



De novo identification of repeat families in large genomes

Alkes L. Price, Neil C. Jones and Pavel A. Pevzner*

Department of Computer Science and Engineering, University of California San Diego, La Jolla, CA 92093-0114, USA

Received on January 15, 2005; accepted on March 27, 2005

Every time we compare two species that are closer to each other than either is to humans, we get nearly killed by unmasked repeats.

Webb Miller (Personal communication)

ABSTRACT

Motivation: *De novo* repeat family identification is a challenging algorithmic problem of great practical importance. As the number of genome sequencing projects increases, there is a pressing need to identify the repeat families present in large, newly sequenced genomes. We develop a new method for *de novo* identification of repeat families via extension of consensus seeds; our method enables a rigorous definition of repeat boundaries, a key issue in repeat analysis.

Results: Our RepeatScout algorithm is more sensitive and is orders of magnitude faster than RECON, the dominant tool for *de novo* repeat family identification in newly sequenced genomes. Using RepeatScout, we estimate that ~2% of the human genome and 4% of mouse and rat genomes consist of previously unannotated repetitive sequence.

Availability: Source code is available for download at <http://www-cse.ucsd.edu/groups/bioinformatics/software.html>

Contact: ppevzner@cs.ucsd.edu

1 INTRODUCTION

Repetitive DNA comprises a significant fraction of eukaryotic genomes, e.g. ~20% of *Caenorhabditis elegans* and *Caenorhabditis briggsae* genomes (Stein *et al.*, 2003) and ~50% of the human genome (International Human Genome Consortium, 2001) have been identified as repetitive DNA. Repeat identification is a critical part of the analysis of a newly sequenced genome, both because repeats drive genome evolution in diverse ways (Kazazian, 2004) and because of a pragmatic need for thorough repeat masking prior to performing homology searches.¹ RepeatMasker (Smit and Green, <http://repeatmasker.org>) is an important and widely used

tool for identifying and masking individual repeat elements given an existing library of repeat families such as Repbase Update (Jurka, 1998, 2000). However, RepeatMasker does not address the need to build such libraries for newly sequenced genomes; it, in fact, highlights this need. As the number of genome sequencing projects increases, there is a pressing need to identify the repeat families present in large, newly sequenced genomes.

All existing algorithms for building a set of repeat families start with a set of pairwise similarities, such as a set of pairwise alignments generated by WU-BLAST (Gish, <http://blast.wustl.edu>) or REPuter (Kurtz and Schleiermacher, 1999; Kurtz *et al.*, 2000). The early single linkage clustering approach (Agarwal and States, 1994) first merges overlapping substrings appearing in the set of pairwise similarities, then uses the pairwise similarities to group the merged substrings into repeat families. The single linkage clustering approach has been significantly extended and improved in two recent algorithms, RepeatFinder (Volfovsky *et al.*, 2001) and RECON (Bao and Eddy, 2002). A different approach to gluing pairwise similarities into repeat families, which captures the mosaic subrepeat structure exhibited by some repeat families, is implemented in the RepeatGluer algorithm (Pevzner *et al.*, 2004). Also of interest is the PILER algorithm (Edgar and Myers, 2005), which achieves high specificity in distinguishing different types of repeats, at the sacrifice of some sensitivity.

Using a set of pairwise similarities as the starting point for building a set of repeat families has two disadvantages. First, as the authors of RECON state, ‘difficulty in defining element boundaries causes . . . problems in clustering related elements into families’ (Bao and Eddy, 2002). Second, for large repeat-rich genomes, producing a set of pairwise similarities can be a prohibitively computationally intensive task. For example, *Alu* repeats, with $>10^6$ copies in human genome, give rise to $\sim 10^{12}$ pairwise alignments, making explicit construction of a set of pairwise similarities computationally infeasible. One way around this problem is to iteratively run on a small sample of the genome and then analyze progressively larger samples after masking the repeat families already identified (Bao and Eddy, 2002). This has shortcomings as well: repeat families constructed from a small sample of the genome will

*To whom correspondence should be addressed.

¹See Bourque *et al.* (2004) and Gibbs *et al.* (2004) for a description of difficulties with the identification of synteny blocks and similarity anchors caused by inadequate repeat masking of the rat genome.

be less accurate (particularly with respect to boundaries), and mosaic subrepeats which have already been masked will be excluded from appearing in other repeat families, leading to fragmentation. Emphasizing these difficulties, Bao and Eddy conclude that ‘the problem of automated repeat sequence family classification is inherently messy and ill-defined and does not appear to be amenable to a clean algorithmic attack.’

In this paper, we describe a surprisingly simple and fast method which addresses these problems. Our RepeatScout algorithm builds a set of repeat families by using high-frequency *l*mers (i.e. short substrings of length *l*) as seeds, and greedily extends each seed to a progressively longer consensus sequence, following the dynamically inferred alignments between the consensus sequence and its occurrences in the genome. The straightforward, transparent nature of our approach is a significant advantage over other repeat family identification algorithms. In addition, our approach utilizes an efficient method of similarity search and enables a rigorous definition of repeat boundaries.

