Package 'PGA'

April 15, 2017

Type Package

Title An package for identification of novel peptides by customized database derived from RNA-Seq

Description This package provides functions for construction of customized protein databases based on RNA-Seq data with/without genome guided, database searching, post-processing and report generation. This kind of customized protein database includes both the reference database (such as Refseq or ENSEMBL) and the novel peptide sequences form RNA-Seq data.

Version 1.4.0

Date 2016-07-26

Author Shaohang Xu, Bo Wen

Maintainer Bo Wen <wenbo@genomics.cn>, Shaohang Xu <xsh.skye@gmail.com>

Depends R (>= 3.0.1), IRanges, GenomicRanges, Biostrings (>= 2.26.3), data.table, rTANDEM

Imports S4Vectors (>= 0.9.25), Rsamtools (>= 1.10.2), GenomicFeatures (>= 1.19.8), biomaRt (>= 2.17.1), stringr, RCurl, Nozzle.R1, VariantAnnotation (>= 1.7.28), rtracklayer, RSQLite, ggplot2, AnnotationDbi, customProDB (>= 1.7.0), pheatmap

Suggests BSgenome.Hsapiens.UCSC.hg19, RUnit, BiocGenerics, BiocStyle, knitr, R.utils

VignetteBuilder knitr

License GPL-2

biocViews Proteomics, MassSpectrometry, Software, ReportWriting, RNASeq, Sequencing

RoxygenNote 5.0.1

NeedsCompilation no

R topics documented:

`addGeneName4Ensembl` *Add gene nemes*

Description

Add "gene name" column to the attachments (eg. snv.tsv and idl.tsv) of report if there is a demand (optional function). This function is useful for "Ensembl" IDs.

Usage

```
addGeneName4Ensembl(mart, report = "report")
```

Arguments

`mart`	See detail in function `PrepareAnnotationEnsembl`.
`report`	The report directory for adding gene names.

Value

none

Examples

```
## Not run:
library(PGA)
# set the biomaRt parameters just as "PrepareAnnotationEnsembl2" did

mart <- biomaRt::useMart("ENSEMBL_MART_ENSEMBL",
        dataset="hsapiens_gene_ensembl",
        host="grch37.ensembl.org",
        path="/biomart/martservice",
        archive=FALSE)
addGeneName4Ensembl(mart=mart,report="report")


## End(Not run)
```

createProDB4DenovoRNASeq *Create protein database based on the transcripts from de novo reconstruction of transcriptome from RNA-seq data*

Description

Create protein database based on the transcripts from de novo reconstruction of transcriptome from RNA-seq data.

Usage

```
createProDB4DenovoRNASeq(infa = "./trinity.fasta", bool_use_3frame = FALSE,
  outmtab = "novel_transcripts_ntx.tab",
  outfa = "./novel_transcripts_ntx.fasta", bool_get_longest = TRUE,
  make_decoy = TRUE, decoy_tag = "#REV#", outfile_name = "test")
```

Arguments

infa	A FASTA format file containing transcript sequences (such as derived from de novo transcriptome assembly by Trinity)
bool_use_3frame	A logical variable indicating whether to translate the raw sequences with 3-frame (forward). Default is 6-frame translation (FALSE).
outmtab	A txt format file containing the novel transcripts information
outfa	The output fasta format protein sequence file
bool_get_longest	When it's set as TRUE, only the longest protein sequences will be retained after the raw sequences are translated with 3-frame or 6-frame. Otherwise, all the protein sequences with longer than 30 aa will be retained.
make_decoy	A logical variable indicating whether to add the decoy sequences.
decoy_tag	The prefix of decoy sequence IDs.
outfile_name	Output file name

Value

The database file(s)

