

Building Paean from source

Paean can only run on the Linux environment currently. Building Paean from source requires GNU compiler which supports C++ standard 11, Cmake \geq 3.8 and CUDA \geq 9.2. Additionally, before building Paean you need install some dependent libraries including `zlib`, `Thrust` and `HTSlib`. Note that please put `HTSlib` to `/usr` or `/usr/local` before compiling Paean. After that, you can run the following commands in the root directory to build Paean.

```
mkdir build
cd build
cmake ../
make -j
```

Running Paean

After building Paean we can get an executable binary file `paean` for transcriptome quantification.

Usage

```
./paean -b <gene annotation> -l <length table> -r <bam> -x <ASE type>,... -y <ASE  
↪ annotation>,... -o <output file> -t <thread number> -m <mode>
```

Arguments

<code>-b,--gene</code>	FILE	Gene annotation file with GFF3 format, required
<code>-l,--length</code>	FILE	Gene length file with csv format, required
<code>-x,--ase-types</code>	STRING	ASE types separated by commas, required
<code>-y,--ase-files</code>	FILE	ASE files separated by commas with csv format, required
<code>-o,--output</code>	FILE	Which directory you want to write the results to (default: the current directory)
<code>-t,--thread</code>	INT	Number of additional threads to use to parse BAM file (default: the number of cores on your computer)
<code>-p,--read-max-gap</code>	INT	Max allowed gap for two reads of a mate (default: 5000000)
<code>-q,--gene-max-gap</code>	INT	Max allowed gap for two genes for fusion detection (default: 500000)
<code>-m,--mode</code>	INT	Single-end or Pair-end mode, represented by 1 and 2 respectively
<code>-h,--help</code>		Print help information

Output format

Paean generates by default six files with corresponding three kinds of results: *TPM*, *PSI* and *Fusion*. It is worth noting that we use the name (denoted as `<BAM_prefix>`) of input read file as the prefix of the name of output files. In this case, `<BAM_prefix>.gene.tsv` outputs the *TPM* result, `<BAM_prefix>.ase.<type>.tsv` outputs *PSI* result in which `type` represents four types of alternative splicing events: SE, A3SS, A5SS and RI, as well as `<BAM_prefix>.fusion.tsv` outputs the *Fusion* result. What's more, Paean generates a plain text file `<BAM_prefix>.log.txt` to show some extra information including time consumption.

Example

We provide default files (gene annotation file and splicing event files) for the user to run Paean. With a bam file you can run Paean in the directory where generated binary file is located as follows:

```
./paean -b ../input/gencode.annotation.gff3 -l ../input/length_table.csv -r some.bam -x  
↪ SE,A3SS -y ../input/csv/SE.annotation.csv,../input/csv/A3SS.annotation.csv -o ./ -t 8  
↪ -m 2
```

Profiling GPU computing

You can profile GPU computing part of Paean by profiling tool `nvprof` provided by CUDA toolkit. Concretely, adding `/usr/local/cuda/bin/nvprof` in the front of command line of Paean:

```
/usr/local/cuda/bin/nvprof [options] ./paean -b <gene annotation> -l <length table> -r  
↪ <bam> -x <ASE type>,... -y <ASE annotation>,... -o <output file> -t <thread number> -m  
↪ <mode>
```

Without options you can profile the time occupancy of each CUDA kernel of Paeon. Additionally, you can know more information of GPU computing part by adding options after `nvprof`. For specific options, please refer to `nvprof` overview.

Multi-GPUs

Paeon shallowly supports Multi-GPUs. To use this feature, you can set environment variable `GPUS` to specify the devices you want to run on. For example, to use the first three GPUs in your computer you can set `GPUS=0,1,2`. We will better support Multi-GPUs in the future.