EasyFuse

Configuration file containing all user input required for running the fusion prediction pipeline within 6 sections: *“general”, “resources”, “commands”, “indices”, “otherFiles”, “easyfuse_helper”*.
Only “general” section might require modifications before a run (e.g. define email sender/receiver or tools to run). Other sections contain path to programs and required files as well as resource definitions and need to be adjusted only upon installation on a new system.

config.ini

File directory

reads{1,2}.fq.gz

Directory containing read files in fastq format.
Read files must end on "*.fastq.gz" or "*.fq.gz"; all other files in the directory are ignored. Only Paired-End data is currently supported and R1/R2 read files may not differ in their names (except for the "R1/R2"-identifier).

Provides methods for reading the config.ini and retrieving information from it

config

iomethods

Static methods collection for:
1) Creating folder
2) Grant specific permissions for files/directories
3) Get fastq files from specified input directory
4) Get icam data from searching a directory tree

processing

Main script to start the pipeline. Creates a folder tree for all data. Creates command line strings for all programs based on the selection of programs and defined resources in the config. Creates "samples.csv“ for progress monitoring. Checks fastq files from the input directory and processes them in pairs. Organizes and prepares slurm jobs.

Accessor to the monitoring file "samples.csv"

samples

Convenience methods for logging infos/debug statements and errors

logger

scheduling

Static methods collection for:
1) Creating an sbatch script and sending it to slurm
2) Command line execution w/o slurm
3) Command line execution w/o slurm with storing stdout
4) Retrieving a list of currently running job ids based on a lookup of a specified job name

Legend

Important I/O files

Main processing script

Helper script

Call external tool

SLURM lower levels depend on higher; everything on the same level (e.g. Kallisto and Seq2hla) and all unconnected (e.g. Mapsplice and Starfusion) jobs may run in parallel

fastqc

Gerneate fastqc report

skewer

Read trimming according to fastqc report (from the end until FRED-score cutoff 28)

Filtering

Star chimeric (1)

Fusionreadfilter (2)

Optional filtering of fastq files to remove fusion non-informative reads:
1) star alignment with chimeric output
2) Parse output.bam from chimeric alignment
3) Generate new filtered input.fastq file with samtools

bam_to_fastq (3)

filtered reads{1,2}.fq.gz

Kallisto

Pizzly

Star

starfusion

starchip

soapfuse

Mapsplice

jaffa

fusioncatcher

infusion

fetchdata

Fusiongrep (1)

liftOver (2)

Fusioninspector (3)

Contextseq (4)

Starindex (5)

Star (6a)

Star (6b)

Requantify (7a)

Requantify (7b)

Fetchdata starts the following four, subsequent steps (fusion inspector is optional so far and the output not parsed):
1) Grep and parse predicted fusion genes from different tools
   IN: specific output files of individual tools; OUT: Detected_Fusions.csv, Fusiongene_list.txt
2) Run liftOver to convert hg38 coordinates for e.g. hg19
   IN: Detected_Fusions.csv; OUT: Detected_Fusions_hg19.csv
3) Run fusion inspector on the list of candidate fusions
   IN: Fusionsgene_list.txt; OUT: Annotated_fusions_FuIn.tdt
4) Create and annotate the breakpoint surrounding fusion sequence and their respective wild type background
   IN: Detected_Fusions.csv; OUT: Context_Seqs.csv, Context_Seqs.csv.fasta, Context_Seqs.csv.fasta.info, Context_Seqs.csv_peptide.fasta, Context_Seqs.csv_transcript.fasta, Context_Seqs.csv.bed)
5) Create a pseudo genome from the context seqs
   IN: Context_Seqs.csv.fasta, Context_Seqs.csv.fasta.info; OUT: STAR_idx
6) Align the filtered (a) and original reads (b) against it
   IN: {filtered}reads{1,2}.fq.gz, Context_Seqs.csv.fasta; OUT: {fltr,org}_Aligned.sortedByCoord.out.bam
7) Compare and count junction and spanning pair mapping to fusion- and/or respective wildtype sequences
   IN: Aligned.sortedByCoord.out.bam; OUT: classification_{fltr,org}.tdt, classification_{fltr,org}.tdt.counts

summarize_data

join_data (1)

R_model_prediction (2)

Summarize_data combines all information and generates a pdf document containing some statistics.
(1) Combine information from various sources and apply R_model_prediction.
   IN: Detected_Fusions.csv, context_seq.csv, requantification.csv, samples.csv,
   OUT: {Samplename}_fusRank.csv
(2) Apply model to available data and attach two columns with labels and predictive values.
   IN: {Samplename}_fusRank.csv, R_model.rds
   OUT: {Samplename}_fusRank.csv

Final output

The final output contains all predicted events, including annotation information, the peptide sequence and a prediction value. The output table is generate per replicate. Replicates, when available, have to be combined subsequently.

TRON