In the past few years, the RECON algorithm (Bao and Eddy, 2002) has become the dominant tool for *de novo* repeat family identification in newly sequenced genomes. For example, RECON has been used to construct a library of *C.briggsae* repeat families (Stein *et al.*, 2003), making this an ideal test bed. A library of *C.briggsae* repeat families can be evaluated by analyzing the set of repeat elements identified by RepeatMasker using that library. This analysis indicates that the library of repeat families produced by RepeatScout is more sensitive than the library of repeat families produced by RECON in identifying *C.briggsae* repeats. In particular, >4% of the *C.briggsae* genome is identified by RepeatScout as repetitive sequence but missed by RECON.

We apply RepeatScout to the human, mouse and rat X chromosomes. We find that ~2% of the human X chromosome and 4% of the mouse and rat X chromosomes consist of previously unannotated repetitive sequence; these percentages serve as estimates for each entire genome. The larger amount of previously unannotated repetitive sequence in mouse and rat reflects the relatively primitive state of existing repeat libraries for these organisms.

Throughout the paper, we use the term repeats to refer to all types of repetitive sequence, including low-complexity repeats, tandem repeats, multicopy genes/pseudogenes, segmental duplications and transposons. Low-complexity repeats and tandem repeats are easily recognized using programs, such as Nseg (Wootton and Federhen, 1993) and Tandem Repeats Finder (Benson, 1999). However, distinguishing between multicopy genes/pseudogenes, segmental duplications and transposons is a hard problem, which is beyond the scope of RECON and beyond the scope of our RepeatScout algorithm. We address this by using existing annotations of genes/pseudogenes and segmental duplications to analyze RepeatScout results for human, mouse and rat X chromosomes.

2 APPROACH

Suppose we observe that a particular *l*mer has many exact matches in the genome, and we observe additional sequence similarity in the regions surrounding the exact *l*mer matches, with the sizes of these similar regions varying considerably. In this scenario, we have roughly identified many approximate partial occurrences of a repeat family plus some spurious sequences which happen to share the same *l*mer. But how do we define the repeat element boundaries and determine the consensus sequence of this repeat family?

We address these questions with a rigorous definition. Let S_1, \dots, S_n denote DNA sequences each of which contains a similar repeat element and extends past the boundary of that repeat element on each side. By ‘similar’ we mean that each repeat element aligns with all or a part of a common consensus sequence, i.e. each one may be only a partial repeat. In practice, S_1, \dots, S_n will typically represent regions surrounding exact matches of a high-frequency *l*mer. We jointly define the underlying consensus sequence Q , and the substrings R_1, \dots, R_n of S_1, \dots, S_n specifying the precise boundaries of the repeat elements, as follows. We define Q to be the sequence which maximizes

$$A(Q; S_1, \dots, S_n) = \left[\sum_k \max\{a(Q, S_k), 0\} \right] - c|Q|,$$

where $a(Q, S_k)$ is a pairwise fit-preferred alignment score and $|Q|$ is the length of Q ; the repeat frequency threshold c imposes a minimum threshold on the number of repeat elements which must align with any given position of Q .² We define R_k to be the substring of S_k appearing in the optimal alignment of Q and S_k with score $a(Q, S_k)$.

Consider the typical case of a repeat family with left and right boundaries shared by some (but not all) of its repeat elements, with many partial repeat elements truncated short of those boundaries (Fig. 1a). The choice of pairwise alignment score $a(Q, S)$ which is used in $A(Q; S_1, \dots, S_n)$ is critical to proper annotation of the repeat boundaries. One possibility is to let $a(Q, S)$ be a local alignment score; letting $f(i, j)$ be the score of the optimal alignment ending at position i of Q and position j of S , we have:

$$\begin{aligned} f(i, 0) &= 0, \\ f(0, j) &= 0, \\ f(i, j) &= \max \begin{cases} f(i-1, j-1) + \mu_{ij} \\ f(i, j-1) - \gamma \\ f(i-1, j) - \gamma \\ 0 \end{cases}, \\ a(Q, S) &= \max_{i,j} f(i, j), \end{aligned}$$

²The adjustment $-c|Q|$ in the formula for $A(Q; S_1, \dots, S_n)$ assumes a nucleotide match score of 1 in the alignment score $a(Q, S)$; if the match score is set to $\mu^+ \neq 1$, then this adjustment should equal $-c\mu^+|Q|$.

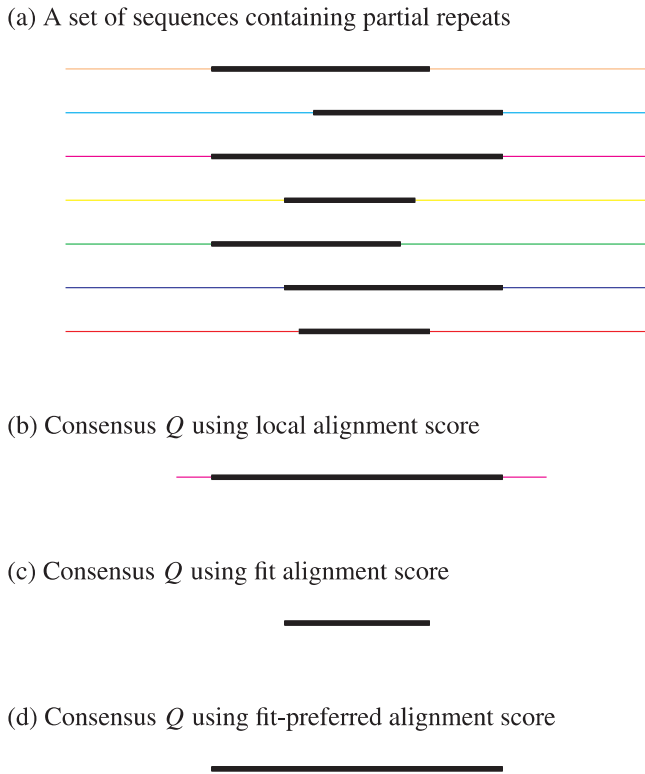
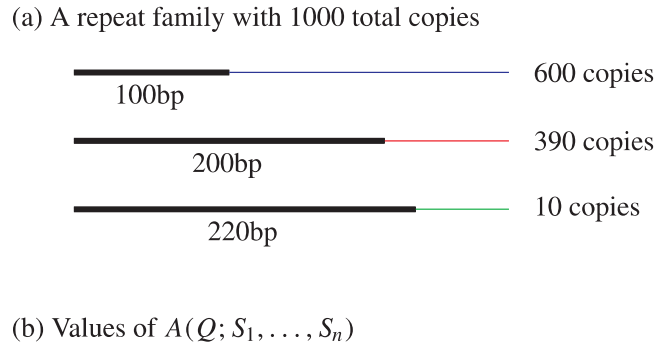


