SparseIso: A novel Bayesian approach to identify alternatively spliced isoforms from RNA-seq data

1. Introduction

The SparseIso package is developed and tested on Linux (Ubuntu) OS. The code is freely available under the MIT License (http://opensource.org/licenses/MIT). This package utilizes Boost and Eigen library for parallel computing and matrix calculation.

Contact: (Jason) Jianhua Xuan at xuan@vt.edu

Version: 1.0

Version Date: Mar. 5th 2017

Developed Platform: Ubuntu 12.04 (64 bit)

2. System Requirements

Requires: *nix OS, Boost library, Eigen library, Bamtools API

Tested with (recommended settings): Ubuntu 12.04 (64 bit), Boost library (1.48.0.2), Eigen library (3.2.4), Bamtools API (2.4.1)

3. Quick Start

We provide precompiled binaries for the user to speed up the setup process. The package (**SparseIso_V1.0.zip**) can be downloaded from http://github.com/henryxushi/SparseIso/releases/tag/V1.0 . After extracting the package, you can see the contents. There are two scripts 'run_SparseIso_paired_end.sh' and 'run_SparseIso_single_end.sh' for users to run SparseIso.

3.1 Run SparseIso with Single-end Data

The script "run_SparseIso_single_end.sh" is an example of running SparseIso on the demo single-end data (demo_single_end.bam available in the demo folder and http://sourceforge.net/projects/sparseiso/files/DemoData/demo_single_end.bam/download). You need to set the following parameters:

- BAMFILE: the path to the bam file aligned by Tophat.
- SPARSEISO_PATH: the path to SparseIso package.
- OUTPUT: the path to the output files (if the folder does not exist, a new folder will be created).
- NUM_OF_PROCESSES: the number of cores you want to use.

After setting the parameters, the script can be run as:

```
Path_To_SparseIso $ sh run_SparseIso_single_end.sh
```

More options of SparseIso are introduced in Section 7.

3.2 Run SparseIso with Paired-end Data

The script "run_SparseIso_paired_end.sh" is an example of running SparseIso on the demo paired-end data (demo_paired_end.bam available in the demo folder and http://sourceforge.net/projects/sparseiso/files/DemoData/demo_paired_end.bam/download). You need to set the following parameters:

- BAMFILE: the path to the bam file aligned by Tophat.
- SPARSEISO_PATH: the path to SparseIso package.
- OUTPUT: the path to the output files (if the folder does not exist, a new folder will be created).
- NUM_OF_PROCESSES: the number of cores you want to use.

After setting the parameters, the script can be run as:

```
Path_To_SparseIso $ sh run_SparseIso_paired_end.sh
```

More options of SparseIso are introduced in Section 7.

4. Build from Source

In this package, we provide both precompiled binaries and source code. Section 4 is for the users that want to compile from the source code. If you want to use precompiled binaries, please read section 3.

4.1 Dependent libraries and packages

Some packages including cmake, Boost library, Eigen library and Bamtools API are needed to install the software.

cmake (https://cmake.org/)

For Ubuntu system, the boost library can be easily installed by:

```
$ sudo apt-get install cmake
```

Boost library (http://www.boost.org/)

For Ubuntu system, the boost library can be easily installed by:

```
$ sudo apt-get install libboost-all-dev
```

More details can be found at http://www.boost.org/doc/libs/1_63_0/more/getting_started/index.html .

Eigen library (http://eigen.tuxfamily.org/)

A detailed installation guide is available at http://eigen.tuxfamily.org/dox/GettingStarted.html .

Bamtools API (http://github.com/pezmaster31/bamtools)

A detailed installation guide is available at http://github.com/pezmaster31/bamtools/wiki/Building-and-installing . You may need to run 'sudo apt-get install cmake' and 'sudo apt-get install libz-dev' to install two prerequisite packages for bamtools.

After installing all dependencies, please make sure the installed libraries are in the system library path. If not, please add the library paths to environment variable (i.e. LD_LIBRARY_PATH in Ubuntu).

Besides the libraries, SparseIso also uses the programs, 'processsam' and 'samtools' for preprocessing. These programs are deposited in the tools folder.

4.2 Compile

The software package is available at https://github.com/henryxushi/SparseIso . You need to specify the following paths of the libraries in the CMakeLists.txt (in the src folder):

- SPARSEISO_DIR: the path to the SparseIso directory.
- BOOST_LIB_DIR: the path to the lib directory of boost library.
- BOOST_INCLUDE_DIR: the path to the include directory of boost library.
- EIGEN_INCLUDE_DIR: the path to the main folder of Eigen library.
- BAMTOOLS_INCLUDE_DIR: path to the include folder of bamtools API.
- BAMTOOLS_LIB_DIR: path to the lib folder of bamtools API.

After setting up the paths, users can compile the program by the following command:

```
Path_To_SparseIso$ cmake ./src
```

If the executable binary file, SparseIso, is created in the SparseIso package folder (the parent folder of src), the installation is successful.

