Evolutionary Bioinformatics

SOFTWARE REVIEW

MiIP: The Monomer Identification and Isolation Program

Christopher Bun[1,§], William Ziccardi[2], Jeffrey Doering[2] and Catherine Putonti[1–3]

[1]Department of Computer Science, Loyola University Chicago, 820 N Michigan Avenue, Chicago, IL 60611 USA. [2]Department of Biology, Loyola University Chicago, 1032 W Sheridan Road, Chicago, IL 60660 USA. [3]Bioinformatics Program, Loyola University Chicago, 1032 W Sheridan Road, Chicago, IL 60660 USA. [§]Current Affiliation: Department of Computer Science, University of Chicago, 1100 E 58th Street, Chicago, IL 60637 USA.
Corresponding author email: cputonti@luc.edu

Abstract: Repetitive elements within genomic DNA are both functionally and evolutionarily informative. Discovering these sequences *ab initio* is computationally challenging, compounded by the fact that selection on these repeats is often relaxed; thus sequence identity between repetitive elements can vary significantly. Here we present a new application, the Monomer Identification and Isolation Program (MiIP), which provides functionality to both search for a particular repeat as well as discover repetitive elements within a larger genomic sequence. To compare MiIP's performance with other repeat detection tools, analysis was conducted for synthetic sequences as well as several α21-II clones and HC21 BAC sequences. The primary benefit of MiIP is the fact that it is a single tool capable of searching for both known monomeric sequences as well as discovering the occurrence of repeats *ab initio*, per the user's required sensitivity of the search. Furthermore, the report functionality helps easily facilitate subsequent phylogenetic analysis.

Keywords: monomer discovery, repetitive element identification, pericentromeric repeats

Evolutionary Bioinformatics 2012:8 293–300

doi: 10.4137/EBO.S9248

This article is available from http://www.la-press.com.

Introduction

Repetitive sequences, be it amino acid or nucleotide, can encode information about functionality as well as the evolution of the sequence. Ubiquitous in the genomes of all living species, these repeats can vary in size (from one to several hundred nucleotides in length) as well as the number of copies, the most well-studied repeats being the 3nt microsatellites associated with human disease.[1] Longer tandemly repeated elements have been found to appear throughout the human genome as well, eg, centromeric satellite sequences. Likewise, repeats within protein sequences are common.[2] While many repetitive elements have been annotated, including those listed in the Tandem Repeats Database (TRDB)[3] and ProtRepeatsDB,[4] they continue to be detected in new sequences and new repetitive elements continue to be discovered.

A number of computational tools have been developed for the identification and discovery of repetitive elements in protein sequences (eg, XSTREAM,[5] T-REKS,[6] and TRUST[7]) and DNA sequences (eg, TRF,[8] STAR,[9] KSA,[10] TRED,[11] SSR Locator,[12] TROLL,[13] mreps,[14] RepeatMasker,[15] and Sputnik).[16] These algorithms can be grouped into one of three approaches: combinatorial algorithms (Sputnik, mreps, TROLL, and XSTREAM), statistical properties (TRF, KSA, and T-REKS), and alignment-based (RepeatMasker, TRUST, TRED, SSR Locator, and STAR). Heuristics are often employed, given the computational complexity of the problem. While expediting the search, this comes at the cost of search sensitivity.

Within the centromeric regions repetitive satellite sequences have been identified including the α-satellite sequences consisting of tandem repetitions of a 171 bp motif (monomer). While monomers arranged within higher-order repeat (HOR) arrays are highly conserved (divergence typically <2%), individual monomers can exhibit significantly more divergence, on the order of 20%–40%.[2,3,17] Furthermore, monomeric tracks near the periphery of centromeric DNA have been found to show even greater divergence.[18] Through the comparison of human centromeric sequences with those of other primate species, it was concluded that the structure and content of these regions are evolving rapidly.[2,7,18]