Fig. 1. Consensus sequence boundaries obtained using various pairwise alignment scores. **(a)** A set of sequences containing partial repeats. Thick black lines denote homologous partial repeat fragments, and thin colored lines denote nonhomologous random sequences. **(b)** The consensus sequence Q optimizing $A(Q; S_1, \dots, S_n)$ using a local alignment score $a(Q, S)$ contains spurious sequence extending past the true repeat boundaries. **(c)** The consensus sequence Q optimizing $A(Q; S_1, \dots, S_n)$ using a fit alignment score $a(Q, S)$ contains only the portion of the repeat family appearing in most repeat fragments. **(d)** Our fit-preferred alignment score $a(Q, S)$ leads to a consensus sequence Q with proper repeat boundaries.

where μ_{ij} is a match/mismatch score between positions i of Q and j of S , and γ is a fixed gap penalty. However, if $a(Q, S)$ is a local alignment score, the consensus sequence Q optimizing $A(Q; S_1, \dots, S_n)$ will contain spurious sequence extending past the true repeat boundaries, which improves a few local alignments without hurting the rest (Fig. 1b). An alternative is to require the boundaries of Q to be shared by all alignments by letting $a(Q, S)$ be a fit alignment score (Waterman, 1995), i.e. the score of the optimal alignment fitting all of Q into S . However, if $a(Q, S)$ is a fit alignment score, the consensus sequence Q optimizing $A(Q; S_1, \dots, S_n)$ will contain only the portion of the repeat family appearing in most repeat fragments (Fig. 1c). What we really want is an alignment score $a(Q, S)$ which encourages boundaries of Q shared by some (but not necessarily all) alignments. Our solution is to use a fit-preferred alignment score $a(Q, S)$ which subtracts a fixed



$a(Q, S)$	$Q=100\text{bp}$	$Q=200\text{bp}$	$Q=220\text{bp}$
local	100,000	140,000	140,200
fit	100,000	80,000	72,400
fit-preferred	100,000	128,000	120,400

Fig. 2. Choices of Q using various pairwise alignment scores. **(a)** A repeat family with 1000 total copies. Thick black lines denote homologous partial repeat fragments and thin colored lines denote nonhomologous random sequences. **(b)** Values of $A(Q; S_1, \dots, S_n)$ for each choice of Q and each choice of alignment score $a(Q, S)$. The optimal Q for each alignment score is shown in bold.

incomplete-fit penalty p if the alignment does not extend all the way to the right end of Q , with an analogous penalty for the left end of Q :³

$$f(i, 0) = \max(-\gamma i, -p),$$

$$f(0, j) = 0,$$

$$f(i, j) = \max \begin{cases} f(i-1, j-1) + \mu_{ij} \\ f(i, j-1) - \gamma \\ f(i-1, j) - \gamma \\ -p \end{cases},$$

$$a(Q, S) = \max_{i,j} \begin{cases} f(i, j) & \text{if } i = |Q| \\ f(i, j) - p & \text{if } i < |Q| \end{cases}.$$

Our fit-preferred alignment score $a(Q, S)$ leads to a consensus sequence Q with proper repeat boundaries (Fig. 1d).

We illustrate with an example. Consider a repeat family with 1000 total copies, in which 600 copies have a length of 100 bp, 390 copies extended on one side to a length of 200 bp and 10 copies extended on one side to a length of 220 bp (Fig. 2a). For simplicity, let us suppose that the copies have no mutations from the consensus sequence. The consensus sequences of length 100, 200 and 220 bp are each candidates for the optimal Q . Assuming match/mismatch scores of

³If Q had only a single boundary, instead of separate left and right boundaries, then the fit-preferred alignment score would equal the larger of either the local alignment score minus p , or the fit alignment score.

1 and -1 and incomplete-fit penalty $p = 20$, the values of $A(Q; S_1, \dots, S_n)$ for each consensus sequence Q , and for each choice of alignment score, are listed in Figure 2b. We see that the local alignment score chooses the 220 bp consensus sequence, even though it is represented in only a few copies; as we will see below, the RECON algorithm (Bao and Eddy, 2002) also has this weakness. However, the fit alignment score chooses the 100 bp consensus sequence, ignoring strong evidence of a 200 bp consensus. Only the fit-preferred alignment score chooses the 200 bp consensus. We note that more complicated examples, in which the length of partial repeat copies is variable, yield similar conclusions.

