Efficient Search of Circular Repeats and MicroDNA Reintegration in DNA Sequences

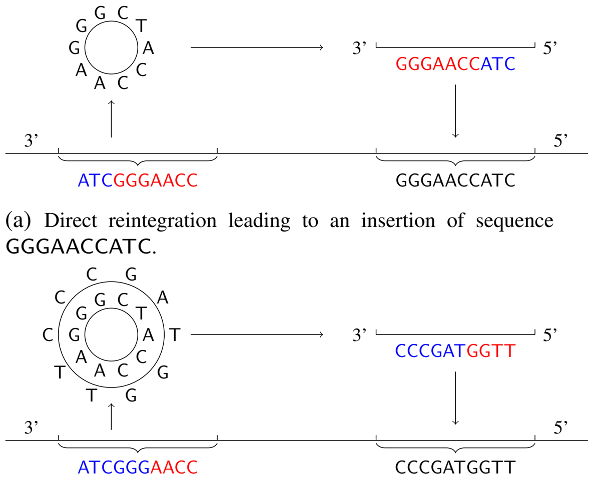
Yiming Wang[*], Hao Lou[†], Pankaj Kumar[‡], Anindya Dutta[‡] Farzad Farnoud[†]
[*]Computer Science, Columbia University, NY. Email: yw3339@columbia.edu
[†] Electrical and Computer Engineering, University of Virginia, VA. Email: {haolou,farzad}@virginia.edu
[‡] Biochemistry and Molecular Genetics, University of Virginia, VA. Email: {pk7z,ad8q}@virginia.edu

***Abstract*—MicroDNAs are a type of extrachromosomal circular DNAs found both in cell nuclei and as cell-free circulating DNA, with links to cancer and genetic mosaicism. Research suggests that microDNAs originate from chromosomal DNA. To better understand the evolutionary role of microDNAs, it is of interest to determine if and how they interact with the chromosomal DNA. In particular, do microDNAs re-integrate back into the chromosomal genome? Given their circular form, if they do, this will lead to a specific form of repeat in the genome, which we term *circular repeat*. Due to the presence of mutations, these repeats are expected to be approximate. Motivated by this question, we develop an efficient *ab initio* algorithm for finding approximate circular repeats in a given genome. The algorithm consists of two main components. First, it performs a two-stage search to locate candidate circular repeat patterns by identifying their substrings. Second, it checks the validity of each candidate by inspecting the flanking sequences of the substrings. By applying our method to human genome chromosomes 21, 22, and Y, we find hundreds of approximate circular repeats. Our simulation shows that the patterns found are unlikely to be purely the result of inherent repetitive structure of the genome, thus suggesting that microDNAs reintegrate back into the genome.**

I. Introduction

Thousands of extrachromosomal circular DNA (eccDNA) have been found in various eukaryotes ranging from yeasts to humans [1]–[6]. The majority of eccDNA in normal cells are small in size and are called microDNA [7]–[9]. MicroDNAs are thought to originate from genomic regions with active chromatin marks [9]. To better understand the function of microDNAs, it is of interest to investigate their interaction with the chromosomal DNA, given the frequent presence of microDNAs in the nuclei of human and mouse cell lines [10]. In particular, do microDNAs reintegrate back into the genome?

The mechanism for the reintegration of microDNAs into the chromosomal DNA may consist of the following stages: i) the circular microDNA splits at a random position and becomes linear, ii) the linear sequence is inserted back into the chromosomal genome at a random position and may subsequently be altered by point mutations, i.e., substitutions and indels. Now if the original microDNA is created as a result of copying a segment of chromosomal DNA rather than excision, this process results in the creation of a *circular repeat* pair. Another possibility is that the microDNA is replicated, which is possible for longer sequences, and two or more copies, after breaking at random positions and becoming linear, reintegrate into the chromosomal genome. Depending on the form of the insertion, the reintegration process can be either *direct* or *inverted*. In Figure 1a, a copy of the segment ATCGGGAACC forms a single-stranded circle. The circle is split between G and C. The linear sequence GGGAACCATC is then inserted back into the genome at some random position, resulting in a *direct circular repeat pair*, namely, ATCGGGAACC and GGGAACCATC. In Figure 1b, a copy of the segment ATCGGGAACC forms a double-stranded circle. The segment ATCGGGAACC is represented by the inner circle and its complement is represented by the outer circle. The complement circle is then split between C and T, becoming the linear sequence CCCGATGGTT, and inserted back into the chromosomal sequence. This results in the *inverted circular repeat pair* ATCGGGAACC and CCCGATGGTT, where ATCGGG is the reverse complement of CCCGAT and AACC of GGTT. Similar repeat patterns emerge if there are multiple copies of the microDNA sequence. Given the presence of mutations, circular repeat pairs resulting from microDNA reintegration will not be exact copies but will rather be approximate. For



(a) Direct reintegration leading to an insertion of sequence GGGAACCATC.

(b) Inverted reintegration leading to an insertion of sequence CCCGATGGTT.

Fig. 1: Reintegration of microDNA ATCGGGAACC.

This work was supported in part by NSF grants under grant nos. CCF-1755773.

simplicity, these mutations are not shown in Figures 1a and 1b.

Motivated by fact that microDNA reintegration may lead to occurrences of approximate circular repeats in genomes, we develop an efficient and parallelizable search algorithm for such repeats and apply it to several chromosomes of the human genome. In our analysis, we found thousands of non-overlapping circular repeats with less than 10% mismatch in chromosomes 21, 22 and Y. Our algorithm finds repeats in an *ab initio* manner, i.e., without any prior knowledge of the sequence. Agreement between properties of sequences that we find and known properties of microDNAs support the reintegration hypothesis.

To the best of our knowledge, the problem of finding circular repeats has not been studied in the literature. The closest related problem is the well-studied problem of finding (linear) repeats. (see surveys [11]–[13]). In particular, maximal repeats can be found using suffix-arrays [14] and suffix-trees [15, Chapter 7].

The rest of the paper is organized as follows. We introduce the necessary notation in Section II. In Section III, we give a detailed description of our search algorithm. Section IV contains analysis results of circular repeats found in human chromosomes. We present a simulation study, which suggests that these repeats are not the result of chance, in Section IV, thus supporting the possibility of reintegration of microDNAs.

II. Preliminaries and Notation

In this paper, all sequences are over the alphabet $\{\mathsf{A},\mathsf{C},\mathsf{G},\mathsf{T}\}$. Vectors and sequences are denoted by boldface letters such as $\boldsymbol{x}$, while scalars and alphabet symbols by plain letters, such as x. For any two sequences $\boldsymbol{w}$ and $\boldsymbol{v}$, the concatenation of $\boldsymbol{w}$ and $\boldsymbol{v}$ is denoted $\boldsymbol{wv}$. The length of $\boldsymbol{w}$ is denoted $|\boldsymbol{w}|$. We shall write $\boldsymbol{w} = w_1 \cdots w_{|\boldsymbol{w}|}$ with w_i denoting the i-th symbol in $\boldsymbol{w}$. For $1 \le i \le j \le |\boldsymbol{w}|$, $\boldsymbol{w}[i:j]$ denotes $w_i w_{i+1} \cdots w_j$. We write $\boldsymbol{w} \in \boldsymbol{v}$ if $\boldsymbol{w}$ is a substring of $\boldsymbol{v}$. We use the terms sequence and string interchangeably.