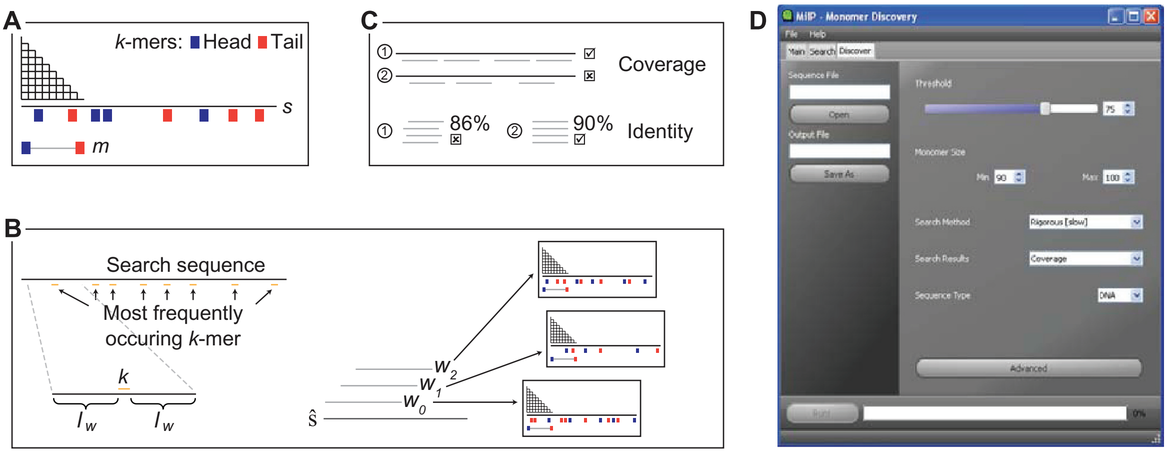
Here we present a new software tool, the Monomer Identification and Isolation Program (MiIP). Although designed specifically for the detection of satellite sequence repeats (monomers) within centromeric DNA, the software is capable of detecting smaller micro- and minisatellites as well as amino acid repeats. Given our focus was on monomers within centromeric sequences, MiIP is fine-tuned to identify larger degenerate repetitive elements. Within this one tool, two options are available given a user-supplied sequence: (1) search for a particular user-defined monomer and (2) discover the occurrence of monomers *ab initio*. Furthermore, robust or heuristic search methods can be conducted, dependent upon the preference of the user.

Implementation

MiIP was developed in C++ and is available as both a command-prompt application and a cross-platform GUI application (Fig. 1D). GUI development was performed using the Qt 4.6.3 framework.[19] Both the command-prompt and GUI application offer the same functionality.

Searching for a particular monomer sequence

Given a monomer sequence m of length l_m and a larger search sequence s of length l_s, the user is prompted for a threshold value t where t equals the minimum sequence identity expected to qualify a "match" as an instance of m in s. The search is "seeded" by first scanning s for instances of the first ("head") and last ("tail") k-mer, where $4^k < l_m$ and $4^{k+1} > l_m$, in m. A window length l_w is calculated as the maximum length of a "match" containing inserted residues while still meeting the user specified threshold of sequence identity. Pair-wise Smith-Waterman sequence alignments are conducted between m and each window i of length l_w in s which starts with the head k-mer or ends with the tail k-mer. Figure 1A illustrates this process. While seeding the search reduces the number of windows for which m is compared, considering windows with either the head or tail k-mer relaxes the condition that both are conserved within each repeat present. Furthermore, if no (or very few relative to the search sequence size l_s) repetitive elements are identified, the size of k is decreased and the process is repeated.



Threshold: 75

Monomer Size: Min 90, Max 100

Search Method: Rigorous [slow]

Search Results: Coverage

Sequence Type: DNA

Figure 1. MiIP search and discovery. (**A**) Search: For a given monomer sequence, *m*, its head (blue) and tail (orange) *k*-mers are located within *s* and alignments, as indicated by the lower triangular matrix, are performed. (**B**) Discovery: The most frequently occurring *k*-mer in *s* (yellow boxes) is identified and the subsequence $\hat{s}$ containing its first appearance is selected; every window in $\hat{s}$ is compared to *s*. (**C**) Selecting the "best" consensus monomer is user-defined as either: "coverage-based", eg, ① over ②; or "identity-based", eg, ② over ①. (**D**) Discovery mode tab of the MiIP GUI.