Now that we have rigorously defined our objective function $A(Q; S_1, \dots, S_n)$, the next question is how to optimize it. The naive approach of solving an n -dimensional dynamic programming problem is clearly computationally intractable. However, the heuristic approach of extending short exact seeds to identify high-scoring alignments has proven successful in pairwise alignment algorithms, such as BLAST (Altschul *et al.*, 1990). We generalize this approach to optimizing our objective function $A(Q; S_1, \dots, S_n)$ as follows. Given a high-frequency l mer, let S_1, \dots, S_n be the regions surrounding exact matches of the l mer, extended far enough on each side (say 10 000 bp) such that they surely extend past the boundaries of the repeat elements in question. We set Q_0 equal to the l mer, and greedily extend the consensus sequence Q maximizing $A(Q; S_1, \dots, S_n)$ one nucleotide at a time: at iteration t , we compute $A(Q_t \cdot N; S_1, \dots, S_n)$ for each nucleotide $N \in \{A, C, G, T\}$ (where $Q_t \cdot N$ represents Q_t with N appended), and set Q_{t+1} equal to the choice of $Q_t \cdot N$ which maximizes $A(Q_t \cdot N; S_1, \dots, S_n)$. We note that the alignment scores $a(Q_t \cdot N, S_k)$ can be computed quickly from alignment scores of the previous iteration by storing alignment scores corresponding to different offsets (see Section 3); this is a straightforward generalization of Smith–Waterman alignment (Smith and Waterman, 1981). Thus, Q is greedily extended 1 nt at a time to a progressively longer consensus sequence. In order to limit the running time, the extension process terminates after a specified number I of iterations fails to improve on the optimal score $A(Q; S_1, \dots, S_n)$. We extend the original l mer seed Q_0 first to the right and then to the left⁴ in this manner, thus obtaining the consensus sequence Q optimizing $A(Q; S_1, \dots, S_n)$. Consensus sequences Q obtained in this way, shorter than a specified length (say 50 bp), are later discarded (see Section 3); this prevents spurious high-frequency l mers (which cannot be extended to a longer repeat family) from entering our repeat family library.

Given a genome and a high-frequency l mer, the above procedure outputs a single repeat family consensus sequence Q . We would like to apply this procedure to all high-frequency

l mers, to identify all repeat families in the genome. However, extending literally every high-frequency l mer would be immensely redundant, as most repeat families spawn a large number of high-frequency l mers (often much larger than the length of the repeat family, due to mutations). A sensible approach is to identify occurrences of a repeat family after its consensus sequence Q is computed and adjust l mer frequency counts to exclude counts from those repeat occurrences, in order to preclude rediscovering the same repeat family. Identifying the occurrences of Q could be done using RepeatMasker (Smit and Green, <http://repeatmasker.org>); however, as a typical genome contains hundreds or thousands of repeat families and this step is run separately for each repeat family, this approach is not practical. Instead, we crudely and efficiently identify most occurrences of Q by using our pairwise fit-preferred alignment score $a(\cdot, \cdot)$ to extend matches of all high-frequency l mers of Q .

The overall algorithm is now clear. We start by building a table of high-frequency l mers. We extend the most frequent l mer to a repeat family Q , identify occurrences of Q in the genome and adjust l mer frequency counts to exclude counts from occurrences of Q and proceed to the most frequent remaining l mer. The algorithm terminates when there are no remaining l mers with frequency at least m , a fixed l mer frequency threshold.

An obvious improvement to the RepeatScout algorithm as described above is to greedily extend the consensus sequence by >1 nb at a time; although a naive implementation of this approach is rather slow, the running time can be significantly improved by precomputing all extensions of the dynamic programming table, as in the Four-Russians Speedup (Myers, 1992). Another possible improvement is to reoptimize Q , refining the consensus sequence after optimal alignment boundaries have been determined. We have chosen to limit our implementation to the greedy single nucleotide extension approach; our experiments indicate that this simpler approach generally produces accurate consensus sequences so long as a good seed exists.

The problem of finding the consensus sequence Q optimizing $A(Q; S_1, \dots, S_n)$ is similar to the consensus alignment problem for sequences S_1, \dots, S_n (Li *et al.*, 1999). Although there exists a polynomial time approximation scheme for the global consensus alignment problem (Li *et al.*, 1999), our problem is more difficult, since (1) the optimal alignment boundaries are different for each sequence S_k and are not known a priori and (2) we are not trying to align all sequences S_1, \dots, S_n , since some of these sequences may be spurious or may belong to distinct repeat families which share the same seed l mer. We note that spurious sequences will be excluded as the consensus sequence is extended, because they will eventually have a negative alignment score (due to the incomplete-fit penalties). The case of repeat families with mosaic subrepeat structure, e.g. which share the same seed l mer as part of a shared mosaic subrepeat, merits further description: in

⁴Since sequences S_k with $a(Q, S_k) < 0$ are excluded from subsequent computations, it could in theory, matter in which order we extend the right or left. However, this does not affect our results in practice.

this case, RepeatScout will initially return one of the families (typically the family with the most occurrences in S_1, \dots, S_n). Although l mer frequency counts will subsequently exclude counts from occurrences of the shared mosaic subrepeat, the other repeat families will be later separately identified—and extended to their full length, including the shared subrepeat—using an l mer seed which is not a part of the shared subrepeat. As with existing algorithms, such as RECON (Bao and Eddy, 2002), RepeatScout does not explicitly illustrate the mosaic subrepeat structure of the repeat families it identifies. However, this mosaic structure can be illustrated by running the A-Bruijn alignment algorithm (Raphael *et al.*, 2004) on the output of RepeatScout, or more generally on any repeat family library (D. Zhi, A.L. Price, B.J. Raphael and P.A. Pevzner, 2005, submitted for publication).

