

# RepeatAssembler: A Package for Annotation of Full-length Repetitive DNA Sequences in Fungal Genomes

Mark L. Farman  
Department of Plant Pathology  
University of Kentucky  
farman@uky.edu

Jerzy W. Jaromczyk  
Department of Computer Science  
University of Kentucky  
jurek@cs.uky.edu

Joshua W. Gilkerson  
Department of Computer Science  
University of Kentucky  
jwgilk0@cs.uky.edu

Chuck Staben  
Department of Biology  
University of Kentucky  
staben@uky.edu

## Abstract

REPEATASSEMBLER aims to automate the identification of full-length consensus sequences for repetitive DNA fragments or the genetic material that is repeated many times throughout the genome (so-called repeats). Although there are packages with similar objectives, e.g. RECON [1], none fully meets the needs of researchers in comparative genomics.

Joining short repeating elements into full-length repeats is particularly intricate and handling this task properly is both the goal and the strength of our solution. REPEATASSEMBLER analyzes as few sequences as possible to screen out all repetitive sequences. Then it builds a set of non-redundant repeats and reports the full-length consensus sequence for each element in the set.

REPEATASSEMBLER consistently delivers repetitive sequences that agree with known full-length repeats for several fungal genomes.

## 1. Background

Much of the genetic material [3] is repeated many times throughout the genome. These repeats arise through various processes within the cell that involve the copying of genetic material.

Some repetitive DNA sequences have important structural and biochemical (protein- or RNA-coding) roles in chromosome function and gene expression. Many repeats, however, are molecular parasites that can insert into chromosomal DNA, where they are replicated as part of the cell's own DNA. These repeats are often capable of becoming

mobilized, often inserting into new chromosomal locations. This transposition process is sometimes accompanied by genome reorganization, as well as amplification of repeat copy number. Another class of repeats, known as retrotransposons, is comprised of elements that are transcribed into RNA copies which are then reverse-transcribed back into DNA, which inserts into new chromosome sites.

Repetitive DNA sequences are often regarded as junk DNA that provides little information on the protein-coding potential of the genome. For this reason, it is useful to screen out the repetitive DNA, leaving behind the more interesting genes. On the other hand, repetitive DNAs can provide information about how genomes evolve, and sometimes have interesting evolutionary histories themselves. Therefore, identification and documentation of their position in the genome can yield important insights into the molecular processes underlying key evolutionary events.

In a real genome, many of the repeated sequences are found not only in their entirety, but also occur as pieces of repeated segments that have been overwritten by the same processes that created them.

Annotation of these sequences has been done with manual methods for some genomes, but it is a long and cumbersome task. A computerized method to either do the complete identification of repeats or to offer some assistance to researchers is needed.

## 2. Evaluation Criteria for Identifying Repeats

Our criteria for defining the consensus sequence for a repeat include the length of a segment, the number of occurrences, and the fidelity of the endpoints among others.

The number of times a sequence occurs in the genome is of course important in deciding if a given sequence is a

repeat.

As well as occurring many times, longer sequences are favored over shorter ones because we want to find a non-redundant set of repeats and a subsequence will always appear at least as many times as its super-sequence. As a result, relying on the number of occurrences without considering the length will result in choosing a large number of short repeats.

We also consider the fidelity of the endpoints of each sequence. Because of the chance of random matches or of instances where two repeats are adjacent in some *contigs*, simply favoring longer and more prevalent sequences would often result in sequences that are too long or reported repeats that are actually two repeats.