Discovering monomer sequences

MiIP includes two options for monomer discovery, referred to as "rigorous" and "heuristic". Both options necessitate the user to define a minimum and maximum value for the size of the repetitive element, $l_w$min and $l_w$max. The rigorous method is similar to that described for the search of a particular monomer sequence; every window *i* in *s* is regarded as a putative monomer and as such, each window is decomposed into head and tail *k*-mers and pair-wise comparisons are conducted. This process is conducted for each size of l_w, where $l_w\text{min} \leq l_w \leq l_w\text{max}$. The run-time estimate for each size l_w examined is $O(xl_w^{\,2})$, where *x* is the number of windows containing a head or tail *k*-mer. Thus as the number of repeats increases, so too will the run-time. Likewise, the run-time will increase as the range of lengths considered expands.

In contrast, the second method of discovery does not assess every window *i*. The most frequently occurring *k*-mer in *s*, where $4^k < l_s$ and $4^{k+1} > l_s$, is identified; thus the assumption is made that this *k*-mer is contained somewhere within the repetitive element. The first location of this *k*-mer *j* is identified and the subsequence $\hat{s}$ of *s* from position $j - l_w$ to $j + k + l_w$ is selected. Every window of length l_w in $\hat{s}$ is then compared to *s* in a manner analogous to the "rigorous" approach. Once again instances of the head and tail of the sliding window to seed the search and all l_w in the user-specified range are evaluated. Figure 1B depicts this process. In the event that no monomers are detected, ie, the *k*-mer selected is not contained within the repetitive element, the next the most frequently occurring *k*-mer is identified and a new $\hat{s}$ is considered.

Thresholding

In both the search and discovery modes, an alignment with a score greater than or equal to the user-defined threshold *t* is considered a putative match. As many putative monomers may be found to exceed *t*, MiIP reports the "best" monomer found. Two different metrics have been implemented and the user can specify to select monomers with the optimal: (1) coverage of *s*, or the number of residues in *s* contained in one or more instances of *m*, and (2) sequence identity between the instances of the putative monomer in *s* (Fig. 1C). In the event of a tie, the other metric is evaluated.

Facilitating phylogenetic analysis of monomers

MiIP was explicitly designed to generate results that can be easily examined and/or manipulated by existing phylogenetic software packages. Execution of either the search or discovery mode generates two separate results files, an MS Excel spreadsheet listing the sequences identified, their position within *s* and their sequence identity to the consensus sequence. The second file is a single FASTA file with all of the monomer sequences. Because researchers may vary in the way in which they derive consensus sequences, MiIP leaves this task to the user. While some of the aforementioned repeat software applications include a finishing step to trim the ends of the repeat sequences reported, MiIP does not. One of the advanced parameters within our application is the ability for the user to permit the occurrence of overlapping repetitive elements. Assessing the biological significance of the occurrence of overlaps and/or the definitive start and stop position of each repetitive element is once again left to the user.

Results and Discussion

Examining the effects of GC content and sequence divergence when searching for repetitive elements

The performance of MiIP was first assessed using synthetic sequences testing the ability to recognize repetitive elements of varying GC contents as well as conservation. Firstly, five sequences each 150 nucleotides long were created, each differing in its overall GC content: 25%, 35%, 50%, 65% and 75%. We refer to each of these sequences as an ancestor repeat. Five synthetic sequences were next generated, one for each GC content tested, in which the 150-mer was repeated each time reducing its sequence identity to the ancestral repeat; a range from 100% sequence identity to 35% sequence identity was included in the synthetic sequence. Each of the repetitive units was then randomly shuffled within the synthetic sequence. The synthetic sequences were generated by code written in-house in C++.

For each of the synthetic sequences, we first used MiIP in the search mode. When supplying both the ancestor repeat sequence as well as the synthetic sequence and specifying a minimum threshold of 30% sequence identity, all of the repeats were found regardless of the GC content of the synthetic sequence. This met our expectations, given that the threshold was lower than the sequence identity between the ancestor repeat sequence and its most divergent repeat within the synthetic sequence (35%). When analyzing these same synthetic sequences using MiIP's discovery mode (heuristic approach, threshold = 30%, repeat size = 150), nearly every repeat was found in less than one minute. MiIP located 99 of the 100 repeats in the synthetic sequences with a GC content of 25%, 50%, and 65% and 98 of the 100 repeats for the synthetic sequences with a GC content of 35% and 75%.

