A Linear-Time Algorithm for the Copy Number Transformation Problem

Ron Shamir[1], Meirav Zehavi[1], and Ron Zeira[1]

1 **School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel**
`rshamir@post.tau.ac.il`
2 **School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel**
`meizeh@post.tau.ac.il`
3 **School of Computer Science, Tel Aviv University, Tel Aviv 69978, Israel**
`ronzeira@post.tau.ac.il`

Abstract

Problems of genome rearrangement are central in both evolution and cancer. Most evolutionary scenarios have been studied under the assumption that the genome contains a single copy of each gene. In contrast, tumor genomes undergo deletions and duplications, and thus the number of copies of genes varies. The number of copies of each gene along a chromosome is called its *copy number profile*. Understanding copy number profile changes can assist in predicting disease progression and treatment. To date, questions related to distances between copy number profiles gained little scientific attention. Here we focus on the following fundamental problem, introduced by Schwarz *et al.* (PLOS Comp. Biol., 2014): given two copy number profiles, u and v, compute the edit distance from u to v, where the edit operations are segmental deletions and amplifications. We establish the computational complexity of this problem, showing that it is solvable in linear time and constant space.

1998 ACM Subject Classification F.2.2 Pattern Matching

Keywords and phrases Genome Rearrangement, Copy Number

Digital Object Identifier 10.4230/LIPIcs.CPM.2016.16

1 Introduction

The genome of a species evolves by undergoing small and large mutations over generations. Large mutations modify genome organization by rearrangement of genomic segments. Computational analysis of the process of *genome rearrangement* has been subject of extensive research over the last two decades [5]. The majority of these studies to date were restricted to a single copy of each gene, and were concerned with the reordering of segments. Extant models that do not make this assumption often result in NP-hard problems [12, 14, 15].

While most work on genome rearrangements to date was done in the context of species evolution, there is today great opportunity in analysis of cancer genome evolution. Cancer is a dynamic process characterized by the rapid accumulation of somatic mutations, which produce complex tumor genomes. Species evolution happens over eons and changes are carried over from one generation to the next. In contrast, cancer evolution happens within a single individual over a few decades. In many tumor genomes, a lot of the changes are segmental deletions and amplifications [16]. As a result, the number of copies of each gene along a chromosome, known as its *copy number profile*, changes during cancer development, compared to the normal genome that has two copies (or *alleles*) for each gene. Understanding these changes can assist in predicting disease progression and the outcome of medical interventions.

However, computational questions related to distances between copy number profiles received little scientific attention to date. Such questions are the topic of this paper.

Over the years, a variety of methods were used to determine the copy number profile of a cancer genome, at different resolutions. G-banding allows viewing the chromosomes bands [11]. FISH measures the copy numbers of tens to hundreds of targeted genes [4]. Array comparative genomic hybridization gives a higher resolution of copy number estimation for a cell population [17]. Most recently, deep sequencing techniques yield copy number profiles by using read depth data [10]. While it would have been preferable to analyze the genome (karyotype) itself and not its copy number profile, detection of structural variations from sequencing data is still problematic [7, 1]. Today it is a routine procedure to obtain detailed copy number profiles of cancer genomes, but utilizing them to understand cancer evolution is still an open problem.

Given two copy number profiles, the healthy tissue's and the tumor's, evaluating the distance between them can help in understanding cancer progression. A naïve measure of distance is the Euclidean distance between the two profiles [13]. Chowdhury *et al.* defined edit distance between copy number profiles obtained from FISH, where the edit operations are amplification or deletion of single genes, single chromosomes or the whole genome [3, 4, 2]. However, calculating these distances requires exponential time in the number of genes and therefore is limited to low resolution FISH data. The *TuMult* algorithm uses the number of breakpoints (loci where the copy numbers change) between two profiles as a simple distance measure [6].

Schwartz *et al.* introduced a model that admits amplification and deletion of contiguous segments [13]. The edit distance between two copy number profiles was defined as the minimum number of segmental deletions and duplications over all separations of the profiles into two alleles (a procedure known as *phasing*). Their algorithm *MEDICC* for computing the edit distance uses finite-state transducers (FSTs) [9] in order to model the profiles and efficiently compute the distance. However, the complexity of this method was not analyzed. Even without the phasing computation, the method needs to compose a 3-state transducer with itself N times, resulting in a transducer with 3^N states [13, 8]. The running time of FST procedures relies on the number of states and transitions, and in some cases may be exponential [9, 8].

Copy Number Transformation. We investigate the following problem, which underlies the model of [13]: Given two copy number profiles (*CNPs*), u and v, compute the minimum number of segmental duplications and deletions needed to transform u into v. We call this problem the Copy Number Transformation Problem (CNTP). A CNP is represented by a vector of nonnegative integers (the number of copies of each gene). A segmental deletion (amplification) decreases (resp. increases) by 1 the values of a contiguous segment of the vector, where zero values are not affected. Formal definitions are given in Section 2.

Our Contribution. We show that CNTP is solvable in linear time and constant space. The algorithm relies on several properties of the problem that we establish in Section 3.1, which may also be relevant to the analysis of other problems involving CNPs. Exploiting these properties results in a pseudo-polynomial dynamic programming algorithm for CNTP, presented in Section 3.2. In Section 3.3, by establishing that a certain function in the dynamic programming recursion is piecewise linear, we improve its performance and obtain our main result. For lack of space, some proofs are omitted.

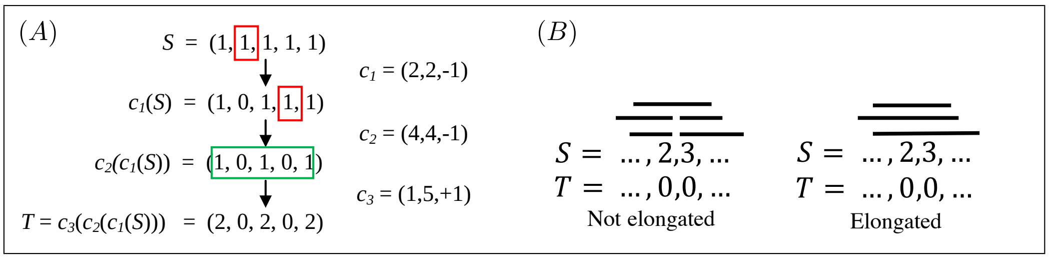


Figure 1 Copy number transformations. (A) The CNT $C = (c_1, c_2, c_3)$ transforms S into T. The size of C is 3. Red and green blocks indicate deletions and amplifications, respectively. (B) Elongated and non-elongated CNTs. Bold lines indicate the range of deletions.