Examples

```
transcript_seq_file <- system.file("extdata/input", "Trinity.fasta",package="PGA")
createProDB4DenovoRNASeq(infa=transcript_seq_file)
```

dbCreator *Create customized protein database from RNA-Seq data*

Description

The main function to create customized protein database from RNA-Seq data

Usage

```
dbCreator(gtfFile = NULL, vcfFile = NULL, bedFile = NULL,
  annotation_path = NULL, outdir, outfile_name, lablersid = FALSE,
  COSMIC = FALSE, bool_get_longest = TRUE, organism = "Homo sapiens",
  make_decoy = TRUE, genome = NULL, ...)
```

Arguments

gtfFile	A GTF format file containing novel transcripts information
vcfFile	A VCF format file containing SNV and INDEL information
bedFile	A BED format file containing juction information
annotation_path	This directory contains numerous pieces of genome annotation information which can be downloaded by PrepareAnnotationEnsembl2 or PrepareAnnotationRefseq2.
outdir	Output directory
outfile_name	Output file name
lablersid	A logical indicating whether to do the SNV annotation(dbSNP)
COSMIC	A logical indicating whether to do the SNV annotation(COSMIC)
bool_get_longest	When it's set as TRUE, the longest sequences will be retained after the DNA sequences are six-frame translated into protein sequences. Otherwise, the protein sequences more than 30 aa are retained.
organism	What is the Genus and species of this organism.Please use proper scientific nomenclature for example: "Homo sapiens" and not "human", default is "Homo sapiens".
make_decoy	A logical indicating whether to add the decoy sequences
genome	Genome information. This is a BSgenome object(e.g. Hsapiens). Default is NULL.
...	Additional arguments

Value

The database file

Examples

```
vcffile <- system.file("extdata/input", "PGA.vcf",package="PGA")
bedfile <- system.file("extdata/input", "junctions.bed",package="PGA")
gtffile <- system.file("extdata/input", "transcripts.gtf",package="PGA")
annotation <- system.file("extdata", "annotation",package="PGA")
outfile_path<-"db/"
outfile_name<-"test"
library(BSgenome.Hsapiens.UCSC.hg19)
dbCreator(gtfFile=gtffile,vcfFile=vcffile,bedFile=bedfile,
          annotation_path=annotation,outfile_name=outfile_name,
          genome=Hsapiens,outdir=outfile_path)
```

---

easyRun *easyRun*

Description

This function is used to automate the peptide identification based on searching the customized database derived from RNA-Seq data.

Usage

```
easyRun(gtfFile = NULL, vcfFile = NULL, bedFile = NULL, spectra = NULL,
  annotation_path = NULL, outdir = "pga_dir", outPrefix = "pga",
  lablersid = FALSE, COSMIC = FALSE, bool_get_longest = TRUE,
  organism = "Homo sapiens", genome = NULL, enzyme = "[KR]|[X]",
  tol = 10, tolu = "ppm", itol = 0.6, itolu = "Daltons",
  varmod = NULL, fixmod = NULL, miss = 2, maxCharge = 8, ti = FALSE,
  cpu = 0, alignment = 1, xmx = 2, ...)
```

Arguments

gtfFile	A GTF format file containing novel transcripts information
vcfFile	A VCF format file containing SNV and INDEL information
bedFile	A BED format file containing juction information
spectra	MS/MS peak list file
annotation_path	This directory contains numerous pieces of genome annotation information which can be downloaded by PrepareAnnotationEnsembl2 or PrepareAnnotationRefseq2.
outdir	Output directory.
outPrefix	The prefix of output file.
lablersid	A logical indicating whether to do the SNV annotation(dbSNP)
COSMIC	A logical indicating whether to do the SNV annotation(COSMIC)
bool_get_longest	When it's set as TRUE, the longest sequences will be retained after the DNA sequences are six-frame translated into protein sequences. Otherwise, the protein sequences more than 30 aa are retained.

