + (x - y_2) \quad (4)$$

Let $y_2 = y_2(x)$ for conciseness. Suppose by contradiction there is some other value in \mathcal{D}_{i-1} (which we can express as $y_2 - kp$ for $k \geq 1$) minimizes equation (4): $D[i-1, y_2 - kp] + (x - (y_2 - kp)) < D[i-1, y_2] + (x - y_2)$. Rearranging, we get

$$D[i-1, y_2] > D[i-1, y_2 - kp] + kp$$

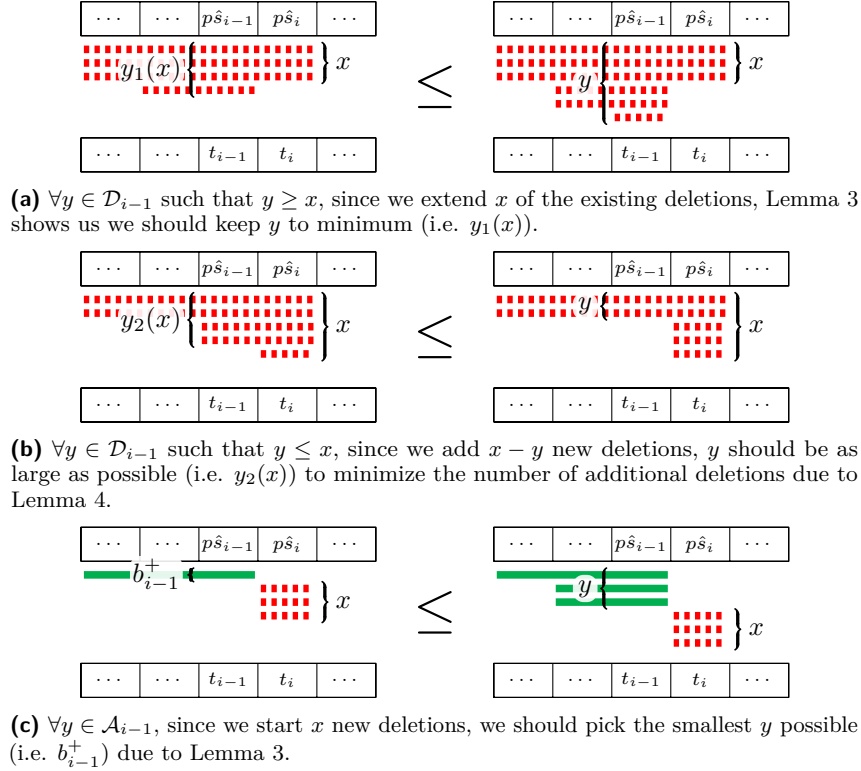
which contradicts Lemma 4. Hence, we can replace the second min terms in equation (3) with $D[i-1, y_2] + (x - y_2)$.

(c) Finally, the following equation hold (Figure 7c):

$$\min_{y \in \mathcal{A}_{i-1}} \{A[i-1, y] + x\} = \min_{y \in \mathcal{A}_{i-1}} \{A[i-1, y]\} + x = A[i-1, b_{i-1}^+] + x$$

This follows from Lemma 3; we pick the smallest y to get some min of $A[i-1, \cdot]$. So, we can replace the third min terms in equation (3) with $A[i-1, b_{i-1}^+] + x$. ◀

Since y_1, y'_1, y_2, y'_2 are computable in $O(1)$ time (Supplemental Proposition S5), we can leverage Theorem 3 to populate each table entry in $O(1)$ time, because we can figure out which three values we need to check using y_j, y'_j and take the minimum of these values. In conclusion, we can populate D and A in $O(n^2)$ time $O(n^2)$ space. However, if we are just interested in calculating the aliquoting distance without finding an optimal preduplication profile, we can use only $O(n)$ space since we simply need $D[i-1, \cdot]$ and $A[i-1, \cdot]$ to compute $D[i, \cdot]$ and $A[i, \cdot]$.



■ **Figure 7** The cases for computing $D[i, x]$ for the improved dynamic programming algorithm (Theorem 3).

4 Experiments

We evaluated our halving and aliquoting algorithms on simulated CNPs in order to assess their ability to recover preduplication profiles. In each simulation, we generate a random preduplication profile $S \in \{1, \dots, 5\}^n$, multiply each entry by p and then apply k amplifications and deletions to create an extant profile T . We then use the aliquoting algorithm on T to estimate the aliquoting distance $\eta_p(T)$ and find a preduplication profile \hat{S} . We run simulations for profile lengths $n \in \{100, 200, 300\}$, polyploidy $p \in \{2, 3, 4\}$ and $k \in \{5, 10, 15\}$ events after polyploidization. To simulate events, we apply k random CNOs with uniform length and position, and with a ratio of deletions/amplifications of 3 to 1. We also make sure that the profiles pS and T generated have no zeros. We implemented the halving and aliquoting algorithms in Python 3, and we ran the simulations on a Thinkpad T470 computer with an Intel i7-7600U processor running Linux 5.11.10. For each configuration n, p, k , 100 instances were simulated.

We use several metrics to measure the algorithm performance. First, we evaluate $d(S, \hat{S})$ to measure how close the aliquoting preduplication profile is to the actual preduplication profile. However, there may exist multiple duplicated profiles with the same copy number distance from T . Therefore, we also compare $\Delta\eta_p(T) := d(pS, T) - \eta_p(T) = d(pS, T) - d(p\hat{S}, T)$, to measure how close the estimated aliquoting distance was from the actual copy number distance between the true duplicated profile pS and T . This is a measure of how accurately we can recover the number of events that have occurred after polyploidization. Finally, we assess the effective run time of our algorithms as we increase the profile sizes.