Using these same synthetic sequences, we next compared MiIP's performance in discovery mode relative to other available tools. Although several of the aforementioned tools for repetitive element detection had to be excluded for various reasons (eg, STAR[9] requires a user defined motif to initiate the search, SSR Locator[12] cannot consider repeats longer than 10 residues, RepeatMasker[15] searches against a library of known repetitive elements, etc.), TRF,[8] TRedD[20] (a new version of TRED),[11] and mreps[14] were each examined. While a variety of parameter values and all five synthetic sequences were tested using mreps, the software (v. 2.5) could not find any repeats in the sequences. In contrast TRedD (maximum number of errors = 20) and TRF (maximum period size = 200) were both able to identify the repeated sequence. As Figure 2 shows, TRF and MiIP both significantly outperformed TRedD. As this figure also reveals, MiIP outperformed TRF for all five synthetic sequences. Given that the threshold supplied by the user is less than the sequence divergence between repetitive elements in the sequence, MiIP is capable of locating the repeat sequence regardless of skews in GC content.

Searching for repeats in pericentromeric regions

To further test the sensitivity and performance of MiIP, several different sequences from pericentromeric regions have been examined looking for α-satellite (171 bp) as well as β-satellite (68 bp)

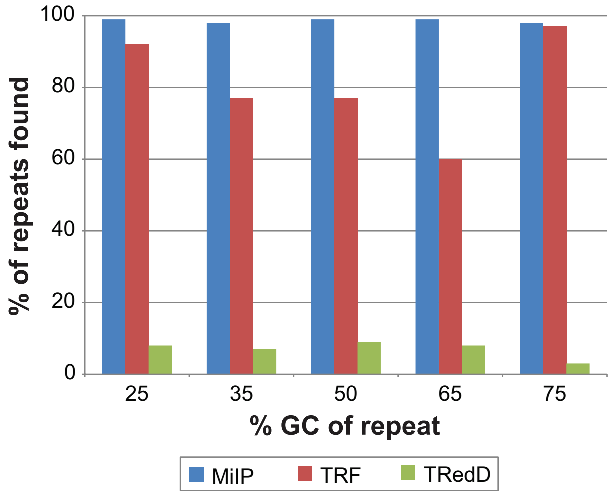


Figure 2. Percentage of repeats found by MiIP, TRF and TRedD within synthetic sequences of different GC contents.

monomeric sequences. Here we will discuss the results of the former, looking specifically at ten sequences from NCBI (1 to 210 Kbp) from the pericentromeric region of the human chromosome 21 (Table 1). Evidence of inter- and intrachromosomal duplications as well as the presence of other repetitive elements within the human centromeres sequences presents a challenge for the identification and isolation of monomers within, in particular, large clone sequences and assemblies.[21,22]

The ten centromeric sequences were first examined for monomers using a monomer from African Green Monkey (AGM) α-satellite [GenBank: V00145].[23] This same sequence has been utilized as the outgroup for phylogenetic analysis of monomers within chromosomes X, 8 and 17.[17] Default parameter values were used and three threshold values were considered and t = 60%, 70% and 80%). All of the occurrences were reported (<1 minute on a standard PC; 2.40 GHz Intel® Core™2 Duo) with no false positives. Utilizing the lower threshold identified more monomers than the higher threshold as expected. By examining the Excel document generated by MiIP, we could identify the locations within the search sequence for each monomer found, thus facilitating the detection of regions of the sequence containing other repetitive elements as well as sites of transposition, partial duplication, etc.

Each genomic sequence was examined again in discovery mode for a size range of 160 to 180 bp. Based upon sequence conservation observed in studies for α-satellites in chromosome 17,[24] a threshold of 68% was used. For the heuristic approach, run-time was comparable to execution of the search mode. While rigorous discovery for short sequences (<3 Kbp) was under five minutes, longer sequences quickly exceeded an hour. This is due to the fact that the number of occurrences of the head and tail k-mers increases rapidly as the number of repeats increases. For three of the longer search sequences considered (>100 Kbp), both the rigorous and heuristic approaches were executed using the same parameters (t = 68%), producing the same number of instances of the monomer. This indicates, at least in the set of pericentromeric sequences examined here, that the heuristic approach is capable of correctly detecting the repetitive elements with essentially the same specificity as the more rigorous option.

Table 1. Pericentromeric alphoid clone sequences of the human chromosome 21.