2 Preliminaries

In this section, we give definitions and notation that are used throughout the paper. Let $n \in \mathbb{N}$. A *CN profile (CNP)* is a vector $V = (v_1, v_2, \ldots, v_n)$, where $v_i \in \mathbb{N} \cup \{0\}$. A *CN operation (CNO)* is a triple $c = (\ell, h, w)$, where $1 \leq \ell \leq h \leq n$ and $w \in \{-1, 1\}$. We say that a CNO $c = (\ell, h, -1)$ is a *deletion* and $c = (\ell, h, 1)$ is an *amplification*. Given a CNP $V = (v_1, v_2, \ldots, v_n)$ and a CNO $c = (\ell, h, w)$, we define the operation $c(V) = (c(v_1), c(v_2), \ldots, c(v_n))$ as follows. For each $i \in \{1, 2, \ldots, n\}$, if $\ell \leq i \leq h$ and $v_i \geq 1$, then $c(v_i) = v_i + w$, otherwise (i.e., if $i < \ell$ or $i > h$ or $v_i = 0$) $c(v_i) = v_i$. A triple $c = (\ell, h, w)$ with $h < \ell$ has no effect on the CNP, i.e., $c(V) = V$. Given two CNPs, $S = (s_1, s_2, \ldots, s_n)$ (source) and $T = (t_1, t_2, \ldots, t_n)$ (target), a *CN transformation (CNT)* is a vector $C = (c_1, c_2, \ldots, c_m)$, where $m \in \mathbb{N}$ and each $c_i = (\ell_i, h_i, w_i)$ is a CNO, such that $C(S) = c_m(c_{m-1}(\cdots(c_1(S)))) = T$. The *size* of C, denoted $|C|$, is m. An example is given in Fig. 1(A). Finally, we denote the number of operations of weight $w \in \{-1, 1\}$ affecting s_i by $op(C, w, i) = |\{(\ell, h, w) \in C : \ell \leq i \leq h\}|$. For example, in Fig. 1(A) $op(C, -1, 2) = 1$.

The *CN distance (CND)* from S to T, $\text{dist}(S, T)$, is the smallest size of a CNT C that satisfies $C(S) = T$, where if no such CNT exists, $\text{dist}(S, T) = \infty$. Note that dist is not symmetric. For example, for $S = (1)$ and $T = (0)$, $\text{dist}(S, T) = 1$ but $\text{dist}(T, S) = \infty$. Given two CNPs, $S = (s_1, s_2, \ldots, s_n)$ and $T = (t_1, t_2, \ldots, t_n)$, the Copy Number Transformation problem, CNTP, seeks $\text{dist}(S, T)$ (if one exists). We say that a CNT C is *optimal* if it realizes $\text{dist}(S, T)$, i.e., $|C| = \text{dist}(S, T)$ (there may exist several optimal CNTs). We let $N = \max\{\max_{i=1}^{n}\{s_i\}, \max_{i=1}^{n}\{t_i\}\}$ denote the maximum copy number in the input. Finally, for all $1 \leq i \leq n$, we define $u_i = s_i - t_i$.

3 An Algorithm for CNTP

We first present an $O(nN^2)$-time, $O(N)$-space algorithm for CNTP that is based on dynamic programming (Sections 3.1 and 3.2). Recall that N is the maximal integer in the input, so that the algorithm is pseudo-polynomial. Then, we modify this algorithm to run in linear time (Section 3.3). On a high level, the modification is based on the observation that the table used by the algorithm to store values of partial solutions can be described by $O(n)$ piecewise linear functions, where each function encapsulates $O(N)$ entries of the table. We show that each function has only three linear segments and so the computation of an entry can be performed in time $O(1)$ rather than $O(N)$. Furthermore, since each function can be represented in a compact manner, the size of table shrinks from $O(nN)$ to $O(n)$. The precise definitions of the table and the functions are given in Sections 3.2 and 3.3. Our proof of the correctness of the use of these functions requires a somewhat extensive case analysis that is presented separately in Section 3.4.

3.1 Key Propositions

We start by developing Alg1, an $O(nN^2)$-time dynamic programming algorithm for CNTP. Let $(S = (s_1, s_2, \ldots, s_n), T = (t_1, t_2, \ldots, t_n))$ be the input. Observe that there exists a CNT C such that $C(S) = T$ if and only if there does not exist an index $1 \leq i \leq n$ such that $s_i = 0$ and $t_i > 0$. Since the existence of such an index can be determined in linear time (where, if such an index is found, we return ∞), we will assume that $\text{dist}(S, T) < \infty$. To simplify the presentation, we further assume w.l.o.g. that $t_1, t_n \neq 0$. Indeed, if $t_1 = 0$ or $t_n = 0$, we can solve the input $(S' = (1, s_1, s_2, \ldots, s_n, 1), T' = (1, t_1, t_2, \ldots, t_n, 1))$ instead, since it holds that $\text{dist}(S, T) = \text{dist}(S', T')$. Finally, we assume w.l.o.g. that for all $1 \leq i \leq n$, $s_i > 0$. Indeed, if there exists $1 \leq i \leq n$ such that $s_i = 0$, then also $t_i = 0$, and we can solve the input $(S' = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n), T' = (t_1 \ldots, t_{i-1}, t_{i+1}, \ldots, t_n))$ since $\text{dist}(S, T) = \text{dist}(S', T')$.

Alg1 exploits four key observations about the nature of the problem at hand, summarized as follows: (1) it is sufficient to examine CNTs where all of the deletions precede all of the amplifications; (2) it is sufficient to examine CNTs that do not contain both a deletion that affects s_i but not s_{i+1} and a deletion that affects s_{i+1} but not s_i, and the same is true for amplifications; (3) when seeking an optimal solution, it is not necessary to store information indicating how many deletions/amplifications affect s_i if $t_i = 0$; (4) the maximum number of deletions/amplifications that affect each s_i can be bounded by N.