5. Run SparseIso with Compiled Binaries

Similar to Section 3, the script "run_SparseIso_single_end.sh" and "run_SparseIso_paired_end.sh" is an example of running SparseIso on the demo data (available in the demo folder and http://sourceforge.net/projects/sparseiso/files/DemoData/). You need to set the following parameters:

- PATH_BAMTOOLS_LIB: the path to the lib folder of bamtools API. (**Not included in Section 3**)
- BAMFILE: the path to the bam file aligned by Tophat.
- OUTPUT: the path to the output files (if the folder does not exist, a new folder will be created).
- NUM_OF_PROCESSES: the number of cores you want to use.
- SPARSEISO_PATH: the path to SparseIso package.

After setting the parameters, the script can be run as:

```
Path_To_SparseIso/scripts$ sh run_SparseIso_single_end.sh
```

```
Path_To_SparseIso/scripts$ sh run_SparseIso_paired_end.sh
```

6. Interpreting the Results

SparseIso will generate a file in GTF format (SparseIso.gtf) in the output folder. For each exon in the transcript, the following information is included:

- genomic location
- gene_id: Splice-graph id.
- transcript_id: Candidate id (unique across splice-graphs).
- FPKM: Mean abundance estimate normalized to effective transcript length and library size.

The GTF file can be loaded in visualization tools such as Integrative Genomics Viewer (IGV, http://software.broadinstitute.org/software/igv/). If you load the results of 'demo_single_end.bam' (Section 3.1) to the IGV, you can see the isoform structure by typing the gene id or transcript id to the genomic location box. For example, Figure1 shows the isoform structures from gene id CUFF.2 that matches to gene EIF2B3 on hg19 reference genome (Figure 1). It can be seen that SparseIso successfully identified all the isoforms of EIF2B3.

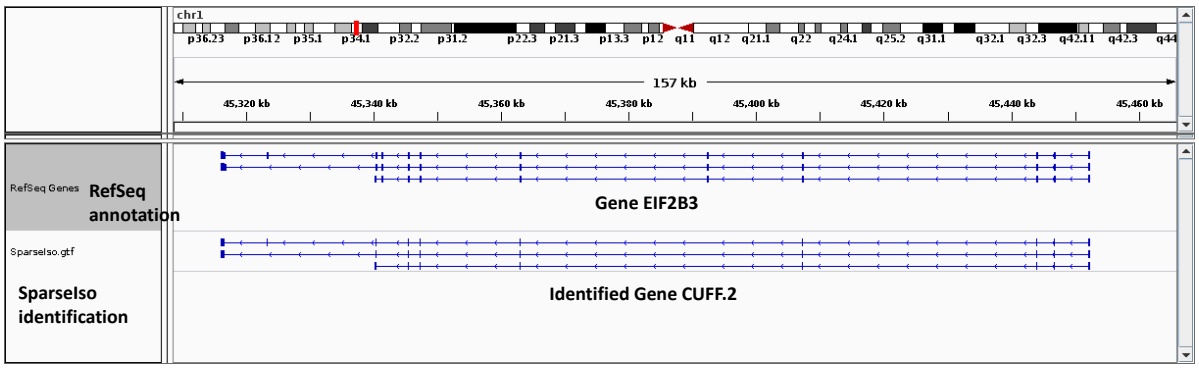

Figure 1. Visualization of identified isoforms from SparseIso

7. SparseIso Options

The full list options of SparseIso can be accessed by:

```
PATH_TO_SparseIso $ ./SparseIso
========================================================================
Usage: SparseIso [--bam/b] <filename>  [opts]
========================================================================
**Required :
--bam/-b <string>        : the name of bam file
--out-dir/-o <string>    : the directory stored all output files
--readtype/-r <p/s>      : the type of reads paired-end(p) or single-end(s)

** Prerequisite program (not needed to set if in system path) :
--cempath <string>     : the path to processsam.
--sampath <string>     : the path to samtools.

** Optional :
--threads/-p <int>       : the number of threads to be used, default: 1.
--outputall              : output all isoforms enumberated from the graph.
--conf/-c <double>       : the confidence level for isoform identification, default: 0.5.
--mean   <double>        : the mean of fragment length distribution, default: 200.
--stdvar  <double>       : the variance of fragment length distribution, default: 20.
--help/-h                : display the help information.

========================================================================
```

The required inputs are the bam file, output directory and the read type (single-end or paired-end). SparseIso uses two prerequisite programs ‘processsam’ and ‘samtools’, which are available in the tool folder. Using the shell scripts provided will automatically set this option. If the two programs are already in your system path, this option can be ignored.

For the Optional parameters, you can set the number of threads or processes. If the ‘outputall’ option is enabled, SparseIso will output all the candidate isoforms. Option ‘--conf’ can help select isoforms with a predefined confidence level. The mean and standard deviation of the fragment length distribution can also be fixed.