The *reverse* of a sequence $\boldsymbol{w} = w_1 w_2 \cdots w_{|\boldsymbol{w}|}$ is the sequence $\boldsymbol{w}^{-1} = w_{|\boldsymbol{w}|} \cdots w_2 w_1$. For any symbol x, we use x^c to denote its complement, with A and T being complements of each other and similarly for C and G. The *reverse complement* of $\boldsymbol{w}$, denoted $\bar{\boldsymbol{w}}$, is obtained by reversing $\boldsymbol{w}$ and complementing each symbol. As an example, the reverse complement of $\boldsymbol{w} = \mathsf{ATCCG}$ is $\bar{\boldsymbol{w}} = \mathsf{G}^c\mathsf{C}^c\mathsf{C}^c\mathsf{T}^c\mathsf{A}^c = \mathsf{CGGAT}$.

Since we consider approximate repeats, we adopt the *Levenshtein distance* [16], also known as the *edit distance*, to measure the similarity between two strings. The Levenshtein distance between $\boldsymbol{w}$ and $\boldsymbol{v}$, denoted $\mathrm{lev}(\boldsymbol{w},\boldsymbol{v})$, is the smallest number of point mutations required to transform $\boldsymbol{w}$ to $\boldsymbol{v}$. We define the *mismatch ratio* between $\boldsymbol{w}$ and $\boldsymbol{v}$ as

$$m(\boldsymbol{w},\boldsymbol{v}) = \mathrm{lev}(\boldsymbol{w},\boldsymbol{v})/\min(|\boldsymbol{w}|,|\boldsymbol{v}|).$$

Two identical substrings that can not be further extended form a *maximal repeat pair*. For example, in the sequence $\mathsf{A}\overline{\mathsf{CTT}}\mathsf{GT}\overline{\mathsf{CTT}}\mathsf{A}$, the overlined substrings form a maximal repeat pair of length 3. Similarly, if two substrings are reverse complements of each other and can not be further extended, then they form an inverted maximal repeats pair.

III. Methods

A. Structure of circular repeats

We start by presenting in further detail the sequence structure of circular repeats. Figure 1a shows two possible linear versions, $\mathsf{ATCGGGAACC}$ and $\mathsf{GGGAACCATC}$, of one circular microDNA. The two linear versions can be obtained from each other by reversing the order of substrings ATC and $\mathsf{GGGAACC}$. It can be seen from this example that any exact direct circular repeat pair must be of the form $\boldsymbol{x} = \boldsymbol{s}_1\boldsymbol{s}_2, \boldsymbol{y} = \boldsymbol{s}_2\boldsymbol{s}_1$. Furthermore, any approximate direct circular repeat pair has the form $(\boldsymbol{s}_1\boldsymbol{s}_2, \boldsymbol{s}_2'\boldsymbol{s}_1')$, where $\boldsymbol{s}_1'$ and $\boldsymbol{s}_2'$ are approximate copies of $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$, respectively.

Similarly, as shown in Figure 1b, $\mathsf{ATCGGGAACC}$ and $\mathsf{CCCGATGGTT}$ are linear versions of two circular sequences that are complements of each other respectively, where ATCGGG is the reverse complement of CCCGAT and AACC is the reverse complement of GGTT. It follows that any inverted circular repeat pair has the form $(\boldsymbol{s}_1\boldsymbol{s}_2, \bar{\boldsymbol{s}}_1\bar{\boldsymbol{s}}_2)$ and any inverted approximate circular repeat pair therefore has the form $(\boldsymbol{s}_1\boldsymbol{s}_2, \bar{\boldsymbol{s}}_1'\bar{\boldsymbol{s}}_2')$, where $\bar{\boldsymbol{s}}_1'$ and $\bar{\boldsymbol{s}}_2'$ are approximate copies of $\bar{\boldsymbol{s}}_1$ and $\bar{\boldsymbol{s}}_2$, respectively.

In the following, we use the notation $(\boldsymbol{s}_1\boldsymbol{s}_2, \boldsymbol{s}_2'\boldsymbol{s}_1')$ to refer to a generic direct circular repeat pair and $(\boldsymbol{s}_1\boldsymbol{s}_2, \bar{\boldsymbol{s}}_1'\bar{\boldsymbol{s}}_2')$ to an inverted circular repeat pair.

B. Edit distance criterion

We now describe more precisely what is meant by an approximate circular repeat pair. A straightforward way to define this term would be to say $\boldsymbol{x},\boldsymbol{y}$ are α_0-approximate direct circular repeats if there exist $\boldsymbol{s}_1,\boldsymbol{s}_2,\boldsymbol{s}_1',\boldsymbol{s}_2'$ such that $\boldsymbol{x} = \boldsymbol{s}_1\boldsymbol{s}_2$, $\boldsymbol{y} = \boldsymbol{s}_2'\boldsymbol{s}_1'$, and $m(\boldsymbol{s}_1\boldsymbol{s}_2, \boldsymbol{s}_1'\boldsymbol{s}_2') \le \alpha_0$. For the sake of computational efficiency, we use a variant of this definition. We assume that all bases are nearly equally likely to mutate, and thus the mismatch ratio of $\boldsymbol{s}_1$ and $\boldsymbol{s}_1'$ is close to the mismatch ratio of $\boldsymbol{s}_2$ and $\boldsymbol{s}_2'$. We thus impose the *α-repeat criterion*, i.e., we search for circular repeats satisfying:

$$m(\boldsymbol{s}_1,\boldsymbol{s}_1') \le \alpha, \quad m(\boldsymbol{s}_2,\boldsymbol{s}_2') \le \alpha.$$

For inverted circular repeats, the definition is similar: $m(\boldsymbol{s}_1,\bar{\boldsymbol{s}}_1') \le \alpha$ & $m(\boldsymbol{s}_2,\bar{\boldsymbol{s}}_2') \le \alpha$. A circular repeat pair satisfying the α-repeat criterion is called a *circular α-repeat*.

C. Subroutines

We use the following existing algorithms as subroutines.

Maximal repeat search: To find circular repeats, we adopt the maximal repeat search algorithm from [15, Chapter 7], which is capable of outputting all maximal repeat pairs of length at least ℓ in a sequence $\mathbf{S}$. The time complexity of this algorithm is $O(|\mathbf{S}| + k)$, where k is the number of maximal repeat pairs in $\mathbf{S}$. We use MAX-REP() to denote this procedure. The output of MAX-REP($\mathbf{S}$,ℓ) is a list of tuples , where each tuple $(\boldsymbol{u},\boldsymbol{v})$ represents a maximal repeat pair (or their positions) in $\mathbf{S}$ with $|\boldsymbol{u}| = |\boldsymbol{v}| \ge \ell$. This search algorithm

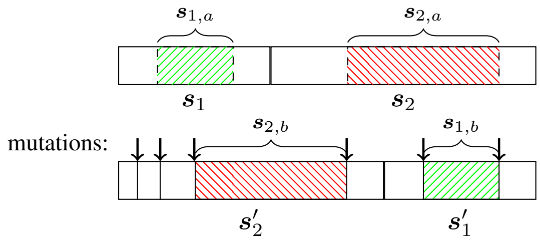


Fig. 2: A visualiztion of s_1 and s'_1 containing common substrings $s_{1,a}$ and $s_{1,b}$, s_2 and s'_2 containing common substrings $s_{2,a}$ and $s_{2,b}$.

can be slightly modified to find maximal repeats between two sequences. In this case, we will write MAX-REP($\mathbf{S}_1\#\mathbf{S}_2$,$\ell$), the output of which is a list of tuples and each tuple $(\boldsymbol{u}, \boldsymbol{v})$ represents a maximal repeat of length $\geq \ell$ with $\boldsymbol{u} \in \mathbf{S}_1$ and $\boldsymbol{v} \in \mathbf{S}_2$. Note that the search of inverted maximal repeat pairs can be achieved by applying MAX-REP() on $\mathbf{S}$ and $\bar{\mathbf{S}}$.

Wagner-Fischer algorithm: We adopt the Wagner-Fischer algorithm for computing edit distances [17].When computing the edit distance between two sequences $\boldsymbol{w}, \boldsymbol{w}'$, Wagner-Fischer generates a matrix $\mathbf{M}_{\boldsymbol{w},\boldsymbol{w}'}$ of size $|\boldsymbol{w}| \times |\boldsymbol{w}'|$ whose (i, j)-th entry equals $\mathrm{lev}(\boldsymbol{w}[1 : i], \boldsymbol{w}'[1 : j])$. After building the matrix, $\mathrm{lev}(\boldsymbol{w}, \boldsymbol{w}')$ is stored in the $(|\boldsymbol{w}|, |\boldsymbol{w}'|)$-th entry. The time complexity is $O(|\boldsymbol{w}| |\boldsymbol{w}'|)$. We write WF() for this procedure; the output of $\mathrm{WF}(\boldsymbol{w}, \boldsymbol{w}')$ is the matrix $\mathbf{M}_{\boldsymbol{w},\boldsymbol{w}'}$.

D. Algorithm outline

For simplicity of presentation, we mainly describe the algorithm for finding direct approximate circular repeats. The search for inverted circular repeats is performed in an analogous way.

Consider the direct approximate circular repeat pair $(\boldsymbol{s}_1\boldsymbol{s}_2, \boldsymbol{s}'_2\boldsymbol{s}'_1)$ given by Figure 2, where we have used arrows to point out positions where $\boldsymbol{s}_1$ and $\boldsymbol{s}'_1$ (respectively, $\boldsymbol{s}_2$ and $\boldsymbol{s}'_2$) differ due to mutations in either sequence (but for clarity let us assume they occurred in $\boldsymbol{s}'_1$). It is clear that although some positions in $\boldsymbol{s}'_1$ are altered, we can still find substrings of $\boldsymbol{s}_1$ and $\boldsymbol{s}'_1$ that form a maximal repeat pair, as the green substrings $\boldsymbol{s}_{1,a}$ and $\boldsymbol{s}_{1,b}$ in Figure 2. Similarly, $\boldsymbol{s}_2$ and $\boldsymbol{s}'_2$ contain a maximal repeat pair $\boldsymbol{s}_{2,a}$ and $\boldsymbol{s}_{2,b}$, marked in red in Figure 2. Taking advantage of this property, our strategy is to first find substrings $\boldsymbol{s}_{1,a}$, $\boldsymbol{s}_{2,a}$, $\boldsymbol{s}_{1,b}$, $\boldsymbol{s}_{2,b}$ and check for ways of extending them to form circular repeats.

The algorithm has two steps, *scanning* and *checking*. In the scanning step, the suffix-tree maximal repeat search algorithm finds in the genome sequence candidates of being substrings of some circular repeat pairs. The checking step then inspects each candidate to determine if it is in fact part of a repeat pair. We present the pseudocode for direct circular repeats search in a given sequence $\mathbf{S}$ in Algorithm 1 and discuss it in detail below.

1) Scanning: Given $\mathbf{S}$, we first search for maximal repeats, as potential substrings of circular repeats, i.e., the strings $\boldsymbol{s}_{1,a}$, $\boldsymbol{s}_{1,b}$, $\boldsymbol{s}_{2,a}$, $\boldsymbol{s}_{2,b}$. The search is composed of two levels. In Algorithm 1, the first-level search is conducted in line 2, where all maximal repeats of length at least ℓ_1 in $\mathbf{S}$ are found. Next, for each maximal repeat pair $(\boldsymbol{u}, \boldsymbol{v})$, we do the second-level search in the neighborhoods of $\boldsymbol{u}$ and $\boldsymbol{v}$ for another pair of maximal repeat which could potentially form a circular repeat with $\boldsymbol{u}$, $\boldsymbol{v}$, and the surrounding elements. The second pair is of length at least ℓ_2, where $\ell_2 \leq \ell_1$. The input L determines the size of these neighborhoods, i.e., the range searched to find the second pair of maximal repeats. Without loss of generality, suppose $\boldsymbol{u}$ is to the left of $\boldsymbol{v}$. We use $\boldsymbol{r}_{\boldsymbol{u}}$ and $\boldsymbol{l}_{\boldsymbol{u}}$ to denote the L-substrings immediately to the right and left of $\boldsymbol{u}$, respectively. Similarly, we use $\boldsymbol{r}_{\boldsymbol{v}}$ and $\boldsymbol{l}_{\boldsymbol{v}}$ to denote the L-substrings immediately to the right and left of $\boldsymbol{v}$, respectively. Guided by the relative position of $\boldsymbol{s}_{1,a}$, $\boldsymbol{s}_{1,b}$, $\boldsymbol{s}_{2,a}$ and $\boldsymbol{s}_{2,b}$ shown in Figure 2, we next search for maximal repeat pairs of length at least ℓ_2 in $\boldsymbol{l}_{\boldsymbol{u}}\#\boldsymbol{r}_{\boldsymbol{v}}$ and $\boldsymbol{r}_{\boldsymbol{u}}\#\boldsymbol{l}_{\boldsymbol{v}}$. The second-level search is conducted in lines 7 and 9. Each $(\boldsymbol{u}, \boldsymbol{v})$ and $(\boldsymbol{w}, \boldsymbol{t})$ found in this way form a candidate for a circular repeat. In line 8 and 10, the candidates are inspected for presence of circular repeats, via the checking module EXT-CHECK().

Algorithm 1 Direct Circular Repeat Pairs Search

```
1:  procedure DI-CIRREP-SEARCH(string $\mathbf{S}$, int $\ell_1$, int $\ell_2$, int $L$, double $\alpha$)
2:      for $(\boldsymbol{u}, \boldsymbol{v})$ in MAX-REP($\mathbf{S}, \ell_1$) do
3:          $\boldsymbol{l}_{\boldsymbol{u}} = L$-substring of $\mathbf{S}$ immediately to left of $\boldsymbol{u}$
4:          $\boldsymbol{r}_{\boldsymbol{u}} = L$-substring of $\mathbf{S}$ immediately to right of $\boldsymbol{u}$
5:          $\boldsymbol{l}_{\boldsymbol{v}} = L$-substring of $\mathbf{S}$ immediately to left of $\boldsymbol{v}$
6:          $\boldsymbol{r}_{\boldsymbol{v}} = L$-substring of $\mathbf{S}$ immediately to right of $\boldsymbol{v}$
7:          for $(\boldsymbol{w}, \boldsymbol{t})$ in MAX-REP($\boldsymbol{l}_{\boldsymbol{u}}\#\boldsymbol{r}_{\boldsymbol{v}}, \ell_2$) do
8:              EXT-CHECK($\mathbf{S}, \alpha, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}, \boldsymbol{t}$)
9:          for $(\boldsymbol{w}, \boldsymbol{t})$ in MAX-REP($\boldsymbol{r}_{\boldsymbol{u}}\#\boldsymbol{l}_{\boldsymbol{v}}, \ell_2$) do
10:             EXT-CHECK($\mathbf{S}, \alpha, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}, \boldsymbol{t}$)
```

The reason for performing maximal repeat search twice with minimum lengths ℓ_1 and ℓ_2, with $\ell_2 \leq \ell_1$, is to improve the efficiency of the algorithm. For the circular repeat $(\boldsymbol{s}_1\boldsymbol{s}_2, \boldsymbol{s}'_2\boldsymbol{s}'_1)$, let c_1 (resp. c_2) denote the length of the longest common substring of $\boldsymbol{s}_1$ and $\boldsymbol{s}'_1$ (resp. $\boldsymbol{s}_2$ and $\boldsymbol{s}'_2$). In our approach, for this repeat to be identified, it suffices to have $\ell_1 \leq \max(c_1, c_2)$ and $\ell_2 \leq \min(c_1, c_2)$. As an alternative to this approach, consider searching for maximal repeats of length at least ℓ to determine the candidate set. In this case, we require $\ell \leq \min(c_1, c_2)$. However, searching for maximal repeats with a smaller minimum length often leads to a much larger number of repeats and thus higher complexity. We overcome this problem by first searching for maximal repeats with length $\geq \ell_1$, leading to a smaller candidate set, and then searching for repeats of length $\geq \ell_2$ only around the repeats in this set, rather than the whole sequence.

2) Checking: After obtaining two maximal repeat pairs $(\boldsymbol{u}, \boldsymbol{v}), (\boldsymbol{w}, \boldsymbol{t})$, where $\boldsymbol{u}$ and $\boldsymbol{v}$ are within the L-neighborhoods of each other, and so are $\boldsymbol{w}$ and $\boldsymbol{t}$, we check if they can be extended to form a circular α-repeat pair.

Let $\mathbf{A}_i, \mathbf{B}_i, 1 \leq i \leq 3$, be as defined in Algorithm 2 and

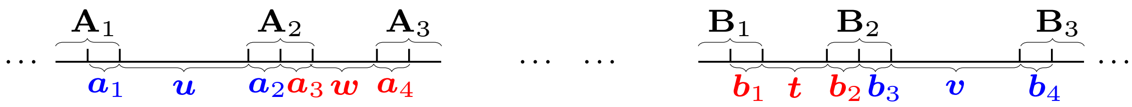


Fig. 3: Relative positions of $\boldsymbol{u}$, $\boldsymbol{v}$, $\boldsymbol{w}$, $\boldsymbol{t}$ and their neighbors.

illustrated in Figure 3. Determining if the candidate is a valid circular repeat is equivalent to finding $\boldsymbol{a}_i, \boldsymbol{b}_i, 1 \leq i \leq 4$ such that $(\boldsymbol{a}_1\boldsymbol{u}\boldsymbol{a}_2\boldsymbol{a}_3\boldsymbol{w}\boldsymbol{a}_4, \boldsymbol{b}_1\boldsymbol{t}\boldsymbol{b}_2\boldsymbol{b}_3\boldsymbol{v}\boldsymbol{b}_4)$ form an approximate circular repeat pair, where $\boldsymbol{a}_2\boldsymbol{a}_3 = \mathbf{A}_2, \boldsymbol{b}_2\boldsymbol{b}_3 = \mathbf{B}_2$, $\boldsymbol{a}_1$ and $\boldsymbol{b}_1$ are suffixes of $\mathbf{A}_1$ and $\mathbf{B}_1$ respectively, and $\boldsymbol{a}_4$ and $\boldsymbol{b}_4$ are prefixes of $\mathbf{A}_3$ and $\mathbf{B}_3$ respectively, as shown in Figure 3. By the α-repeat criterion, we need to check

$$m(\boldsymbol{a}_1\boldsymbol{u}\boldsymbol{a}_2, \boldsymbol{b}_3\boldsymbol{v}\boldsymbol{b}_4) \leq \alpha, \quad m(\boldsymbol{a}_3\boldsymbol{w}\boldsymbol{a}_4, \boldsymbol{b}_1\boldsymbol{t}\boldsymbol{b}_2) \leq \alpha. \qquad (1)$$

Note that the ways of extending two pairs of maximal repeats with (1) satisfied may not be unique. So for circular repeats that overlap significantly, we only output one. On the other hand, the exact edit distance is an important biological statistic. Therefore, the circular repeat pair with minimum total edit distance, i.e.,

$$\operatorname{lev}(\boldsymbol{a}_1\boldsymbol{u}\boldsymbol{a}_2, \boldsymbol{b}_3\boldsymbol{v}\boldsymbol{b}_4) + \operatorname{lev}(\boldsymbol{a}_3\boldsymbol{w}\boldsymbol{a}_4, \boldsymbol{b}_1\boldsymbol{t}\boldsymbol{b}_2), \qquad (2)$$

that satisfies the α-repeat criterion (1) should be identified.

The extension checking can be performed in a brute-force way. More specifically, we can exhaustively compute $\operatorname{lev}(\boldsymbol{a}_1\boldsymbol{u}\boldsymbol{a}_2,\ \boldsymbol{b}_3\boldsymbol{v}\boldsymbol{b}_4)$ and $\operatorname{lev}(\boldsymbol{a}_3\boldsymbol{w}\boldsymbol{a}_4, \boldsymbol{b}_1\boldsymbol{t}\boldsymbol{b}_2)$ for all choices of $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_4, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_4$, and choose the one that minimizes the total edit distance. However, the time complexity is prohibitive. There are in total $O(L)$ ways of splitting $\mathbf{A}_2$ to obtain $\boldsymbol{a}_2, \boldsymbol{a}_3$ and also $O(L)$ ways of splitting $\mathbf{B}_2$ to obtain $\boldsymbol{b}_2, \boldsymbol{b}_3$. Fixing $\boldsymbol{a}_2$, $\boldsymbol{a}_3$, $\boldsymbol{b}_2$, $\boldsymbol{b}_3$, we have $O(L)$ ways of choosing each of $\boldsymbol{a}_1$, $\boldsymbol{b}_4$, $\boldsymbol{b}_1$, $\boldsymbol{a}_4$ to compute the edit distances. Thus, the process of computing edit distance needs to be performed $O(L^2(L^2 + L^2)) = O(L^4)$ times. Since each edit distance takes $O(L^2)$ to compute by the Wagner-Fischer algorithm, the complexity of checking for extension on each candidate $(\boldsymbol{u}, \boldsymbol{v}), (\boldsymbol{w}, \boldsymbol{t})$ is $O(L^6)$, which is unfeasible even for moderate values of L.

To reduce running time, we adopt two heuristic assumptions, described below. The pseudocode of the checking procedure utilizing these heuristics, for the case when the order of appearance of these substrings in $\mathbf{S}$ is $\boldsymbol{u}$, $\boldsymbol{w}$, $\boldsymbol{t}$, $\boldsymbol{v}$, is shown in Algorithm 2. First, we assume

$$\begin{aligned} \operatorname{lev}(\boldsymbol{a}_1\boldsymbol{u}\boldsymbol{a}_2, \boldsymbol{b}_3\boldsymbol{v}\boldsymbol{b}_4) &= \operatorname{lev}(\boldsymbol{a}_1, \boldsymbol{b}_3) + \operatorname{lev}(\boldsymbol{a}_2, \boldsymbol{b}_4), \\ \operatorname{lev}(\boldsymbol{a}_3\boldsymbol{w}\boldsymbol{a}_4, \boldsymbol{b}_1\boldsymbol{t}\boldsymbol{b}_2) &= \operatorname{lev}(\boldsymbol{a}_3, \boldsymbol{b}_1) + \operatorname{lev}(\boldsymbol{a}_4, \boldsymbol{b}_2). \end{aligned} \qquad (3)$$

Note that $\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}, \boldsymbol{t}$ are candidates for substrings of the circular repeat not affected by mutations ($\boldsymbol{s}_{1,a}, \boldsymbol{s}_{1,b}, \boldsymbol{s}_{2,a}, \boldsymbol{s}_{2,b}$ in Figure 2). Furthermore, the substrings on their sides mutate independently from each other. Thus, the right side of (3) also provides a reasonable proxy for the number of mutations in the repeat. Using (3), (2) can be replaced by

$$\operatorname{lev}(\boldsymbol{a}_1, \boldsymbol{b}_3) + \operatorname{lev}(\boldsymbol{a}_2, \boldsymbol{b}_4) + \operatorname{lev}(\boldsymbol{a}_3, \boldsymbol{b}_1) + \operatorname{lev}(\boldsymbol{a}_4, \boldsymbol{b}_2). \qquad (4)$$