3 METHODS

Iterative computation of alignment scores. The alignment score $a(Q_t \cdot N, S_k)$ can be computed quickly from alignment scores of the previous iteration: letting the offset of an alignment be the net number of insertions minus deletions at iteration t , we compute and store alignment scores $a_f(Q_t, S_k)$ for each offset $f \in [-b, b]$, where b is a banding constraint. Then the values of $a_f(Q_t \cdot N, S_k)$ for $f \in [-b, b]$ can be computed easily from the values of $a_f(Q_t, S_k)$, and

$$a(Q_t \cdot N, S_k) = \max_{f \in [-b, b]} a_f(Q_t \cdot N, S_k).$$

In practice, a carefully chosen banding constraint b speeds up the algorithm without significantly affecting the results.

RepeatScout parameter settings. In each case, we ran RepeatScout with seed length $l = 15$; l mer frequency threshold $m = 3$, repeat frequency threshold $c = 3$, alignment match, mismatch scores and gap penalty equal to 1, -1 and -5 , respectively, incomplete-fit penalty $p = 20$, and banding constraint $b = 5$. To minimize the computational burden imposed by low-complexity and tandem repeats, we disallowed multiple occurrences of the same l mer within 500 bp of sequence by including only the first occurrence in our l mer frequency count. We extended high-frequency l mer seeds up to 10 000 bp on each side, terminating the extension process whenever $I = 500$ consecutive iterations failed to improve on the optimal score $A(Q; S_1, \dots, S_n)$. We removed repeat families with consensus sequence <50 bp long, and removed repeat families with at most 10 copies identified by RepeatMasker. We note that the number of repeat family copies identified by RepeatMasker is typically larger than the number of copies containing exact occurrences of the original l mer seed.

We now explain our choice of parameter settings. First, $l = 15$ was the smallest seed length which did not lead to an excessive number of spurious frequent l mers (e.g. if we had

used $l = 13$, then even a random l mer would be expected to have frequency >1 in any of our input sequences). Larger values of l slightly reduced the chances of a given repeat family having a frequent l mer seed and thus slightly reduced sensitivity. In general, we have found $l = \lceil \log_4 L + 1 \rceil$ to be a suitable choice of seed length for an input sequence of arbitrary length L . Next, $m = c = 3$ were the smallest values which allowed a reasonable running time; higher values of m and c reduced running time somewhat and slightly reduced sensitivity. We set $m = c$ in all of our experiments because both these parameters reflect our desired lower bound on the number of repeat elements which will contribute to our objective function. Next, we extended high-frequency l mer seeds up to 10 000 bp on each side and removed repeat families with consensus sequence <50 bp long, because most known transposon families have length between 50 and 10 000 bp. All other parameter choices were somewhat arbitrary and (within a reasonable choice) do not affect the algorithm's performance significantly. No effort was made to fine tune the algorithm's parameters to suit the datasets we analyzed.

RepeatMasker parameter settings. In each case, RepeatMasker was run using default scoring matrices. We used the least sensitive setting of RepeatMasker, in order to limit the computation time of the RepeatMasker runs; we note that running RepeatMasker at its most sensitive setting on the human X chromosome would take months on a single processor. Tests on a limited amount of sequence data indicate that running RepeatMasker at more sensitive settings increases the amount of repetitive sequence identified by all RepeatMasker runs but has little effect on the relative results.

Removal of low-complexity repeats, tandem repeats, multicopy gene/pseudogene families and segmental duplications. Repeat families with $>50\%$ of their length annotated as low-complexity by Nseg (Wootton and Federhen, 1993) were removed. Similarly, repeat families with $>50\%$ of their length annotated as tandem repeats by Tandem Repeat Finder (Benson, 1999) were removed. In human, mouse and rat analyses, repeat families whose occurrences (as identified by RepeatMasker) collectively had $>50\%$ of their sequence annotated as either gene/pseudogene families or segmental duplications were removed.

