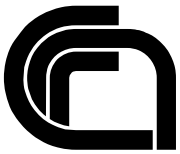


Consiglio Nazionale delle Ricerche

TRStalker: an Efficient Heuristic for Finding NP-Complete Tandem Repeats

M. Pellegrini, M. E. Renda, A. Vecchio

IIT TR-08/2009

Technical report

Ottobre 2009

Istituto di Informatica e Telematica

TRStalker: an Efficient Heuristic for Finding NP-Complete Tandem Repeats

Marco Pellegrini[1], M. Elena Renda[1], and Alessio Vecchio[2]

[1] CNR, Istituto di Informatica e Telematica, Via Moruzzi 1, 56124 Pisa (Italy).
marco.pellegrini@iit.cnr.it, elena.renda@iit.cnr.it
[2] Univ. di Pisa, Dip. Ingegneria dell'Informazione, Via Diotisalvi 2, 56122 Pisa (Italy).
a.vecchio@ing.unipi.it

Abstract. Genomic sequences in higher eucaryotic organisms contain a substantial amount of (almost) repeated sequences. Tandem Repeats (TRs) constitute a large class of repetitive sequences that are originated via phenomena such as replication slippage, are characterized by close spatial contiguity, and play an important role in several molecular regulatory mechanisms. Certain types of tandem repeats are highly polymorphic and constitute a fingerprint feature of individuals. Abnormal TRs are known to be linked to several diseases. Researchers in bio-informatics in the last 20 years have proposed many formal definitions for the rather loose notion of a Tandem Repeat and have proposed exact or heuristic algorithms to detect TRs in genomic sequences. The general trend has been to use formal (implicit or explicit) definitions of TR for which verification of the solution was easy (with complexity linear, or polynomial in the TR's length and substitution+indel rates) while the effort was directed towards identifying efficiently the sub-strings of the input to submit to the verification phase (either implicitly or explicitly). In this paper we take a step forward: we use a definition of TR for which also the verification step is difficult (in effect, NP-complete) and we develop new filtering techniques for coping with high error levels. The resulting heuristic algorithm, christened *TRStalker*, is approximate since it cannot guarantee that all NP-Complete Tandem Repeats satisfying the target definition in the input string will be found. However, in synthetic experiments with 30% of errors allowed, TRStalker has demonstrated a very high recall (ranging from 100% to 60%, depending on motif length and repetition number) for the NP-complete TRs. TRStalker has consistently better performance than some state-of-the-art methods for a large range of parameters on the class of NP-complete Tandem Repeats. TRStalker aims at improving the capability of TR detection for classes of TRs for which existing methods do not perform well.

1 Introduction

Tandem Repeats (TRs) are multiple (two or more) duplications of substrings in the DNA that occur contiguously, and may involve some base mutations (such as substitutions, insertions, and deletions). Tandem Repeats of several forms (satellites, microsatellites, minisatellites, and others) have been studied extensively because of their role in several biological processes[3]. In fact, TRs are privileged targets in activities such as fingerprinting or tracing the evolution of populations [KTCM08,VKN$^+$06]. Several diseases, disorders and addictive behaviors are linked to specific TRs loci [WCJC$^+$94]. The role of TRs has been studied also within coding regions [OEPS05] and in relation to gene functions [LPPV07]. Large scale Tandem Repeats comparative studies of the human genome are described in [DNT$^+$08,WHG$^+$08]. Data Bases of repetitive elements such as RepBase [JKP$^+$05] and TRDB [GRB07] are now available; and the detection of repetitive elements via library-based similarity matching, for example by using the tool Repeatmasker[4], is a popular practice. However, tools for *ab initio* detection of repetitive elements that are not based on prior knowledge accumulated in data bases, are still important in order to extend our comprehension of the role of TRs in biological mechanisms. Existing *ab initio* tools are successful when the TR exhibit a moderate amount of noise and when the TR is easily certified. However, there is an emerging need for new tools that are able to cope with higher levels of noise and/or TR computationally more difficult to detect. For example, Boeva et al. [BRPM06] study so called Fuzzy Tandem Repeats and their role in gene expression, but their technique is limited since they do not handle insertions/and deletions.

[3] As of today, in the PubMed bibliographic database 39214 articles mention the expression "tandem repeat" in their title or abstract (1270 in their title).
[4] http://www.repeatmasker.org/

Some of the most successful *ab initio* tools, such as TRF [Ben99] and ATRHunter [WYKG05], are based on a filtering multi-stage approach (see also [PSdL$^+$09]). In the first stage the input sequence is analyzed so to detect via statistical criteria likely position and length of candidate subsequences. The final stage is the validation one in which a more expensive validation test is applied to candidate substrings passing the first stages, so to determine an output matching the implicit definition of TR and the user-defined filtering parameters.

Our contribution is a novel filtering multi-stage algorithm *TRStalker* that introduces new techniques in all stages. For the first stage, where over-represented distances between probes are sought, we employ *gapped q-grams* [BK03] in place of the standard *q-grams* in order to collect evidence on the candidate substrings. Gapped q-grams have been used before in the context of textual and biological database searching, but less so in the area of tandem repeats detection (with the exception of the system TEIRESIAS [SGFR99]). Because of errors due to insertion/deletions the *period* of a TR is subject to fluctuations, we employ a weighting scheme with exponential decay so to reinforce the signal even in presence of this smearing effect. Finally we use *ranking* instead of *thresholds* when deciding the substrings to pass to the next phases, in order to concentrate the computational effort on the zones with candidates with higher weight. For the final validation stage we employ an NP-complete definition of TR involving the concept of *generalized median string* under edit distance [dlHC00,SP03], together with an efficient heuristic for computing such median strings [JABC03].

By extensive experimental comparisons of *TRStalker* with two state-of-the-art tools, namely TRF and ATRHunter, we did find out that TRStalker has consistently better performance for a large range of error and length parameters for the class of NP-complete Tandem Repeats, with a recall ranging from 100% to 60%. Thus TRStalker improves the capability of TR detection for classes of TRs for which existing methods do not perform well. Tests performed on standard TRs definitions (verifiable in polynomial time) also show recall performance close to 100%. Incidentally, this result confirms of the power of the new techniques developed for the initial filtering phase.

The paper is organized as follows: in Section 2 we describe some basic definitions of TRs; in Section 3 we survey the state of the art; in Section 4 we describe in detail the algorithmic principles of TRStalker; in Section 5 we describe the outcome of the experimental validation. Preliminary experiments with biological data are under way.

2 Basic Definitions

A Tandem Repeat in a DNA sequence is the repetition of two or more contiguous exact or approximate copies of a substring (called the *motif*) of the tandem repeat.