Algorithm 2 Checking

```
procedure EXT-CHECK(string $\mathbf{S}$, double $\alpha$, string $\boldsymbol{u}$, string $\boldsymbol{v}$, string $\boldsymbol{w}$, string $\boldsymbol{t}$)
    $\mathbf{A}_1$ = substring of $\mathbf{S}$ immediately to the left of $\boldsymbol{u}$
    $\mathbf{A}_2$ = substring of $\mathbf{S}$ between $\boldsymbol{u}$ and $\boldsymbol{w}$
    $\mathbf{A}_3$ = substring of $\mathbf{S}$ immediately to the right of $\boldsymbol{w}$
    $\mathbf{B}_1$ = substring of $\mathbf{S}$ immediately to the left of $\boldsymbol{t}$
    $\mathbf{B}_2$ = substring of $\mathbf{S}$ between $\boldsymbol{t}$ and $\boldsymbol{v}$
    $\mathbf{B}_3$ = substring of $\mathbf{S}$ immediately to the right of $\boldsymbol{v}$
    $\mathbf{M}_1 = \mathrm{WF}(\mathbf{A}_1^{-1}[1 : |\mathbf{B}_2|], \mathbf{B}_2^{-1})$
    $\mathbf{M}_2 = \mathrm{WF}(\mathbf{A}_2, \mathbf{B}_3[1 : |\mathbf{A}_2|])$
    $\mathbf{M}_3 = \mathrm{WF}(\mathbf{A}_2^{-1}, \mathbf{B}_1^{-1}[1 : |\mathbf{A}_2|])$
    $\mathbf{M}_4 = \mathrm{WF}(\mathbf{A}_3[1, |\mathbf{B}_2|], \mathbf{B}_2)$
    $m = \arg\min_x \mathbf{M}_2(x) + \mathbf{M}_3(|\mathbf{A}_2| - x)$
    $n = \arg\min_x \mathbf{M}_1(|\mathbf{B}_2| - x) + \mathbf{M}_4(x)$
    $d_1 = \mathbf{M}_1(|\mathbf{B}_2| - n) + \mathbf{M}_2(m)$
    $d_2 = \mathbf{M}_3(|\mathbf{A}_2| - m) + \mathbf{M}_4(n)$
    $r_1 = d_1/\min(|\boldsymbol{a}_1\boldsymbol{s}_{i,1}\boldsymbol{a}_2|, |\boldsymbol{b}_3\boldsymbol{s}_{i,2}\boldsymbol{b}_4|)$
    $r_2 = d_2/\min(|\boldsymbol{a}_3\boldsymbol{s}_{j,a}\boldsymbol{a}_4|, |\boldsymbol{b}_1\boldsymbol{s}_{j,b}\boldsymbol{b}_2|)$
    if $r_1 \leq \alpha$ and $r_2 \leq \alpha$ then
        Output $(\boldsymbol{a}_1\boldsymbol{s}_{i,a}\boldsymbol{a}_2\boldsymbol{a}_3\boldsymbol{s}_{j,a}\boldsymbol{a}_4, \boldsymbol{b}_1\boldsymbol{s}_{j,b}\boldsymbol{b}_2\boldsymbol{b}_3\boldsymbol{s}_{i,b}\boldsymbol{b}_4)$, $d_1 + d_2$
```

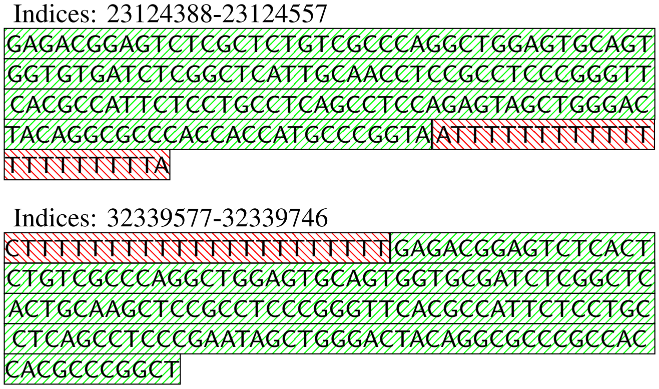
Second, we also assume that for every valid circular repeat,

$$|\boldsymbol{a}_1| = |\boldsymbol{b}_3|, \quad |\boldsymbol{a}_2| = |\boldsymbol{b}_4|, \quad |\boldsymbol{a}_3| = |\boldsymbol{b}_1|, \quad |\boldsymbol{a}_4| = |\boldsymbol{b}_2|. \qquad (5)$$

The purpose of making this assumption is to avoid dealing with all possible $\boldsymbol{a}_1$, $\boldsymbol{a}_4$, $\boldsymbol{b}_1$, $\boldsymbol{b}_4$ of different lengths. This assumption is rather intuitive: we can extend strings that are not of the same length with the cost of edit distance. So if there is some underlying repeat, e.g., with $|\boldsymbol{a}_1| \neq |\boldsymbol{b}_3|$, it can still be reported by a suitable α. With assumption (5), to minimize (4), it suffices to find the optimal $|\boldsymbol{a}_2|$ and $|\boldsymbol{b}_2|$ (the splitting point for $\mathbf{A}_2$ and $\mathbf{B}_2$) since $\boldsymbol{a}_2, \boldsymbol{a}_3, \boldsymbol{b}_4, \boldsymbol{b}_1$ (resp. $\boldsymbol{b}_2, \boldsymbol{b}_3, \boldsymbol{a}_1, \boldsymbol{a}_4$) are uniquely determined by $|\boldsymbol{a}_2|$ (resp. $|\boldsymbol{b}_2|$). Moreover, it is clear that the choice of $|\boldsymbol{a}_2|$ is independent of $|\boldsymbol{b}_2|$.

In order to find the optimal $|\boldsymbol{a}_2|$ and $|\boldsymbol{b}_2|$, we first construct the edit distance matrices by Wagner-Fischer algorithm of the following four pairs of strings: $(\mathbf{A}_1^{-1}[1 : |\mathbf{B}_2|], \mathbf{B}_2^{-1})$, $(\mathbf{A}_2, \mathbf{B}_3[1 : |\mathbf{A}_2|])$, $(\mathbf{A}_2^{-1}, \mathbf{B}_1^{-1}[1 : |\mathbf{A}_2|])$ and $(\mathbf{A}_3[1, |\mathbf{B}_2|], \mathbf{B}_2)$. Denote the matrices by $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$, $\mathbf{M}_4$, respectively. For a matrix $\mathbf{M}$, let $\mathbf{M}(k)$ denote the k-th diagonal entry of $\mathbf{M}$. It is easy to see that for $|\boldsymbol{a}_2| = m, |\boldsymbol{b}_2| = n$,

$$\begin{aligned} \mathbf{M}_1(|\mathbf{B}_2| - n) &= \operatorname{lev}(\boldsymbol{a}_1, \boldsymbol{b}_3), & \mathbf{M}_2(m) &= \operatorname{lev}(\boldsymbol{a}_2, \boldsymbol{b}_4), \\ \mathbf{M}_3(|\mathbf{A}_2| - m) &= \operatorname{lev}(\boldsymbol{a}_3, \boldsymbol{b}_1), & \mathbf{M}_4(n) &= \operatorname{lev}(\boldsymbol{a}_4, \boldsymbol{b}_2). \end{aligned}$$



Indices: 23124388-23124557

GAGACGGAGTCTCGCTCTGTCGCCCAGGCTGGAGTGCAGT
GGTGTGATCTCGGCTCATTGCAACCTCCGCCTCCCGGGTT
CACGCCATTCTCCTGCCTCAGCCTCCAGAGTAGCTGGGAC
TACAGGCGCCCACCACCATGCCCCGGTA ATTTTTTTTTTTT
TTTTTTTTTA

Indices: 32339577-32339746

CTTTTTTTTTTTTTTTTTTTTTT GAGACGGAGTCTCACT
CTGTCGCCCAGGCTGGAGTGCAGTGGTGCGATCTCGGCTC
ACTGCAAGCTCCGCCTCCCGGGTTCACGCCATTCTCCTGC
CTCAGCCTCCCGAATAGCTGGGACTACAGGCGCCCGCCAC
CACGCCCGGCT

Fig. 4: Display of a circular repeat in chr-21.

After obtaining the matrices, we go over all values of m and n separately and find the values that minimize the sum of corresponding entries in $\mathbf{M}_1$, $\mathbf{M}_2$, $\mathbf{M}_3$, $\mathbf{M}_4$.

E. Complexity analysis

Let $\mathbf{S}$, L, ℓ_1, ℓ_2 be as defined in §III-D. We use k_1 to denote the number of maximal repeat pairs found in $\mathbf{S}$ with length at least ℓ_1 and use k_2 to denote the maximum (worst-case) number of maximal repeats found in second-level search for all $(\boldsymbol{u}, \boldsymbol{v})$. The overall time complexity of our algorithm is $O(|\mathbf{S}| + k_1(L + k_2 L^2)) = O(|\mathbf{S}| + k_1 k_2 L^2)$.

The space complexity is dominated by the size of the suffix tree used for the first-level search, which is $O(|\mathbf{S}|)$. The constant in $O(|\mathbf{S}|)$ is important and given the size of some genomes, space complexity may become the bottleneck. We thus provide a partitioning method that alleviates this issue and also allows us to parallelize the algorithm, described in §III-H.

F. Program output

For each circular repeat $(\boldsymbol{s}_1\boldsymbol{s}_2, \boldsymbol{s}_2'\boldsymbol{s}_1')$ that it finds, our program outputs the following information:

- Starting indices of $\boldsymbol{s}_1\boldsymbol{s}_2$ and $\boldsymbol{s}_2'\boldsymbol{s}_1'$.
- Lengths of $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$, (equal to $|\boldsymbol{s}_1'|, |\boldsymbol{s}_2'|$, respectively).
- Mismatch ratio of the pair $(\boldsymbol{s}_1\boldsymbol{s}_2, \boldsymbol{s}_2'\boldsymbol{s}_1')$, the pair $(\boldsymbol{s}_1, \boldsymbol{s}_1')$, and the pair $(\boldsymbol{s}_2, \boldsymbol{s}_2')$.

As an example, the following 7-tuple,

$$(23124388, 32339577, 147, 23, 0.071, 0.068, 0.087, 170),$$

represents a direct circular repeat we found in chromosome 21. Indices in the tuple are indices in the sequence after tandem repeat removal (See §III-G). We show this repeat in Figure 4, where counterparts are marked in the same colored patterns.

G. Preprocessing

A tandem repeat is a string of nucleotides consisting of multiple consecutive occurrences of a substring called the motif. For example, the sequence ACTACTACT is a tandem repeat with motif ACT. Substrings of tandem repeats can form circular repeats. In this example, the underlined part and the overlined part in $\overline{\text{ACT}}\text{AC}\underline{\text{TAC}}\text{T}$ are a direct circular repeat pair. To avoid reporting tandem repeats as circular repeat pairs, we provide the option of removing tandem repeats before circular repeat search by using Tandem Repeat Finder (TRF) [18]. In all results presented in this paper, tandem repeats are removed first.

H. Partitioning and parallelization

Depending on the length of the input sequence, the bottleneck of our algorithm may be the space required to build the suffix trees. To overcome this problem, we devise a divide and conquer approach. A helper program is first used to equally divide the long input sequence $\mathbf{S}$ into n chunks, i.e., $\mathbf{S} = \mathbf{S}_1\mathbf{S}_2 \cdots \mathbf{S}_n$, with n being large enough so that the lengths of $\mathbf{S}_i$'s are manageable.

The algorithm is first executed on each $\mathbf{S}_i$ separately, as described in Section III. In this way, we are able to find all circular repeat pairs with both components in the same chunk. In order to find repeat pairs with components located in different chunks, we perform a slightly modified search. For pairs $(\mathbf{S}_j, \mathbf{S}_k)$ where $k > j$, in the first-level search in Step 2 of Algorithm 1, we only process maximal repeat pairs $(\boldsymbol{u}, \boldsymbol{v})$ with $\boldsymbol{u}$ in $\mathbf{S}_j$ and $\boldsymbol{v}$ in $\mathbf{S}_k$. We note that a small number of circular repeats at the boundary of chunks may be left out in this method. This issue can be resolved with a slightly more complex approach of dividing $\mathbf{S}$ into overlapping substrings.

This strategy allows us to search for circular repeats multiple times on short segments instead of on the entire genome. Note that the parallelization will increase the time complexity because maximal repeats completely contained in each $\mathbf{S}_i$ will be visited multiple times during the first-level search. However, this increase is limited, because these repeat pairs will not be processed in second-level search and checking, which cost most of the computation.

IV. Results

In this section, we present the results of applying the proposed algorithm to chromosomes in the human genome. We will also present a simulation study aimed at investigating the statistical significance of the findings.

A. Circular repeat search in the human genome

We apply our program to three chromosomes of the human genome: chromosome-Y (chr-Y), chromosome-21 (chr-21) and chromosome-22 (chr-22).[1] We present below the results with algorithm parameters $\ell_1 = 40$, $\ell_2 = 20$, $L = 800$ and mismatch ratio $\alpha = 0.1$. Note that L is the size of the neighborhood for the second level search, and is not an upper bound for the length of circular repeats.

The lengths of the three genome sequences after removing tandem repeat regions are given in Table I. Table II gives the number of maximal repeats and maximal inverted repeats of lengths over 40. The total number of direct (resp. inverted) circular repeats found in the three genome sequences is given

[1] The DNA sequences are available on https://www.ncbi.nlm.nih.gov/nuccore. The NCBI reference number for chr-Y, chr-21, chr-22 are NC_000024.9, NC_000021.8 and NC_000022.9, respectively.

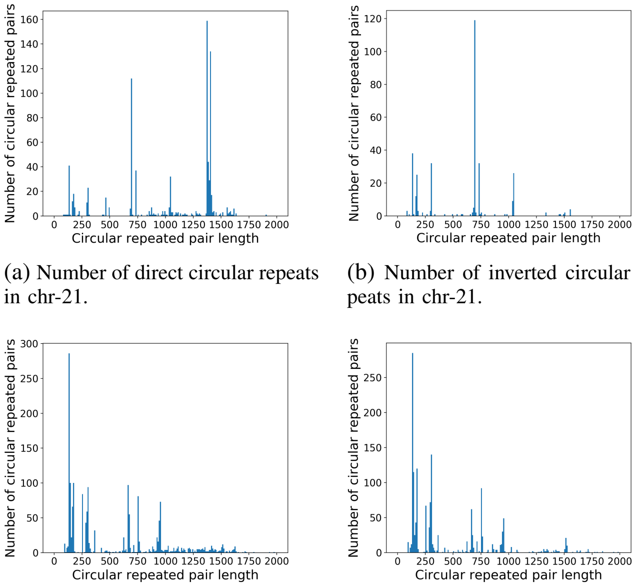
Genome	Length after removing tandem repeat regions (bp)
chr-Y	23974895
chr-22	35018961
chr-21	36454203

TABLE I: Lengths of chr-Y, chr-22 and chr-21 after tandem repeat regions removal.

