The SINE_Scan (version 1.1.1) Tutorial

Last updated: October 6, 2016

Hongliang Mao and Hao Wang

Contents

What is SINE_Scan?

SINE_Scan is a program to identify short interspersed nuclear elements (SINEs) from genomic DNA datasets. SINE_Scan verified SINE elements by multiple lines of evidence, including transpositionhallmark, structural signals (RNA POLIII binding domain, TSD, size etc) and copy number.

Prerequisites

The following software tools/modules should be installed in your system:

1. Perl and BioPerl Modules. All modules can be obtained from CPAN (http://search.cpan.org/).
    1) Statistics::Basic; (we have put this module into SINE_Scan subdirectory 'modules'. The user does not need to install it)
    2) Parallel::ForkManager; (we have put this module into SINE_Scan subdirectory 'modules'. The user does not need to install it)

3) Bio::AlignIO;
4) Bio::Align;
5) Bio::PairwiseStatistics;
6) Bio::SimpleAlign;
7) Getopt::Long;
8) File::Basename;
9) Get::Sdt;
10) Bio::SeqIO;
11) Bio::Seq;
12) Bio::Tools::Run::StandAloneBlast;
13) Bio::SearchIO;

2. Python. SINE_Scan uses an improved version of SINE-Finder as the default SINE candidatesidentification tool. SINE-Finderis written with Python.

3. SINE-Finder.py: The python script performs structural based*de novo*SINEs discovery.The original SINE-Finder.py can only find plant tRNA type SINEs. Wehave improved the algorithm. The enhanced version now can detect all three types (7SL, 5s and tRNA) of SINEs and is used by SINE_Scan. Here we prepared two patches (SINE_Finder-v1.1-7SLandtRNA.patch and SINE_Finder-v1.1-5sRNA.patch) to build the enhanced SINE-Finder from the original version. The enhanced SINE-Finder is automatically generated during the installation of SINE_Scan. What the user need to do are as follows:
   1) Download original version of SINE-Finder from http://www.plantcell.org/content/suppl/2011/08/29/tpc.111.088682.DC1/Supplemental_Data_Set_1-sine_finder.txt).
   2) Rename the file “Supplemental_Data_Set_1-sine_finder.txt” as “SINE-Finder.py” and put it to a directory. The path of SINE-Finder is needed in installation.
   3) Make sure that two LINUX commands “unix2dos” and “patch” have been

installed in the operating system.

4) Install SINE_Scan (see below).

4. NCBI Blast+: http://www.ncbi.nlm.nih.gov/.

5. Muscle: http://www.drive5.com/muscle/.

6. Bedtools:http://bedtools.readthedocs.org/en/latest/

7. Stretcher: http://emboss.open-bio.org/.

8. cd-hit: http://weizhong-lab.ucsd.edu/cd-hit/.

SINE_Scan Installation

To install SINE_Scan, go to the directory containing SINE_Scan program (called SINE_ScanDirectory hereafter) and run SINE_Scan_Installer.pl. Values of the following parameters should be provided:

1) **-d** the full path to SINE_Scan directory
2) **-a** the full path to the original version of SINE-Finder.py script
3) **-f** the full path to makeblastdb program of blast+ package
4) **-b** the full path to blastn program of blast+ package
5) **-M** the full path to muscle program
6) **-e** the full path to EMBOSS 'stretcher' program
7) **-c** the full path to cd-hit-est program
8) **-S** the full path to Known SINEs database file (FASTA format)
9) **-R** the full path to tRNA, 7SLRNA and 5SRNA database file (FASTA format)
10) **–l** the full path to bedtools program

We provide pre-equipped "known SINEs database" and/or "RNA databases" which

can be directly used as values of -S and -R. The paths of the two databases are SINE_ScanDirectory/SINEBase/SineDatabase.fasta and SINE_ScanDirectory/RNABase/RNAbase.fasta, repectively.

An example of installation of SINE_Scan:
perl SINE_Scan_Installer.pl -d /Home/SINE/SINE_Scan/ -a /Home/SINE-Finder.py -f /Home/blast+/bin/makeblastdb -b /Home/blast+/bin/blastn -M /Home/software/Muscle/muscle -e /Home/software/EMBOSS/emboss/stretcher-c /Home/software/cd-hit/cd-hit-est -l /Home/software/bedtools2/bin/bedtools-S /Home/database/SINE/sines.fasta -R /Home/database/RNA/rna.intergated.fasta

Running SINE_Scan

Input

A file containing genomic DNA (FASTA format)is the only required input file if SINE_Scan runs in the fully automatic mode (controlled by parameter "-s 123", which is the default mode).

SINE_Scan is very flexible. Users can customize it to fulfill multiple purposes. When doing this, additional input files are required. Two frequently ways of using SINE_Scan are: (1) If the purpose is to verify a set of SINE candidates, the user can run "-s 23" and use a FASTA file containing these candidates as the additional input file. (2) If SINEs are known, and the purpose is to perform classification and investigate their genomic distribution, the user can run "-s 3" mode. Now these SINE sequencesare used as the additional input file.

Output

Module 1 outputs a FASTA file containing all SINE candidates.