To formally state the first observation, we need the following definition.

▶ **Definition 1.** A CNT $C = (c_1, c_2, \ldots, c_m)$ is *ordered* if for all $1 \leq i < j \leq m$, if c_j is a deletion, then c_i is also a deletion.

▶ **Proposition 2.** *There exists an optimal ordered CNT.*

We note that the "opposite" proposition, stating that there exists an optimal CNT where all of the amplifications precede all of the deletions, does not hold: consider, e.g., $S = (1, 1, 1, 1, 1)$ and $T = (2, 0, 2, 0, 2)$. To prove this proposition, we will need the following claim.

▶ **Claim 3.** *Let $C = (c_1, c_2, \ldots, c_m)$ be an optimal CNT and let i be an index such that $c_i = (\ell_i, h_i, 1)$ and $c_{i+1} = (\ell_{i+1}, h_{i+1}, -1)$. Then, there exists an optimal CNT $C' = (c_1, \ldots, c_{i-1}, c'_i, c'_{i+1}, c_{i+2}, \ldots, c_m)$, where $c'_i = (\ell'_i, h'_i, w'_i)$ and $c'_{i+1} = (\ell'_{i+1}, h'_{i+1}, w'_{i+1})$, such that one of the following conditions holds.*

1. $(h'_i - \ell'_i) + (h'_{i+1} - \ell'_{i+1}) < (h_i - \ell_i) + (h_{i+1} - \ell_{i+1})$.
2. $(h'_i - \ell'_i) + (h'_{i+1} - \ell'_{i+1}) = (h_i - \ell_i) + (h_{i+1} - \ell_{i+1})$ *and* $w'_i = -1$.

Proof. Consider the following exhaustive case-analysis.

1. $h_i < \ell_{i+1}$ or $h_{i+1} < \ell_i$: In this case, the segments corresponding to c_i and c_{i+1} are disjoint. Thus, we can simply define $c'_i = c_{i+1}$ and $c'_{i+1} = c_i$. Then, Condition 2 is satisfied.
2. $\ell_i \leq \ell_{i+1} \leq h_i \leq h_{i+1}$: Define $c'_i = (h_i + 1, h_{i+1}, -1)$ and $c'_{i+1} = (\ell_i, \ell_{i+1} - 1, 1)$. For any CNP $V = (v_1, v_2, \ldots, v_n)$, $c'_{i+1}(c'_i(V)) = c_{i+1}(c_i(V))$. This argument holds because an application of c_i which is followed by an application of c_{i+1} does not change any entry v_k such that $\ell_{i+1} \leq k \leq h_i$. We have that $C'(S) = T$. Since $|C'| = |C|$, C' is an optimal CNT. Now, Condition 1 is satisfied.

3. $\ell_{i+1} \leq \ell_i \leq h_{i+1} \leq h_i$: Define $c'_i = (\ell_{i+1}, \ell_i - 1, -1)$ and $c'_{i+1} = (h_{i+1} + 1, h_i, 1)$. As in the second case, we obtain an optimal CNT that satisfies Condition 1.
4. $\ell_i \leq \ell_{i+1} \leq h_{i+1} \leq h_i$: Define $c'_i = (\ell_i, \ell_{i+1} - 1, 1)$ and $c'_{i+1} = (h_{i+1} + 1, h_i, 1)$. As in the second case, we obtain an optimal CNT that satisfies Condition 1.
5. $\ell_{i+1} \leq \ell_i \leq h_i \leq h_{i+1}$: Define $c'_i = (\ell_{i+1}, \ell_i - 1, -1)$ and $c'_{i+1} = (h_i + 1, h_{i+1}, -1)$. As in the second case, we obtain an optimal CNT that satisfies Condition 1. ◀

As we show below, Claim 3 implies the existance of an ordered optimal CNT. In each of the cases in Claim 3, a local change is made in the CNT. Note however that just performing enough local operations does not guarantee reaching an ordered optimal CNT. For example, in a CNT with three consecutive CNOs, $c_i = (\ell_i, h_i, 1), c_{i+1} = (\ell_{i+1}, h_{i+1}, 1), c_{i+2} = (\ell_{i+2}, h_{i+2}, -1)$, one may loop between changing c_{i+1} into a deletion and then into an amplification.

Proof of Proposition 2. Let $\mathcal{C}$ be the set of optimal CNTs, and suppose, by way of contradiction, that it does not contain an ordered CNT. The three following phases sieve some solutions out of $\mathcal{C}$. Informally, we initially consider only optimal CNTs that minimize the sum of the sizes of the segments corresponding to their CNOs ($\mathcal{C}^1$); then, we further consider only the CNTs whose first amplification is as late as possible ($\mathcal{C}^2$); finally, we only take the CNTs whose first deletion after their first amplification is as early as possible ($\mathcal{C}^3$).

- Given $C = (c_1, c_2, \ldots, c_m) \in \mathcal{C}$, define $x(C) = \sum_{i=1}^{m}(h_i - \ell_i)$. Let $\mathcal{C}^1$ be the set of every $C \in \mathcal{C}$ for which there does not exist $C' \in \mathcal{C}$ such that $x(C) > x(C')$.
- Given $C = (c_1, c_2, \ldots, c_m) \in \mathcal{C}^1$, let $y(C)$ be the largest index $0 \leq i \leq m$ such that for all $1 \leq j \leq i$, c_j is a deletion. Note that $y(C) = 0$ if and only if c_1 is an amplification. Let $\mathcal{C}^2$ be the set of every $C \in \mathcal{C}^1$ for which there does not exist $C' \in \mathcal{C}^1$ such that $y(C) < y(C')$.
- Given $C = (c_1, c_2, \ldots, c_m) \in \mathcal{C}^2$, let $z(C)$ be the smallest index $i \in \{y(C) + 1, \ldots, m\}$ such that c_i is a deletion. By the definition of $y(C)$ and since C is not ordered, we have that $z(C)$ is well-defined and $z(C) \geq y(C) + 2$. Let $\mathcal{C}^3$ be the set of every $C \in \mathcal{C}^2$ for which there does not exist $C' \in \mathcal{C}^2$ such that $z(C) > z(C')$.