Clone/region	GenBank accession	Length (Kbp)
pN23	D29746	0.6
CEN 2-4	EU597835	1.5
pTRA-1	X55370	1.2
CEN 3-1	EU597837	2.2
CEN 3-4	GU047352	2.2
pTRA-4	X55370	5
pIA1	AF105153	42.9
CH507-239L24	CU638690	128.3
CH507-478D3	CT476838	129.5
CTD-2503J9	AF254982	211.3

The analysis of the ten centromeric sequences using the discovery mode identified more monomers than were found using the AGM sequence. As each sequence was run independently, the threshold value of t = 68% applied only to the threshold of sequence identity within the search sequence. Over 1,800 instances of the monomer were detected in the ten sequences examined. The consensus sequence was derived for each search sequence's results using BioEdit and the sequence identity for each monomer was computed. Relative to each respective consensus, the monomers had on average 73.8% sequence identity. Those monomers which were only discovered through the discovery mode were compared individually to AGM, exhibiting anywhere from 24.3% to 59.4% sequence identity to AGM. Comparison of monomers isolated from one search sequence with those monomers from another search sequence exhibited an average sequence identity of 70.2%; individual pairwise similarity scores ranged from 17.7% to 100%.

To compare MiIPs performance with other repeat detection tools, analysis was conducted using MiIP, RepeatMasker[15] and TRF.[8] RepeatMasker was selected because it screens its search sequence against a predefined library of repetitive elements which includes the α-satellite monomeric sequence. Based upon its performance when analyzing the synthetic sequences (Fig. 2), we once again chose to compare MiIP with TRF. For example, in the analysis of the 1.5 Kbp sequence CEN 2-4 [GenBank: EU597835], all of the repeats found by MiIP were also found by these two tools. The following parameters were selected for RepeatMasker: the cross_match search engine and the "slow" option for speed/sensitivity. This provides the greatest sensitivity for the search of the 1.5 Kbp sequence CEN 2-4. Two repeats were reported for the search, one to ALR and one to ALR_ (offset from ALR by 84 bp). The ALR repeat identified is identical to the repeat found by MiIP in the discovery mode using the heuristic search option. The same CEN 2-4 sequence was examined by TRF. Like MiIP, TRF necessitates user input regarding the expected size of the repetitive element. In our search, we set the maximum period size to 200 bp. The same repeat set found by MiIP (in the same frame) was identified by TRF. Minor variations in the exact 'start' and 'end' positions of the repeat within the search sequence were observed between all three result sets.

In our evaluation of MiIP's discovery mode in comparison to RepeatMasker and TRF using the human chromosome 21 pericentric sequences, we encountered several instances in which MiIP identified a repeat not reported by either of the other tools. Upon investigation of these repeats, a common trend was identified; the repeats exhibited low sequence identity to the other repeats in the sequence. This corresponds with what was observed during our prior comparison of TRF and MiIP (Fig. 2). Further investigation of these instances revealed a high degree of degeneracy, particularly at the 5′ end. Detecting these degenerate repeats, however, is very informative with respect to the evolution of the alphoid monomeric repeats warranting further investigation.

Identifying highly degenerate repeats

The memory usage is uniform for all approaches implemented, and is dependent upon the size of the search sequence, $O(l_s)$. The utilization of *k*-mers to seed the searches has been employed by other software tools for expediting alignments (eg, YASS,[25] Patternhunter[26] and BLAST)[27] and significant investigation of the limitations of this approach given seed selection and sequence homology has been conducted.[28] Several of the aforementioned pattern recognition algorithms employ a similar *k*-mer approach to seeding the search. With respect to conducting a search given a user-provided monomer sequence, either the head or tail *k*-mer is expected to be conserved. In the event that an occurrence of the monomer is missing the first *k* and last *k* residues of the monomer sequence, MiIP will not be able to identify the partial monomer. In the event that the first and last *k* residues are not well conserved, MiIP will consider smaller sizes of *k*. For the rigorous approach in the discovery mode, a similar approach is employed. All sliding windows are considered within the search sequence, thus the conservation of the head or tail *k*-mers is relaxed. In the event that an occurrence of the monomer is missing the first *k* and last *k* residues of the monomer sequence, MiIP will not be able to identify the partial monomer. In the event that the first and last *k* residues are not well conserved, MiIP will consider smaller sizes of *k*. The size of *k* initially selected is relatively small, eg, for the 171 bp AGM monomer $k = 3$. The heuristic approach for monomer discovery takes a slightly different approach. In this search strategy, the most frequently occurring *k*-mer, or the most conserved *k*-mer, is selected regardless of its location within the monomer. Once again the *k*-mer size is relatively small. Numerous *k*-mer selection strategies were considered during development for all three search strategies.