Exact TR. Formally, given an alphabet Σ, and a set of strings $x_i \in \Sigma^*$, consider the concatenation $X = x_1 x_2 ..x_t$. The string X is a *exact tandem repeat* (ETR) of *period* k and *repeat number* t, when $|x_i| = k$ and $x_i = x_1$, for each $i \in [1, ..t]$. In general we may suppose there is a longer string Y of which X is a substring. The string x_1 that is repeated exactly is called the *motif* of the ETR. A TR X is called maximal if it cannot be extended in Y while still being a TR.

Exact tandem repeats are sometimes found in biological sequences, but they tell us only part of the story, thus several notions of an *approximate tandem repeat* have been developed. Denote with $D_H(a, b)$ the hamming distance of two strings with equal length. If the length of a and b is different we consider the smallest possible mismatch in an alignment of the two strings without gaps. Denote with $D_E(a, b)$ the edit distance of the two strings a and b.

Approximate TR (ATR). Several different definitions of Approximate Tandem Repeats proposed in the literature differ by the choices made along a few dimensional axes. The most important are the following:

1. The distance function (or similarity function) used to compare pairs of strings (repeats or motifs). Classical distance functions are (weighted or unweighted) Hamming and edit distances, but also more complex functions with block-operations have been considered. Usually distance functions of choice are also metrics. Sequence pair alignment scoring functions are often used as similarity functions.
2. The choice of pairing policy (adjacent pairs, all pairs, motif-instance pairs, etc..) is the most critical one, in particular the use of motifs not present in the input string (Steiner motifs) may easily change the validation task from polynomial to np-complete.
3. Different functionals for the final error score (typically the sum and the maximum functionals are used).
4. Use of an absolute error threshold or relative to the TR length. More generally the error thresholds can be complex functions of the structure of the TR.

5. Role of the ambient string Y. In most definition a string X is a TR regardless of the properties of the string Y containing it. In a few definitions a statistical analysis of Y is used to determine some of parameters thus the property of being a TR depends on both Y and X

In [SBT07] it is used the following definition: X is called a *k-edit Approximate Tandem Repeat* when $\sum_{i=1}^{t-1} D_E(x_i, x_{i+1}) \leq k$, where the last repeat x_t might be incomplete so $D_E(x_{t-1}, x_t)$ is computed as the minimum edit distance of x_t and the prefixes of x_{t-1}. This definition is inspired by the evolutionary model of TRs in which it is assumed TRs are generated by duplicating the last copy of a previous TR, possibly with duplication errors that truncates it. A k-edit repeat is *maximal* if it cannot be extended either to the left or to the right without violating its definition.
In [WYKG04] for a similarity function ϕ that measures the alignment score of two sequences, it is defined a *η-Simple Approximate Tandem Repeat* (η-SATR) a string $X = x_1...x_t$ such that: there exists a motif $\bar{x} \in \Sigma^*$ so that for every $i \in [1, .., t]$, $\phi(\bar{x}, x_i) \geq \eta$. In other words the TR consists of t duplications of a single consensus string $\bar{x}$ with mutations. Such string $\bar{x}$ is also called a *Steiner motif* if $\bar{x}$ is not constrained to be equal to some repeat x_j. Often in practice $\bar{x}$ is chosen as the repeat x_j that minimizes the error function, and is called a *Pivot motif*. The distinction is critical since, as mentioned before, Steiner motifs lead to NP-complete recognition problems, while Pivot motifs do not.
The *η-Neighboring Approximate Tandem Repeat* (η-NATR) is a string X, so that for each $i \in [1, .., t-1]$, $\phi(x_i, x_{i+1}) \geq \eta$ [WYKG04]. The *Pairwise Approximate Tandem Repeat* (PATR) is a string X, such that for every pair of indices $i, j \in [1, ..t]^2$ with $i \neq j$ we have $\phi(x_i, x_j) \geq \eta_{ij}$, where η_{ij} is set to be a monotonically decreasing function of $|i-j|$, thus allowing more slackness when comparing distant copies of the basic motif.
In [KT04] it is used a definition similar to that of the Neighboring Approximate Tandem Repeat, except that the Hamming distance is used and that the threshold is not absolute but relative to the length. A γ-HATR (γ-Hamming Approximate Tandem Repeat) is a string X such that: for each $i \in [1, .., t-1]$, $D_H(x_i, x_{i+1}) \leq |x_i|\gamma$.
In [SGFR99] a more complex definition is given that takes into account the substring alignment score density function for pairs of random substrings of a given length. Here the definition of a TR X depends on the properties of the longer string Y into which X is embedded. In particular a (μ, p)-TR must comply to two conditions: 1) $\sum_{i=0}^{t-1} \phi(x_i, x_{i+1}) \geq (t-1)\mu$, that imposes an average high similarity score for adjacent repeats, and 2) define $\alpha(p, k)$ as the value of similarity such that there is probability p that two random substrings of length k in Y have similarity above $\alpha(p, k)$. There must be an index $q \in [1, .., t]$ such that $\phi(x_q, x_j) \geq \alpha(p, k)$ for all $j \in [1, ..t]$. Note that this condition limits the dispersion of the similarity with respect to one of the copies (called the *pivot*).
TRF [Ben99] uses as final validation algorithm the Wraparound Dynamic Programming technique (WDP) that tests efficiently the alignments of a given candidate motif with the surrounding portions of the input sequence, so to determine the maximum number of adjacent repetitions within a user-defined score bound. This implies a notion of TR akin to that of simple approximate tandem repeats (SATR) with Pivot motif. Classical results on string alignments [Gus97, page 351] ensures that, for the metric score given by the sum of motif-repeats distances, the solution found using the optimal Pivot motif has a score within a factor $(2 - 1/t)$ of the score induced by the optimal Stainer motif. For low levels of errors one could use a Pivot-SATR definition doubling the error threshold to capture a Steiner-SATR, however for higher error levels (say, above 25%), doubling the error threshold forces the existing systems to work in a range of values (say, above 50%) where most methods do not perform well.
Precision and Recall. For all of the above definitions once the *tandem repeat and its motif* are found it is possible to validate them according to the chosen definition. Thus we can assume that in principle any Approximate TR finder algorithm will report only legitimate TRs (thus *precision* is always 100%) and the only important output quality measure is *recall* (that is how many of the implanted ATR's present in the input string have been reported as such by the algorithm).