Since $\mathcal{C} \neq \emptyset$, we have that $\mathcal{C}^3 \neq \emptyset$. Thus, we can let $C = (c_1, c_2, \ldots, c_m)$ be a solution in $\mathcal{C}^3$. Let i be the smallest index such that c_i is an amplification and c_{i+1} is a deletion. Now, consider the conditions in Claim 3: if Condition 1 holds, we have a contradiction to the fact that $C \in \mathcal{C}^1$, while if Condition 2 holds, we have a contradiction either to the fact that $C \in \mathcal{C}^2$ (if $i = 1$ or c_{i-1} is a deletion) or to the fact that $C \in \mathcal{C}^3$ (otherwise). Thus, we conclude that $\mathcal{C}$ contains an ordered CNT. ◀

The other three propositions are stated without proof.

▶ **Definition 4.** A CNT C is *elongated* if for all $1 \leq i < n$ and $w \in \{-1, 1\}$,

$$\min\{op(C, w, i), op(C, w, i + 1)\} = |\{(\ell, h, w) \in C : \ell \leq i, i + 1 \leq h\}|.$$

Equivalently, C is elongated if no two amplifications (or deletions) "dovetail", i.e., one ending at i and the other starting at i+1. It is clear that for any CNT C, the inequality $\geq$ holds above (since $\{(\ell, h, w) \in C : \ell \leq i, i + 1 \leq h\}$ is a subset of both $\{(\ell, h, w) \in C : \ell \leq i \leq h\}$ and $\{(\ell, h, w) \in C : \ell \leq i + 1 \leq h\}$). Our second key proposition implies the inequality $\leq$ holds as well. An example for an elongated CNT is given in Fig. 1(B).

▶ **Proposition 5.** *Every ordered optimal CNT is elongated.*

To formalize our third key proposition, we need the following definition.

▶ **Definition 6.** A CNT C *skips zeros* if for every $1 \leq i < j \leq n$ such that for all $i < r \leq j, t_r = 0$ we have

$$op(C,-1,j) = \max\{\max_{r=i+1}^{j}\{s_r\}, op(C,-1,i)\}, \text{ and } op(C,1,j) = op(C,1,i).$$

In words, for a block of consecutive zeros in the target profile, all deletions that span the block also include its flanking positions. An example of a CNT that skips zeros is given in Fig. 2(A).

▶ **Proposition 7.** *There exists an optimal ordered CNT that skips zeros.*

For a position with positive target value, knowing the number of deletions that affected it uniquely determines the number of amplifications that affected it. This simple fact will help the efficiency of our procedures. Formally:

▶ **Observation 8.** *Let $1 \leq i \leq n$ be an index such that $t_i > 0$, and let $C = (c_1, c_2, \ldots, c_m)$ be a CNT such that $C(S) = T$. Then, $op(C,1,i) = -u_i + op(C,-1,i)$.*

Finally, we formalize our fourth key proposition.

▶ **Definition 9.** A CNT C is *bounded* if for all $1 \leq i \leq n$ and every $w \in \{-1,1\}$, we have $op(C,w,i) \leq N$.

▶ **Proposition 10.** *Every optimal ordered CNT that skips zeros is bounded.*

3.2 An $O(nN^2)$-Time Algorithm for CNTP

On a high-level, the dynamic programming algorithm works as follows. It considers increasing prefixes $S^i = (s_1, s_2, \ldots, s_i)$ and $T^i = (t_1, t_2, \ldots, t_i)$ of the input. It computes a table M having $n(N+1)$ entries where $\mathrm{M}[i,d]$ is the best value of a solution on (S^i, T^i) that uses exactly d deletions that affect the i^{th} position. The parameter d ranges between zero and N, and the values for each i are computed based on values $\mathrm{M}[j,\cdot]$ for a single specific $j < i$. In particular, at each point of time, only two rows of the table M are stored. By Propositions 2–10, the algorithm considers only ordered, elongated, zero-skipping and bounded solutions. We call such solutions *good*.

More formally, given $1 \leq i \leq n$ and $0 \leq d \leq N$, we say that a CNT C is an *(i,d)-CNT* if $C(S^i) = T^i$, $d = op(C,-1,i)$, and C is good. We say that an (i,d)-CNT C is *optimal* if there is no (i,d)-CNT C' such that $|C'| < |C|$. Our goal will be to ensure that each entry $\mathrm{M}[i,d]$ stores the size of an optimal (i,d)-CNT, where if no such CNT exists, it stores ∞. We do not compute entries $\mathrm{M}[i,d]$ such that $t_i = 0$; indeed, by relying on Property 7, we are able to skip such entries (though our recursive formula does consider CNs s_i referring to indices i such that $t_i = 0$). In this context, observe that any ordered CNT C such that $C(S) = T$ consists of at least u_i deletions that affect s_i, and if $t_i > 0$, it cannot consist of more than $s_i - 1$ such deletions (since after decreasing s_i to 0, it remains 0). Moreover, if $u_i \leq d < s_i$, there exists an (i,d)-CNT – by independently adjusting the value of each position $< i$ to its target position and the value at position i with d deletions, using operations of span 1.

▶ **Observation 11.** *Given $1 \leq i \leq n$ such that $t_i > 0$ and $0 \leq d \leq N$, there exists an (i,d)-CNT if and only if $u_i \leq d < s_i$.*

In case $s_i < t_i$, Observation 11 states that there exists an (i,d)-CNT if and only if $d < s_i$. In light of this observation, we will use the following assumption.

▶ **Assumption 12.** *In the computation below, we assume that* $\max\{u_i, 0\} \leq d < s_i$*. Entries* $M[i, d]$ *for which it is not true that* $\max\{u_i, 0\} \leq d < s_i$ *store* ∞*.*

By Observation 8, if a solution involved d deletions at position i with $t_i > 0$, then it involved $-u_i + d$ amplifications at that position. For convenience denote that number by $a(i, d) = -u_i + d$ for all $1 \leq i \leq n$ satisfying $t_i > 0$ and $\max\{u_i, 0\} \leq d < s_i$, and $a(i, d) = \infty$ otherwise.