4 RESULTS

We benchmarked RepeatScout using the *C. briggsae* genome (Stein *et al.*, 2003), a recently assembled genome of length 108 Mb, which has already been analyzed using RECON (Bao and Eddy, 2002). We ran RepeatScout on *C. briggsae* using default parameter settings (explained in Section 3). RepeatScout's running time was 7 h on a single 0.5 GHz DEC Alpha processor. To insure a fair comparison with the RECON *C. briggsae* library, which restricted itself to repeat families with >10 copies (Stein *et al.*, 2003), we subsequently discarded repeat families ≤ 10 copies (see Section 3). Our

Table 1. Results of RepeatMasker runs on the *C.briggsae* genome using the library of repeat families produced by RepeatScout versus the raw library produced by RECON

	Masked by RepeatScout (Mb)	Not masked by RepeatScout (Mb)	Total (Mb)
Masked by RECON	23.1	2.0	25.1
Not masked by RECON	4.8	78.5	83.3
Total	28.0	80.5	108.4

We list the amount of genomic sequence masked by both, neither, or just one of the libraries.

resulting library contains 1391 repeat families of total length 0.65 Mb. In comparison, the raw library produced by RECON⁵ contains 723 repeat families of total length 0.43 Mb (Stein *et al.*, unpublished data).⁶ We ran RepeatMasker on the *C.briggsae* genome using each repeat library (Section 3), and compared the two runs (Table 1). RepeatScout identified 4.8 Mb of repetitive sequence missed by RECON, and RECON identified 2.0 Mb of repetitive sequence missed by RepeatScout. These differences are primarily due to different repeat families, not to different boundaries of the same repeat family: 3.8 Mb of the 4.8 Mb identified by RepeatScout only, and 1.5 Mb of the 2.0 Mb identified by RECON only, occur in blocks of length at least 50 bp which are identified by only one of the algorithms. Both the RepeatScout library and the raw RECON library contain low-complexity and tandem repeat families. However, removing low-complexity and tandem repeat families from both libraries (see Section 3) has little effect on the relative results.

We investigated why some repeat families are identified by RECON but missed by RepeatScout. In each case, occurrences of the repeat family are sufficiently diverged from its consensus sequence (the consensus sequence *l*mers either do not appear in the genome or have low frequency), which explains why RepeatScout cannot identify the repeat family. (An exception is low-complexity consensus sequence *l*mers, whose frequency is initially high but later decremented when low-complexity repeat families are identified, preventing these *l*mers from being chosen subsequently as seeds.) Relatively few *C.briggsae* repeat families are sufficiently diverged to prevent their identification via extension of

⁵The raw RECON library has been filtered of multicopy genes/pseudogenes, via a complex series of filters, to produce a filtered RECON library (Stein *et al.*, 2003). This biologically important step is beyond the scope of either RECON or RepeatScout. Because our intent is simply to benchmark the ability of RepeatScout to identify repetitive sequence, our comparison is based on raw output of RECON and RepeatScout, prior to filtering of multicopy genes/pseudogenes.

⁶We remark that the number of repeat families is not an adequate measure of the quality of different repeat family libraries. Repeats often have mosaic subrepeat structure and can thus be represented in many different ways, with a varying number of reported repeat families.

Table 2. Number and total length of repeat families in our RepeatScout library versus Repbase Update (Jurka, 1998, 2000) for human, mouse and rat

	RepeatScout Number of repeat families	Total length (Mb)	Repbase Number of repeat families	Total length (Mb)
Human	1010	0.64	620	0.97
Mouse	886	1.21	478	0.49
Rat	831	0.51	478	0.49

Repbase entries for mouse and rat are identical, since Repbase contains a single library for rodents.

a high-frequency consensus sequence *l*mer; it may be possible to identify such repeats by using approximate (rather than exact) high-frequency *l*mers as seeds.

We applied RepeatScout to the human, mouse and rat X chromosomes. Our goal was to identify repetitive sequence, excluding low-complexity or tandem repeats which is not already annotated as transposon families, multicopy gene/pseudogene families or segmental duplications. We ran RepeatScout on each X chromosome using default parameter settings. RepeatScout's running time was 8 h on a single 0.5 GHz DEC Alpha processor for the human X chromosome, 8 h for mouse and 10 h for rat. We removed low-complexity and tandem repeat families (see Section 3). Using genomic annotations of genes/pseudogenes (obtained from NCBI) and segmental duplications (Eichler, unpublished data), we subsequently removed repeat families whose occurrences predominantly correspond to annotated multicopy gene/pseudogene families or segmental duplications (see Section 3). We also removed repeat families with 10 copies or less. In Table 2, we list for each organism the number of repeat families in our resulting library and their total length, as compared with the Repbase Update library of transposon families (Jurka, 1998, 2000). We note that RepeatScout correctly identified the human *Alu* repeat family as a consensus sequence of length 282 bp followed by a poly-A tail; in contrast, the *Alu* consensus sequence produced by RECON has a length of 424 bp, reflecting the trimeric (LLR) configuration of a tiny fraction of *Alu* elements rather than the canonical dimeric (LR) configuration (Bao and Eddy, 2002). This example illustrates the effectiveness of RepeatScout's fit-preferred alignment approach in annotating repeat boundaries shared by many repeat elements.

For each X chromosome, we ran RepeatMasker (Smit and Green, <http://repeatmasker.org>) using either our RepeatScout library or the Repbase library for that organism (Section 3) and compared the two runs. The results are listed in Table 3. In each case, we restricted the comparison to the X chromosome excluding low-complexity or tandem repeats, annotated genes/pseudogenes and annotated

Table 3. Results of RepeatMasker runs on human X chromosome, mouse X chromosome and rat X chromosome, using the library of repeat families produced by RepeatScout versus Repbase Update (Jurka, 1998, 2000)

	Masked by RepeatScout (Mb)	Not masked by RepeatScout (Mb)	Total (Mb)
Human X chromosome			
Masked by Repbase	51.7	7.2	58.9
Not masked by Repbase	2.9	62.0	64.9
Total	54.6	69.2	123.8
Mouse X chromosome			
Masked by Repbase	43.2	2.7	45.9
Not masked by Repbase	6.4	62.4	68.7
Total	49.6	65.1	114.6
Rat X chromosome			
Masked by Repbase	49.8	3.2	53.0
Not masked by Repbase	7.2	77.3	84.5
Total	56.9	80.6	137.5