organism	What is the Genus and species of this organism.Please use proper scientific nomenclature for example: "Homo sapiens" and not "human", default is "Homo sapiens".
genome	Genome information. This is a BSgenome object(e.g. Hsapiens).
enzyme	Specification of specific protein cleavage sites. Default is "[KR]|[X]".
tol	Parent ion mass tolerance (monoisotopic mass).
tolu	Parent ion M+H mass tolerance window units.
itol	Fragment ion mass tolerance (monoisotopic mass).
itolu	Unit for fragment ion mass tolerance (monoisotopic mass).
varmod	Specificiation of potential modifications of residues.
fixmod	Specification of modifications of residues.
miss	The number of missed cleavage sites. Default is 2.
maxCharge	The Maximum parent charge, default is 8
ti	anticipate carbon isotope parent ion assignment errors. Default is false.
cpu	The number of CPU used for X!Tandem search. Default is 1.
alignment	0 or 1 to determine if peptide should be alignment or not. Default is 0.
xmx	The maximum Java heap size. The unit is "G".
...	Additional arguments

Value

none

Examples

```
vcffile <- system.file("extdata/input", "PGA.vcf",package="PGA")
bedfile <- system.file("extdata/input", "junctions.bed",package="PGA")
gtffile <- system.file("extdata/input", "transcripts.gtf",package="PGA")
annotation <- system.file("extdata", "annotation",package="PGA")
library(BSgenome.Hsapiens.UCSC.hg19)
msfile <- system.file("extdata/input", "pga.mgf",package="PGA")
easyRun(gtfFile=gtffile,vcfFile=vcffile,bedFile=bedfile,spectra=msfile,
        annotation_path=annotation,genome=Hsapiens,cpu = 6,
        enzyme = "[KR]|[X]", varmod = "15.994915@M",itol = 0.05,
        fixmod = "57.021464@C", tol = 10, tolu = "ppm", itolu = "Daltons",
        miss = 2, maxCharge = 8, ti = FALSE,xmx=1)
```

---

parserGear *Post-processing for the identification result*

Description

This function is mainly for q-value calculation, protein inference and novel peptides spectra annotation.

Usage

```
parserGear(file = NULL, db = NULL, outdir = "parser_outdir",
  prefix = "pga", novelPrefix = "VAR", decoyPrefix = "###REV###",
  alignment = 1, xmx = NULL, thread = 1, msfile = NULL)
```

Arguments

file	MS/MS search file. Currently, only XML format file of X!Tandem, DAT result of Mascot and mzIdentML result file (MS-GF+, MyriMatch, IPeak, OMSSA, ...) are supported.
db	A FASTA format database file used for MS/MS searching. Usually, it is from the output of the function dbCreator.
outdir	Output directory.
prefix	The prefix of output file.
novelPrefix	The prefix of novel protein ID. Default is "VAR". "VAR" is the prefix which used by function dbCreator. This value should be left to the default when your database is constructed by the function getTrinityDB.
decoyPrefix	The prefix of decoy sequences ID. Default is "###REV###". "###REV###" is the prefix which used by function dbCreator.
alignment	0 or 1 to determine if peptide should be alignment or not. Default is 1.
xmx	The maximum Java heap size. The unit is "G".
thread	This parameter is used to specify the number of threads. "0" represents that all of the available threads are used; "1" represents one thread is used; "2" represents two threads are used,and so on. Default is 1.
msfile	The MS/MS data (mgf format) used.

Value

none

Examples

```
vcffile <- system.file("extdata/input", "PGA.vcf",package="PGA")
bedfile <- system.file("extdata/input", "junctions.bed",package="PGA")
gtffile <- system.file("extdata/input", "transcripts.gtf",package="PGA")
annotation <- system.file("extdata", "annotation",package="PGA")
outfile_path<-"db/"
outfile_name<-"test"
library(BSgenome.Hsapiens.UCSC.hg19)
dbfile <- dbCreator(gtfFile=gtffile,vcfFile=vcffile,bedFile=bedfile,
                    annotation_path=annotation,outfile_name=outfile_name,
                    genome=Hsapiens,outdir=outfile_path)

msfile <- system.file("extdata/input", "pga.mgf",package="PGA")

## X!Tandem as the peptide identification software
idfile <- runTandem(spectra = msfile, fasta = dbfile, outdir = "./", cpu = 6,
                    enzyme = "[KR]|[X]", varmod = "15.994915@M",itol = 0.05,
                    fixmod = "57.021464@C", tol = 10, tolu = "ppm",
                    itolu = "Daltons", miss = 2, maxCharge = 8, ti = FALSE)
parserGear(file = idfile, db = dbfile, decoyPrefix="#REV#",xmx=1,thread=8,
```

```
            outdir = "parser_outdir")

## Mascot as the peptide identification software
dat_file <- system.file("extdata/input", "mascot.dat",package="PGA")
parserGear(file = dat_file, db = dbfile, decoyPrefix="#REV#",xmx=1,thread=8,
            outdir = "parser_outdir_mascot")
```