Investigating how repetitive elements arise

One of the key considerations taken into the development of MiIP was to facilitate phylogenetic analysis. The FASTA files generated, listing each instance of the monomer within the search sequence, can be examined using any sequence analysis tool. To test the ease of analysis of the MiIP results, we here present an example using the results from the discovery mode search for monomers within the pIA1 region [GenBank: AF105153] located in the p arm of Chromosome 21. The AGM sequence was added to the results, serving as the outgroup. The 91 monomers identified in this search sequence and the AGM reference sequence were aligned using ClustalW2 through the SeaView tool.[29] The tree for these sequences was then derived using the PhyML v3.0.1, once again through SeaView.[29] AGM was specified as the root of the tree and visualized by Phylowidget.[30] Figure 3 illustrates the tree generated for the monomers found by MiIP in this search sequence. By creating a standard FASTA format file of the results, the user can utilize software tools of their choosing for evolutionary studies.

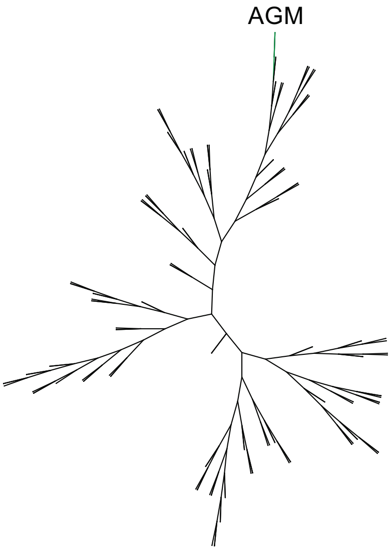


Figure 3. Maximum likelihood tree of pIA1 HC21 alphoid monomers.

Conclusions

The primary benefit of MiIP is the fact that it is a single tool capable of searching for both known monomeric sequences as well as discovering the occurrence of repeats *ab initio*, be it nucleotide or amino acid. Furthermore, the report functionality helps easily facilitate subsequent phylogenetic analysis. Despite the fact that MiIP was developed with the worst-case scenario in mind, choosing sensitivity over rapidity, the heuristic approach for discovery retains sensitivity at a reasonable run-time. Because MiIP's performance is dependent upon the number of alignments executed, a parallel architecture like GPU computing is an attractive avenue for future development.

Availability and Requirements

Project name: MiIP; Project home page: miip.googlecode.com; Operating systems(s): Platform independent; Programming language: C++, Qt; License: GNU GPL.

Authors' Contributions

CB designed and developed the GUI as well as writing of the user documentation. WZ and JD participated in the design and testing of the software. CP designed and implemented the algorithms and data structures and wrote the manuscript. All authors have read and approved the final manuscript.

Acknowledgements

The authors would like to thank Mr. Robert Ennesser for his assistance in the design and testing of the application.

Disclosures and Ethics

As a requirement of publication author(s) have provided to the publisher signed confirmation of compliance with legal and ethical obligations including but not limited to the following: authorship and contributorship, conflicts of interest, privacy and confidentiality and (where applicable) protection of human and animal research subjects. The authors have read and confirmed their agreement with the ICMJE authorship and conflict of interest criteria. The authors have also confirmed that this article is unique and not under consideration or published in any other publication, and that they have permission from rights holders to reproduce any copyrighted material. Any disclosures are made in this section. The external blind peer reviewers report no conflicts of interest.