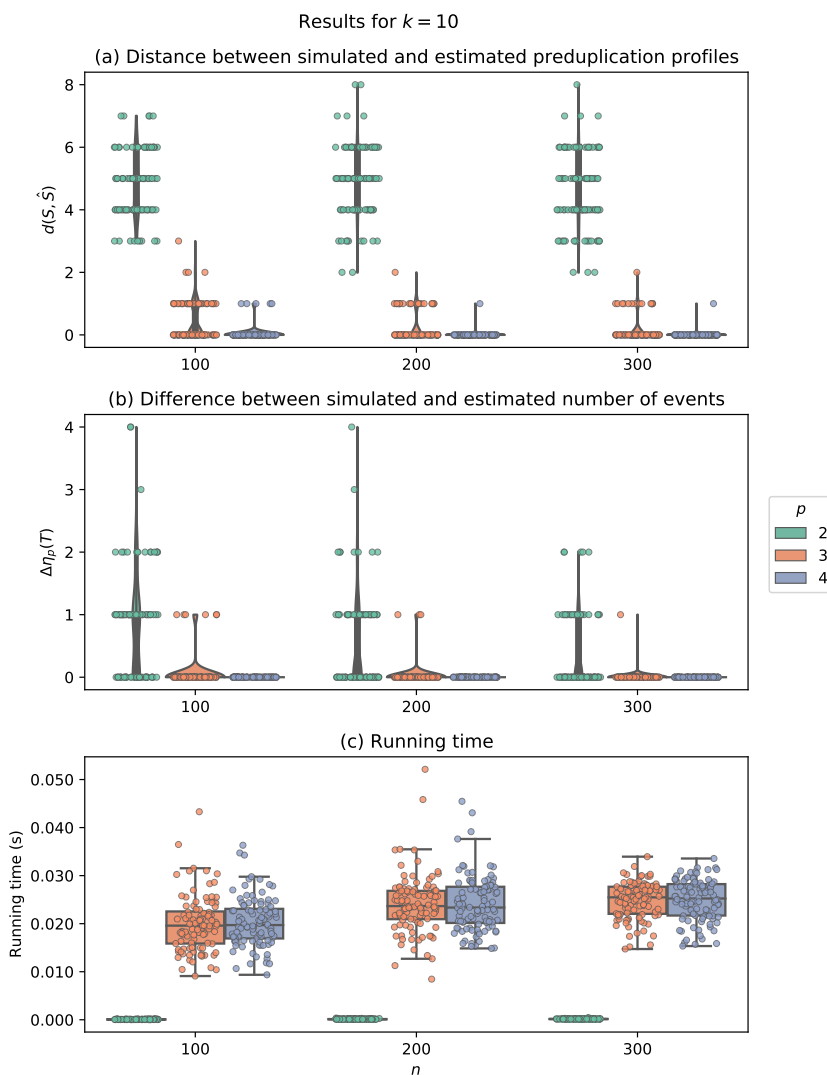
We find that CNP aliquoting is more accurate than halving (Figures 8, S2 and S2). When aliquoting $p \geq 3$, we have $\Delta\eta_p(T) \leq 2$ and $d(S, \hat{S}) \leq 4$ for every simulated CNP, which suggests that not only are we able to estimate the number of post-polyploidization events, but we can also recover a very close profile to the original preduplication profile S by solving the aliquoting problem. On the other hand, for halving ($p = 2$), $\Delta\eta_2(T)$ and $d(S, \hat{S})$ are larger, which is likely because there are many different preduplication profiles for halving. For instance, in Corollary 2 we show that there exists a preduplication profile that maximizes the number of deletions in the optimal transformation, but in Proposition 4 we prove that there exists a different preduplication profile that maximizes the number of amplifications. Notice however, that as the length of the profile increases, we do estimate the number of post-duplication events more accurately. This is partially because with larger CNPs, there is a smaller chance for events to cancel one another.

Finally, we measured the running time of our halving and aliquoting algorithms from Theorem 1 and Theorem 3, respectively (Figures 8c, S2c, and S3c). Notice that our implementation of the aliquoting algorithm contracts runs that have the same value modulo p , using Proposition 2. This preprocessing is done in linear time and does not affect the overall worst-case asymptotic time complexity. However, we find that in practice, it improves the run time significantly. Although the aliquoting dynamic programming algorithm is quadratic in the worst case, we see that on simulated profiles, the increase in running time is much lower. This is because the effective profile length for which we solve the problem depends on the number of runs modulo p and not the original length of the profile.

5 Discussion

In this paper, we formulate and solve the genome halving and genome aliquoting problems for CNPs under the CNT model. For the halving distance we derive a simple linear time algorithm and show how to obtain preduplicated genomes having a transformation with maximum number of deletions or amplifications. For the CNP aliquoting distance we derive a quadratic time dynamic programming algorithm by showing several properties of an optimal preduplication profile and carefully analyzing aliquoting sub problems. We further note that with $O(n)$ time preprocessing and postprocessing, the latter algorithm is quadratic in the number of distinct runs modulo p which can be effectively quite lower than the length of the profile, as we show on simulated CNPs. Finally, our simulations show that we are able accurately estimate the number of events post-duplication.

There are several directions for further research. First, for some CNPs there be many optimal preduplication profiles and our algorithm will not distinguish between these solutions, selecting one arbitrarily based on the implementation. It is therefore interesting to further explore the space of optimal preduplication profiles. Similar ambiguity in selecting a preduplication genome arises in other rearrangement distances, and one solution is to use an out-group in order to further constrain the preduplication genome. This modification, called



■ **Figure 8** Simulation results for using $k = 10$ events after duplication. (a) $d(S, \hat{S})$ - the distance between simulated and estimated preduplication profiles. (b) $\Delta\eta_p(T)$ - the difference between the simulated and estimated number of events after duplication. (c) running time in seconds.

the *guided genome halving problem*, seeks to find a preduplication genome that minimizes the distance to a given out-group plus the distance from the duplicated genome to the extant genome [47]. An alternative solution which might be more biologically relevant in some cases such as cancer samples is to extend the copy number distance to also include a WGD event [30]. In this case, the copy number distance between a pair of profiles would be the minimum number of amplifications, deletions and WGDs that transform profile into the other. Third, these algorithms could be extended to address the issues of normal cell admixture and subclonality that arise in analyzing cancer sequencing data as has been previously done for CNPs [11, 44]. Finally, applying our algorithms to real cancer genomes with high tumor ploidy could help identify genomes with strong evidence for polyploidization during cancer evolution and provide new insights into highly aneuploid cancer genomes.

References

- 1 Max A. Alekseyev and Pavel A. Pevzner. Genome halving problem revisited. In Kamal Lodaya and Meena Mahajan, editors, *FSTTCS 2004: Foundations of Software Technology and Theoretical Computer Science*, pages 1–15, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.
- 2 Max A Alekseyev and Pavel A Pevzner. Colored de Bruijn graphs and the genome halving problem. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 4(1):98–107, 2007. doi:10.1109/TCBB.2007.1002.
- 3 Anne Bergeron, Julia Mixtacki, and Jens Stoye. A unifying view of genome rearrangements. In Philipp B ucher and Bernard M.E. Moret, editors, *Proc. Workshop on Algorithms in Bioinformatics*, volume 4175 of *LNCS*, pages 163–173. Springer, 2006. doi:10.1007/11851561_16.
- 4 Craig M Bielski, Ahmet Zehir, Alexander V Penson, Mark TA Donoghue, Walid Chatila, Joshua Armenia, Matthew T Chang, Alison M Schram, Philip Jonsson, Chaitanya Bandlamudi, et al. Genome doubling shapes the evolution and prognosis of advanced cancers. *Nature genetics*, 50(8):1189–1195, 2018.
- 5 John E Bowers, Brad A Chapman, Junkang Rong, and Andrew H Paterson. Unravelling angiosperm genome evolution by phylogenetic analysis of chromosomal duplication events. *Nature*, 422(6930):433, 2003.
- 6 Peter J. Campbell, Gad Getz, Jan O. Korbel, Joshua M. Stuart, Jennifer L. Jennings, Lincoln D. Stein, Marc D. Perry, Hardeep K. Nahal-Bose, B. F. Francis Ouellette, Constance H. Li, Esther Rheinbay, G. Petur Nielsen, Dennis C. Sgroi, Chin-Lee Wu, William C. Faquin, Vikram Deshpande, Paul C. Boutros, Alexander J. Lazar, Katherine A. Hoadley, David N. Louis, L. Jonathan Dursi, Christina K. Yung, Matthew H. Bailey, Gordon Saksena, Keiran M. Raine, Ivo Buchhalter, Kortine Kleinheinz, Matthias Schlesner, Junjun Zhang, Wenyi Wang, David A. Wheeler, Li Ding, Jared T. Simpson, Brian D. O’Connor, Sergei Yakneen, Kyle Ellrott, Naoki Miyoshi, Adam P. Butler, Romina Royo, Solomon I. Shorser, Miguel Vazquez, Tobias Rausch, Grace Tiao, Sebastian M. Waszak, Bernardo Rodriguez-Martin, Suyash Shringarpure, Dai-Ying Wu, German M. Demidov, Olivier Delaneau, Shuto Hayashi, Seiya Imoto, Nina Habermann, Ayellet V. Segre, Erik Garrison, Andy Cafferkey, Eva G. Alvarez, Jos eMar ia Heredia-Genestar, Francesc Muyas, Oliver Drechsel, Alicia L. Bruzos, Javier Temes, Jorge Zamora, Adrian Baez-Ortega, Hyung-Lae Kim, R. Jay Mashl, Kai Ye, Anthony DiBiase, Kuan-lin Huang, Ivica Letunic, Michael D. McLellan, Steven J. Newhouse, Tal Shmaya, Sushant Kumar, David C. Wedge, Mark H. Wright, Venkata D. Yellapantula, Mark Gerstein, Ekta Khurana, Tomas Marques-Bonet, Arcadi Navarro, Carlos D. Bustamante, Reiner Siebert, Hidewaki Nakagawa, Douglas F. Easton, Stephan Ossowski, Jose M. C. Tubio, Francisco M. De La Vega, Xavier Estivill, Denis Yuen, George L. Mihaiescu, Larsson Omberg, Vincent Ferretti, Radhakrishnan Sabarinathan, Oriol Pich, Abel Gonzalez-Perez, Amaro Taylor-Weiner, Matthew W. Fittall, Jonas Demeulemeester, Maxime Tarabichi, Nicola D. Roberts, Peter Van Loo, Isidro Cort es-Ciriano, Lara Urban, Peter Park, Bin Zhu, Esa Pitk anen, Yilong Li, Natalie Saini, Leszek J. Klimczak, Joachim Weischenfeldt, Nikos Sidiropoulos, Ludmil B. Alexandrov, Raquel Rabionet, Georgia Escaramis, Mattia Bosio, Aliaksei Z. Holik, Hana Susak, Aparna Prasad, Serap Erkek, Claudia Calabrese, Benjamin Raeder, Eoghan Harrington, Simon Mayes, Daniel Turner, Sissel Juul, Steven A. Roberts, Lei Song, Roelof Koster, Lisa Mirabello, Xing Hua, Tomas J. Tanskanen, Marta Tojo, Jieming Chen, Lauri A. Aaltonen, Gunnar R atsch, Roland F. Schwarz, Atul J. Butte, Alvis Brazma, Stephen J. Chanock, Nilanjan Chatterjee, Oliver Stegle, Olivier Harismendy, G. Steven Bova, Dmitry A. Gordenin, David Haan, Lina Sieverling, Lars Feuerbach, Don Chalmers, Yann Joly, Bartha Knoppers, Fruzsina Moln ar-G abor, Mark Phillips, Adrian Thorogood, David Townend, Mary Goldman, Nuno A. Fonseca, Qian Xiang, Brian Craft, Elena Pi eiro-Y a nez, Alfonso Mu oz, Robert Petryszak, Anja F ullgrabe, Fatima Al-Shahrour, Maria Keays, David Haussler, John Weinstein, Wolfgang Huber, Alfonso Valencia, Irene Papatheodorou, Jingchun Zhu, Yu Fan,

- David Torrents, Matthias Bieg, Ken Chen, Zechen Chong, Kristian Cibulskis, Roland Eils, Robert S. Fulton, Josep L. Gelpi, Santiago Gonzalez, Ivo G. Gut, Faraz Hach, Michael Heinold, Taobo Hu, Vincent Huang, Barbara Hutter, Natalie Jäger, Jongsun Jung, Yogesh Kumar, Christopher Lalansingh, Ignaty Leshchiner, Dimitri Livitz, Eric Z. Ma, Yosef E. Maruvka, Ana Milovanovic, Morten Muhlig Nielsen, Nagarajan Paramasivam, Jakob Skou Pedersen, Montserrat Puiggròs, S. Cenk Sahinalp, Iman Sarrafi, Chip Stewart, Miranda D. Stobbe, Jeremiah A. Wala, Jiayin Wang, Michael Wendl, Johannes Werner, Zhenggang Wu, Hong Xue, Takafumi N. Yamaguchi, Venkata Yellapantula, Brandi N. Davis-Dusenbery, Robert L. Grossman, Youngwook Kim, Michael C. Heinold, Jonathan Hinton, David R. Jones, Andrew Menzies, Lucy Stebbings, Julian M. Hess, Mara Rosenberg, Andrew J. Dunford, Manaswi Gupta, Marcin Imielinski, Matthew Meyerson, Rameen Beroukhim, Jüri Reimand, Priyanka Dhingra, Francesco Favero, Stefan Dentro, Jeff Wintersinger, Vasilisa Rudneva, Ji Wan Park, Eun Pyo Hong, Seong Gu Heo, André Kahles, Kjong-Van Lehmann, Cameron M. Soulette, Yuichi Shiraishi, Fenglin Liu, Yao He, Deniz Demircioğlu, Natalie R. Davidson, Liliana Greger, Siliang Li, Dongbing Liu, Stefan G. Stark, Fan Zhang, Samirkumar B. Amin, Peter Bailey, Aurélien Chateigner, Milana Frenkel-Morgenstern, Yong Hou, Matthew R. Huska, Helena Kilpinen, Fabien C. Lamaze, Chang Li, Xiaobo Li, Xinyue Li, Xingmin Liu, Maximillian G. Marin, Julia Markowski, Tannistha Nandi, Akinyemi I. Ojesina, Qiang Pan-Hammarström, Peter J. Park, Chandra Sekhar Pedamallu, Hong Su, Patrick Tan, Bin Tean Teh, Jian Wang, Heng Xiong, Chen Ye, Christina Yung, Xiuqing Zhang, Liangtao Zheng, Shida Zhu, Philip Awadalla, Chad J. Creighton, Kui Wu, Huanming Yang, Jonathan Göke, Zemin Zhang, Angela N. Brooks, Matthew W. Fittall, Iñigo Martincorena, Carlota Rubio-Perez, Malene Juul, Steven Schumacher, Ofer Shapira, David Tamborero, Loris Mularoni, Henrik Hornshøj, Jordi Deu-Pons, Ferran Muiños, Johanna Bertl, Qianyun Guo, and The ICGC/TCGA Pan-Cancer Analysis of Whole Genomes Consortium. Pan-cancer analysis of whole genomes. *Nature*, 578(7793):82–93, 2020. doi:10.1038/s41586-020-1969-6.
- 7 Scott L Carter, Kristian Cibulskis, Elena Helman, Aaron McKenna, Hui Shen, Travis Zack, Peter W Laird, Robert C Onofrio, Wendy Winckler, Barbara A Weir, et al. Absolute quantification of somatic dna alterations in human cancer. *Nature biotechnology*, 30(5):413, 2012.
 - 8 Salim Akhter Chowdhury, Stanley E Shackney, Kerstin Heselmeyer-Haddad, Thomas Ried, Alejandro A Schäffer, and Russell Schwartz. Algorithms to model single gene, single chromosome, and whole genome copy number changes jointly in tumor phylogenetics. *PLoS computational biology*, 10(7):e1003740, 2014.
 - 9 Stefan C. Dentro, Ignaty Leshchiner, Kerstin Haase, Maxime Tarabichi, Jeff Wintersinger, Amit G. Deshwar, Kaixian Yu, Yulia Rubanova, Geoff Macintyre, Jonas Demeulemeester, Ignacio Vázquez-García, Kortine Kleinheinz, Dimitri G. Livitz, Salem Malikic, Nilgun Donmez, Subhajit Sengupta, Pavana Anur, Clemency Jolly, Marek Cmero, Daniel Rosebrock, Steven E. Schumacher, Yu Fan, Matthew Fittall, Ruben M. Drews, Xiaotong Yao, Thomas B. K. Watkins, Juhee Lee, Matthias Schlesner, Hongtu Zhu, David J. Adams, Nicholas McGranahan, Charles Swanton, Gad Getz, Paul C. Boutros, Marcin Imielinski, Rameen Beroukhim, S. Cenk Sahinalp, Yuan Ji, Martin Peifer, Inigo Martincorena, Florian Markowetz, Ville Mustonen, Ke Yuan, Moritz Gerstung, Paul T. Spellman, Wenyi Wang, Quaid D. Morris, David C. Wedge, Peter Van Loo, Stefan C. Dentro, Amit G. Deshwar, Santiago Gonzalez, David J. Adams, Paul C. Boutros, David D. Bowtell, Peter J. Campbell, Shaolong Cao, Elizabeth L. Christie, Yupeng Cun, Kevin J. Dawson, Ruben M. Drews, Roland Eils, Dale W. Garsed, Gavin Ha, Lara Jerman, Henry Lee-Six, Dimitri G. Livitz, Thomas J. Mitchell, Layla Oesper, Myron Peto, Benjamin J. Raphael, S. Cenk Sahinalp, Adriana Salcedo, Steven E. Schumacher, Ruihan Shi, Seung Jun Shin, Lincoln D. Stein, Oliver Spiro, Shankar Vembu, David A. Wheeler, Tsun-Po Yang, Quaid D. Morris, Paul T. Spellman, and David C. Wedge. Characterizing genetic intra-tumor heterogeneity across 2,658 human cancer genomes. *Cell*, 2021. doi:10.1016/j.cell.2021.03.009.