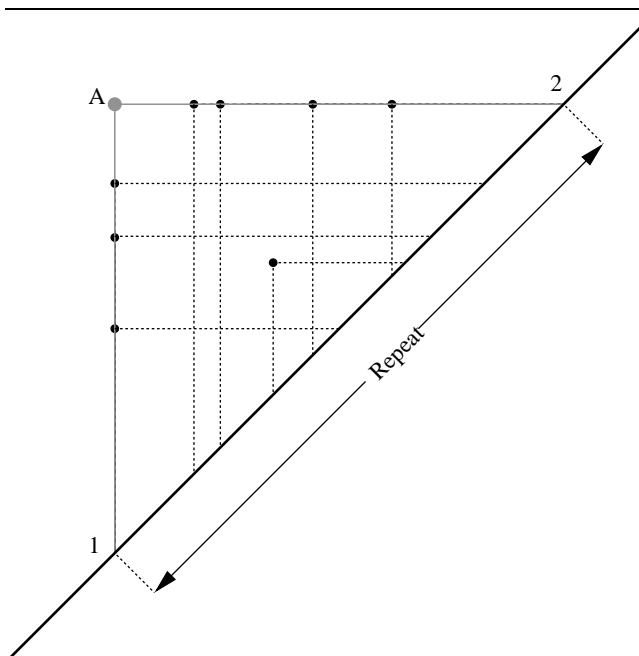


Figure 1. A Simple Example

Figures 1 and 2, are schematic representations of two situations that sometimes arise when searching a genome for sequences that are repeated. The diagonal line represents a portion of the genome sequence and each black dot occupies the Cartesian coordinate corresponding to the endpoints of a region that has a match somewhere else in the genome. In example 1, many of the repeats in the genome are not full-length and so their endpoints will vary relative to the reference. Nevertheless, the true termini of a repeat are identifiable as the position where multiple matches share the same endpoint (indicated as 1 and 2). Figure 2 shows a situation in which several matches support one pair of endpoints (1 and 2), while another set supports a third endpoint (3). This is expected when a large repeat contains a sub-

repeat sequence that exists in the genome independent from the larger element.

This difficulty and ambiguity over what constitutes a repetitive sequence is the motivation to make our solution as flexible as possible and to give the user the ability to select the set of repeats based on their criteria.

### 3. Algorithm

The most challenging aspect of repeat annotation is the lack of a precise definition of what constitutes a repeat. In particular, automatic computer methods lend themselves better to solving problems that can be precisely and unambiguously stated. To further confound the search for a solution, not only is the existing definition not concise, but there is much disagreement among geneticists concerning even an ambiguous definition, specifically when it concerns how to select exact boundaries [5].

One of the answers to such a situation is to design a solution that allows the user to easily modify and tune the decision rules and to include many tunable options so that a user may customize the decision-making process to meet their own definition or their needs for a specific application.

The underlying algorithm of REPEATASSEMBLER began as our effort to emulate a manual identification method developed and used by one of us (MF) when constructing sets of repeats. Over time, many changes have been made that take advantage of the capabilities of computer systems. However, the resulting algorithm is still very intu-

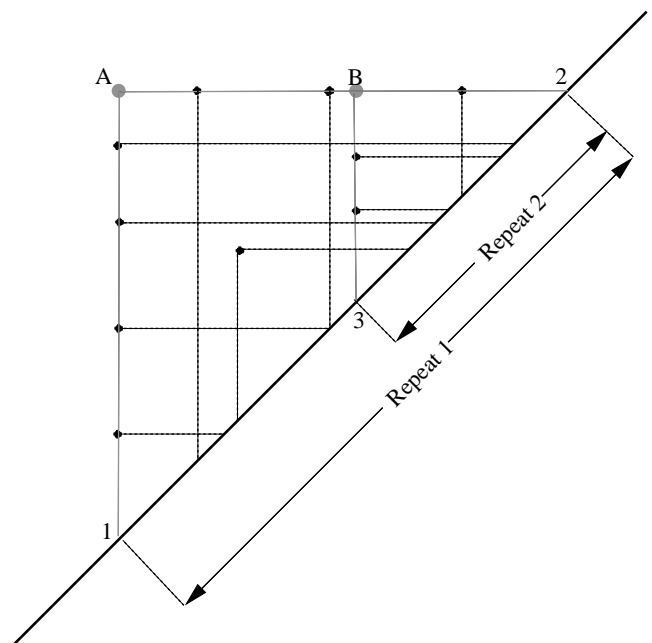


Figure 2. An Ambiguous Example

itive and easily understandable. These properties are important because one of the premises is to provide a tool that researchers can tune to meet their unique needs.

The algorithm follows these steps:

**BLAST** The genome is blasted against itself.

**DIVIDE** The genome is divided into segments.

**EVALUATE** Each segment is evaluated with respect to the plausibility that it is part of a repeat.

**ADD AND PRUNE** using an iterative process, the best segments are chosen and added to the set of repeats. Each time a segment is chosen, it and other segments are removed from consideration.

They are described in more details below.

### 3.1. BLAST

Initially, the genome is blasted against itself to identify segments that match other segments (i.e., candidates for repeats) in the genome and to produce tabular output that will be read by REPEATASSEMBLER.

Other tools than BLAST can be used for the same purpose. BLAST provides for a good integration with the Generic Genome Browser [2].