For input profiles S, T, the algorithm precomputes two vectors .Given an index $1 < i \leq n$ such that $t_i > 0$, let $\text{prev}(i)$ denote the largest index $j < i$ such that $t_j > 0$. Moreover, if $\text{prev}(i) = i - 1$, let $Q_i = 0$, and otherwise let $Q_i = \max_{\text{prev}(i)<j<i}\{s_j\}$. A skipping zero solution will skip the positions between i and $prev(i)$ in the computation, but will make sure to perform at least Q_i deletions spanning the skipped positions.

Initialization. The initialization step sets all entries $\mathrm{M}[1, d]$ as follows.

$$\mathrm{M}[1, d] \leftarrow d + a(1, d).$$

Recursion. If $t_i = 0$ position i is skipped. Suppose that $i > 1$, $t_i > 0$ and $\max\{u_i, 0\} \leq d < s_i$. The order of the computation is determined by the first argument. The computation is summarized in the following formula.

$$\mathrm{M}[i, d] \leftarrow \min_{0\leq d'\leq N}\{\mathrm{M}[\text{prev}(i), d'] + \max\{d - d', 0\} + \max\{a(i, d) - a(\text{prev}(i), d'), 0\}$$

$$+ \max\{Q_i - \max\{d, d'\}, 0\}\}.$$

Roughly speaking, to compute $M[i, d]$ we look back to the previous non zero position in T, and for each value d' in that position add the difference from d if needed, the number of amplifications to be added if needed, and the number of additional deletions if such are needed to take care of the skipped zero positions. After filling the table M, Alg1 returns $\min_{0\leq d\leq N} \mathrm{M}[n, d]$. An example if a filled table is given in Fig. 2(B).

Correctness. First, we claim that the entries of the table M are computed properly.

▶ **Lemma 13.** *For all* $1 \leq i \leq n$ *such that* $t_i > 0$ *and for all* $0 \leq d \leq N$*,* $M[i, d]$ *stores the size of an optimal* (i, d)*-CNT, where if no such CNT exists, it stores* ∞*.*

Proof. We prove the lemma by induction on the order of the computation.

The correctness of the initialization step follows from the definition of an (i, d)-CNT and Observation 8.

Now, fix $1 < i \leq n$ such that $t_i > 0$, and fix $\max\{u_i, 0\} \leq d < s_i$. Let m be the size of an optimal (i, d)-CNT. Suppose that the lemma is correct for all $i' < i$ and $0 \leq d' \leq N$. We need to show that $\mathrm{M}[i, d] = m$.

First Direction. First, we show that $\mathrm{M}[i, d] \leq m$. Let $C = (c_1, c_2, \ldots, c_m)$ be an optimal (i, d)-CNT, and for all $1 \leq j \leq m$, denote $c_j = (\ell_j, h_j, w_j)$. For all $1 \leq j \leq m$, let $c'_j = (\ell_j, \min\{h_j, \text{prev}(i)\}, w_j)$. Now, define $C' = (c'_1, c'_2, \ldots, c'_m)$. We further let $\widehat{C} = (\widehat{c}_1, \widehat{c}_2, \ldots, \widehat{c}_q)$ denote the CNT obtained from C' by removing all of the CNOs $c = (\ell, h, w)$ such that $h < \ell$. Denote $\widehat{d} = op(\widehat{C}, -1, \text{prev}(i))$. Observe that $\widehat{d} \leq N$ and that $\widehat{C}$ is a $(\text{prev}(i), \widehat{d})$-CNT (because C is an (i, d)-CNT). Therefore, by the induction hypothesis, $\mathrm{M}[\text{prev}(i), \widehat{d}] \leq q$ (recall that $q = |\widehat{C}|$). If $\text{prev}(i) = i - 1$, then $Q_i = 0$ and since C is ordered and elongated, by Observation 8 we have that $m-q=\max\{d-\widehat{d},0\}+\max\{a(i,d)-a(\mathrm{prev}(i),\widehat{d}),0\}$. Thus, by the recursive formula, in this case we get that $\mathrm{M}[i,d]\leq m$

Now, suppose that $\mathrm{prev}(i)<i-1$. Then, since C is ordered and skips zeros, and by the definition of Q_i, the two following conditions hold.

1. $op(C,-1,i-1)=\max\{Q_i,op(C,-1,\mathrm{prev}(i))\}$.
2. $op(C,1,i-1)=op(C,1,\mathrm{prev}(i))$.

Thus, since C is ordered and elongated, by Observation 8 we have that $m-q=\max\{d-\widehat{d},0\}+\max\{a(i,d)-a(\mathrm{prev}(i),\widehat{d}),0\}+\max\{Q_i-\max\{d,\widehat{d}\},0\}$. Again, by the recursive formula, this implies that $\mathrm{M}[i,d]\leq m$.

Second Direction. Next, we show that $\mathrm{M}[i,d]\geq m$. To this end, it is sufficient to show that there exists an (i,d)-CNT C such that $\mathrm{M}[i,d]\geq|C|$. Let $\widehat{d}$ be an argument d' at which the value computed by using the recursive formula is minimized. By the inductive hypothesis, there exists a $(\mathrm{prev}(i),\widehat{d})$-CNT $\widehat{C}=(\widehat{c}_1,\widehat{c}_2,\dots,\widehat{c}_q)$ such that $\mathrm{M}[\mathrm{prev}(i),\widehat{d}]\geq q$. For all $1\leq j\leq q$, denote $\widehat{c}_j=(\ell_j,h_j,w_j)$. Now, if $\mathrm{prev}(i)=i-1$, define $\widetilde{C}=\widehat{C}$, and else define $\widetilde{C}$ as follows. For all $1\leq j\leq q$, let $\widetilde{c}_j=(\ell_j,\widetilde{h},w_j)$, where $\widetilde{h}=h_j$ if $h_j<\mathrm{prev}(i)$ and $\widetilde{h}=i-1$ otherwise. Let $\widetilde{C}=(\widetilde{c}_1,\widetilde{c}_2,\dots,\widetilde{c}_q)$. Moreover, as long as there exists $\mathrm{prev}(i)<j<i$ such that $op(\widetilde{C},-1,j)<s_j$, choose the smallest such j, and append to the beginning of $\widetilde{C}$ the CNO $(j,i-1,-1)$. Let C' be the CNT obtained at the end of this process. Denote $C'=(c'_1,c'_2,\dots,c'_r)$, and for all $1\leq j\leq r$, denote $c'_j=(\ell'_j,h'_j,w'_j)$. Now, let p and q be the number of deletions and amplifications in C' whose segments include $i-1$, respectively. If $p<d$, append to the beginning of C' $d-p$ "dummy" deletions of the form $(i,i-1,-1)$, and if $a(i,d)<q$, append to the end of C' $a(i,d)-q$ "dummy" amplifications of the form $(i,i-1,1)$. Let $C''=(c''_1,c''_2,\dots,c''_k)$ be the resulting CNT, and for all $1\leq j\leq k$, denote $c''_j=(\ell''_j,h''_j,w''_j)$. Finally, we define C as follows. Let D (A) be a set of exactly d deletions (resp. amplifications) in C'' whose second argument is $i-1$. We let C be defined as C'', except that each CNO $(\ell,h,w)\in D\cup A$ is replaced by the CNO (ℓ,i,w). It is straightforward to verify that C is an (i,d)-CNT such that $|C|=q+\max\{d-\widehat{d},0\}+\max\{a(i,d)-a(\mathrm{prev}(i),\widehat{d}),0\}+\max\{Q_i-\max\{d,\widehat{d}\},0\}$, which concludes the correctness of the second direction. ◀