- 10 Li Ding, Timothy J Ley, David E Larson, Christopher A Miller, Daniel C Koboldt, John S Welch, Julie K Ritchey, Margaret A Young, Tamara Lamprecht, Michael D McLellan, Joshua F McMichael, John W Wallis, Charles Lu, Dong Shen, Christopher C Harris, David J Dooling, Robert S Fulton, Lucinda L Fulton, Ken Chen, Heather Schmidt, Joelle Kalicki-Veizer, Vincent J Magrini, Lisa Cook, Sean D McGrath, Tammi L Vickery, Michael C Wendl, Sharon Heath, Mark A Watson, Daniel C Link, Michael H Tomasson, William D Shannon, Jacqueline E Payton, Shashikant Kulkarni, Peter Westervelt, Matthew J Walter, Timothy A Graubert, Elaine R Mardis, Richard K Wilson, and John F DiPersio. Clonal evolution in relapsed acute myeloid leukaemia revealed by whole-genome sequencing. *Nature*, 481(7382):506–10, 2012. doi:10.1038/nature10738.
- 11 Mohammed El-Kebir, Benjamin J Raphael, Ron Shamir, Roded Sharan, Simone Zaccaria, Meirav Zehavi, and Ron Zeira. Complexity and algorithms for copy-number evolution problems. *Algorithms for Molecular Biology*, 12(1):13, 2017.
- 12 Nadia El-Mabrouk, Joseph H. Nadeau, and David Sankoff. Genome halving. In Martin Farach-Colton, editor, *Proc. Combinatorial Pattern Matching*, pages 235–250, Berlin, Heidelberg, 1998. Springer Berlin Heidelberg.
- 13 Pedro Feijão and Joao Meidanis. SCJ: a breakpoint-like distance that simplifies several rearrangement problems. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 8(5):1318–29, 2011. doi:10.1109/TCBB.2011.34.
- 14 Guillaume Fertin, Anthony Labarre, Irena Rusu, Stéphane Vialette, and Eric Tannier. *Combinatorics of genome rearrangements*. MIT press, 2009.
- 15 Andrej Fischer, Ignacio Vázquez-García, Christopher JR Illingworth, and Ville Mustonen. High-definition reconstruction of clonal composition in cancer. *Cell reports*, 7(5):1740–1752, 2014.
- 16 Moritz Gerstung, Clemency Jolly, Ignaty Leshchiner, Stefan C. Dentre, Santiago Gonzalez, Daniel Rosebrock, Thomas J. Mitchell, Yulia Rubanova, Pavana Anur, Kaixian Yu, Maxime Tarabichi, Amit Deshwar, Jeff Wintersinger, Kortine Kleinheinz, Ignacio Vázquez-García, Kerstin Haase, Lara Jerman, Subhajit Sengupta, Geoff Macintyre, Salem Malikic, Nilgun Donmez, Dimitri G. Livitz, Marek Cmero, Jonas Demeulemeester, Steven Schumacher, Yu Fan, Xiaotong Yao, Juhee Lee, Matthias Schlesner, Paul C. Boutros, David D. Bowtell, Hongtu Zhu, Gad Getz, Marcin Imielinski, Rameen Beroukhim, S. Cenk Sahinalp, Yuan Ji, Martin Peifer, Florian Markowitz, Ville Mustonen, Ke Yuan, Wenyi Wang, Quaid D. Morris, Stefan C. Dentre, Amit G. Deshwar, David J. Adams, Paul C. Boutros, David D. Bowtell, Peter J. Campbell, Shaolong Cao, Elizabeth L. Christie, Yupeng Cun, Kevin J. Dawson, Ruben M. Drews, Roland Eils, Matthew Fittall, Dale W. Garsed, Gavin Ha, Henry Lee-Six, Dimitri G. Livitz, Inigo Martincorena, Thomas J. Mitchell, Layla Oesper, Myron Peto, Benjamin J. Raphael, S. Cenk Sahinalp, Adriana Salcedo, Ruian Shi, Seung Jun Shin, Oliver Spiro, Lincoln D. Stein, Shankar Vembu, David A. Wheeler, Tsun-Po Yang, Quaid D. Morris, Paul T. Spellman, David C. Wedge, Peter Van Loo, Paul T. Spellman, David C. Wedge, PCAWG Evolution & Heterogeneity Working Group, and PCAWG Consortium. The evolutionary history of 2,658 cancers. *Nature*, 578(7793):122–128, 2020. doi:10.1038/s41586-019-1907-7.
- 17 Gavin Ha, Andrew Roth, Jaswinder Khattra, Julie Ho, Damian Yap, Leah M Prentice, Nataliya Melnyk, Andrew McPherson, Ali Bashashati, Emma Laks, et al. Titan: inference of copy number architectures in clonal cell populations from tumor whole-genome sequence data. *Genome research*, 24(11):1881–1893, 2014.
- 18 Sridhar Hannenhalli and Pavel 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. doi:10.1145/225058.225112.
- 19 Sridhar Hannenhalli and Pavel 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. doi:10.1109/SFCS.1995.492588.

- 20 Jun Inoue, Yukuto Sato, Robert Sinclair, Katsumi Tsukamoto, and Mutsumi Nishida. Rapid genome reshaping by multiple-gene loss after whole-genome duplication in teleost fish suggested by mathematical modeling. *Proceedings of the National Academy of Sciences*, 112(48):14918–14923, 2015. doi:10.1073/pnas.1507669112.
- 21 Olivier Jaillon, Jean-Marc Aury, Frédéric Brunet, Jean-Louis Petit, Nicole Stange-Thomann, Evan Mauceli, Laurence Bouneau, Cécile Fischer, Catherine Ozouf-Costaz, Alain Bernot, et al. Genome duplication in the teleost fish tetraodon nigroviridis reveals the early vertebrate proto-karyotype. *Nature*, 431(7011):946–957, 2004.
- 22 Jakub Kováč. On the complexity of rearrangement problems under the breakpoint distance. *Journal of Computational Biology*, 21(1):1–15, 2014. doi:10.1089/cmb.2013.0004.
- 23 Yilong Li, Nicola D. Roberts, Jeremiah A. Wala, Ofer Shapira, Steven E. Schumacher, Kiran Kumar, Ekta Khurana, Sebastian Waszak, Jan O. Korb, James E. Haber, Marcin Imielinski, Kadir C. Akdemir, Eva G. Alvarez, Adrian Baez-Ortega, Rameen Beroukhi, Paul C. Boutros, David D. L. Bowtell, Benedikt Brors, Kathleen H. Burns, Peter J. Campbell, Kin Chan, Ken Chen, Isidro Cortés-Ciriano, Ana Dueso-Barroso, Andrew J. Dunford, Paul A. Edwards, Xavier Estivill, Dariush Etemadmoghadam, Lars Feuerbach, J. Lynn Fink, Milana Frenkel-Morgenstern, Dale W. Garsed, Mark Gerstein, Dmitry A. Gordenin, David Haan, James E. Haber, Julian M. Hess, Barbara Hutter, David T. W. Jones, Young Seok Ju, Marat D. Kazanov, Leszek J. Klimczak, Youngil Koh, Jan O. Korb, Eunjung Alice Lee, Jake June-Koo Lee, Andy G. Lynch, Geoff Macintyre, Florian Markowetz, Iñigo Martincorena, Alexander Martinez-Fundichely, Matthew Meyerson, Satoru Miyano, Hidewaki Nakagawa, Fabio C. P. Navarro, Stephan Ossowski, Peter J. Park, John V. Pearson, Montserrat Puiggròs, Karsten Rippe, Nicola D. Roberts, Steven A. Roberts, Bernardo Rodriguez-Martin, Steven E. Schumacher, Ralph Scully, Mark Shackleton, Nikos Sidiropoulos, Lina Sieverling, Chip Stewart, David Torrents, Jose M. C. Tubio, Izar Villasante, Nicola Waddell, Jeremiah A. Wala, Joachim Weischenfeldt, Lixing Yang, Xiaotong Yao, Sung-Soo Yoon, Jorge Zamora, Cheng-Zhong Zhang, Peter J. Campbell, PCAWG Structural Variation Working Group, and PCAWG Consortium. Patterns of somatic structural variation in human cancer genomes. *Nature*, 578(7793):112–121, 2020. doi:10.1038/s41586-019-1913-9.
- 24 Stefano Mangiola, Matthew KH Hong, Marek Cmero, Natalie Kurganovs, Andrew Ryan, Anthony J Costello, Niall M Corcoran, Geoff Macintyre, and Christopher M Hovens. Comparing nodal versus bony metastatic spread using tumour phylogenies. *Scientific reports*, 6:33918, 2016.
- 25 Julia Mixtacki. Genome halving under DCJ revisited. In Xiaodong Hu and Jie Wang, editors, *Proc. Computing and Combinatorics*, volume 5092 of *Lecture Notes in Computer Science*, pages 276–286. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. doi:10.1007/978-3-540-69733-6.
- 26 Serena Nik-Zainal, Peter Van Loo, David C Wedge, Ludmil B Alexandrov, Christopher D Greenman, King Wai Lau, Keiran Raine, David Jones, John Marshall, Manasa Ramakrishna, et al. The life history of 21 breast cancers. *Cell*, 149(5):994–1007, 2012.
- 27 Layla Oesper, Ahmad Mahmood, and Benjamin J Raphael. Theta: inferring intra-tumor heterogeneity from high-throughput dna sequencing data. *Genome biology*, 14(7):R80, 2013.
- 28 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. URL: <http://www.ncbi.nlm.nih.gov/pubmed/3148746>.
- 29 Alexandre Pelé, Mathieu Rousseau-Gueutin, and Anne-Marie Chèvre. Speciation success of polyploid plants closely relates to the regulation of meiotic recombination. *Frontiers in Plant Science*, 9:907, 2018. doi:10.3389/fpls.2018.00907.
- 30 Marina Petkovic, Thomas BK Watkins, Emma C Colliver, Sofya Laskina, Charles Swanton, Kerstin Haase, and Roland F Schwarz. Whole-genome doubling-aware copy number phylogenies for cancer evolution with medicc2. *bioRxiv*, 2021. doi:10.1101/2021.02.28.433227.

- 31 Pavel Pevzner and Glenn 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. doi:10.1145/640075.640108.
- 32 David Sankoff and Mathieu Blanchette. Multiple genome rearrangement and breakpoint phylogeny. *Journal of computational biology*, 5(3):555–570, 1998.
- 33 Roland F Schwarz, Charlotte KY Ng, Susanna L Cooke, Scott Newman, Jillian Temple, Anna M Piskorz, Davina Gale, Karen Sayal, Muhammed Murtaza, Peter J Baldwin, et al. Spatial and temporal heterogeneity in high-grade serous ovarian cancer: a phylogenetic analysis. *PLoS medicine*, 12(2):e1001789, 2015.
- 34 Roland F Schwarz, Anne Trinh, Botond Sipos, James D Brenton, Nick Goldman, and Florian Markowetz. Phylogenetic quantification of intra-tumour heterogeneity. *PLoS computational biology*, 10(4):e1003535, 2014.
- 35 Ronglai Shen and Venkatraman E Seshan. Facets: allele-specific copy number and clonal heterogeneity analysis tool for high-throughput dna sequencing. *Nucleic acids research*, 44(16):e131–e131, 2016.
- 36 Andrea Sottoriva, Haeyoun Kang, Zhicheng Ma, Trevor A Graham, Matthew P Salomon, Junsong Zhao, Paul Marjoram, Kimberly Siegmund, Michael F Press, Darryl Shibata, et al. A big bang model of human colorectal tumor growth. *Nature genetics*, 47(3):209, 2015.
- 37 Steven H Strauss, Jeffrey D Palmer, Glen T Howe, and Allan 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.
- 38 Alfred H Sturtevant and Th 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.
- 39 Eric Tannier, Chunfang Zheng, and David Sankoff. Multichromosomal median and halving problems under different genomic distances. *BMC Bioinformatics*, 10(1):120, 2009. doi:10.1186/1471-2105-10-120.
- 40 Robert Warren and David Sankoff. Genome halving with double cut and join. *Journal of Computational Biology*, 7(2):357–371, 2009.
- 41 Robert Warren and David Sankoff. Genome aliquoting revisited. *Journal of Computational Biology*, 18(9):1065–1075, 2011. URL: <http://online.liebertpub.com/doi/abs/10.1089/cmb.2011.0087>.
- 42 Kenneth H. Wolfe and Denis C. Shields. Molecular evidence for an ancient duplication of the entire yeast genome. *Nature*, 387:708 EP–, 1997. doi:10.1038/42711.
- 43 Sophia Yancopoulos, Oliver Attie, and Richard Friedberg. Efficient sorting of genomic permutations by translocation, inversion and block interchange. *Bioinformatics*, 21(16):3340–3346, 2005. doi:10.1093/bioinformatics/bti535.
- 44 Simone Zaccaria, Mohammed El-Kebir, Gunnar W. Klau, and Benjamin J. Raphael. Phylogenetic copy-number factorization of multiple tumor samples. *Journal of Computational Biology*, 25(7):689–708, 2018. PMID: 29658782. doi:10.1089/cmb.2017.0253.
- 45 Travis I Zack, Steven E Schumacher, Scott L Carter, Andrew D Cherniack, Gordon Saksena, Barbara Tabak, Michael S Lawrence, Cheng-Zhong Zhang, Jeremiah Wala, Craig H Mermel, et al. Pan-cancer patterns of somatic copy number alteration. *Nature genetics*, 45(10):1134, 2013.
- 46 Ron Zeira, Meirav Zehavi, and Ron Shamir. A linear-time algorithm for the copy number transformation problem. *Journal of Computational Biology*, 24(12):1179–1194, 2017.
- 47 Chunfang Zheng, Qian Zhu, Zaky Adam, and David Sankoff. Guided genome halving: hardness, heuristics and the history of the hemiascomycetes. *Bioinformatics (Oxford, England)*, 24(13):i96–i104, July 2008. doi:10.1093/bioinformatics/btn146.

S1 Appendix

S1.1 Reducing input profile size

In order to show these Proposition 1 and Proposition 2, we define two ways of modifying profiles that “preserve” the validity of a transformation between them. Let $S = \langle s_i \rangle$ and $T = \langle t_i \rangle$ be CNPs with n genes, and let C be a transformation for $S \rightarrow T$. We *remove* a gene i from S and T to obtain new CNPs S' and T' , where

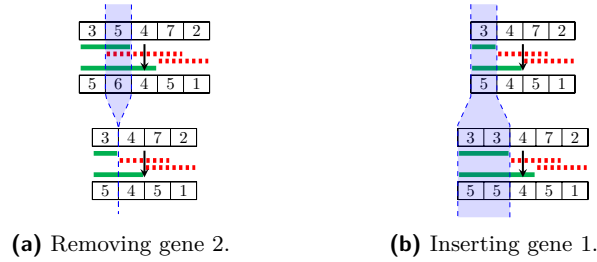
$$\begin{aligned} S' &= \langle s_1, s_2, \dots, s_{i-1}, s_{i+1}, \dots, s_n \rangle \\ T' &= \langle t_1, t_2, \dots, t_{i-1}, t_{i+1}, \dots, t_n \rangle \end{aligned}$$

Then, we can modify C to obtain a new transformation C' for $S' \rightarrow T'$, where $|C'| \leq |C|$, because we can modify each event in C to “skip” gene i (Figure S1a).

Similarly, we can *insert* a gene i to S and T to get S^* and T^* , such that

$$\begin{aligned} S^* &= \langle s_1, s_2, \dots, s_i, \mathbf{s}_i, s_{i+1}, \dots, s_n \rangle \\ T^* &= \langle t_1, t_2, \dots, t_i, \mathbf{t}_i, t_{i+1}, \dots, t_n \rangle \end{aligned}$$

That is, we create a new gene next to gene i , with the same value as the gene at i . Then, we can modify a transformation C for $S \rightarrow T$ to create a transformation C^* for $S^* \rightarrow T^*$, where $|C^*| \leq |C|$, because we can modify each event in C to “stretch over” a new gene at i (Figure S1b).



■ **Figure S1** Modifying profiles and transformations by removing and inserting genes.

S1.1.1 Proof of Proposition 1

Proof. Let \hat{S} be an optimal preduplication profile for $\eta_p(T)$ and let C be a transformation for $p\hat{S} \rightarrow T$ of size $\eta_p(T)$. We can delete gene i from \hat{S} to get \hat{S}' and since T' can be formed from T by also removing gene i , we can obtain a transformation C' for $p\hat{S}' \rightarrow T'$ such that $|C'| \leq |C|$. This implies $\eta_p(T') \leq \eta_p(T)$.

To show the other direction, we insert a gene at i into \hat{S}' and T' to get \hat{S}^* and T^* , respectively. Then, we have some transformation C^* for $p\hat{S}^* \rightarrow T^*$ such that $|C^*| \leq |C'|$. Finally, we can set $\hat{s}_i^* = 0$ and $t_i^* = 0$, such that \hat{S}^* is now equal to \hat{S} and T^* is now equal to T . But, C^* is still valid for $p\hat{S} \rightarrow T$, because modifying \hat{S}^* in such a manner does not affect any other gene, and preserves the validity of C^* at gene i , which has value zero in both $p\hat{S}$ and T . So, $\eta_p(T) \leq \eta_p(T')$, which completes the proof. ◀

S1.1.2 Proof of Proposition 2

Proof. First, we show $\eta_p(T') \leq \eta_p(T)$. Let \hat{S} be an optimal preduplication profile for $\eta_p(T)$, and let C be a transformation $p\hat{S} \rightarrow T$. We can remove gene $i + 1$ from \hat{S} to get \hat{S}' , and so there is a valid transformation C' for $p\hat{S}' \rightarrow T'$ that is no larger than C . This implies that $\eta_p(T') \leq \eta_p(T)$, and that we can get an optimal preduplication profile for T' by deleting a gene of T .

Next, we show $\eta_p(T') \geq \eta_p(T)$. Let \hat{S}' be an optimal preduplication profile (with $n - 1$ genes) for $\eta_p(T')$, and let C' be a transformation $p\hat{S}' \rightarrow T'$ of size $\eta_p(T')$. Then, we can insert a gene at i into \hat{S}' to get \hat{S}^* . Similarly, we can insert a gene at i into T' to get T^* . Then, we obtain a transformation C^* for $p\hat{S}^* \rightarrow T^*$ such that $|C^*| \leq |C'|$. Finally, because $t_{i+1} \geq t_i$, we need to modify \hat{S}^* and T^* such that $T^* = T$, while preserving the validity of C^* for $p\hat{S}^* \rightarrow T^*$. Note that because $t_i \equiv t_{i+1} \pmod{p}$ and $t_i \leq t_{i+1}$, it follows that $(t_{i+1} - t_i) = kp$ for some $k \geq 0$. Since S^* and T^* are a result of gene insertion at i , we have $\hat{s}_{i+1}^* = \hat{s}_i^* \neq 0$ and $t_{i+1}^* = t_i \neq 0$. Thus, we can increase \hat{s}_{i+1}^* by k and increase t_{i+1}^* by kp to get two new profiles, \tilde{S} and \tilde{T} . After doing so, C^* is still a valid transformation for $p\tilde{S} \rightarrow \tilde{T}$ because:

- We did not change the values of genes other than $i + 1$ in either profile and we did not modify C^* , so C^* is still valid at these genes.
- Consider gene $i + 1$. Before modifying \hat{S}^* and T^* , we had that $\hat{s}_{i+1}^* = \hat{s}_i^* \neq 0$ and $t_{i+1}^* = t_i \neq 0$. Hence, by Observation 1, if a is the number of amplifications in C^* that target gene $i + 1$ and if d is the number of deletions that target gene $i + 1$, then

$$t_{i+1}^* = p\hat{s}_{i+1}^* + a - d$$

Now, when we modify \hat{S}^* and T^* to get \tilde{S} and \tilde{T} , we set $\tilde{s}_{i+1} = \hat{s}_{i+1}^* + k$ and $\tilde{t}_{i+1} = t_{i+1}^* + kp$. Note that $\tilde{t}_{i+1} = p\tilde{s}_{i+1} + a - d$, because we can add kp to both sides of the previous equation. In addition, both \tilde{t}_{i+1} and \tilde{s}_{i+1} are nonzero, so C^* is valid at gene $i + 1$.

Together, we have $T^* = T$, a preduplication profile S^* and transformation $p\hat{S}^* \rightarrow T$ of size $\eta_p(T')$. This implies $\eta_p(T) \leq \eta_p(T')$. ◀

S1.2 Computing $y_1(x), y'_1(x), y_2(x), y'_2(x)$

► **Proposition S5.** $y_1(x), y'_1(x), y_2(x), y'_2(x)$ are all computable in $O(1)$ time.

Proof. We show each computation separately.

- To compute $y_1(x)$, note that

$$\begin{aligned} y_1(x) &= \min\{y \mid y \in \mathcal{D}_{i-1}, y \geq x\} \\ &= \min\{b_{i-1}^- + kp \mid k \geq 0 \wedge b_{i-1}^- + kp \leq np \wedge b_{i-1}^- + kp \geq x\} \\ &= b_{i-1}^- + p \cdot \min\{k \geq 0 \mid np \geq b_{i-1}^- + kp \geq x\} \end{aligned}$$

So, we can find the min k that satisfies these conditions in order to compute y_1 .

$$\begin{aligned} b_{i-1}^- + kp &\geq x \\ kp &\geq x - b_{i-1}^- \\ k &\geq (x - b_{i-1}^-)/p \end{aligned}$$

We can compute the right hand side, take the ceiling; the result is always non-negative because $|x - b_{i-1}^-|$ cannot exceed p . Afterwards, we double check that this satisfies the bound $b_{i-1}^- + kp \leq np$ that we get from bounding the number of total events. If it does, we have all we need to compute $y_1(x)$. If it does not, then $y_1(x)$ does not exist, and we ignore its term in the $\min\{\dots\}$ when computing $D[i, x]$.

18:24 CND Halving and Aliquoting

- To compute $y_2(x)$, we use a similar approach.

$$\begin{aligned} y_2(x) &= \max\{y \mid y \in \mathcal{D}_{i-1}, y \leq x\} \\ &= \max\{b_{i-1}^- + kp \mid k \geq 0 \wedge b_{i-1}^- + kp \leq np \wedge b_{i-1}^- + kp \leq x\} \\ &= b_{i-1}^- + p \cdot \max\{k \geq 0 \mid b_{i-1}^- + kp \leq np \wedge b_{i-1}^- + kp \leq x\} \end{aligned}$$

So, we can find the max k that satisfies the conditions to compute y_2 .

$$\begin{aligned} b_{i-1}^- + kp &\leq x \\ kp &\leq x - b_{i-1}^- \\ k &\leq (x - b_{i-1}^-)/p \end{aligned}$$

We can pick the max k that works by computing the right hand side and taking the floor to get k . We do not need to check that $b_{i-1}^- + kp \leq np$, because we get this from $x \leq np$, but we do need to be careful to check if $k \geq 0$; if not, then x is small enough that no choice of k works. If such a k exists, then $y_2(x) = b_{i-1}^- + kp$. Note that $y_2(x) = y_1(x) - p$, because y_2 and y_1 are computed from the closest multiples of p that “sandwich” x .

- Computing $y'_1(x)$ is similar to computing $y_1(x)$, with the additional condition that $t_{i-1} - b_{i-1}^+ - kp \geq p$ to ensure that $\hat{s}_{i-1} \geq 1$.

$$\begin{aligned} y'_1(x) &= \min\{y \mid y \in \mathcal{A}_{i-1}, y \geq x\} \\ &= \min\{b_{i-1}^+ + kp \mid k \geq 0 \wedge b_{i-1}^- + kp \leq np \wedge b_{i-1}^- + kp \geq x \wedge t_{i-1} - b_{i-1}^+ - kp \geq p\} \\ &= b_{i-1}^+ + p \cdot \min\{k \geq 0 \mid np \geq b_{i-1}^+ + kp \geq x \wedge t_{i-1} - b_{i-1}^+ \geq (k+1)p\} \end{aligned}$$

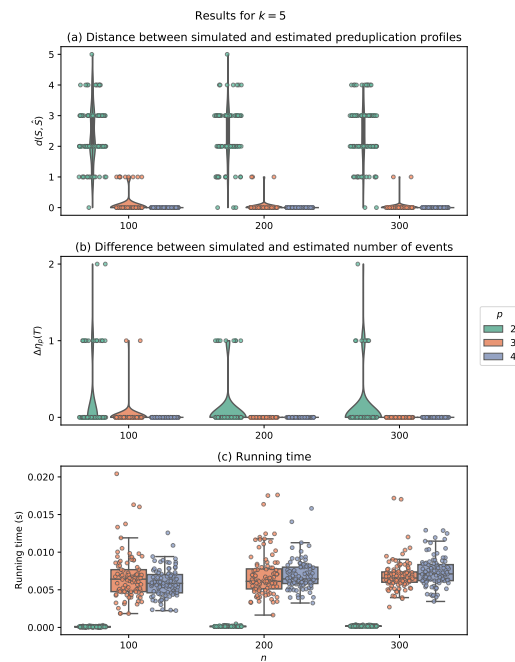
So, we can find the min k that satisfies the first condition in the same way that we computed $y_1(x)$, and check that this k satisfies the additional condition which enforces $\hat{s}_i \geq 1$.

- Computing $y'_2(x)$ is similar to computing $y_2(x)$, with the additional condition as mentioned in the case for $y'_1(x)$.

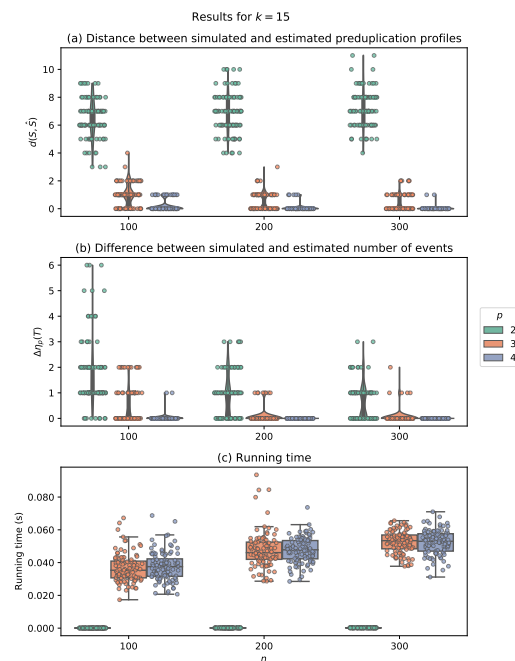
$$\begin{aligned} y'_2(x) &= \max\{y \mid y \in \mathcal{A}_{i-1}, y \leq x\} \\ &= \max\{b_{i-1}^+ + kp \mid k \geq 0 \wedge b_{i-1}^- + kp \leq np \wedge b_{i-1}^- + kp \leq x \wedge t_{i-1} - b_{i-1}^+ - kp \geq p\} \\ &= b_{i-1}^+ + p \cdot \max\{k \geq 0 \mid b_{i-1}^+ + kp \leq np \wedge b_{i-1}^- + kp \leq x \wedge t_{i-1} - b_{i-1}^+ \geq (k+1)p\} \end{aligned}$$

We compute the max k that satisfies all but the last condition in the same way that we computed $y_2(x)$. Then, compute the max k that satisfies the second condition, which is an upper bound on k . We can take the min of these two values to get the max k that satisfies both. ◀

S1.3 Supplemental results



■ **Figure S2** Simulation results for using $k = 5$ events after duplication. (a) $d(S, \hat{S})$ - the distance between simulated and estimated preduplication profiles. (b) $\Delta\eta_p(T)$ - the difference between the simulated and estimated number of events after duplication. (c) running time in seconds.



■ **Figure S3** Simulation results for using $k = 15$ events after duplication. (a) $d(S, \hat{S})$ - the distance between simulated and estimated preduplication profiles. (b) $\Delta\eta_p(T)$ - the difference between the simulated and estimated number of events after duplication. (c) running time in seconds.