Creation of a comprehensive repeat library for a newly sequenced parasitic worm genome

Matthew Berriman (mb4@sanger.ac.uk)
Berriman Lab Group, Sanger Institute
Avril Coghlan
Berriman Lab Group, Sanger Institute
Isheng Jason Tsai
Berriman Lab Group, Sanger Institute

Method Article

Keywords: repeat-finding, repeat library, repeat libraries, RepeatMasker, RepeatModeler, Transposon PSI, LTRharvest, LTRdigest, transposons, transposable elements, repetitive elements, repetitive DNA, repeats, parasites, genomes, assemblies, parasitic worms, helminths, nematodes, flatworms, platyhelminths, Nematoda, Platyhelminthes

Posted Date: May 15th, 2018

DOI: https://doi.org/10.1038/protex.2018.054

Abstract

It is useful to mask repeat elements within the assembly of a newly sequenced genome for tasks such as gene-finding, but also for the purpose of understanding evolution of repetitive elements. To do this, one must first construct a library of the repeat elements in the genome. We present a protocol for creating a comprehensive repeat library for the genome of a newly sequenced parasitic worm, based on merging repeat libraries made by RepeatModeler, TransposonPSI and LTRharvest.

Introduction

It is useful to mask repeat elements within the assembly of a newly sequenced genome for tasks such as gene-finding, but also for the purpose of understanding evolution of repetitive elements. To do this, one must first construct a library of the repeat elements in the genome. We present a protocol for creating a comprehensive repeat library for the genome of a newly sequenced parasitic worm, based on merging repeat libraries made by RepeatModeler, TransposonPSI and LTRharvest. RepeatModeler uses results from three other programs, RECON \(Bao & Eddy 2002), RepeatScout \(Price _et al._ 2005), and Tandem Repeats Finder \(TRF; Benson, 1999) to build a repeat library for an assembly. RepeatScout is good for finding highly conserved repetitive elements, while RECON can find less highly conserved elements. TransposonPSI is a program for finding repeats in genomes, by using PSI-BLAST \(Altschul _et al._ 1997) to search for sequence matches at the DNA or protein level to proteins encoded by transposable elements. It can be useful for finding transposable elements that are very degenerate, and so may not be found by other repeat-finding software. The types of transposable elements found by TransposonPSI include LTR retroelements \(e.g. gypsy and copia), non-LTR retroelements \(e.g. LINE retrotransposon ORFs), DNA transposons \(e.g. cryptons and other families), and helitrons. The LTRharvest software can be used to find LTR retrotransposons in a genome assembly. While many analyses of genomes use a single repeat-finding approach to produce a repeat library, we believe that by making a non-redundant library based on a combination of three different software packages \(LTRharvest, TransposonPSI, RepeatModeler), our protocol produces a more complete repeat library.

Reagents

Pfam database website GyDB database wbsite WormBase database website GeneDB database website

Equipment

Computer cluster.

Procedure

1. Build a repeat library for the newly sequenced genome using RepeatModeler \(RepeatModeler). 2. Build a second repeat library for the newly sequenced genome using TransposonPSI \(TransposonPSI).

TransposonPSI uses PSI-BLAST to search for sequence matches at the DNA or protein level to proteins encoded by transposable elements. Remove sequences of <50 bp from the TransposonPSI repeat library. 3. Build a third repeat library for the newly sequenced genome, by using LTRharvest \(Ellinghaus _et al._ 2008) to identify long terminal repeat \(LTR) retrotransposons. To remove likely false positives, and make a library of full-length \(or mostly full-length) LTR retrotransposon sequences, run LTRdigest on the LTRharvest results, with protein HMMs from Pfam \(Finn _et al._ 2016) \(those Pfam domains listed in tables B1 and B2 of Steinbiss _et al._ 2009) and GyDB \(Llorens _et al._ 2011). To further remove likely false positives, remove any LTR retrotransposon candidates without domain hits. 4. For each of the three repeat libraries from \(1), \(2), \(3), classify the repeats using RepeatClassifier \(part of the RepeatModeler software). 5. Merge the repeat libraries from RepeatModeler, TransposonPSI and LTRharvest by using USEARCH v7 \(Edgar 2010) to cluster the candidate sequences with ≥80% identity, and remove all but one sequence for each cluster. The resultant repeat library should be non-redundant. 6. To try to remove protein-coding genes from non-transposable element genes \(e.g. collagen repeats), then filter the non-redundant library by removing repeats that were classified as 'unknown' \(by RepeatClassifier) and that have BLASTN \(Altschul _et al._ 1997) hits of E≤0.001 against known protein-coding genes and ncRNA genes from nematodes and platyhelminths \(DNA sequences for _C. elegans_ protein-coding and ncRNA transcripts from WormBase WS235 \(Howe _et al._ 2016), and for _S. mansoni_ and _E. multilocularis_ protein-coding transcripts from GeneDB \(Logan-Klumpler _et al._ 2012)). 7. If it is required to repeat-mask the genome of interest, the non-redundant library from \(6) above can be used to mask repeats in the genome using RepeatMasker \(www.repeatmasker.org), with the –s \(sensitive) option. This will mask low complexity DNA and simple repeats as well as transposable elements.