We list the amount of genomic sequence masked by both, neither or just one of the libraries. Low-complexity or tandem repeats, annotated genes/pseudogenes and annotated segmental duplications are excluded from each X chromosome in this analysis. Total X chromosome lengths are 153.7 Mb for human, 160.6 Mb for mouse and 160.8 Mb for rat.

segmental duplications.⁷ As Table 3 indicates, we find that 2.9 Mb or 2% of the human X chromosome, 6.4 Mb or 4% of the mouse X chromosome and 7.2 Mb or 4% of the rat X chromosome consist of previously unannotated repetitive sequence; these percentages serve as estimates for each entire genome. The previously unannotated repetitive sequence could be due to transposon families not present in Repbase, incomplete gene/pseudogene or segmental duplication annotations. Since the rodent genomes were assembled quite recently, the manually curated Repbase library of rodent transposon families is likely to be incomplete. This explains both the relatively large amount of repetitive sequence identified by RepeatScout alone, and the relatively small amount of repetitive sequence identified by Repbase alone, in mouse and rat. We speculate that much of the repetitive sequence identified by RepeatScout alone in mouse and rat consists of transposon families not present in Repbase. Indeed, further analysis of rodent RepeatScout results (Zhi *et al.*, 2005, submitted for publication) has identified both previously unannotated segmental duplications and transposons not present in Repbase. However, the Repbase library of human transposon families is based on many years of manual curation. Preliminary results indicate that much of the repetitive sequence identified by RepeatScout alone in human may consist of

diverged duplication units (Tang *et al.*, 2005, submitted for publication) which are not presently annotated as segmental duplications. The relatively large amount of human repetitive sequence identified by Repbase alone may be due not only to more thorough repeat family identification but also to further partitioning of repeat families into subfamilies. RepeatMasker typically masks more repeats when repeat families are partitioned into subfamilies, since such partitioning allows RepeatMasker to detect more diverged repeat elements. We have recently developed an efficient tool for repeat subfamily classification and applied it to *Alu* subfamily classification (Price *et al.*, 2004), but have not yet applied it to further process the RepeatScout library.

5 DISCUSSION

We have developed an accurate and efficient repeat family identification algorithm, as indicated by our benchmarking analysis on *C. briggsae*. Repetitive sequence can be further classified into low-complexity repeats, tandem repeats, multicopy gene/pseudogene families, segmental duplications and transposons. Low-complexity repeats and tandem repeats are easily recognized using programs such as Nseg (Wooton and Federhen, 1993) and Tandem Repeats Finder (Benson, 1999). However, distinguishing between multicopy gene/pseudogene families, segmental duplications and transposons remains a problem, one which we do not claim to solve. For human, mouse and rat, we elected to filter out repeat families corresponding to previously annotated multicopy gene/pseudogene families and segmental duplications, producing a library containing transposons and previously unannotated multicopy gene/pseudogene families and segmental duplications; for mouse and rat, there is reason to believe that our library contains a significant number of transposon families not present in Repbase. We note that the PILER algorithm (Edgar and Myers, 2005) achieves high specificity in distinguishing different types of repeats, at the sacrifice of some sensitivity; the methods of PILER may be useful in further classifying repeat families identified by RepeatScout.

RepeatScout is orders of magnitude faster than RECON, the dominant tool for *de novo* repeat family identification in newly sequenced genomes. For example, Bao and Eddy (2002), report that RECON required 4 h on a single 1.7 GHz Intel Xeon processor to run on 3.0 Mb of human sequence and 39 h to run on 9.0 Mb of human sequence. In contrast, RepeatScout took 6 min on a single 0.5 GHz DEC Alpha processor to run on 3.0 Mb of human sequence and 21 min to run on 9.0 Mb of human sequence and ran on the entire human X chromosome in just 8 h. Thus, RepeatScout is particularly well suited to analyzing large repeat-rich genomes. Although RepeatScout's running time is slightly superlinear in the size of the input sequence, efforts to optimize its implementation are nearly complete, enabling RepeatScout to analyze entire mammalian genome sequences. Our experiments on

⁷Although human, mouse and rat X chromosomes are of similar size, the sizes of the unannotated parts appearing in our comparison differ significantly for human, mouse and rat. This is due to large differences in the amount of segmental duplication annotations for these organisms, a consequence of the different parameter settings used to generate those annotations (Eichler, personal communication).

chromosomes of different lengths have shown that the quality of the results scales with the amount of input sequence; thus, we believe that running RepeatScout on an entire mammalian genome will improve our ability to identify low copy number or highly diverged repeat families which we currently miss.

With an increase in the number of genome sequencing projects, there is a pressing need to identify the repeat families present in large, newly sequenced genomes. For example, a growing amount of sequence data is now available for each of 28 mammalian species (NIH Intramural Sequencing Center, <http://www.nisc.nih.gov>). Application of RepeatScout to certain species with no existing repeat library has contributed to efforts to infer the phylogeny of these species using orthologous repeats (Bashir *et al.*, 2005). The benefits of such applications will surely multiply in the years ahead.