Module 2 generates verified SINEs and useful information for manual inspection. For each SINEcandidate, its highly similar sequences in the genomeare grouped as one cluster. For each cluster, SINE_Scan generates a directory and two files arestored in the subdirectory:(1) a FASTA file (name ends with “for_annotation.fa”) containing SINE_Scan verified SINE candidate. The file is empty if the candidate is “bad”; (2) multiple sequence alignments (MSA) of this cluster, based on which the user can perform manual inspection (see below).

Module 3 outputs three files: (1) The distribution of SINEs in the genomic DNA (GFF format); (2) SINE family representative sequences (FASTA format); (3) All identified SINE sequences in the genomic DNA (FASTA format).

The files generated by Module 3 are final output, which are deposited in the output directory, controlled by parameter “-z” (default: ./). Others are intermediate files. All intermediate files were placed in the working directory, which is set by parameter “-d”.

Parameters of SINE_Scan

The main program of SINE_Scan is a Perl script calledSINE_Scan_process.pl. Parameters controlling performance of this program are listed below. The user can also type SINE_Scan_process.pl in the command line environment to get the information.

1. Mandatory parameters:

-gThe file of genomic sequences (required).

-s Mode of running SINE_Scan (default: 123).

-d Working directory in which intermediate files are stored (required).

-o Genome identifier used as prefix of final output files. Format: genus_speices_strain

(or version). (required).

-kThe number of CPU used when performing blast search (default: 2).

-w Continue to run the pipeline from the point it stops. The value of –w only can be assigned as 2 or 3, which means the pipeline begins from Module 2 or 3. Use this parameter only based on an aborted previous run. If this parameter is used, no other parameters need to be provided. The programwill read parameters of a log file named "para.log" which records information of the previous run.

2. Parameters related to SINE verification in Module 2:

-iThe file of customized candidate SINE sequences in FASTA format file (only used with -s 2 or -s 23).

-n Minimal copy number of valid SINEs (default: 5).

-F The length of the flanking sequences used to findthe SINE transpositionhallmark (default: 60).

-E The length of SINE ends used to find the SINE transpositionhallmark (default: 25).

-D Minimum difference between SAQs of SINE ends and flanking regions (default: 0.3).

-S Maximum SAQ of flanking regions (default: 0.6).

-I Minimum SAQ of TE ends (default: 0.75).

-C Minimum proportion of identical basesto define a conserved sitein MSA (default: 0.8).

-l Maximum distance between two conserved sites (default: 4).

-L Minimum length of a highly conserved block (default: 10).

-f Minimum percent of conserved sites in highly conserved block(default: 0.6).

3. Parameters related to classification and homology search in Module 3:

-aThe file of SINE candidates for classification and homology search in FASTA format file (only used with -s 3).

-cMinimum sequence similarity of grouping SINEs into a family (default: 0.8).

-p Minimum proportion of overlap between SINEs within a family (default: 0.8).

-b Minimum sequence similarity to classify one verified SINE candidate intoa known SINE family (default: 0.8).

-t Minimum similarity between short RNA and RNAhead of SINE. The value is used to find the RNAtype of SINEs (default: 0.6).

-r Minimum overlapped proportion between short RNA and SINERNApart. The value is used to find the RNAtype of SINEs (default: 0.6).

Examples of running SINE_Scan

(1) To perform fully automatic annotation of SINEs in a genome assembly:

perl SINE_Scan_process.pl -s 123 -g Rice.fasta -o Oryza_sative_v7.0 -d Workdir

Note: Suffix of FASTA files should be 'fasta', 'fa' or 'fas'.

(2) To verified a set of sequences (stored in the file SINEs_Candiates.fasta)

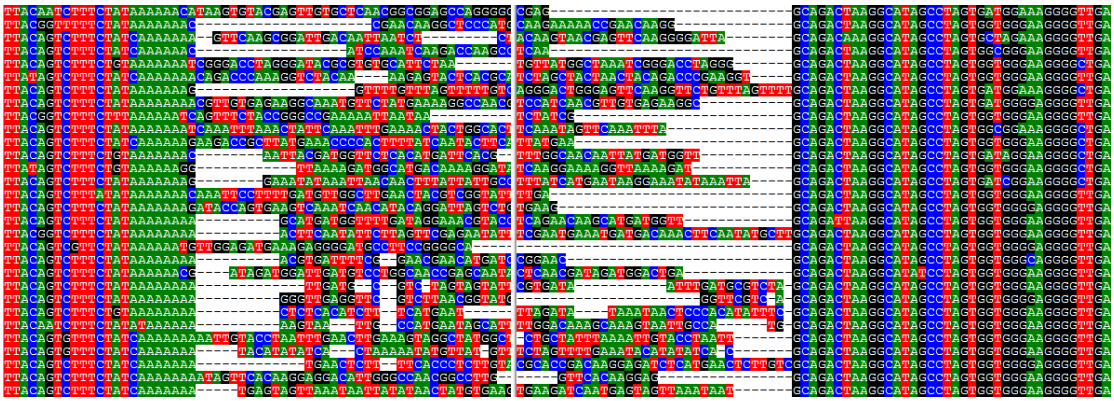
perl SINE_Scan_process.pl -s 23 -i SINEs_Candiates.fasta -g Zmays.fasta -o Zea_mays_v3.0 -d Workdir;

Note: TSDsof SINE candidates should be removed.