PrepareAnnotationEnsembl2 *Prepare annotation from ENSEMBL*

Description

Prepare the annotation from ENSEMBL through biomaRt. This function is modified from the function PrepareAnnotationEnsembl in **customProDB**.

Usage

```
PrepareAnnotationEnsembl2(mart, annotation_path, splice_matrix = FALSE,
  dbsnp = NULL, transcript_ids = NULL, COSMIC = FALSE, ...)
```

Arguments

mart	See detail in function PrepareAnnotationEnsembl.
annotation_path	See detail in function PrepareAnnotationEnsembl.
splice_matrix	See detail in function PrepareAnnotationEnsembl.
dbsnp	See detail in function PrepareAnnotationEnsembl.
transcript_ids	See detail in function PrepareAnnotationEnsembl.
COSMIC	See detail in function PrepareAnnotationEnsembl.
...	Additional arguments

Value

Several .RData file containing annotations needed for following analysis.

See Also

PrepareAnnotationRefseq2.

Examples

```
ensembl <- biomaRt::useMart("ENSEMBL_MART_ENSEMBL", dataset="hsapiens_gene_ensembl",
                  host="grch37.ensembl.org", path="/biomart/martservice",
                  archive=FALSE)

annotation_path <- tempdir()
transcript_ids <- c("ENST00000234420", "ENST00000269305", "ENST00000445888")

PrepareAnnotationEnsembl2(mart=ensembl, annotation_path=annotation_path,
                        splice_matrix=FALSE, dbsnp=NULL, transcript_ids=transcript_ids,
                         COSMIC=FALSE)
```

PrepareAnnotationRefseq2 *Prepare annotation from Refseq*

Description

Prepare the annotation for Refseq through UCSC table browser. This function is modified from the function PrepareAnnotationRefseq in **customProDB**.

Usage

```
PrepareAnnotationRefseq2(genome = "hg19", CDSfasta, pepfasta, annotation_path,
  dbsnp = NULL, transcript_ids = NULL, splice_matrix = FALSE,
  COSMIC = FALSE, ...)
```

Arguments

genome	See detail in function PrepareAnnotationRefseq.
CDSfasta	See detail in function PrepareAnnotationRefseq.
pepfasta	See detail in function PrepareAnnotationRefseq.
annotation_path	See detail in function PrepareAnnotationRefseq.
dbsnp	See detail in function PrepareAnnotationRefseq.
transcript_ids	See detail in function PrepareAnnotationRefseq.
splice_matrix	See detail in function PrepareAnnotationRefseq.
COSMIC	See detail in function PrepareAnnotationRefseq.
...	Additional arguments

Value

Several .RData file containing annotations needed for following analysis.

See Also

PrepareAnnotationEnsembl2.