3 State of the Art

We will briefly survey the state of the art in finding tandem repeats. First we will describe methods that for a given definition of TR are able to find all maximal substrings in the input that match the definition (*exhaustive algorithms*). Often exhaustive algorithms may not be available, or when available are too slow in practice, thus several *heuristic algorithm* have been developed which are shown experimentally to be able

to detect a large fraction of TRs efficiently. Note that the time/precision trade-off is severely influenced by the allowed error thresholds. Performance often degrades quickly with increasing error levels.

Exhaustive algorithms. When we allow no error, it is possible to find all maximal exact TRs in a string of length n in time $O(n)$ [KK99,GS04]. When we allow two consecutive repeats to differ by an amount at most k (either in Hamming or in edit distance) Landau, Schmidt and Sokol [LSS01] give exhaustive algorithms running in time $O(nk \log(n/k))$ for Hamming distance, and $O(nk \log k \log(n/k))$ for edit distance. A simpler algorithm with the same asymptotic complexity for the edit distance is proposed by Sokol, Benson and Tojeira [SBT07]. Kolpakov and Kucherov [KK03] improved the bound for the Hamming distance to $O(nk \log k + s)$ where s is the number of TR found. For the Hamming distance, Krishnan and Tang [KT04] give an exhaustive method running sequentially in time $O(n^3)$, that can be easily implemented onto a parallel architecture, since every possible pattern length is searched independently.

Heuristic algorithms. The algorithmic techniques in [KK99,KK03] has been extended in the tool `mreps` [KBK03] so to be able to handle approximate TRs under edit distance, with some additional heuristic filtering steps.

The tool TRF (Tandem Repeat Finder) developed by Benson [Ben98,Ben99], based on statistical filtering of zones of DNA likely to contain TRs, is currently one of the standard heuristic methods. ATRHunter [WYKG04] by Wexler et al. is also based on a statistical filtering approach, placing greater emphasis in techniques for designing thresholds for the quantities of interest. Other proposed heuristics for finding TRs are REPuter [KS99,KCO$^+$01], STRING [PFAP03], TEIRESIAS [SGFR99], TandemSWAN [BRPM06]. A class of papers (see e.g. [SIRR04], [Bro07],[BJ03], [GSMS07]) tackle the problem of finding tandem repeats as a problem in signal processing theory and usually map the input string into a time-signal in a suitable numerical domain for which several spectral techniques can be used, such as the *Periodicity Transform* or the *Fourier Transform*. Other methods use data compression techniques to detect repetitive elements [RDD$^+$97].

The methods cited above are rather general since they aim at treating efficiently TRs in a wide range of length values. There is also a large class of methods that are aimed at handling particular or special classes of TRs such as: microsatellites (e.g. IMEx [MN07]), palindromic repeats (e.g. CRISPFinder [GVP07]), Variable Length Tandem Repeats (VLTR) and Multi-period Tandem Repeats (MPTR) [HJ02], Variable Number Tandem Repeats (VNTR) [SS06]. Since the focus of our research on TRs at present is on the more classical forms of TRs we do not dwell longer on them. However, we just note that often methods for MPTR, VNTR, VLTR use standard TR finding as a subroutine, thus our proposed algorithm can increase also our ability to detect such higher order structures. Other methods need as input additional parameters, such as the target period of the TR [BW94].

Systematic comparison among TR finding tools and algorithms operating "ab initio", that is without support of specific biological data bases has been tackled in recent years [LRJ07,SBMP08]. A survey of problems on Tandem Repeats in the context of evolutionary mechanisms, such as the construction of TR Evolutionary Trees, is proposed in [Riv04] (see also [EG02]).

4 Our algorithm: TRStalker

Our definitions of TR. We used two different definitions of TRs:

- **Neighboring TR (NTR)**: it is a string X, so that for each $i \in [1, .., t-1]$, $D_E(x_i, x_{i+1}) \leq \mu |x_i|$, for a user defined parameter $0 \leq \mu \leq 1$
- **Stainer-STR with sum**: it is a string $X = x_1 x_2 .. x_t$ for which there exists a Steiner string $\bar{x} \in \Sigma^*$ so that $\sum_{i \in [1,..,t]} D_E(\bar{x}, x_i) \leq \mu |X|$, for a user defined parameter $0 \leq \mu \leq 1$. In other words the TR consists of t duplications of a single Steiner consensus string $\bar{x}$ with mutations.

The initial filtering phase is the same in either case, while the verification phase will be different for the two possible definitions.

An example To focus on the main ideas, let us consider the very simple case of Exact TR. Given a string $X = x_1 x_2 .. x_t$, embedded in a random string Y, where $x_i = x_1$ for all i and $|x_1| = k$. Consider the class of ungapped q-grams, and the distance between any two occurrences of a given q-gram (also called probes) in Y. For q-grams in X, the period k will appear at least $(k-q+1)(t-1)$ times as the distance between homologous probes. More generally the distance hk, an integer multiple of k, will appear at least $(k-q+1)(t-h)$ times for each value $h = 1, .., t-1$. For gapped q-grams similar formulae hold. For values of k and t large enough,

the period k and its integer multiples will occur more frequently than the expected number of occurrences in a random string, thus the empirical number of occurrences of the value k and its multiple will tend to be in the higher part of a ranking by frequency. This observation holds true as long as the length of the superstring Y is sufficiently limited so that the frequencies generated by the random portion of Y do not overrun the frequencies generated by X. An exact characterization of such a distribution in terms of the parameters k, t, q and $|Y|$ is complex since it can be characterized as the sum of *non-independent* random variables each with a negative binomial distribution. However we avoid the issue of characterizing exactly such a distribution (1) by splitting the input string into blocks of predefined length and limiting the analysis to each block separately, with mechanisms to deal with TR stranded across the block boundaries; (2) by ranking the periods by weighted frequency and exploring only the top L positions (for $L = 50$ in our experiments). Note that in most cases the top ranking periods not corresponding to tandem repeats will be discarded quickly when the positional density is considered, thus we can be very slack in choosing L without incurring in a computational cost. The choice of block-length could be more critical, but experiments show that blocks of length within factor 10-20 of the length of the TR sought do work well, so here too we do not need to define a sharp threshold.

Gapped q-grams The presence of substitutions/insert/deletions has the effect that many instances of q-grams will be affected by error and a match will be missed, thus reducing the frequency counts for the period k. To cope with this effect we use *gapped q-grams* [BK03,BK02] that are more resilient to the presence of substitutions/insert/deletions. As suggested by experiments in [BK02] just few gaps are sufficient to be effective, thus we will use the patterns all gapped q-grams with at most 2 gaps and 3 letters.

Period detection. Our overall strategy for period detection is as follows. We split the input string into blocks of predefined length and we scan the block Y and record for each gapped q-gram P in Y the distance of each occurrence of P to the 5 preceding and following occurrences (10 in total). If a distance and its integer multiples has been detected we form a *composite period* by summing the occurrences of the single simple periods. The candidate periods (simple and composite) are then weighted with the anti-smearing procedure described below, sorted by weight. The candidate periods are processed in order from the highest weight.

Anti-smearing weighting. If q_1 and q_2 are occurrences of the same probe at distance k, before the implant of mutations, the effect of insertion and deletions on the positions between q_1 and q_2 is to alter their distance so that a different period k' is detected. The difference $k - k'$ is equal to the algebraic sum of number of insertions and deletions in the positions between q_1 and q_2. Assuming that any such position can be an insert or a deletion independently with the same probability, the random variable $k - k'$ is distributed as a sum of independent r.v. of value $+1/-1$ with the same probability [MR95,Mul93], thus, by a Chernoff argument, its tail distribution decays exponentially, for any fixed number of terms in the summation[5]. Inspired by the above observation we devise a weighting scheme that increments the total weight of period k if another period of value k' is discovered in a near-by position, with weights that decay exponentially with $|k - k'|$. More formally, at position i of the input string we have pattern $P(i) = P$, let $P(j_1)$,..$P(j_5)$ be the next 5 occurrence of P and $j_w - i = x_w$ the detected distances for $w = 1, ..5$. We give to the period x_w a weight $1 + \sum_{y \in Q} 2^{-|x_w - y|}$, that favors the presence of nearby distances with similar values, and this weight is added to the cumulative array indexed by period value. Afterwards we enqueue all values x_w in the queue position buffer Q. The queue is of a fixed length $|Q| = 20$ and the last elements (older) are dequeued and discarded[6]. The weighting scheme can be easily adapted to the case in which the insert and delete probabilities are different.

Positional density. The second property that TRs exhibit is that of having the same period (simple or composite) detected by probes in near-by positions, so that we can define a notion of k-density, that is the density of probes that contribute to the counter for the candidate period k. We search for position of high k-density as candidates for the starting point of a TR. More formally, let p be the simple period under investigation. Consider the set Q_p of probes (i.e. substrings of Y that contribute to the popularity of p (taking of the two matching probes only the first one) compute the union $\bigcup Q_p$, thus if a position is shared by several probes it will be counted only once. Let $f : N \to \{0, 1\}$ the characteristic function that

[5] Technically, in our case, also the number of terms in the summation is a random variable distributed as a binomial distribution, thus a full formal proof would need to take this into account.

[6] Although the value of the queue length of $|Q|$ could be made dependent of other parameters, we noticed that good performance could be attained even with this simple choice.

for each position in Y denote the membership of that position to $\bigcup Q_p$. Consider the p-window smoothing of f: $F(i) = \sum_{j=i}^{i+p} f(j)$ that computes the p-density of the function f. Finally we define a threshold $t(p)$ proportional to the average p-density by a user-defined constant, and we consider as a candidate position set $CP(Y,p) = \{i \in N | F(i) \geq t(p)\}$. The output of the second phase is a sequence of pairs (p,i) where p is a candidate period and i a candidate position.

Validation. In the third phase we take each candidate pair (p,i) and we test explicitly whether there is a tandem repeat of period p starting in position i according to the definitions above. When using the definition STR we use the incremental method for computing generalized media strings proposed in [JABC03]. When using the definition NTR we use a standard wraparound dynamic programming technique (WDP) [FLSS93]. In this phase we discover the (maximum) repetition number of the TR eventually extracted. As a post processing we check for inclusion the TRs found and we filter out those TR completely enclosed in another one. For TR in the same position and length but different period we report all of them.

5 Experiments

5.1 Measurements of quality.

Let a TR be characterized by the triple: (b,p,r), where b is the initial position, p the period, r the repetition number. Thus the TR covers the positions in Y from index b to $b+rp-1$, we identify the TR with its segment $Seg(TR) = [b, b+rp-1]$. Given two tandem repeats TR_1 and TR_2 their Jaccard coefficient JC is:

$$JC(TR_1, TR_2) = \frac{|Seg(TR_1) \cap Seg(TR_2)|}{|Seg(TR_1) \cup Seg(TR_2)|}.$$

Given a TR TR_0 and a set of TRs: $T = \{TR_1, ..., TR_s\}$ we define the best-match $BM(TR_0, T)$:

$$BM(TR_0, T) = \arg\max_{t \in T} JC(TR_0, t),$$

and the best-match-score BMS:

$$BMS(TR_0, T) = \max_{t \in T} JC(TR_0, t).$$

In our controlled experiments we will have the ground truth TR_0, and we will score the set T of output TRs by the best match score. For a series of experiments we will take the average of the BMS. At first sight one might consider this metric as overly generous. However, consider two algorithms A and B having as input the same string Y into which TR_0 has been embedded returning two candidate sets T_A and T_B. Because of the above considerations, if A and B use the same definition of TR with the same parameters, the two sets are both composed of valid TRs (no false positives). We cannot rule out the existence of other TRs in Y besides the embedded ones, so we cannot penalize the presence in T_A and T_B of elements different from TR_0 provided they are valid. The BMS score measures how well the best choice in T_A and T_B approximates TR_0.

Modified Jaccard Coefficient. Even if X is a TR according to the definition, when we embed X in a string Y, it is well possible that X is not maximal in Y, thus if an algorithm reports $X' \supset X$ there will be a slight penalization in the JC measure. This problem arises a number of times, thus we decided to use a customized version of JC where the denominator is changed as $max(|Seg(TR_1)|, |Seg(TR_2)|)$.

5.2 Experimental Setup

TRStalker has been compared with respect to TRF and ATRHunter. We used the on-line version of the ATRHunter program that is available at its website[7]. The operation of the ATRHunter program can be customized according to the following parameters:

- *match, mismatch, gap, terminal gap*: scores for match, mismatch, indel (insertion or deletion) and terminal indels;

[7] http://bioinfo.cs.technion.ac.il/atrhunter/

– *maximum motif length*: only repeats with motif length shorter than or equal to this value are reported;
– *definition of approximate tandem repeat*: it is possible to select the definition of TRs among the following ones:
  • similarity between adjacent copies;
  • average similarity between adjacent copies;
  • minimum alignement score with a repeating copy.

The operation of the TRF program can be customized through the following parameters:

– *match, mismatch, indel score*: weights used to compute the alignement score;
– *match and indel probability*: the expected value of matches and indels between two motif repetitions;
– *minimum score*: the program reports only the TRs that obtain an alignement score higher than this one;
– *maximum period*: the program looks for TRs where the period size is no larger than this value.

Since the parameters of operation of the TRF web version cannot be completely customized, we used both the web-based tool and the downloadable binary (version 4.00).

5.3 Synthetic Data

We carried out a first set of experiments by using synthetic data. This allows a fine grained control on the amount of mutations introduced within the regions covered by the TRs. The sequences we gave as input to the programs have been built according to the following steps:

1. the background sequence is generated by selecting the four bases A,C,G, and T with equal probability;
2. a perfect TR is embedded within the previous sequence, the TR is generated as r repetitions of a motif with length l;
3. the region covered by the TR is mutated according to substitution, insertion and deletion probabilities (p_s, p_i, and p_d); the number of substitutions, insertions and deletion for every repetition of the motif is exactly equal to lp_s, lp_i, and lp_d;
4. if the TR is a Steiner-STR mutations are introduced in every repeat with respect to the consensus motif, if the TR is a Neighboring-TR mutations are introduced with respect to the previous repeat.

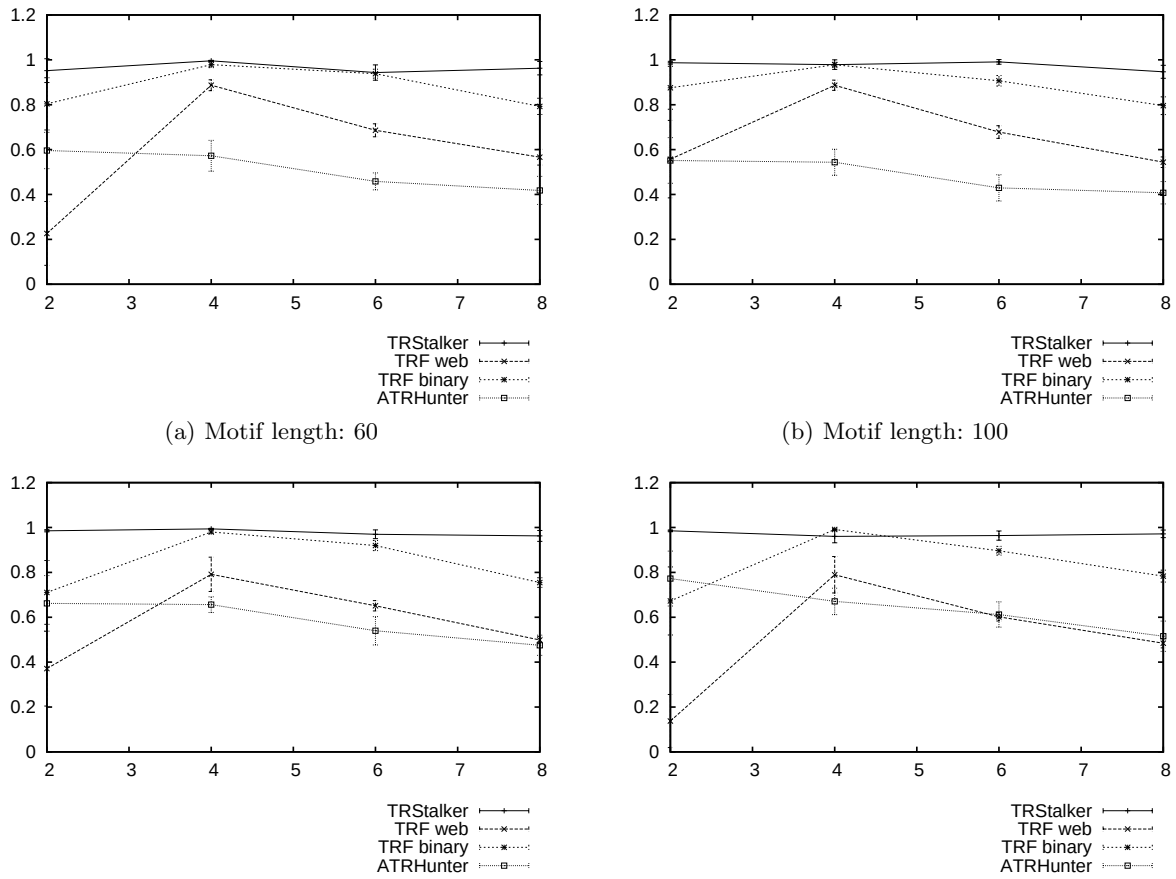
The experiments have been carried out running ATRHunter with these values: match, mismatch, gap and terminal gap score equal to 1 0 -1 0 (the most permissive setting on the website); maximum motif length equal to 500bp. In order to select the definition of TRs, we performed a preliminary set of experiments: the definition that gave the best results was the third one (minimum alignement score). In this case, ATRHunter reports only the TRs that have a score higher than a given threshold. The value of the threshold has been set to 30.

For the web-based version of TRF all the experiments have been carried out with these parameters: match, mismatch, and indel score equal to 2, 3, and 5 respectively; maximum period equal to 500; minimum score equal to 30. For the binary version we used the following ones: match, mismatch, and indel score equal to 2, 3, and 3 respectively; match and indel probability equal to 0.75 and 0.20; maximum period equal to 500; minimum score equal to 30. The parameters of the experiments have been set so to make sure that the minimum allowed score for all the tools tested is attained on the input data.

Results For the experiments on Neighboring-TR (Figure 1) we tested TR with motifs of length from 60 to 300, and a number of repeats from 2 to 8. TRStalker has recall always above 95%. TRF (binary) has a recall always above 80% except for TR with repeat number 2 for which the recall drops to 60%. ATRHunter has recall of about 60%. These experiments confirm the effectiveness of the new techniques for the initial filtering steps.

Results on Steiner Simple TR with motifs of length from 60 to 300, and a number of repeats from 2 to 8 are shown in figure (Figure 2). Here we notice that all methods have degraded performance for longer motifs (above 200 bases) while TRStalker still manages to have recall above 60%. For shorter motifs (of less than 100 bases) TRF (binary) is able to match TRStalker only when the repeat number is above 6. Thus for a large range of values TRStalker attains the best performance in recall, or a matching one, always above 80%.

The time performance of TRStalker has not been yet optimized. At the moment it is within an order of magnitude of TRF and ATRHunter.



(a) Motif length: 60

(b) Motif length: 100

(c) Motif length: 200

(d) Motif length: 300

Fig. 1. The total length of the input sequence is 10000bp; the type of TR is Neighboring-TR (NTR); the amount of substitutions, insertions, and deletions are equal to 10% of the motif length each (thus with total error allowed of 30%). Every point is the average of 30 measurements and the 95% confidence intervals are shown.