ACKNOWLEDGEMENTS

We are grateful to Lincoln Stein for providing us with the raw library of *C.briggsae* repeats produced by the RECON algorithm, prior to filtering of genes/pseudogenes, to Evan Eichler for providing us with segmental duplication annotations of human, mouse and rat X chromosomes and to Brian Haas for testing RepeatScout and offering numerous helpful comments and suggestions.

REFERENCES

- Agarwal,P. and States,D.J. (1994) The Repeat Pattern Toolkit (RPT): analyzing the structure and evolution of the *C.elegans* genome. In *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology (ISMB-94)*, AAAI Press, Stanford, CA, pp. 1–9.
- Altschul,S.F., Gish,W., Miller,W., Myers,E.W. and Lipman,D.J. (1990) Basic local alignment search tool. *J. Mol. Biol.*, **215**, 403–410.
- Bao,Z. and Eddy,S.R. (2002) Automated *de novo* identification of repeat sequence families in sequenced genomes. *Genome Res.*, **12**, 1269–1276.
- Bashir,A., Ye,C., Price,A.L. and Bafna,V. (2005) Orthologous repeats and mammalian phylogenetic inference. *Genome Res.* (To appear)
- Benson,G. (1999) Tandem repeats finder—a program to analyze DNA sequences. *Nucleic Acids Res.*, **27**, 573–580.
- Bourque,G., Pevzner,P.A. and Tesler,G. (2004) Reconstructing the genomic architecture of ancestral mammals: lessons from human, mouse, and rat genomes. *Genome Res.*, **14**, 507–516.
- Edgar,R.C. and Myers,E.W. (2005) PILER: identification and classification of genomic repeats. To appear in *Proceedings of the Thirteenth International Conference on Intelligent Systems for Molecular Biology (ISMB-05)*, Detroit, Michigan.
- Gibbs,R.A., Weinstock,G.M., Metzker,M.L., Muzny,D.M., Sodergren,E.J., Scherer,S., Scott,G., Steffen,D., Worley,K.C., Burch,P.E. *et al.* (2004) Genome sequence of the Brown Norway rat yields insights into mammalian evolution. *Nature*, **428**, 493–521.
- International Human Genome Consortium. (2001) Initial sequencing and analysis of the human genome. *Nature*, **409**, 860–921.
- Jurka,J. (1998) Repeats in genomic DNA: mining and meaning. *Curr. Opin. Struct. Biol.*, **8**, 333–337.
- Jurka,J. (2000) Repbase Update: a database and an electronic journal of repetitive elements. *Trends Genet.*, **9**, 418–420.
- Kazazian,H.H.,Jr (2004) Mobile elements: drivers of genome evolution. *Science*, **303**, 1626–1632.
- Kurtz,S. and Schleiermacher,C. (1999) REPuter: fast computation of maximal repeats in complete genomes, *Bioinformatics*, **15**, 426–427.
- Kurtz,S., Ohlebusch,E., Schleiermacher,C., Stoye,J. and Giegerich,R. (2000) Computation and visualization of degenerate repeats in complete genomes. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology (ISMB-00)*, AAAI Press, San Diego, CA, pp. 269–278.
- Li,M., Ma,B. and Wang,L. (1999) Finding similar regions in many strings. In *Proceedings of the 31st ACM Annual Symposium on Theory of Computing*. Atlanta, Georgia, pp. 473–482.
- Myers,E.W. (1992) A Four Russians algorithm for regular expression pattern matching. *J. ACM*, **39**, 430–448.
- Pevzner,P.A., Tang,H. and Waterman,M.S. (2001) An Eulerian path approach to DNA fragment assembly. *Proc. Natl Acad. Sci. USA*, **98**, 9748–9753.
- Pevzner,P.A., Tang,H. and Tesler,G. (2004) *De novo* repeat classification and fragment assembly. *Genome Res.*, **14**, 1786–1796.
- Price,A.L., Eskin,E. and Pevzner,P.A. (2004) Whole-genome analysis of *Alu* repeat elements reveals complex evolutionary history. *Genome Res.*, **14**, 2245–2252.
- Raphael,B., Zhi,D., Tang,H. and Pevzner,P.A. (2004) A novel method for multiple alignment of sequences with repeated and shuffled elements. *Genome Res.*, **14**, 2336–2346.
- Smit,A.F.A. and Green,P. RepeatMasker, <http://repeatmasker.org>
- Smith,T.F. and Waterman,M.S. (1981) Identification of common molecular subsequences. *J. Mol. Biol.*, **147**, 195–197.
- Stein,L.D., Bao,Z., Blasiar,D., Blumenthal,T., Brent,M.R., Chen,N., Chinwalla,A., Clarke,L., Clee,C., Coghlan,A. *et al.* (2003) The genome sequence of *C.briggsae*: a platform for comparative genomics. *PLoS Biol.*, **1**, E45.
- Volfovsky,N., Haas,B.J. and Salzberg,S.L. (2001) A clustering method for repeat analysis in DNA sequences. *Genome Biol.*, **2**, RESEARCH0027.
- Waterman,M.S. (1995) *Introduction to Computational Biology: Sequences, Maps and Genomes*. Chapman and Hall/CRC Press, Boca Raton, FL.
- Wootton,J.C. and Federhen,S. (1993) Statistics of local complexity in amino acid sequences and sequence databases. *Computers and Chemistry*, **17**, 149–163.