Genome	Number of maximal repeat pairs of length ≥ 40	Number of maximal inverted repeat pairs of length ≥ 40
chr-Y	544030	540270
chr-22	1764437	1762105
chr-21	672453	670857

TABLE II: Number of maximal repeat and inverted repeat pairs found in chr-Y, chr-22 and chr-21.

in the first column of Table III (resp. Table IV). Note that our algorithm reports circular repeats that overlap with others. However, overlapping circular repeats are likely to be parts of a single reintegration pattern. To achieve more accuracy, for two circular repeats, e.g., $(\boldsymbol{x}_1, \boldsymbol{y}_1)$ and $(\boldsymbol{x}_2, \boldsymbol{y}_2)$ (assuming $\boldsymbol{x}_1$ is to the left of $\boldsymbol{y}_1$ and $\boldsymbol{x}_2$ is to the left of $\boldsymbol{y}_2$), if $\boldsymbol{x}_1$ overlaps with $\boldsymbol{x}_2$ and $\boldsymbol{y}_1$ overlaps with $\boldsymbol{y}_2$, then we count only one of them. The second columns of Table III and IV show the number of direct and inverted circular repeats after removing the overlaps, respectively.



(a) Number of direct circular repeats in chr-21.

(b) Number of inverted circular repeats in chr-21.

(c) Number of direct circular repeats in chr-22.

(d) Number of inverted circular repeats in chr-22.

Fig. 5: Length distribution of circular repeat (non-overlapping) found with program parameters $\ell_1 = 40$, $\ell_2 = 20$, $L = 800$, $\alpha = 0.1$.

Length Distribution and GC Content: Figures 5 and 6 show the distributions of length and GC content of the non-overlapping circular repeats found in chr-21 and chr-22. It can be observed that in chr-21 and chr-22, for both direct and inverted circular repeats, the length distribution has peaks around 150 bp and 300 bp, as well as around 700 bp . These peaks coincide with those in the length distribution of microDNAs found experimentally as reported in [8] and also given in Figure 7).[2] This agreement provides evidence that microDNAs do reintegrate back into the genome. However, we can also observe that the circular repeat lengths in our result also concentrate around some other values, e.g., 1000 and 1400 bp, which are absent from Figure 7. Furthermore, there are peaks in Figure 7 at 500-600 bp, which are absent from all but one of our plots. The absence of long microDNAs from experimental results may be due to the difficulty of their amplification from random primer. Given that these longer microDNAs can independently replicate, it seems possible that a larger number of them would be reintegrated back into the chromosomal genome.

Genome	Direct circular repeats	Non-overlapping repeats (NOR)	Percentage of NOR with microhomology
chr-Y	6137	2126	62.5%
chr-22	3419	1749	53.1%
chr-21	4410	846	58.2%

TABLE III: Number of direct circular repeats found in chr-Y, chr-22 and chr-21.

Genome	Inverted circular repeats	Non-overlapping repeats (NOR)	Percentage of NOR with microhomology
chr-Y	6626	1816	17.1%
chr-22	3091	1525	30.9%
chr-21	427	341	17.9%

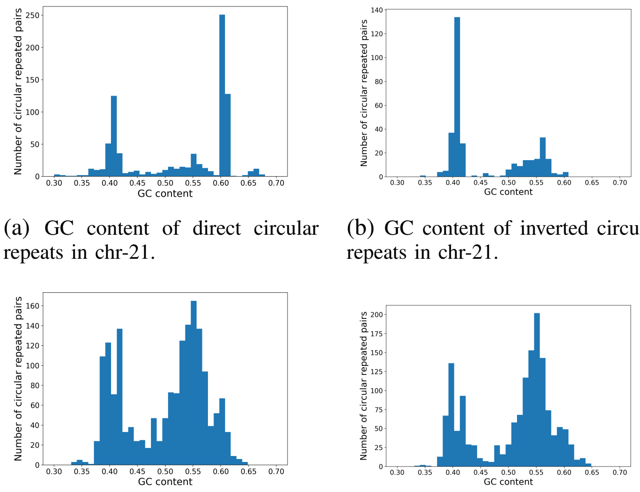
TABLE IV: Number of inverted circular repeats found in chr-Y, chr-22 and chr-21.

Our GC content distribution has peaks at around 40% and around 55%. The results reported in [8] only show a peak at 55%. The discrepancies in length distribution and GC content may result from the presence of other mechanisms for the creation of circular repeats, factors affecting their re-integration frequency, as well as the effect of the choice of the program parameters. Overall the agreement between our computational results and the available data on length distribution and GC content suggests that microDNAs interact with the genome.

Microhomology: It is reported in [19]–[21] that some microDNA sequences are flanked on both sides by a repeat of an average length of 9-11 bp. Moreover, 2 to 15 bp repeats of microhomology at both ends of microDNAs are found to be enriched in all mouse tissues and human cell lines [8, Fig. 2D]. The presence of microhomology suggests production mechanisms for microDNAs [8], which is of significant biological importance.

Therefore, we also study the presence of microhomology in circular repeats found by our program. We adopt the setting

[2] The data is available at: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSM880928



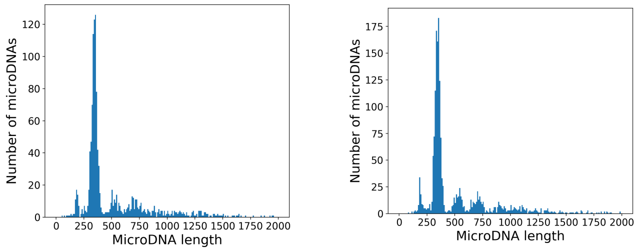
(a) GC content of direct circular repeats in chr-21.

(b) GC content of inverted circular repeats in chr-21.

(c) GC content of direct circular repeats in chr-22.

(d) GC content of inverted circular repeats in chr-22.

Fig. 6: GC content distribution of circular repeated pairs (non-overlapping) found with program parameters $\ell_1 = 40$, $\ell_2 = 20$, $L = 800$, $\alpha = 0.1$.



(a) MicroDNA length distribution in chr-21.

(b) MicroDNA length distribution in chr-22.

Fig. 7: Length distribution of microDNAs reported in [8].

from [8] that a (direct or inverted) circular repeat pair $(\boldsymbol{x}, \boldsymbol{y})$ contains microhomology if any one of $\boldsymbol{x}, \boldsymbol{y}$ has the form shown in Figure 8: That is, repeats of 2 to 15 bp (red letters) exist at the junction of the repeat string (uppercase) with flanking string (lowercase). The percentage of non-overlapping circular repeats found by our algorithm that contain microhomology is given in the third columns of Table III and IV.

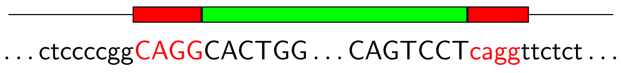


Fig. 8: An example of a circular repeat string containing a repeat of microhomology CAGG at the beginning and the end.

B. Simulation

In this section, we investigate whether the reported circular repeat pairs are in fact results of microDNA reintegration. An alternative hypothesis is circular repeats occur due to a prevalence of repeated sequences. As an extreme example, consider a sequence that is the concatenation of copies of $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$, for some $\boldsymbol{s}_1, \boldsymbol{s}_2$. In such a sequence, any occurrence of $\boldsymbol{s}_1\boldsymbol{s}_2$ and any occurrence of $\boldsymbol{s}_2\boldsymbol{s}_1$ form a circular repeat pair. Similarly, for a repeat-rich genome, any two maximal repeat pairs which happen to be located close enough may also form approximate circular repeat pairs. Therefore, we study whether it is likely to observe as many repeats as we reported by chance.