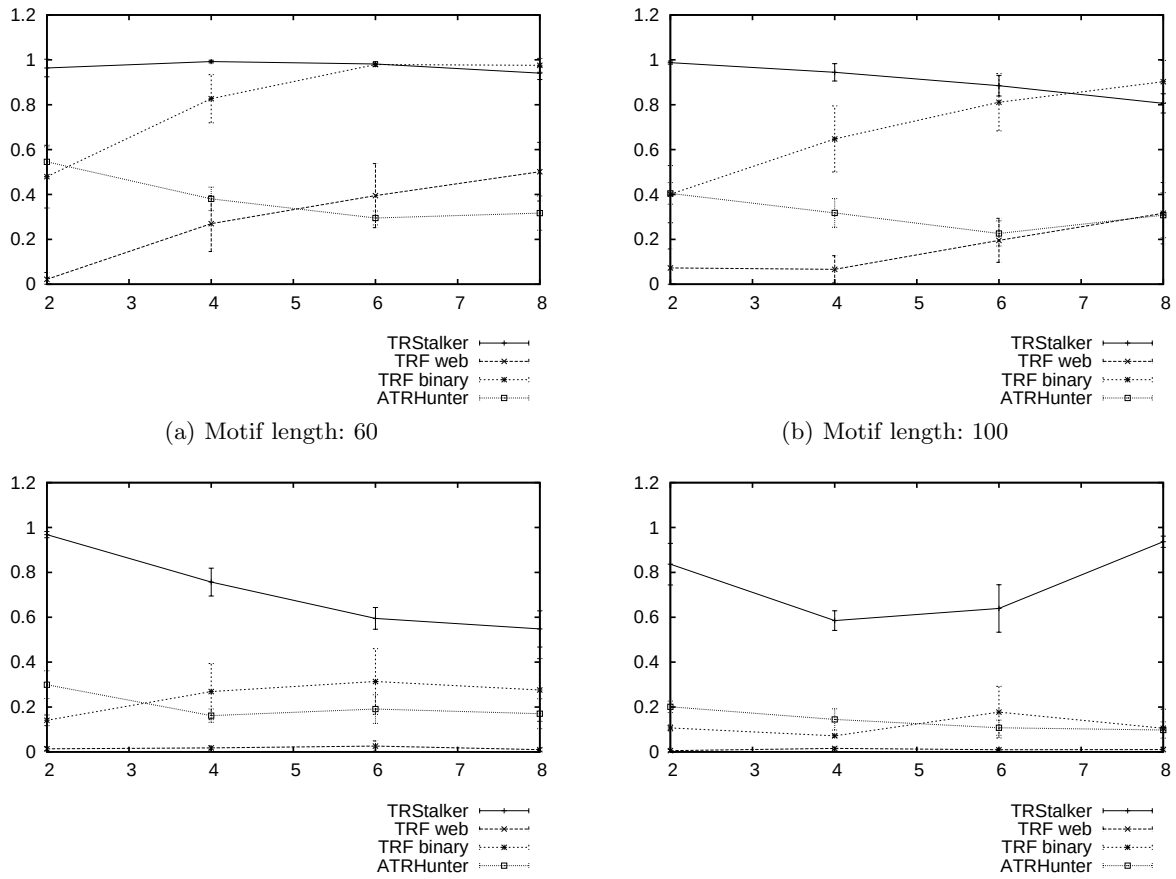
5.4 Biological data

A very preliminary testing of TRStalker on biological sequences has confirmed the potential of our method for finding very fuzzy TRs not detected by TRF and ATRHunter. We tested the following sequences:

1. L3609 Homo sapiens germline T-cell receptor beta chain, complete gene - 684,973 bp long.
2. U43748 Homo sapiens frataxin (FRDA) gene, promoter region and exon - 2,465 bp long.
3. Saccharomyces cerevisiae Chromosome I - 230,208 bp long.

The three algorithms have been run with the setting used in the synthetic experiments on Steiner Simple TRs (thus with a very permissive acceptance policy). In general, none of the three algorithms generates all TRs found by the two others. For our purposes we concentrate on those found by TRStalker only. In Table 1 we report some very long TRs that were detected by TRStalker but missed by the other two methods. We check the motif/repeat alignments using the tool `jaligner`[8] using the BLOSUM62, that confirms the good quality of the motifs found (see table 2, figure 3).

[8] http://jaligner.sourceforge.net/



(a) Motif length: 60

(b) Motif length: 100

(c) Motif length: 200

(d) Motif length: 300

Fig. 2. Total length of the input sequence is 10000bp; the type of TR is a Steiner Simple TR (Steiner-STR); the amount of substitutions, insertions, and deletions are equal to 10% of the motif length each (thus with total error allowed of 30%). Every point is the average of 30 measurements and the 95% confidence intervals are shown.

6 Conclusions

TRStalker is a novel efficient heuristic algorithm for finding NP-complete tandem repeats in biological sequences. TRStalker aims at improving the capability of TR detection for a class of TRs and error range for which existing methods do not perform well. Initial testing on synthetic data are encouraging and we plan extensive testing on biological data as future work.

References

[Ben98] Gary Benson. An algorithm for finding tandem repeats of unspecified pattern size. In *Proceedings of RECOMB*, pages 20–29, 1998.

[Ben99] G. Benson. Tandem repeats finder: A program to analyze DNA sequences. *Nucleic Acids Research*, 27(2):573–580, 1999. http://tandem.bu.edu/trf/trf.html.

[BJ03] M. Buchner and S. Janjarasjitt. Detection and visualization of tandem repeats in dna sequences. *IEEE Trans. Signal Process*, 51:2280–2287, September 2003.

[BK02] Stefan Burkhardt and Juha Kärkkäinen. One-gapped q-gram filtersfor levenshtein distance. In *CPM*, pages 225–234, 2002.