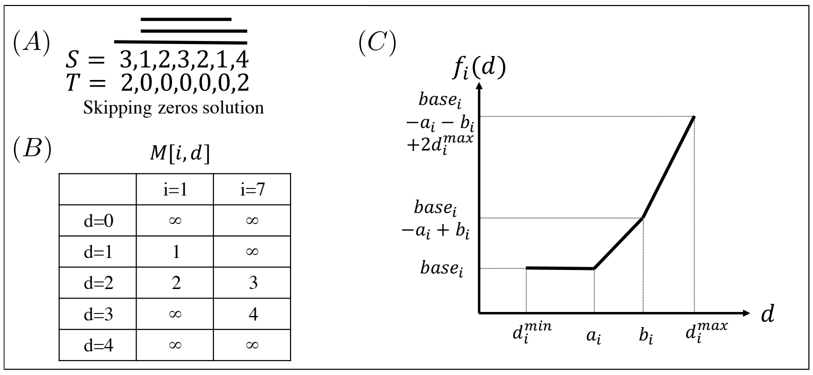
Now, we turn to consider the correctness and running time of Alg1.

▶ **Theorem 14.** Alg1 *solves* CNTP *in time* $O(nN^2)$ *and space* $O(N)$.

Proof. The table M contains $O(nN)$ entries, and each entry can be computed in time $O(N)$. Therefore, the time complexity of Alg1 is bounded by $O(nN^2)$. Moreover, for the computation of $\mathrm{M}[i,\cdot]$, it is only necessary to keep $O(N)$ entries for position $\mathrm{prev}(i)$, and therefore the space complexity is bounded by $O(N)$. Since every (n,d)-CNT C satisfies $C(S)=T$, and since for every good optimal CNT C, there exists $0\leq d\leq N$ such that C is an (n,d)-CNT, we have that Lemma 13 implies that Alg1 returns the smallest size of a good optimal CNT (if such a CNT exists). By Propositions 2–10, such a CNT indeed exists, and therefore Alg1 solves CNTP. ◀

3.3 A Linear-Time Algorithm for CNTP

In this section we show how to modify Alg1 in order to obtain an algorithm, called Alg2, that solves CNTP in linear time. The central lemma that leads to this improvement states each column in the table M can be described by a piecewise linear function of at most three segments.



	i=1	i=7
d=0	∞	∞
d=1	1	∞
d=2	2	3
d=3	∞	4
d=4	∞	∞

Figure 2 (A) A skipping-zeros solution. Bold lines indicate deletions. (B) The DP $M[i, d]$ matrix for the two CNPs in (A). (C) An example of the piecewise linear function $f_i(d)$ described in Lemma 15. The number of segments is three but can be smaller, depending on the values involved.

To present this lemma, we need the following notation. For all $i \in \{1, 2, \ldots, n\}$ such that $t_i > 0$, let $d_i^{min} = \max\{u_i, 0\}$ and $d_i^{max} = \max\{s_i - 1, 0\}$ be the least and largest values of d for which $M[i, d]$ is finite. Now, the function $f_i : \{d_i^{min}, \ldots, d_i^{max}\} \to \mathbb{N} \cup \{0\}$ will satisfy $f_i(d) = \mathrm{M}[i, d]$. Observe that the function f_i is discrete. We stress that in this section, we do not explicitly compute the entries of M – the definition of the functions concerns the values that would have been stored in these entries if they were computed by using Alg1.

▶ **Lemma 15.** *For each $i \in \{1, 2, \ldots, n\}$ such that $t_i > 0$, there exist $base_i, a_i, b_i \in \mathbb{N} \cup \{0\}$ such that for all $d \in \{d_i^{min}, \ldots, d_i^{max}\}$:*

$$f_i(d) = \begin{cases} base_i & \text{if } d_i^{min} \le d \le a_i \\ (base_i - a_i) + d & \text{if } a_i \le d \le b_i \\ (base_i - a_i - b_i) + 2d & \text{if } b_i \le d \le d_i^{max} \end{cases}$$

Moreover, $base_1, a_1$ and b_1 can be computed in constant time, and for each $i \in \{2, 3, \ldots, n\}$ such that $t_i > 0$, given $base_{\mathrm{prev}(i)}, a_{\mathrm{prev}(i)}$ and $b_{\mathrm{prev}(i)}$, $base_i, a_i$ and b_i can be computed in constant time.

An example is given in Fig. 2(C). The proof is based on Lemma 13 and an exhaustive case analysis, which, for the sake of clarity of presentation, is handled separately in Section 3.4.

Our algorithm, Alg2, performs the following computation:

1. Let $base_0 = a_0 = b_0 = 0$.
2. For $i = 1, 2, \ldots, n$:
  a. If $t_i = 0$, skip the rest of the current iteration.
  b. Compute $base_i, a_i$ and b_i using $base_{\mathrm{prev}(i)}, a_{\mathrm{prev}(i)}$ and $b_{\mathrm{prev}(i)}$.
3. Return $base_n$.