References

1. Orr HT, Zoghbi HY. Trinucleotide repeat disorders. *Annu Rev Neurosci*. 2007;30:575–621.
2. Andrade MA, Perez-Iratxeta C, Ponting CP. Protein repeats: structures, functions, and evolution. *J Struct Biol*. 2001;134:117–31.
3. Gelfand Y, Rodriguez A, Benson G. TRDB—the Tandem Repeats Database. *Nucleic Acids Res*. 2007;35:D80–7.
4. Kalita MK, Ramasamy G, Duraisamy S, Chauhan VS, Gupta D. ProtRepeatsDB: a database of amino acid repeats in genomes. *BMC Bioinformatics*. 2006;7:336.
5. Newman AM, Cooper JB. XSTREAM: A practical algorithm for identification and architecture modeling of tandem repeats in protein sequences. *BMC Bioinformatics*. 2007;8:382.
6. Jorda J, Kajava AV. T-REKS: identification of Tandem REpeats in sequences with a K-meanS based algorithm. *Bioinformatics*. 2009;25:2632–8.
7. Szklarczyk R, Heringa J. Tracking repeats using significance and transitivity. *Bioinformatics*. 2004;20 Suppl 1:i311–7.
8. Benson G. Tandem repeats finder: a program to analyze DNA. *Nucleic Acids Res*. 1999;27:573–80.
9. Delgrange O, Rivals E. STAR: an algorithm to search for tandem approximate repeats. *Bioinformatics*. 2004;20:2812–20.
10. Rosandić M, Paar V, Basar I. Key-string segmentation algorithm and higher-order repeat 16 mer (54 copies) in human alpha satellite DNA in chromosome 7. *J Theor Biol*. 2003;221:29–37.
11. Sokol D, Benson G, Tojeira J. Tandem repeats over the edit distance. *Bioinformatics*. 2006;23:e30–5.
12. de Maia LC, Palmieri DA, de Souza VQ, Kopp MM, de Carvalho FIF, de Oliveira AC. SSR Locator: Tool for simple sequence repeat discovery integrated with primer design and PCR simulation. *Int J Plant Genomics*. 2008;2008:412696.
13. Castelo AT, Martins W, Gao GR. TROLL—tandem repeat occurrence locator. *Bioinformatics*. 2002;18:634–6.

14. Kolpakov R, Bana G, Kucherov G. mreps: efficient and flexible detection of tandem repeats in DNA. *Nucleic Acids Res*. 2003;31:3672–8.
15. Smit AFA, Hubley R. *Repeat Masker*. [http://www.repeatmasker.org].
16. Abajian C. *Sputnik—DNA microsatellite repeat search utility*. [http://espressosoftware.com/sputnik/index.html].
17. Shepelev VA, Alexandrov AA, Yurov YB, Alexandrov IA. The evolutionary origin of man can be traced in the layers of defunct ancestral alpha satellites flanking the active centromeres of human chromosomes. *PLoS Genet*. 2009;5:e1000641.
18. Alkan C, Ventura M, Archidiacono N, Rocchi M, Sahinalp SC, Eichler EE. Organization and evolution of primate centromeric DNA from whole-genome shotgun sequence data. *PLoS Comput Biol*. 2007;3:e181.
19. Qt Framework [http://qt.nokia.com/products].
20. Sokol D, Atagun F. TRedD: A database for tandem repeats over the edit distance. *Database*. 2010;2010:baq003.
21. Rudd MK, Willard HF. Analysis of the centromeric regions of the human genome assembly. *Trends Genet*. 2004;20:529–33.
22. Lee C, Wevrick R, Fisher RB, Ferguson-Smith MA, Lin CC. Human centromeric DNAs. *Hum Genet*. 1997;100:291–304.
23. Rosenberg H, Singer M, Rosenberg M. Highly reiterated sequences of SIMIANSIMIANSIMIANSIMIANSIMIAN. *Science*. 1978;200:394–402.
24. Rudd MK, Wray GA, Willard HF. The evolutionary dynamics of α-satellite. *Genome Res*. 2006;16:88–96.
25. Noé L, Kucherov G. YASS: enhancing the sensitivity of DNA similarity search. *Nucleic Acids Res*. 2005;33:W540–3.
26. Li M, Ma B, Kisman D, Tromp J. Patternhunter II: highly sensitive and fast homology search. *J Bioinform Comput Biol*. 2004;2:417–39.
27. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. *J Mol Biol*. 1990;215:403–10.
28. Chung WH, Park SB. Hit integration for identifying optimal spaced seeds. *BMC Bioinformatics*. 2010;11 Suppl 1:S37.
29. Gouy M, Guindon S, Gascuel O. SeaView version 4: a multiplatform graphical user interface for sequence alignment and phylogenetic tree building. *Mol Biol Evol*. 2010;27:221–4.
30. Jordan GE, Piel WH. PhyloWidget: web-based visualizations for the tree of life. *Bioinformatics*. 2008;24:1641–2.