[BK03] Stefan Burkhardt and Juha Kärkkäinen. Better filtering with gapped q-grams. *Fundam. Inform.*, 56(1-2):51–70, 2003.

```
repeat_1          1 CCACAC-CCACACACACACATCCTAA--CACTACCCTAACACAGCCCTA-     46
                    |..||| ||||.||.|| |||||.||  ||||||||||.|||.||||||
consensus         1 CTGCACTCCACTCATAC-CATCCCAATGCACTACCCTACCACTGCCCTAC     49

repeat_1         47 -ATCTAACCCTGGCCAACCTGTCTCTCAACTTACCCTCCAT-TACC     90
                     ||| ||||.|  ||||||| .|||||||| |||||||||| ||||
consensus        50 CATC-AACCAT--CCAACCT-ACTCTCAAC-TACCCTCCATCTACC     90
```

(a) First period (score: 415.50).

```
repeat_2          1 CTGC-CTCCACTCGTTACCCTGTCCCATT---CAACCATACCACT-CCGA     45
                    |||| |||||||| .||||  .|||||.|   |.|||.||||||| ||..
consensus         1 CTGCACTCCACTC-ATACC--ATCCCAATGCACTACCCTACCACTGCCCT     47

repeat_2         46 ACCACCATCCATCCCT-CTACTTACTACCACTCACCCACCGTTACC     90
                    ||||.||.||||||.. |||||..|.||.||.|.||..|   ||||
consensus        48 ACCATCAACCATCCAACCTACTCTCAACTACCCTCCATC---TACC     90
```

(b) Second period (score: 384.50).

```
repeat_3          1 CTCCAATTACCCATATCCAACCCACTGCCACTTACCCTACCATTACCCTA     50
                    ||||    ||.|||| |||.||||.|| ||| ||||||||||.|.|||||
consensus         6 CTCC----ACTCATA-CCATCCCAATG-CAC-TACCCTACCACTGCCCTA     48

repeat_3         51 CCATCCACCAT--GACCTACTCACCA-TACTGTTCTTCTACC     89
                    |||||.|||||  .||||||||.|.| |||..|.|.||||||
consensus        49 CCATCAACCATCCAACCTACTCTCAACTACCCTCCATCTACC     90
```

(c) Third period (score: 399.00).

Fig. 3. TRs found by TRStalker in the Yeast Chromosome I sequence as reported in Table 1. Alignment with the consensus string of the first (a), second (b), and third repeat found (c). The score reported here has been computed with a BLOSUM62 score matrix.

N.	Sequence	Sequence Length	TR Start	TR End	TR Length	Repetitions
1	Human Beta T-cell receptor (L36092)	684,973	411,000	413,128	2,128	2.00
2	Human Beta T-cell receptor (L36092)	684,973	448,000	449,684	1,684	2.00
3	Human Beta T-cell receptor (L36092)	684,973	636,116	638,622	2,506	2.00
4	Yeast Chromosome I	230,208	43	314	271	3.01
5	Homo Sapiens Frataxin (U43748)	2,465	2,036	2,414	377	2.01

Table 1. Examples of TRs found by TRStalker and missed by TRF and ATRHunter.

N.	Repeat	Length	Identity	Gaps	Score
1	1	1107	805/1107 (72.72%)	91/1107 (8.22%)	3657.00
1	2	1093	895/1093 (81.88%)	70/1093 (6.40%)	4291.00
2	1	878	638/878 (72.67%)	85/878 (9.68%)	3045.50
2	2	866	716/866 (82.68%)	52/866 (6.00%)	3568.00
3	1	1300	1000/1300 (76.92%)	94/1300 (7.23%)	5206.00
3	2	1313	1004/1313 (76.47%)	120/1313 (9.14%)	5176.50
4	1	96	75/96 (78.12%)	12/96 (12.50%)	415.50
4	2	96	65/96 (67.71%)	12/96 (12.50%)	384.50
4	3	92	69/92 (75.00%)	10/92 (10.87%)	399.00
5	1	193	149/193 (77.20%)	10/193 (5.18%)	723.50
5	2	191	146/191 (76.44%)	5/191 (2.62%)	765.00

Table 2. Motif/repeats alignment scores computed by `jaligner` using the BLOSUM62 score matrix with gap open penalty set to 10.0 and gap extend penalty set to 0.5.

[Bro07] Andrzej K. Brodzik. Quaternionic periodicity transform: an algebraic solution to the tandem repeat detection problem. *Bioinformatics*, 23(6):694–700, 2007.

[BRPM06] Valentina Boeva, Mireille Régnier, Dmitri A. Papatsenko, and Vsevolod Makeev. Short fuzzy tandem repeats in genomic sequences, identification, and possible role in regulation of gene expression. *Bioinformatics*, 22(6):676–684, 2006.

[BW94] Gary Benson and Michael S. Waterman. A method for fast database search for all k-nucleotide repeats. *Nucl. Acids Res.*, 22(22):4828–4836, 1994.

[dlHC00] Colin de la Higuera and Francisco Casacuberta. Topology of strings: Median string is np-complete. *Theor. Comput. Sci.*, 230(1-2):39–48, 2000.

[DNT+08] Ames D., Murphy N., Helentjaris T., Sun N., and Chandler V. Comparative analyses of human single- and multilocus tandem repeats. *Genetics*, 179(3):1693–1704, July 2008.

[EG02] Olivier Elemento and Olivier Gascuel. An efficient and accurate distance based algorithm to reconstruct tandem duplication trees. In *Proceedigns of ECCB*, pages 92–99, 2002.

[FLSS93] Vincent A. Fischetti, Gad M. Landau, Peter H. Sellers, and Jeanette P. Schmidt. Identifying periodic occurrences of a template with applications to protein structure. *Inf. Process. Lett.*, 45(1):11–18, 1993.

[GRB07] Yevgeniy Gelfand, Alfredo Rodriguez, and Gary Benson. TRDB - the tandem repeats database. *Nucleic Acids Research*, 35(Database-Issue):80–87, 2007.

[GS04] Dan Gusfield and Jens Stoye. Linear time algorithms for finding and representing all the tandem repeats in a string. *J. Comput. Syst. Sci.*, 69(4):525–546, 2004.

[GSMS07] Ravi Gupta, Divya Sarthi, Ankush Mittal, and Kuldip Singh. A novel signal processing measure to identify exact and inexact tandem repeat patterns in dna sequences. *EURASIP J Bioinform Syst Biol.*, 2007. Published online 2007 March 13. doi: 10.1155/2007/43596.

[Gus97] Dan Gusfield. *Algorithms on Strings, Trees, and Sequences - Computer Science and Computational Biology*. Cambridge University Press, 1997.

[GVP07] Ibtissem Grissa, Gilles Vergnaud, and Christine Pourcel. The crisprdb database and tools to display crisprs and to generate dictionaries of spacers and repeats. *BMC Bioinformatics*, 8, 2007.

[HJ02] Amy M. Hauth and Deborah Joseph. Beyond tandem repeats: complex pattern structures and distant regions of similarity. In *Proceedings of ISMB*, pages 31–37, 2002.

[JABC03] Xiaoyi Jiang, Karin Abegglen, Horst Bunke, and János Csirik. Dynamic computation of generalised median strings. *Pattern Anal. Appl.*, 6(3):185–193, 2003.

[JKP+05] J. Jurka, V. V. Kapitonov, A. Pavlicek, P. Klonowski, O. Kohany, and J. Walichiewicz. Repbase update, a database of eukaryotic repetitive elements. *Cytogenet Genome Res*, 110(1-4):462–467, 2005.

[KBK03] Roman Kolpakov, Ghizlane Bana, and Gregory Kucherov. mreps: efficient and flexible detection of tandem repeats in DNA. *Nucleic Acids Research*, 31(13):3672–3678, 2003. http://bioinfo.lifl.fr/mreps/.

[KCO+01] S. Kurtz, J.V. Choudhuri, E. Ohlebusch, C. Schleiermacher, J. Stoye, and R. Giegerich. Reputer: the manifold applications of repeat analysis on a genomic scale. *Nucleic Acids Res*, 29(22):4633–42, 2001.

[KK99] Roman M. Kolpakov and Gregory Kucherov. Finding maximal repetitions in a word in linear time. In *FOCS*, pages 596–604, 1999.

[KK03] Roman Kolpakov and Gregory Kucherov. Finding approximate repetitions under Hamming distance. *Theoretical Computer Science*, 303(1):135–156, June 2003.

[KS99] Stefan Kurtz and Chris Schleiermacher. Reputer: fast computation of maximal repeats in complete genomes. *Bioinformatics*, 15(5):426–427, 1999.

[KT04] Arun Krishnan and Francis Tang. Exhaustive whole-genome tandem repeats search. *Bioinformatics*, 20(16):2702–2710, 2004.

[KTCM08] Yogeshwar D D. Kelkar, Svitlana Tyekucheva, Francesca Chiaromonte, and Kateryna D D. Makova. The genome-wide determinants of human and chimpanzee microsatellite evolution. *Genome Research*, 18:30–38, November 2008.

[LPPV07] Matthieu Legendre, Nathalie Pochet, Theodore Pak, and Kevin J. Verstrepen. Sequence-based estimation of minisatellite and microsatellite repeat variability. *Genome Research*, 17(12):1787–1796, 2007.

[LRJ07] Sébastien Leclercq, Eric Rivals, and Philippe Jarne. Detecting microsatellites within genomes: significant variation among algorithms. *BMC Bioinformatics*, 125(8), April 2007.

[LSS01] Gad M. Landau, Jeanette P. Schmidt, and Dina Sokol. An algorithm for approximate tandem repeats. *Journal of Computational Biology*, 8(1), 2001.

[MN07] Suresh B. Mudunuri and Hampapathalu A. Nagarajaram. Imex: Imperfect microsatellite extractor. *Bioinformatics*, 23(10):1181–1187, 2007.

[MR95] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[Mul93] K. Mulmuley. *Computational Geometry, an Introduction through Randomized Algorithms*. Prentice Hall, 1993.

[OEPS05] Colm O'Dushlaine, Richard Edwards, Stephen Park, and Denis Shields. Tandem repeat copy-number variation in protein-coding regions of human genes. *Genome Biology*, 6(8):R69, 2005.

[PFAP03] Valerio Parisi, Valeria De Fonzo, and Filippo Aluffi-Pentini. String: finding tandem repeats in dna sequences. *Bioinformatics*, 19(14):1733–1738, 2003.

[PSdL+09] Pierre Peterlongo, Gustavo Akio Tominaga Sacomoto, Alair Pereira do Lago, Nadia Pisanti, and Marie-France Sagot. Lossless filter for multiple repeats with bounded edit distance. *Algorithms for Molecular Biology*, 4, 2009.

[RDD+97] E. Rivals, O. Delgrange, J.-P. Delahaye, M. Dauchet, M.-O. Delorme, A. Henaut, and E. Ollivier. Detection of significant patterns by compression algorithms: the case of approximate tandem repeats in DNA sequences. *Comput. Appl. Biosci.*, 13(2):131–136, 1997.

[Riv04] Eric Rivals. A survey on algorithmic aspects of tandem repeats evolution. *International J. Foundations of Computer Science*, 15:225–257, 2004.

[SBMP08] Surya Saha, Susan Bridges, Zenaida V. Magbanua, and Daniel G. Peterson. Empirical comparison of ab initio repeat finding programs. *Nucl. Acids Res.*, 36:2284–2294, 2008.

[SBT07] Dina Sokol, Gary Benson, and Justin Tojeira. Tandem repeats over the edit distance. *Bioinformatics*, 23(2):30–35, 2007.

[SGFR99] G. Stolovitzky, Y. Gao, A. Floratos, and I. Rigoutsos. Tandem repeat detection using pattern discovery with applications to the identification of yeast satellites. Technical Report RC21508, IBM T.J. Watson Research Labs, 1999.

[SIRR04] Deepak Sharma, Biju Issac, G. P. S. Raghava, and R. Ramaswamy. Spectral repeat finder (srf): identification of repetitive sequences using fourier transformation. *Bioinformatics*, 20(9):1405–1412, 2004.

[SP03] Jeong Seop Sim and Kunsoo Park. The consensus string problem for a metric is np-complete. *J. Discrete Algorithms*, 1(1):111–117, 2003.

[SS06] Michael Sammeth and Jens Stoye. Comparing tandem repeats with duplications and excisions of variable degree. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 3(4):395–407, 2006.

[VKN+06] A.J. Vogler, C. Keys, Y. Nemoto, R.E. Colman, Z. Jay, and P. Keim. Effect of repeat copy number on variable-number tandem repeat mutations in escherichia coli O157:H7. *Journal of Bacteriology*, 188(12):4253–63, June 2006.

[WCJC+94] R. Wooster, A.-M. Cleton-Jansen, N. Collins, R.S. Mangion, J. Cornelis, C.S. Cooper, B.A. Gusterson, B.A.J. Ponder, A. von Deimling, O.D. Wiestler, C.J. Cornelisse, P. Devilee, and M.R. Stratton. Instability of short tandem repeats (microsatellites) in human cancers. *Nature Genetics*, 6(2):152–156, 1994.

[WHG+08] Peter Warburton, Dan Hasson, Flavia Guillem, Chloe Lescale, Xiaoping Jin, and Gyorgy Abrusan. Analysis of the largest tandemly repeated dna families in the human genome. *BMC Genomics*, 9(1):533, 2008.

[WYKG04] Ydo Wexler, Zohar Yakhini, Yechezkel Kashi, and Dan Geiger. Finding approximate tandem repeats in genomic sequences. In *Proceedings of the eighth annual international conference on resaerch in computational molecular biology (RECOMB 2004)*, pages 223–232, New York, NY, USA, 2004. ACM.

[WYKG05] Ydo Wexler, Zohar Yakhini, Yechezkel Kashi, and Dan Geiger. Finding approximate tandem repeats in genomic sequences. *Journal of Computational Biology*, 12(7):928–942, 2005.