We run simulations on chr-21 to test the validity of circular repeats being results of reintegration. Our strategy is perturbing the sequence of chr-21 in a way that eliminates existing circular repeat structures while keeping the number of maximal repeats unchanged. We will then perform our algorithm again on the modified sequence with the same parameters, i.e., $\ell_1 = 40, \ell_2 = 20, L = 800, \alpha = 0.1$, and record the number of circular repeats found. We perform this several times. If the number of circular repeats found after perturbation is significantly smaller than that of the original sequence, this provides evidence that the circular repeats found in the original genome are not the result of the prevalence of repeats and random chance.

In the simulation, we decompose chr-21 sequence into segments of repeat regions and non-repeat regions. Specifically, we locate all maximal repeats and decompose chr-21 into segments that are either covered by some maximal repeat or contain no repeats. We choose maximal repeats of minimum length 40 to be consistent with our choice of $\ell_1 = 40$. After decomposition, we generate new sequences by randomly permuting the segments. We obtained 10 perturbed sequences by this procedure. On average, 265 non-overlapping direct circular repeats are found with a standard deviation of 17.

Due to the high computational cost of the simulation, the number of perturbed sequences is small, so we do not report the p-values, but the magnitude of the difference provides evidence against the null hypotheses. In particular, in the original chr-21 sequence, 846 non-overlapping direct circular repeats were found, nearly 3 times the average number in the simulation. In summary, the number of circular repeats found in perturbed sequences is much less than that in the original genome sequence, indicating that circular repeat pairs are unlikely to be solely products of the inherent repetitive structure in genomic sequences.

V. Conclusion

In this paper, we described a model for microDNA reintegration and the resulting insertions of circular repeats, and presented an algorithm that efficiently searches for such circular repeats in genomic sequences. We performed a search in human genome chr-Y, chr-22 and chr-21 and found thousands of circular repeats. We also performed simulations that indicate that it is unlikely that circular repeats are the result of high repeat content and random chance.

References

[1] H. D. Mller, C. E. Larsen, L. Parsons, A. J. Hansen, B. Regenberg, and T. Mourier, "Formation of extrachromosomal circular dna from long terminal repeats of retrotransposons in saccharomyces cerevisiae," *G3: Genes, Genomes, Genetics*, vol. 6, no. 2, pp. 453–462, 2016.

[2] H. D. Mller, L. Parsons, T. S. Jrgensen, D. Botstein, and B. Regenberg, "Extrachromosomal circular dna is common in yeast," *Proceedings of the National Academy of Sciences*, vol. 112, no. 24, E3114–E3122, 2015.

[3] H. D. Mller, M. Mohiyuddin, I. Prada-Luengo, M. R. Sailani, J. F. Halling, P. Plomgaard, L. Maretty, A. J. Hansen, M. P. Snyder, H. Pilegaard, *et al.*, "Circular dna elements of chromosomal origin are common in healthy human somatic tissue," *Nature communications*, vol. 9, no. 1, pp. 1–12, 2018.

[4] A. C. Decarvalho, H. Kim, L. M. Poisson, M. E. Winn, C. Mueller, D. Cherba, J. Koeman, S. Seth, A. Protopopov, M. Felicella, *et al.*, "Discordant inheritance of chromosomal and extrachromosomal dna elements contributes to dynamic disease evolution in glioblastoma," *Nature genetics*, vol. 50, no. 5, pp. 708–717, 2018.

[5] M. J. Shoura, I. Gabdank, L. Hansen, J. Merker, J. Gotlib, S. D. Levene, and A. Z. Fire, "Intricate and cell type-specific populations of endogenous circular dna (eccdna) in caenorhabditis elegans and homo sapiens," *G3: Genes, Genomes, Genetics*, vol. 7, no. 10, pp. 3295–3303, 2017.

[6] K. M. Turner, V. Deshpande, D. Beyter, T. Koga, J. Rusert, C. Lee, B. Li, K. Arden, B. Ren, D. A. Nathanson, *et al.*, "Extrachromosomal oncogene amplification drives tumour evolution and genetic heterogeneity," *Nature*, vol. 543, no. 7643, pp. 122–125, 2017.

[7] P. Kumar, L. W. Dillon, Y. Shibata, A. A. Jazaeri, D. R. Jones, and A. Dutta, "Normal and cancerous tissues release extrachromosomal circular dna (eccdna) into the circulation," *Molecular Cancer Research*, vol. 15, no. 9, pp. 1197–1205, 2017.

[8] Y. Shibata, P. Kumar, R. Layer, S. Willcox, J. R. Gagan, J. D. Griffith, and A. Dutta, "Extrachromosomal microdnas and chromosomal microdeletions in normal tissues," *Science*, vol. 336, no. 6077, pp. 82–86, 2012.

[9] L. W. Dillon, P. Kumar, Y. Shibata, Y.-H. Wang, S. Willcox, J. D. Griffith, Y. Pommier, S. Takeda, and A. Dutta, "Production of extrachromosomal microdnas is linked to mismatch repair pathways and transcriptional activity," *Cell reports*, vol. 11, no. 11, pp. 1749–1759, 2015.

[10] P. Kumar, S. Kiran, S. Saha, Z. Su, T. Paulsen, A. Chatrath, Y. Shibata, E. Shibata, and A. Dutta, "Atac-seq identifies thousands of extrachromosomal circular dna in cancer and cell lines," *Science Advances*, vol. 6, no. 20, eaba2489, 2020.

[11] E. Lerat, "Identifying repeats and transposable elements in sequenced genomes: How to find your way through the dense forest of programs," *Heredity*, vol. 104, no. 6, pp. 520–533, 2010.

[12] S. Saha, S. Bridges, Z. V. Magbanua, and D. G. Peterson, "Computational approaches and tools used in identification of dispersed repetitive dna sequences," *Tropical Plant Biology*, vol. 1, no. 1, pp. 85–96, 2008.

[13] ——, "Empirical comparison of ab initio repeat finding programs," *Nucleic acids research*, vol. 36, no. 7, pp. 2284–2294, 2008.

[14] V. Becher, A. Deymonnaz, and P. Heiber, "Efficient computation of all perfect repeats in genomic sequences of up to half a gigabyte, with a case study on the human genome," *Bioinformatics*, vol. 25, no. 14, pp. 1746–1753, 2009.

[15] D. Gusfield, "Algorithms on stings, trees, and sequences: Computer science and computational biology," *Acm Sigact News*, vol. 28, no. 4, pp. 41–60, 1997.

[16] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, vol. 10, 1966, pp. 707–710.

[17] R. A. Wagner and M. J. Fischer, "The string-to-string correction problem," *Journal of the ACM (JACM)*, vol. 21, no. 1, pp. 168–173, 1974.

[18] G. Benson, "Tandem repeats finder: A program to analyze dna sequences," *Nucleic acids research*, vol. 27, no. 2, pp. 573–580, 1999.

[19] S. W. Stanfield and J. A. Lengyel, "Small circular dna of drosophila melanogaster: Chromosomal homology and kinetic complexity," *Proceedings of the National Academy of Sciences*, vol. 76, no. 12, pp. 6142–6146, 1979.

[20] P. Sunnerhagen, R.-M. Sj
berg, A.-L. Karlsson, L. Lundh, and G. Bjursell, "Molecular cloning and characterization of small polydisperse circular dna from mouse 3t6 cells," *Nucleic acids research*, vol. 14, no. 20, pp. 7823–7838, 1986.

[21] S. W. Stanfield and D. R. Helinski, "Cloning and characterization of small circular dna from chinese hamster ovary cells.," *Molecular and cellular biology*, vol. 4, no. 1, pp. 173–180, 1984.