Troubleshooting

RepeatModeler When running RepeatModeler, it may be necessary to rename the sequences in assembly to have simple names eg. C1, C2, C3, etc. as RepeatModeler gets confused by unusual characters or sequence names. If your assembly has lots of smallish scaffolds \(eg. average scaffold size 10kb), then it might be a good idea to cut down run-time by running RepeatModeller on just the subset of scaffolds that are quite long \(eg. >=100 kb). We find RepeatModeler can take several days or even a week or two to run on a large genome. For a 65-Mbase genome, it ran overnight and we reserved 3000 Mbyte of memory \(RAM) for it. If RepeatModeler crashes for no obvious reason, it may have ran out of memory, so it may be worth trying re-running it with lots \(e.g. 20,000 Mbyte) of memory. RepeatModeler uses RepeatScout, and RepeatScout will often find tandem repeats and low complexity sequences, which may be present in the RepeatModeler output file, and you might want to remove them. **TransposonPSI** We find TransposonPSI takes a long time \(e.g. 1 week) for some assemblies with lots of scaffolds \(e.g. ~100,000 scaffolds). To get around this, we glue together short scaffolds into long scaffolds, with 1000 'N's between each pair of short scaffolds. **RepeatClassifier** Running RepeatClassifier seems to require a lot of memory \(RAM), e.g. if a repeat library is $_{0.5\ \text{Mbase, it needs}}$ 100 Mbyte RAM and takes $_{1\ \text{day; if}}$ 5-10 Mbase then 5000 Mbyte and 2 days; if 20-50 Mbase then 5000 Mbyte

and >2 days. Sometimes RepeatClassifier takes a long time and produces a lot of large temporary output files; in this case the repeat library can be split up into smaller files of 500 sequences each, and RepeatClassifier run on each separately. **Merged repeat library** We find that for very fragmented genome assemblies, there appear to be quite a lot of low complexity sequences in the merged repeat library, which you may wish to identify \(e.g. using Tandem Repeats Finder; Benson 1999) and remove. **RepeatMasker** When running RepeatMasker, it may be necessary to rename the repeats in the repeat library if the sequence names are too long to use for RepeatMasker \(does not accept names of >80 characters).

Anticipated Results

The output is a repeat library. This is a fasta file of repeats for the genome of interest, each repeat classified by type \(by RepeatClassifier). If RepeatMasker is used to mask the genome with the repeat library, it produces a masked version of the genome, as well as a table showing the positions of repeats in the genome assembly.

References

Altschul, S. F. _et al._ Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. _Nucleic acids res._ **25**, 3389-402 \(1997). Bao, Z. & Eddy S. R. Automated _de novo_ identification of repeat sequence families in sequenced genomes. _Genome res._ **12**, 1269-76 \(2002). Benson, G. Tandem repeats finder: a program to analyze DNA sequences. _Nucleic acids res._ **27**, 573-80 \(1999). Edgar, R. C. Search and clustering orders of magnitude faster than BLAST. _Bioinformatics_ **26**, 2460-2461 \(2010). Ellinghaus, D. _et al._ LTRharvest, an efficient and flexible software for de novo detection of LTR retrotransposons. _BMC bioinformatics_ **9**, 18 \(2008). Finn, R. D. _et al._ The Pfam protein families database: towards a more sustainable future. _Nucleic acids res._ **44**, D279-285 \(2016). Howe, K. L. _et al._ WormBase 2016: expanding to enable helminth genomic research. _Nucleic acids res._ **44**, D774-780 \(2016). Llorens, C. _et al._ The Gypsy Database \(GyDB) of mobile genetic elements: release 2.0. _Nucleic acids res._ **39**, D70-74 \(2011). Logan-Klumpler, F. J. _et al._ GeneDB–an annotation database for pathogens. _Nucleic acids res._ **40**, D98-108 \(2012). Price, A. L. _et al._ _De novo_ identification of repeat families in large genomes. _Bioinformatics_ **Suppl 1**, i351-8 \(2005). Steinbiss, S. _et al._ Fine-grained annotation and classification of de novo predicted LTR retrotransposons. _Nucleic acids res._ **37**, 7002-7013 \(2009).

Acknowledgements

We thank James Cotton and Diogo Ribeiro for advice on running RepeatModeler, and to Sascha Steinbiss for advice on LTRharvest and LTRdigest.