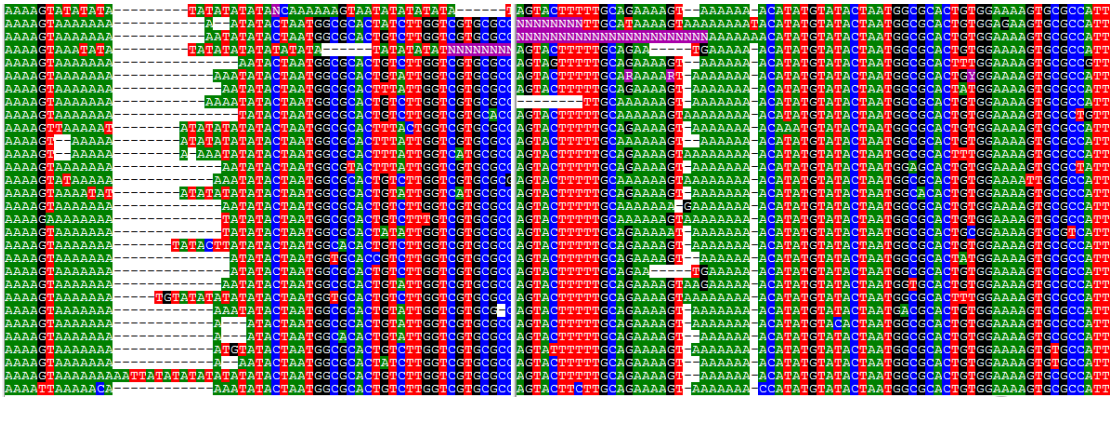
Guide of manual inspection

SINE_Scan shows high sensitivity and specificity in our test. However, a few artificial results have been found. These false results can be easily identified by manual inspection using MSA files generated byModule 2. Here we show an example of good SINE candidate and several typical false results. In the following figures, the left part represents alignments around 3' insertion site, and right represents 5' insertion site (separated by white stripes).All MSAs are viewedin BioEdit (http://www.mbio.ncsu.edu/bioedit/bioedit.html).

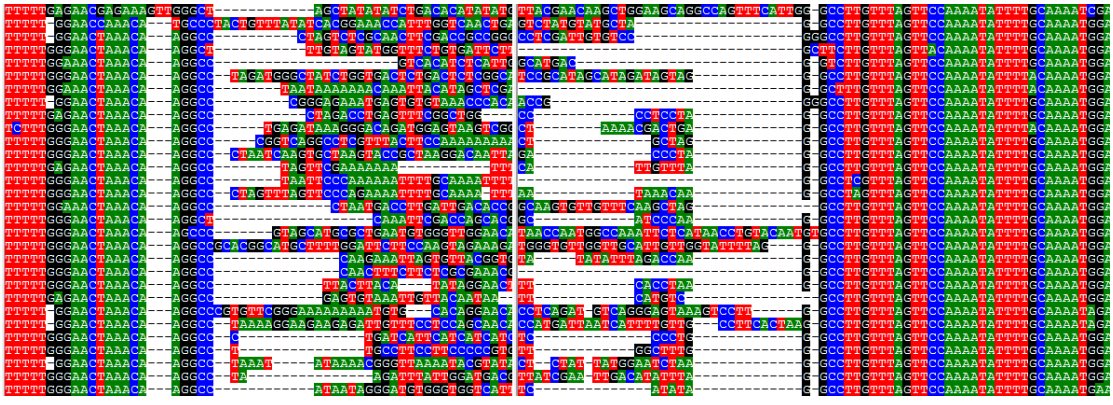
1. A good SINE candidate:



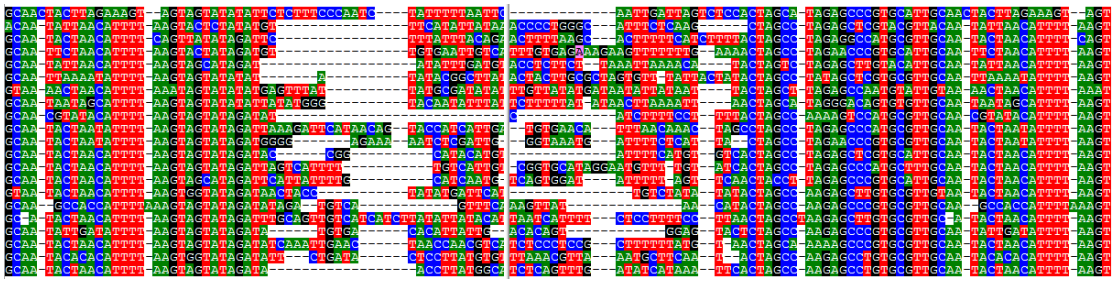
2. A false candidate (no alignment quality drop):



3. Perhaps a MITE:



4. Perhaps a *Helitron*:



5. Perhaps a solo-LTR: