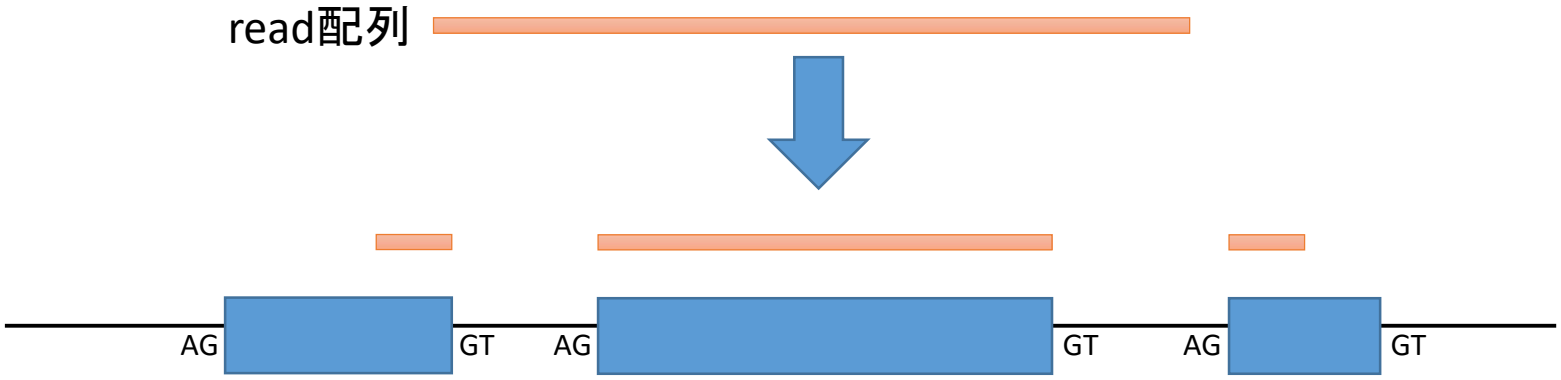
基礎生物学研究所ゲノムインフォマティックス・トレーニングコース　2020
RNA-seq入門：RNA-seq解析パイプライン
2021.03.10-2021.03.11

RNA-Seqパイプライン
ゲノムベースの解析法

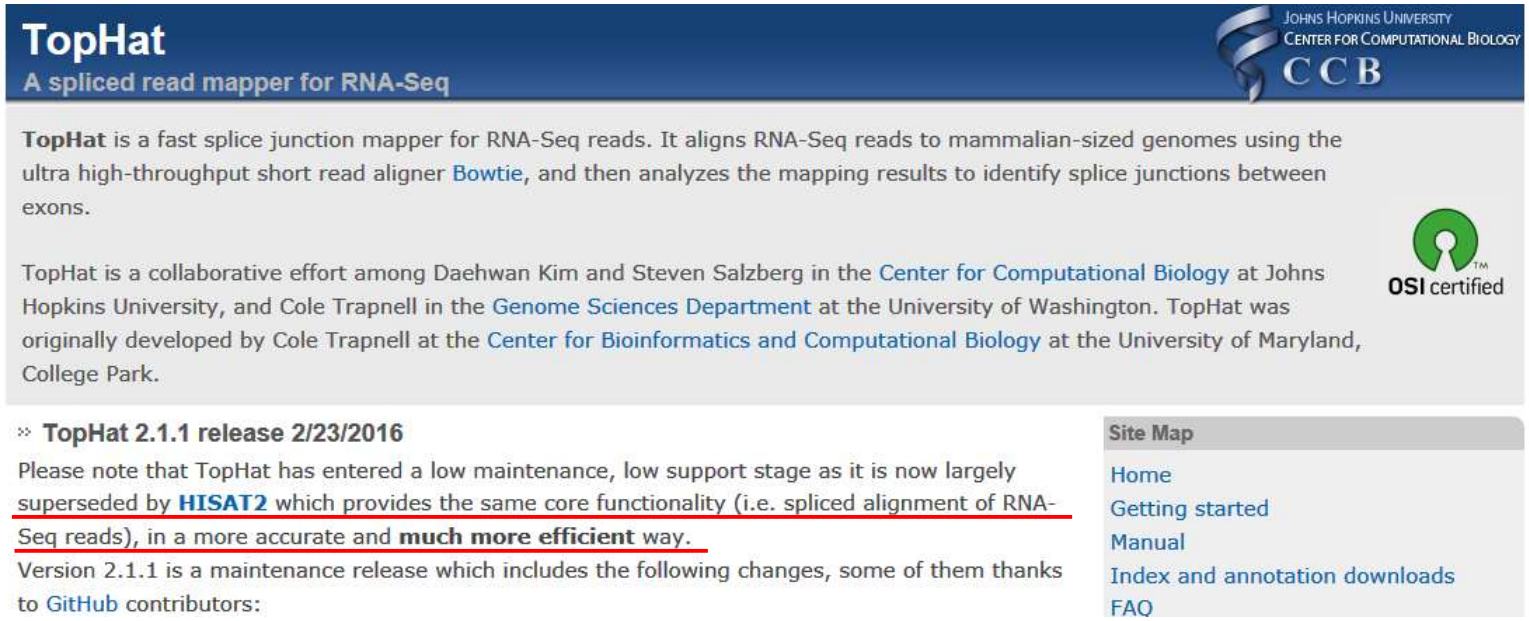
基礎生物学研究所
生物機能解析センター
山口勝司

genomeをレファレンスとする場合

レファレンスがゲノム配列の場合、
イントロン配列のスプライシングを考慮した
アライメントを行う必要がある。
今回はHISATを用いる
他　Tophat, Blat, SpliceMap, MapSplice, GSMAP, QPALMA



実際こんな感じにアラインされる



Traditional 'Tuxedo' package

New 'Tuxedo' package

TopHat2

↓

Cufflinks

HISAT

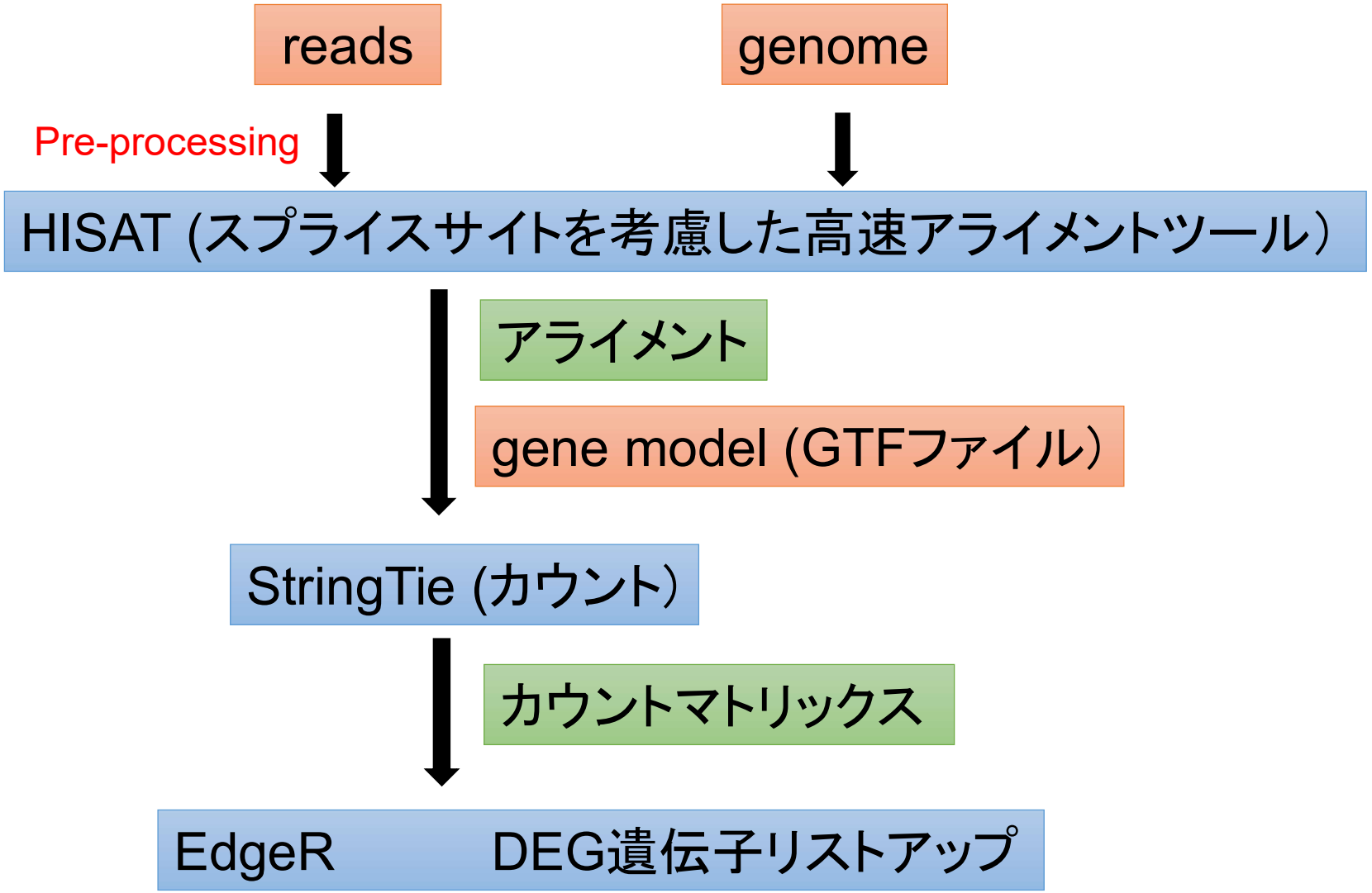
↓

StringTie

Ballgown

劇的に解析速度が速くなった

本トレーニングコースでの流れ



HISAT

HISAT2 is a fast and sensitive alignment program for mapping next-generation sequencing reads (both DNA and RNA) to a population of human genomes as well as to a single reference genome. Based on an extension of BWT for graphs (Sirén et al. 2014), we designed and implemented a graph FM index (GFM), an original approach and its first implementation. In addition to using one global GFM index that represents a population of human genomes, **HISAT2** uses a large set of small GFM indexes that collectively cover the whole genome. These small indexes (called local indexes), combined with several alignment strategies, enable rapid and accurate alignment of sequencing reads. This new indexing scheme is called a Hierarchical Graph FM index (HGFM).

TopHat2と比較して速い

HISAT-3N beta release 12/14/2020

HISAT-3N is a software system for analyzing nucleotide conversion sequencing reads. See the HISAT-3N for more details.

Index files are moved to the AWS Public Dataset Program. 9/3/2020

We have moved HISAT2 index files to the AWS Public Dataset Program. See the link for more details.

HISAT 2.2.1 release 7/24/2020

This patch version includes the following changes.

- Python3 support
- Remove the HISAT-genotype related scripts. HISAT-genotype moved to http://daehwankimlab.github.io/hisat-genotype/
- Fixed bugs related to `--read-lengths` option

Search

Funding

This work was supported in part by the National Human Genome Research Institute under grants R01-HG006102 and R01-HG006677, and NIH grants R01-LM06845 and R01-GM083873 and NSF grant CCF-0347992 to Steven L. Salzberg and by the Cancer Prevention Research Institute of Texas under grant RR170068 and NIH grant R01-GM135341 to Daehwan Kim

Manual

Introduction

パラメータの意味など
詳しく知るためには、
必ずManualを見る

What is HISAT2?

HISAT2 is a fast and sensitive alignment program for mapping next-generation sequencing reads (whole-genome, transcriptome, and exome sequencing data) against the general human population (as well as against a single reference genome). Based on GCSA (an extension of BWT for a graph), we designed and implemented a graph FM index (GFM), an original approach and its first implementation to the best of our knowledge. In addition to using one global GFM index that represents general population, HISAT2 uses a large set of small GFM indexes that collectively cover the whole genome (each index representing a genomic region of 56 Kbp, with 55,000 indexes needed to cover human population). These small indexes (called local indexes) combined with several alignment strategies enable effective alignment of sequencing reads. This new indexing scheme is called Hierarchical Graph FM index (HGFM). We have developed HISAT 2 based on the HISAT and Bowtie2 implementations. HISAT2 outputs alignments in SAM format, enabling interoperation with a large number of other tools (e.g. SAMtools, GATK) that use SAM. HISAT2 is distributed under the GPLv3 license, and it runs on the command line under Linux, Mac OS X and Windows.

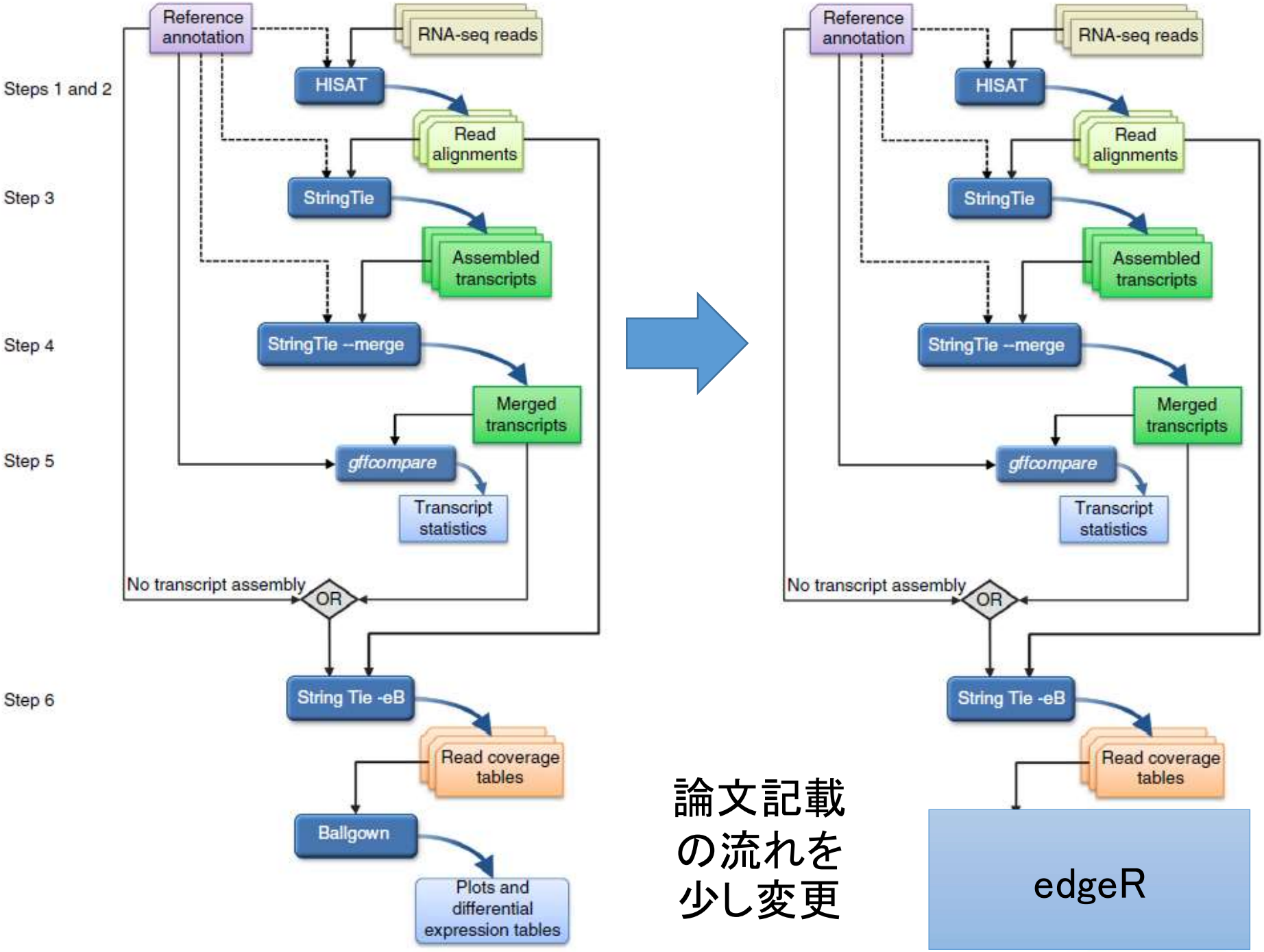
PROTOCOL

Transcript-level expression analysis of RNA-seq experiments with HISAT, StringTie and Ballgown

Mihaela Pertea[1,2], Daehwan Kim[1], Geo M Pertea[1], Jeffrey T Leek[3] & Steven L Salzberg[1–4]

[1]Center for Computational Biology, McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins School of Medicine, Baltimore, Maryland, USA. [2]Department of Computer Science, Whiting School of Engineering, Johns Hopkins University, Baltimore, Maryland, USA. [3]Department of Biostatistics, Bloomberg School of Public Health, Johns Hopkins University, Baltimore, Maryland, USA. [4]Department of Biomedical Engineering, Johns Hopkins University, Baltimore, Maryland, USA. Correspondence should be addressed to S.L.S. (salzberg@jhu.edu).

Published online 11 August 2016; doi:10.1038/nprot.2016.095


Steps 1 and 2
Step 3
Step 4
Step 5
Step 6
Reference annotation
RNA-seq reads
HISAT
Read alignments
StringTie
Assembled transcripts
StringTie --merge
Merged transcripts
gffcompare
Transcript statistics
No transcript assembly
OR
String Tie -eB
Read coverage tables
Ballgown
Plots and differential expression tables
論文記載の流れを少し変更
edgeR

hisat-buildでリファレンスのインデックスを作る

```
$ hisat2-build -h
HISAT2 version 2.1.0 by Daehwan Kim (infphilo@gmail.com, http://www.ccb.jhu.edu/people/infphilo)
Usage: hisat2-build [options]* <reference_in> <ht2_index_base>
    reference_in            comma-separated list of files with ref sequences
    hisat2_index_base       write ht2 data to files with this dir/basename
Options:
    -c                      reference sequences given on cmd line (as
                            <reference_in>)
    --large-index           force generated index to be 'large', even if ref
                            has fewer than 4 billion nucleotides
    -a/--noauto             disable automatic -p/--bmax/--dcv memory-fitting
    -p                      number of threads
    --bmax <int>            max bucket sz for blockwise suffix-array builder
    --bmaxdivn <int>        max bucket sz as divisor of ref len (default: 4)
    --dcv <int>             diff-cover period for blockwise (default: 1024)
    --nodc                  disable diff-cover (algorithm becomes quadratic)
    -r/--noref              don't build .3/.4.ht2 (packed reference) portion
    -3/--justref            just build .3/.4.ht2 (packed reference) portion
    -o/--offrate <int>      SA is sampled every 2^offRate BWT chars (default: 5)
    -t/--ftabchars <int>    # of chars consumed in initial lookup (default: 10)
    --localoffrate <int>    SA (local) is sampled every 2^offRate BWT chars (default: 3)
    --localftabchars <int>  # of chars consumed in initial lookup in a local index (default: 6)
    --snp <path>            SNP file name
    --haplotype <path>      haplotype file name
    --ss <path>             Splice site file name
    --exon <path>           Exon file name
    --seed <int>            seed for random number generator
    -q/--quiet              verbose output (for debugging)
    -h/--help               print detailed description of tool and its options
    --usage                 print this usage message
    --version               print version information and quit
```

一部のモデル生物種以外は、リファレンス配列のインデックスを作る必要がある

実習1　hisat2-build

genome.faはArabidopsis thaliana (シロイヌナズナ）のレファレンスゲノム配列である。

中身を閲覧、query名およびreads数を確認せよ。

```
$ less genome.fa
$ grep '>' genome.fa
$ grep '>' genome.fa|wc
```

indexを作製せよ。

```
$ hisat2-build genome.fa genome
```

新たに作製されたファイルを確認せよ。

```
$ ls
```

HISAT基本コマンド

```
$ hisat2 -h
HISAT2 version 2.1.0 by Daehwan Kim (infphilo@gmail.com,
www.ccb.jhu.edu/people/infphilo)
Usage:
  hisat2 [options]* -x <ht2-idx> {-1 <m1> -2 <m2> | -U <r> | --sra-acc <SRA accession
number>} [-S <sam>]

  <ht2-idx>  Index filename prefix (minus trailing .X.ht2).
  <m1>       Files with #1 mates, paired with files in <m2>.
             Could be gzip'ed (extension: .gz) or bzip2'ed (extension: .bz2).
  <m2>       Files with #2 mates, paired with files in <m1>.
             Could be gzip'ed (extension: .gz) or bzip2'ed (extension: .bz2).
  <r>        Files with unpaired reads.
             Could be gzip'ed (extension: .gz) or bzip2'ed (extension: .bz2).
  <SRA accession number>        Comma-separated list of SRA accession numbers, e.g. --
sra-acc SRR353653,SRR353654.
  <sam>      File for SAM output (default: stdout)

  <m1>, <m2>, <r> can be comma-separated lists (no whitespace) and can be
  specified many times.  E.g. '-U file1.fq,file2.fq -U file3.fq'.
```

結果はsamファイルで出力される

実習2　hisat2

read結果
2D2L_rep1_R1.fastq
2D2L_rep1_R2.fastq
を先にindexを作製したリファレンスにmapさせよ。

```
$ hisat2 -p 4 --dta ¥
-x genome ¥
-1 2D2L_rep1_R1.fastq ¥
-2 2D2L_rep1_R2.fastq ¥
-S 2D2L_rep1.sam
```

samファイルの内容を確認しよう

```
$ less 2D2L_rep1.sam
```

StringTie

Transcript assembly and quantification for RNA-Seq

JOHNS HOPKINS UNIVERSITY
CENTER FOR COMPUTATIONAL BIOLOGY
CCB

Overview

StringTie is a fast and highly efficient assembler of RNA-Seq alignments into potential transcripts. It uses a novel network flow algorithm as well as an optional *de novo* assembly step to assemble and quantitate full-length transcripts representing multiple splice variants for each gene locus. Its input can include not only alignments of short reads that can also be used by other transcript assemblers, but also alignments of longer sequences that have been assembled from those reads. In order to identify differentially expressed genes between experiments, StringTie's output can be processed by specialized software like Ballgown, Cuffdiff or other programs (DESeq2, edgeR, etc.).

News

▸ **5/12/2020 - v2.1.3 release** new features and fixes

- added the `--viral` option for long reads from viral data where splice sites do not follow consensus
- adjustments to the assembly of long read alignment data
- fixed an occasional issue with the `--merge` option
- made the `-e` compatible with long reads (`-L` option)

ダウンロードリンクはv2.1.4

StringTieを用いてアラインされたreadを数える

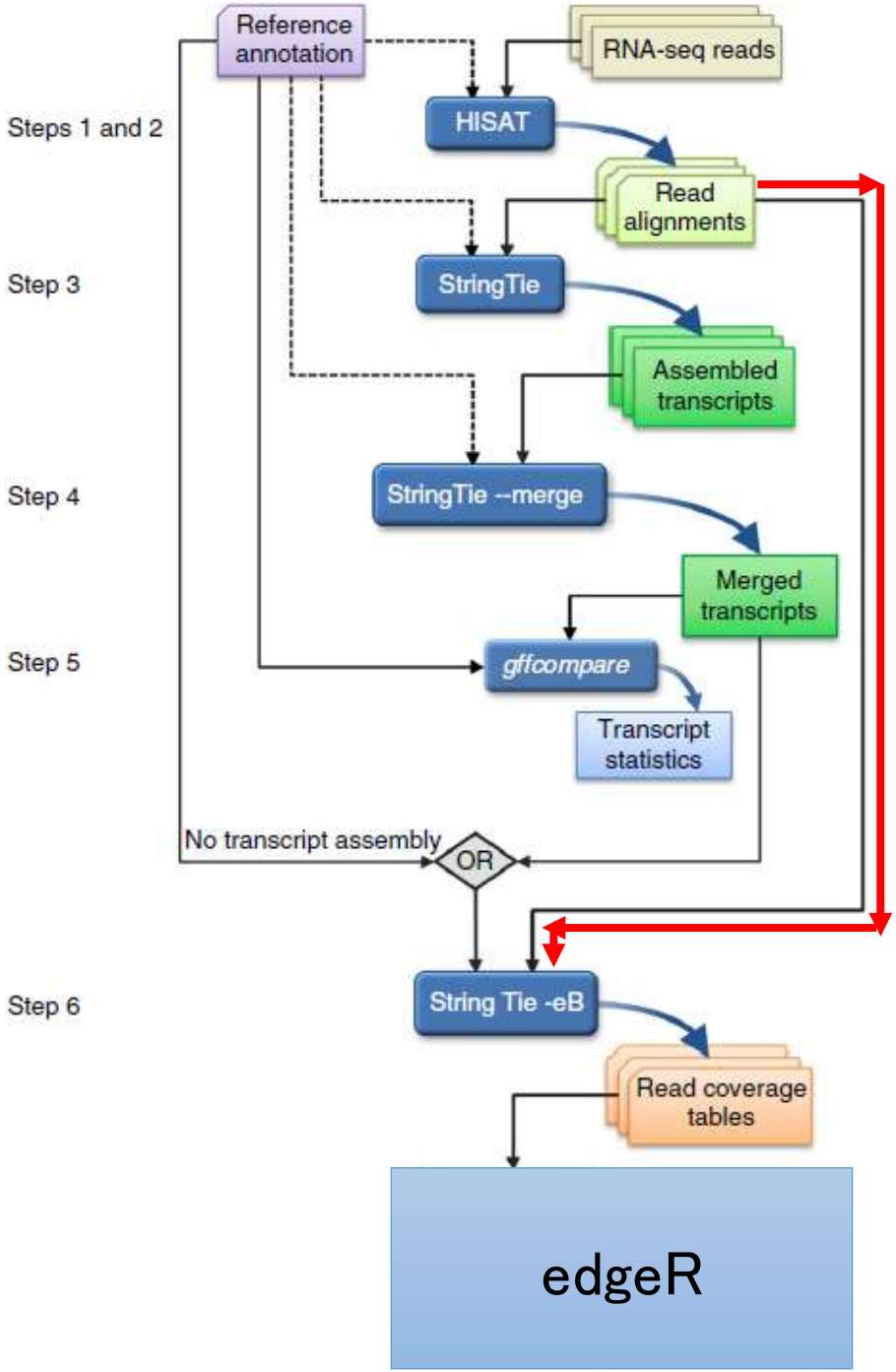
StringTieの解析の方向性として大きく2つある

・GTFファイルに記載された遺伝子モデルのみを数える

・新規な遺伝子モデルを見出し、それも数える
新規な遺伝子モデルはサンプルによって異なりうるので、
個々のモデルをStringTieのmerge modeでmergeし、
それを含めた、新しい遺伝子モデルを作製できる

StringTie基本コマンド

```
$ stringtie
StringTie v1.3.4d usage:
 stringtie <input.bam ..> [-G <guide_gff>] [-l <label>] [-o <out_gtf>] [-p
<cpus>]
  [-v] [-a <min_anchor_len>] [-m <min_tlen>] [-j <min_anchor_cov>] [-f
<min_iso>]
  [-C <coverage_file_name>] [-c <min_bundle_cov>] [-g <bdist>] [-u]
  [-e] [-x <seqid,..>] [-A <gene_abund.out>] [-h] {-B | -b <dir_path>}
Assemble RNA-Seq alignments into potential transcripts.
         :
         :
         :
Transcript merge usage mode:
  stringtie --merge [Options] { gtf_list | strg1.gtf ...}
With this option StringTie will assemble transcripts from multiple
input files generating a unified non-redundant set of isoforms. In this mode
the following options are available:
  -G <guide_gff>   reference annotation to include in the merging (GTF/GFF3)
  -o <out_gtf>     output file name for the merged transcripts GTF
                    (default: stdout)
         :
         :
         :
```



Case A

GTFファイルに記載された
遺伝子モデルのみを数える場合

StringTie

Transcript assembly and quantification for RNA-Seq

JOHNS HOPKINS UNIVERSITY
CENTER FOR COMPUTATIONAL BIOLOGY
CCB

CCB » Software » StringTie

Home | Manual | FAQ

Running StringTie

Run `stringtie` from the command line like this:

```
stringtie <aligned_reads.bam> [options]*
```

inputはsortされたBAM

The main input of the program is a BAM file with RNA-Seq read mappings which must be sorted by their genomic location (for example the `accepted_hits.bam` file produced by TopHat or the output of HISAT2 after sorting and converting it using samtools as explained below).

The following optional parameters can be specified when running `stringtie`:

`-h/--help`	Prints help message and exits.
`-v`	Turns on verbose mode, printing bundle processing details.
`-o [<path/>]<out.gtf>`	Sets the name of the output GTF file where StringTie will write the assembled transcripts. This can be specified as a full path, in which case directories will be created as needed. By default StringTie writes the GTF at standard output.
`-p <int>`	Specify the number of processing threads (CPUs) to use for transcript assembly. The default is 1.
`-G <ref_ann.gff>`	Use the reference annotation file (in GTF or GFF3 format) to guide the assembly process. The output will include expressed reference transcripts as well as any novel transcripts that are assembled. This option is required by options -B, -b, -e, -C (see below).

```
$ stringtie ¥
-e ¥
-p 4 ¥
-G genes.gtf ¥
-o count_genes.gtf ¥
hoge.sort.bam
```

-G reference annotation to use for guiding the assembly process (GTF/GFF3)
-e only estimate the abundance of given reference transcripts (requires -G)
-p number of threads (CPUs) to use (default: 1)
-o output path/file name for the assembled transcripts GTF (default: stdout)
-B enable output of Ballgown table files which will be created in the
same directory as the output GTF (requires -G, -o recommended)

個々のサンプルごとに行う

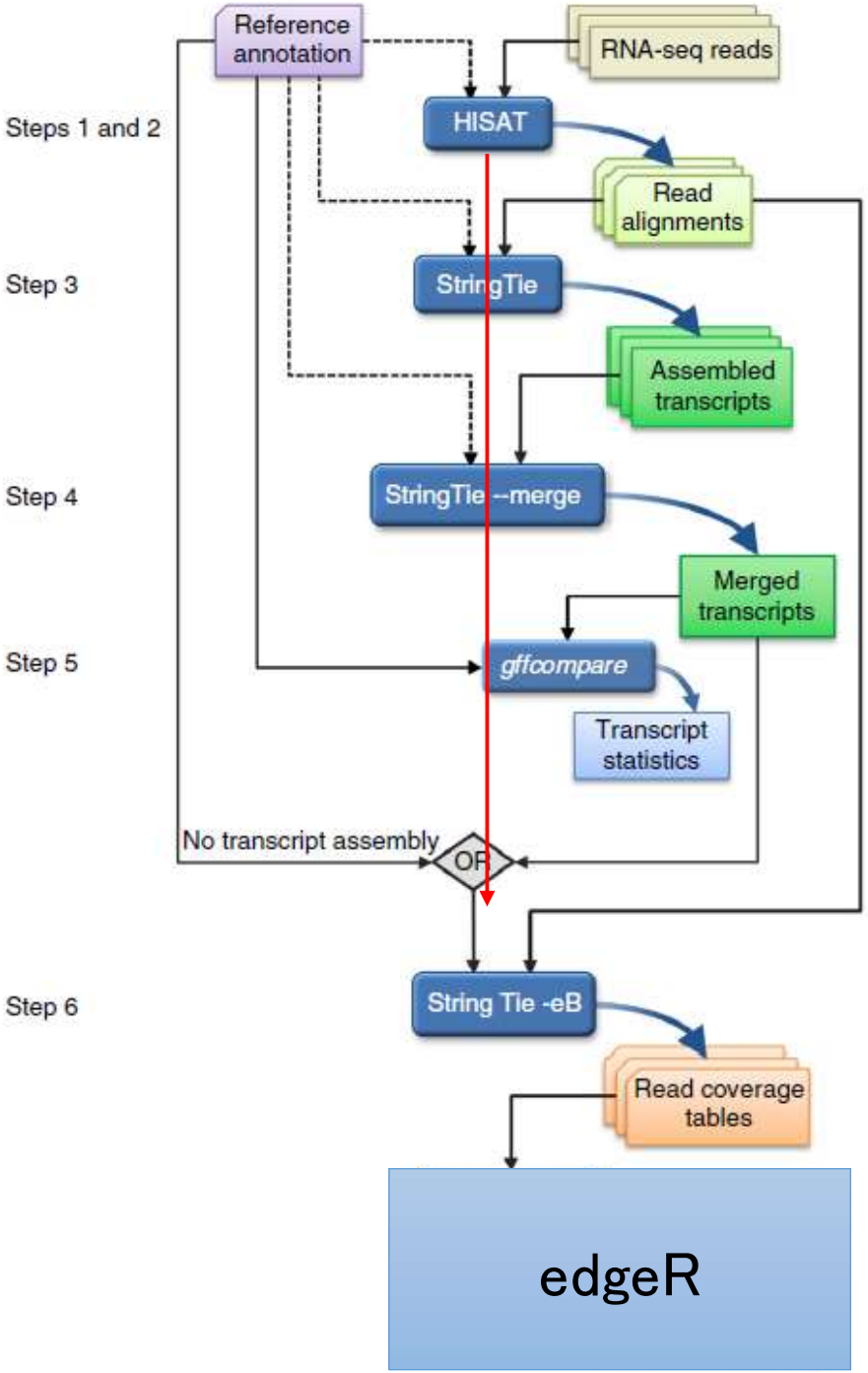
実習3　stringtie

HISATで作製したsamをsort.bamにし、StringTieにかける
hisat結果　2D2L_rep1.sam

```
$ samtools sort ¥
-@ 4 ¥
-o 2D2L_rep1.sort.bam ¥
2D2L_rep1.sam

$ stringtie -e -p 4 ¥
-G genes.gtf ¥
-o count_2D2L_rep1.gtf ¥
2D2L_rep1.sort.bam
```

samtools v1.3以降、
samファイルのsort,bam化
は同時にできる



Case B

新規な遺伝子モデルを見出し、
それも数える場合

```
$ stringtie ¥
-p 4 ¥
-G genes.gtf ¥
-o count_genes.gtf ¥
hoge.sort.bam
```

-G reference annotation to use for guiding the assembly process (GTF/GFF3)
-p number of threads (CPUs) to use (default: 1)
-o output path/file name for the assembled transcripts GTF (default: stdout)

-e の指定はなし

個々のサンプルごとに行う

StringTieのmerge modeでmerged_gtfファイルを作製する

```
$ stringtie ¥
--merge ¥
-p 4 ¥
-G genes.gtf ¥
-o stringtie_merged.gtf ¥
sample.list   #個々のgtfファイルのスペース区切りでの羅列も可
```

Sample.list　　gtfファイルの場所を指定

Ex)
count_2D_rep1.gtf
count_2D_rep2.gtf
count_2D_rep3.gtf
count_2D2L_rep1.gtf
count_2D2L_rep2.gtf
count_2D2L_rep3.gtf

mergeしたgtfファイルを-Gで指定して、先と同様-eを指定し、
個々のbamからカウントデータを得る

```
$ stringtie ¥
-e ¥
-p 4 ¥
-G stringtie_merged.gtf ¥
-o count_genes.gtf ¥
hoge.sort.bam
```

-G reference annotation to use for guiding the assembly process (GTF/GFF3)
-e only estimate the abundance of given reference transcripts (requires -G)
-p number of threads (CPUs) to use (default: 1)
-o output path/file name for the assembled transcripts GTF (default: stdout)

個々のサンプルでおこなう

gtfファイルを比較するツール

The gffcompare utility

The program gffcompare can be used to compare, merge, annotate and estimate accuracy of one or more GFF files (the "query" files), when compared with a reference annotation (also provided as GFF/GTF). A more detailed documentation for the program and its output files can be found here (gffcompare documentation page)

https://ccb.jhu.edu/software/stringtie/gff.shtml#gffcompare

```
gffcompare ¥
-r gene.gtf ¥
-o merged ¥
stringtie_merged.gtf
```

gene.gtf <-既知model
stringtie_merged.gtf <-含新規model
この両者を比較できる

```
# gffcompare v0.10.4 | Command line was:
#gffcompare -r genes.gtf -o merged stringtie_merged.gtf
#

#= Summary for dataset: stringtie_merged.gtf
#     Query mRNAs :   42241 in   33367 loci  (30667 multi-exon
transcripts)
#            (6233 multi-transcript loci, ~1.3 transcripts per
locus)
# Reference mRNAs :   41607 in   33350 loci  (30127 multi-exon)
# Super-loci w/ reference transcripts:    33240
#-----------------| Sensitivity | Precision  |
        Base level:   100.0     |     99.8    |
        Exon level:   100.0     |     99.4    |
      Intron level:   100.0     |     99.8    |
Intron chain level:   100.0     |     98.2    |
  Transcript level:   100.0     |     98.5    |
       Locus level:   100.0     |     99.9    |

     Matching intron chains:   30127
       Matching transcripts:   41607
              Matching loci:   33350

          Missed exons:       0/169264   (  0.0%)
           Novel exons:     102/170581   (  0.1%)
        Missed introns:       0/127896   (  0.0%)
         Novel introns:      55/128111   (  0.0%)
           Missed loci:       0/33350    (  0.0%)
            Novel loci:      37/33367    (  0.1%)
```

Differential expression analysis^

Differential expression analysis

Together with HISAT and Ballgown, StringTie can be used for estimating differential expression across multiple RNA-Seq samples and generating plots and differential expression tables as described in our protocol paper.

⋮

Using StringTie with DESeq2 and edgeR

DESeq2 and edgeR are two popular Bioconductor packages for analyzing differential expression, which take as input a matrix of read counts mapped to particular genomic features (e.g., genes). We provide a Python script (`prepDE.py`, or the Python 3 version: `prepDE.py3`) that can be used to extract this read count information directly from the files generated by StringTie (run with the `-e` parameter).

カウントマトリックス作製

```
$ python prepDE.py -h
Usage: prepDE.py [options]

Generates two CSV files containing the count matrices for genes and
transcripts, using the coverage values found in the output of `stringtie -e`

Options:
  -h, --help            show this help message and exit
  -i INPUT, --input=INPUT, --in=INPUT
                        the parent directory of the sample sub-directories or
                        a textfile listing the paths to GTF files [default:
                        ballgown]
  -g G                  where to output the gene count matrix [default:
                        gene_count_matrix.csv
  -t T                  where to output the transcript count matrix [default:
                        transcript_count_matrix.csv]
  -l LENGTH, --length=LENGTH
                        the average read length [default: 75]
  -p PATTERN, --pattern=PATTERN
                        a regular expression that selects the sample
                        subdirectories
  -c, --cluster         whether to cluster genes that overlap with different
                        gene IDs, ignoring ones with geneID pattern (see
                        below)
  -s STRING, --string=STRING
                        if a different prefix is used for geneIDs assigned by
                        StringTie [default: MSTRG]
  -k KEY, --key=KEY     if clustering, what prefix to use for geneIDs assigned
                        by this script [default: prepG]
  --legend=LEGEND       if clustering, where to output the legend file mapping
                        transcripts to assigned geneIDs [default: legend.csv]
```

```
$ python prepDE.py
```

gene_count_matrix.csv
transcript_count_matrix.csv

Case study 2: Genome-based RNA-Seq pipeline
を進め、確認してみよう。

```
gene_id,2D2L_rep1,2D2L_rep2,2D2L_rep3,2D2L_rep4,2D_rep1,2D_rep2,2D_rep3,4D_rep1,4
D_rep2,4D_rep3,4D_rep4
AT4G22890,295,204,203,154,20,22,17,35,26,17,22
AT1G38440,0,0,0,0,0,0,0,0,0,0,0
AT3G27910,0,0,0,0,0,0,0,0,0,0,0
AT1G06620,3,0,6,0,0,3,4,9,0,3,0
AT5G54067,0,0,0,0,0,0,0,0,0,0,0
AT2G34630,52,13,10,18,9,0,3,11,7,12,11
AT2G46660,0,0,0,3,4,0,0,16,23,3,6
AT2G25590,13,7,7,12,3,4,7,21,15,13,15
AT1G43171,0,0,0,0,0,0,0,0,0,0,0
AT5G25130,3,5,3,5,0,0,0,0,0,0,0
AT2G32280,6,0,7,0,5,0,15,0,5,6,0
AT3G15020,5,0,4,7,40,9,23,9,18,10,0
AT5G61100,0,0,0,0,0,0,0,0,0,0,0
AT5G01650,42,15,27,13,35,19,33,0,23,10,18
AT5G05570,6,8,4,4,3,5,3,0,11,9,3
AT3G09770,47,30,25,10,3,14,14,38,46,13,26
AT3G10210,9,0,5,12,0,7,12,20,9,9,3
AT5G06000,0,0,0,5,7,0,5,0,0,0,0
AT5G64620,40,31,20,31,64,35,41,21,37,41,36
AT1G75280,36,45,36,44,8,11,14,16,10,4,11
```

このカウントマトリックスファイルをedgeRへのinputとして、transcript base解析で扱った同一の方法で解析を進める。

edgeRでの解析

このケースでは　,が区切りのテキストとして得られているので、read.csvを用いる。

```
$ R
> library(edgeR)
> dat<-read.csv("gene_count_matrix.csv",row.names=1)
> group <- c(rep("2D",3),rep("2D2L",3))
> D<-DGEList(dat,group=group)
> D<-calcNormFactors(D)
> D<-estimateCommonDisp(D)
> D<-estimateTagwiseDisp(D)
```

2D vs 2D2Lの比較

```
> de_2D_2D2L <- exactTest(D,pair=c("2D","2D2L"))
> tmp <- topTags(de_2D_2D2L, n=nrow(de_2D_2D2L$table))
> write.table(tmp$table, "de.tagwise2.txt", sep="¥t", quote=F)
```

まとめ

HISAT

StringTie

edgeR

上記の流れを基盤にした、
genome baseのDEG解析を紹介した