Examples

```
transcript_ids <- c("NM_001126112", "NM_033360", "NR_073499")
pepfasta <- system.file("extdata", "refseq_pro_seq.fasta",
                        package="customProDB")
CDSfasta <- system.file("extdata", "refseq_coding_seq.fasta",
                        package="customProDB")
annotation_path <- tempdir()
PrepareAnnotationRefseq2(genome='hg19', CDSfasta, pepfasta, annotation_path,
                        dbsnp=NULL, transcript_ids=transcript_ids,
                        splice_matrix=FALSE, COSMIC=FALSE)
```

reportGear *The main function for report generation*

Description

The main function for report generation

Usage

```
reportGear(parser_dir, tab_dir, report_dir)
```

Arguments

parser_dir	The directory which contains the peptide identification results
tab_dir	The directory which contains the database annotation files
report_dir	The report output directory

Value

none

Examples

```
vcffile <- system.file("extdata/input", "PGA.vcf",package="PGA")
bedfile <- system.file("extdata/input", "junctions.bed",package="PGA")
gtffile <- system.file("extdata/input", "transcripts.gtf",package="PGA")
annotation <- system.file("extdata", "annotation",package="PGA")
outfile_path<-"db/"
outfile_name<-"test"
library(BSgenome.Hsapiens.UCSC.hg19)
dbfile <- dbCreator(gtfFile=gtffile,vcfFile=vcffile,bedFile=bedfile,
                    annotation_path=annotation,outfile_name=outfile_name,
                    genome=Hsapiens,outdir=outfile_path)

msfile <- system.file("extdata/input", "pga.mgf",package="PGA")
idfile <- runTandem(spectra = msfile, fasta = dbfile, outdir = "./", cpu = 6,
                    enzyme = "[KR]|[X]", varmod = "15.994915@M",itol = 0.05,
                    fixmod = "57.021464@C", tol = 10, tolu = "ppm",
                    itolu = "Daltons", miss = 2, maxCharge = 8, ti = FALSE)
parserGear(file = idfile, db = dbfile, decoyPrefix="#REV#",xmx=1,thread=8,
           outdir = "parser_outdir")
reportGear(parser_dir = "parser_outdir", tab_dir = outfile_path,
           report_dir = "report")
```

runTandem *run X!Tandem*

Description

run X!Tandem

Usage

```
runTandem(spectra = "", fasta = "", outdir = ".", cpu = 1,
  enzyme = "[KR]|[X]", tol = 10, tolu = "ppm", itol = 0.6,
  itolu = "Daltons", varmod = NULL, fixmod = NULL, miss = 2,
  maxCharge = 8, ti = FALSE)
```

Arguments

spectra	MS/MS peak list file
fasta	Protein database file for searching.
outdir	The output directory.
cpu	The number of CPU used for X!Tandem search. Default is 1.
enzyme	Specification of specific protein cleavage sites. Default is "[KR]|[X]".
tol	Parent ion mass tolerance (monoisotopic mass).
tolu	Parent ion M+H mass tolerance window units.
itol	Fragment ion mass tolerance (monoisotopic mass).
itolu	Unit for fragment ion mass tolerance (monoisotopic mass).
varmod	Specificiation of potential modifications of residues.
fixmod	Specification of modifications of residues.
miss	The number of missed cleavage sites. Default is 2.
maxCharge	The Maximum parent charge, default is 8
ti	anticipate carbon isotope parent ion assignment errors. Default is false.

Value

The search result file path

Examples

```
vcffile <- system.file("extdata/input", "PGA.vcf",package="PGA")
bedfile <- system.file("extdata/input", "junctions.bed",package="PGA")
gtffile <- system.file("extdata/input", "transcripts.gtf",package="PGA")
annotation <- system.file("extdata", "annotation",package="PGA")
outfile_path<-"db/"
outfile_name<-"test"
library(BSgenome.Hsapiens.UCSC.hg19)
dbfile <- dbCreator(gtfFile=gtffile,vcfFile=vcffile,bedFile=bedfile,
                    annotation_path=annotation,outfile_name=outfile_name,
                    genome=Hsapiens,outdir=outfile_path)
```

```
msfile <- system.file("extdata/input", "pga.mgf",package="PGA")
runTandem(spectra = msfile, fasta = dbfile, outdir = "./", cpu = 6,
          enzyme = "[KR]|[X]", varmod = "15.994915@M",
          fixmod = "57.021464@C", tol = 10, tolu = "ppm", itol = 0.05,
          itolu = "Daltons", miss = 2, maxCharge = 8, ti = FALSE)
```

Index