### 3.2. Divide

During the DIVIDE stage of execution, the genome is broken down into segments. This segmentation is based on the endpoints of the BLAST results.

There are a number of ways to do segmentation. One simply takes each piece of the genome involved in a match. This results in segments that may overlap, but each segment is involved in at most one match. Alternatively, the genome can be divided such a way that segments do not overlap. Each segment may be involved in multiple matches, however. Currently, the former is our preferred method.

For the remainder of the execution, the genome is only considered in terms of the segments generated in this DIVIDE step and their relationships to one another.

### 3.3. Evaluate

Each of the segments is modified in an attempt to find its true endpoints and evaluated with regard to the plausibility that it is a repeat or part of one.

The tunable parameters come into play in this step. At present all tuning is done directly in the code, but as our research and testing continues, it will be possible to set parameters at invocation.

The parameters used for evaluation include the length of the segment, the total number of matched bases ( the sum of the length of each match within the segment ), the surety of

the endpoints, and the number of times the entire segment matches another.

### 3.4. Add and Prune

Using an iterative process, the segments best meeting the rules are selected and used to update the set of candidate repeats. Updating may create new candidate repeats or extend existing ones. Once selected, segments are removed from further consideration. This removal from consideration may also entail reevaluating related segments.

If the selected segment overlaps with a current candidate repeat, they may be merged to form one later. If it is contained within an existing candidate repeat, it is neglected. Otherwise, the segment will become a new candidate repeat.

## 4. Implementation and Interface

REPEATASSEMBLER integrates with BLAST, the Generic Genome Browser, and any applications that read GFF or FASTA file formats. Input is accepted in multiple formats, including FASTA and BLAST report.

BLAST is used to preprocess the genome by blasting it against itself. The output from BLAST is then used by REPEATASSEMBLER.

The internal design of REPEATASSEMBLER is such that data manipulation and searching can be done using either an external SQL server, an embedded SQL engine, or a custom internal data structure. This gives us great flexibility in deployment options.

Output can be given in either an easily understandable report that summarizes information on the repeats identity, or in one of several machine-readable formats including FASTA and the General Feature Format (GFF). These output formats allow our application to integrate easily with other software. The GFF output from REPEATASSEMBLER may be loaded into the Generic Genome Browser [2] to allow visual examination. We have also included a number of scripts that generate GFF files to display the self-BLAST results, and the results of any other BLAST query.

It is also possible to output a FASTA file containing the reference sequence for each repeat. This can then be used to BLAST for the repeats or compare to other sequences.

Thanks to full integration with existing tools including NCBI BLAST [4], and the Generic Genome Browser [2], the REPEATASSEMBLER provides an intuitive interface that allows the user to quickly identify biologically-relevant information.

## 5. Results

As test cases we are using genome data [6] for the fungi *Magnaporthe Grisea* and *Aspergillus Nidulans*. These were chosen because of our experience with them and because the repeats in these genomes are relatively well characterized. Manual annotation reveals that there are probably only 17 repeats (with a copy number of 10 or greater) in the *Magnaporthe grisea* genome (and certainly no more than 20).

REPEATASSEMBLER has succeeded in finding the majority of repeats with zero false positives and consistently delivered repetitive sequences that agree with known full-length repeats for the above fungal genomes.

## 6. Acknowledgments

This work was supported by a subcontract to the University of Kentucky from the KBRIN grant 5P20RR016481-03 to Nigel Cooper at the University of Louisville from the NCRN and by KY NSF EPSCOR grant EPS-0132295. M. Farman acknowledges support from the National Science Foundation MCB-0135462.

## References

- [1] Bao, Zhiron and Eddy, Sean R. Automated De Novo Identification of Repeat Sequence Families in Sequenced Genomes. *Genome Research*, Vol. 12, Issue 8, 1269-1276, August 2002. Available online at: <http://www.genome.org/cgi/content/abstract/12/8/1269>
- [2] GMOD: Generic Genome Browser: <http://www.gmod.org/ggb/>.
- [3] Lewin, B., *Genes VII*, Oxford University Press, 2000.
- [4] WU-BLAST. <http://blast.wustl.edu>
- [5] Tchourbanov et al. A New Approach for Gene Annotation Using Unambiguous Sequence Joining. *Proceedings of the 2003 IEEE Bioinformatics Conference, CSB2003*, 353-362, 11-14 August 2003.
- [6] Fungal Genome Databases: <http://www.hgmp.mrc.ac.uk/GenomeWeb/fungal-gen-db.html>