We are now ready to prove our main result.

▶ **Theorem 16.** Alg2 *solves* CNTP *in time $O(n)$ and space $O(1)$.*

Proof. According to Lemma 15, the function $f_i(d) = \mathrm{M}[i, d]$ is a piecewise linear function described by three values. The correctness of Lemma 15 shows that step 3 calculates these values in constant time and space given the previous values. The time and space complexity of Alg2 follow directly.

Now, by the correctness of Alg1, it is sufficient to prove that Alg2 returns the value $\min_{0\le d\le N} \mathrm{M}[n,d]$. By Observation 11, $\min_{0\le d\le N} \mathrm{M}[n,d] = \min_{d_n^{min}\le d\le d_n^{max}} \mathrm{M}[n,d]$. By Lemma 15, we further have that $\min_{d_n^{min}\le d\le d_n^{max}} \mathrm{M}[n,d] = base_n$. Thus, by the inductive proof of Lemma 15, we conclude that Alg2 solves CNTP. ◀

3.4 Case Analysis

The purpose of this section is to prove the correctness of Lemma 15. That is, we want to show that $f_i(d)$ is a piecewise linear function described by three parameters, and these parameters can be calculated in constant time. To this end, let $j = \mathrm{prev}(i)$ and $R_i = u_j - u_i$. Accordingly, the term $a(i,d) - a(j,d')$ can be written as $R_i + d - d'$. Moreover, let d'_{opt} be the argument d' that minimizes the recursive formula we use to compute $\mathrm{M}[i,d]$ under certain conditions that will be clear from context.

We prove Lemma 15 by induction on i. To simplify the proof, let $a_0 = b_0 = base_0 = 0$ and $f_0(d) = 2d$ for every $0 \le d \le N$. This definition is equivalent to adding the new entries $s_0 = t_0 = N+1$ (which do not affect the distance from S to T), and thus, it can serve as the basis of our induction. Next, suppose that Lemma 15 holds for $j = \mathrm{prev}(i) < i$, and we will prove that it holds for i.

The proof is based on an exhaustive case analysis that examines the position of Q_i relative to d_j^{min}, a_j, b_j and d_j^{max}, as well as the sign of R_i. For example, one of the cases is defined by the conditions $d_j^{min} \le Q_i \le a_j$, $R_i \ge 0$ and $a_j - R_i \le Q_i$. In each case, we analyze the behavior of $\mathrm{M}[i,d]$ as we increase d. More precisely, we examine several intervals that together contain all of the values that can be assigned to d. For example, in the above mentioned case, we consider the intervals $d \le a_j - R_j$, $a_j - R_j \le d \le Q_i$ and $Q_i \le d$. For each interval, we let d'_{opt} be an argument d' that minimizes $\mathrm{M}[i,d]$ under the conditions of the examined case. These conditions along with d'_{opt} allow us to remove the minimization and maximization functions from the formula defining $\mathrm{M}[i,d]$, and thus we obtain $f_i(d)$. In the latter example, if $d \le a_j - R_j$ we can choose $d'_{opt} = a_j$ and get $f_i(d) = \mathrm{M}[i,d] = \mathrm{M}[j,a_j] + \max\{d-a_j,0\} + \max\{R_i + d - a_j, 0\} + \max\{Q_i - \max\{d,a_j\},0\}\} = base_j$. As a corollary of the analysis, we get that indeed $f_i(d)$ is piecewise linear, and that a_i, b_i and $base_i$ can be calculated in constant time given a_j, b_j, $base_j$, R_i and Q_i.

Due to lack of space, the details of the case analysis are omitted. The analysis shows that in all cases, $f_i(d)$ is indeed a piecewise linear function with at most three linear segments defined by some a_i, b_i, $base_i$. After applying straightforward operations that reorganize the analysis (to present the results in a compact manner), we obtain the algorithm PiecewiseAlg, whose pseudocode is given below. This algorithm performs step 2b of Alg2, i.e., it calculates a_i, b_i, $base_i$ given a_j, b_j, $base_j$ and Q_i in constant time and space.

PiecewiseAlg first calculates R_i, d_i^{min} and d_i^{max} based on s_i and t_i. Next, according to the sign of R_i and the relative position of Q_i in comparison to the previous a_j and b_j, the algorithm calculates the structure of $f_i(d)$ defined by a_i and b_i. Finally, since $f_i(d)$ is defined only for the range $d_i^{min} \le d \le d_i^{max}$, we calculate $base_i = f_i(d_i^{min})$. Similarly, we limit the values of a_i and b_i to that range.

4 Conclusion

In this paper, we initiated the study of distances between CNPs from a theoretical point of view. We focused on one fundamental problem, CNTP, and showed that it is solvable in linear time and constant space. To this end, we proved several properties of CNTP that may be useful in solving other problems involving CNPs. Our algorithm can be modified to return

Algorithm 1 PiecewiseAlg

```
Input: $s_i, t_i, Q_i, a_j, b_j, base_j$
Output: $a_i, b_i, base_i$
  $R_i \leftarrow u_j - u_i$
  $d_i^{min} \leftarrow \max\{u_i, 0\}$
  $d_i^{max} \leftarrow \max\{s_i - 1, 0\}$
  if $R_i \geq 0$ then
    if $Q_i \leq a_j$ then
      $a_i \leftarrow a_j - R_i$; $b_i \leftarrow b_j$.
    else if $a_j < Q_i \leq b_j$ then
      $a_i \leftarrow Q_i - R_i$; $b_i \leftarrow b_j$.
    else if $b_j < Q_i$ then
      $a_i \leftarrow b_j - R_i$; $b_i \leftarrow Q_i$.
    end if
  else if $R_i < 0$ then
    if $Q_i \leq a_j$ then
      $a_i \leftarrow a_j$; $b_i \leftarrow b_j - R_i$.
    else if $a_j < Q_i \leq b_j$ then
      $a_i \leftarrow Q_i$; $b_i \leftarrow b_j - R_i$.
    else if $b_j < Q_i$ then
      $a_i \leftarrow \min\{Q_i, b_j - R_i\}$; $b_i \leftarrow \max\{Q_i, b_j - R_i\}$.
    end if
  end if
  $base_i \leftarrow base_j + \max\{Q_i - a_j, 0\} + \begin{cases} 0 & \text{if } d_i^{min} \leq a_i \\ d_i^{min} - a_i & \text{if } a_i < d_i^{min} \leq b_i \\ 2d_i^{min} - a_i - b_i & \text{if } b_i < d_i^{min} \leq d_i^{max} \end{cases}$
  $a_i \leftarrow \max\{d_i^{min}, \min\{a_i, d_i^{max}\}\}$; $b_i \leftarrow \max\{a_i, \min\{b_i, d_i^{max}\}\}$.
```

a transformation that realizes dist(S, T) in linear time and linear space by backtracking the dynamic programming vector. We have implemented the algorithm as well as an ILP formulation of CNTP (the implementations are available upon request), and we intend to assess the performance of these approaches.

Many computational and combinatorial aspects in the analysis of distances between CNPs require further research. Indeed, this paper can be viewed as a first step towards understanding them. We intend to investigate variants of CNTP where one seeks a CNP that minimizes the overall distance from it to two (or more) CNPs that are given as input. Such variants are relevant to phylogenetic reconstruction in cancer (see [13]). Additional directions for further research involve the introduction of edit operations other than basic segmental deletions and amplifications, dealing with phasing of the profiles, as well as the handling of noise.

Acknowledgment. We thank the referees for many helpful 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 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). MZ was supported by a fellowship from the I-CORE in Algorithms and the Simons

Institute for the Theory of Computing in Berkeley and by the Postdoctoral Fellowship for Women of Israel's Council for Higher Education.

References

1 Ryan P Abo, Matthew Ducar, Elizabeth P Garcia, Aaron R Thorner, Vanesa Rojas-Rudilla, Ling Lin, Lynette M Sholl, William C Hahn, Matthew Meyerson, Neal I Lindeman, Paul Van Hummelen, and Laura E MacConaill. BreaKmer: detection of structural variation in targeted massively parallel sequencing data using kmers. *Nucleic Acids Research*, nov 2014. doi:10.1093/nar/gku1211.

2 Salim Akhter Chowdhury, E Michael Gertz, Darawalee Wangsa, Kerstin Heselmeyer-Haddad, Thomas Ried, Alejandro A Schäffer, and Russell Schwartz. Inferring models of multiscale copy number evolution for single-tumor phylogenetics. *Bioinformatics*, 31(12):i258–67, jun 2015. doi:10.1093/bioinformatics/btv233.

3 Salim Akhter Chowdhury, Stanley E Shackney, Kerstin Heselmeyer-Haddad, Thomas Ried, Alejandro A Schäffer, and Russell Schwartz. Phylogenetic analysis of multiprobe fluorescence in situ hybridization data from tumor cell populations. *Bioinformatics*, 29(13):i189–98, jul 2013. doi:10.1093/bioinformatics/btt205.

4 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, jul 2014. doi:10.1371/journal.pcbi.1003740.

5 Guillaume Fertin, Anthony Labarre, Irena Rusu, Eric Tannier, and Stéphane Vialette. *Combinatorics of Genome Rearrangements*. MIT Press, 2009.

6 Eric Letouzé, Yves Allory, Marc A Bollet, François Radvanyi, and Frédéric Guyon. Analysis of the copy number profiles of several tumor samples from the same patient reveals the successive steps in tumorigenesis. *Genome Biology*, 11(7):R76, 2010. doi:10.1186/gb-2010-11-7-r76.

7 Andrew McPherson, Chunxiao Wu, Alexander W Wyatt, Sohrab Shah, Colin Collins, and S Cenk Sahinalp. nFuse: discovery of complex genomic rearrangements in cancer using high-throughput sequencing. *Genome Research*, 22(11):2250–61, nov 2012. doi:10.1101/gr.136572.111.

8 M Mohri. Weighted finite-state transducer algorithms. An overview. *Formal Languages and Applications*, 2004. URL: http://link.springer.com/chapter/10.1007/978-3-540-39886-8_29.

9 Mehryar Mohri. Edit-distance of weighted automata: General definitions and algorithms. *International Journal of Foundations of Computer Science*, 14(06):957–982, 2003.

10 Layla Oesper, Anna Ritz, Sarah J Aerni, Ryan Drebin, and Benjamin J Raphael. Reconstructing cancer genomes from paired-end sequencing data. *BMC Bioinformatics*, 13 Suppl 6(Suppl 6):S10, jan 2012. doi:10.1186/1471-2105-13-S6-S10.

11 D. Pinkel, T. Straume, and J. W. Gray. Cytogenetic analysis using quantitative, high-sensitivity, fluorescence hybridization. *Proceedings of the National Academy of Sciences*, 83(9):2934–2938, may 1986. doi:10.1073/pnas.83.9.2934.

12 Olivier Tremblay Savard, Yves Gagnon, Denis Bertrand, and Nadia El-Mabrouk. Genome halving and double distance with losses. *Journal of Computational Biology*, 18(9):1185–99, 2011. doi:10.1089/cmb.2011.0136.

13 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, apr 2014. doi:10.1371/journal.pcbi.1003535.

14 Mingfu Shao and Yu Lin. Approximating the edit distance for genomes with duplicate genes under DCJ, insertion and deletion. *BMC Bioinformatics*, 13(Suppl 19):S13, 2012. doi:10.1186/1471-2105-13-S19-S13.

15 Eric Tannier, Chunfang Zheng, and David Sankoff. Multichromosomal median and halving problems under different genomic distances. *BMC Bioinformatics*, 10(1):120, 2009. URL: http://www.biomedcentral.com/1471-2105/10/120, doi:10.1186/1471-2105-10-120.

16 The Cancer Genome Atlas Research Network. Integrated genomic analyses of ovarian carcinoma. *Nature*, 474(7353):609–15, jun 2011. doi:10.1038/nature10166.

17 Alexander Eckehart Urban, Jan O Korbel, Rebecca Selzer, Todd Richmond, April Hacker, George V Popescu, Joseph F Cubells, Roland Green, Beverly S Emanuel, Mark B Gerstein, Sherman M Weissman, and Michael Snyder. High-resolution mapping of DNA copy alterations in human chromosome 22 using high-density tiling oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, 103(12):4534–9, mar 2006. doi:10.1073/pnas.0511340103.