Presenting Semi-Streaming Algorithms and Future Implementations for Speeding Up RNA-Seq Assembly graph

Roberto Carlos López Rivera: roberto.lopez16@upr.edu
Advisor: Humberto Ortiz Zuazaga: humberto@hpcf.upr.edu
Computer Science Department
University of Puerto Rico - Río Piedras
May 17, 2019

ABSTRACT

This is my first semester investigating with Dr. Humberto Ortiz Zuazaga, PhD. I was assigned to read and present a paper to my lab peers called Crossing the streams: a framework for streaming analysis of short DNA sequencing reads, by Qingpeng Zhang, Sherine Awad, and C. Titus Brown. I was also asked to work with a R package called Polyester that is used to simulate RNA sequencing experiments with differential transcript expression https://bioconductor.org/packages/release/bioc/vignettes/polyester/inst/doc/polyester.html, so that I could get some first hand experience when it comes to simulating reads and last but not least I was assigned a to do a modified tutorial by Dr. Humberto Ortiz Zuazaga named The Eel Pond mRNAseq Protocol that was made by C. Titus Brown, Camille Scott, and Leigh Sheneman. This tutorial is a lightweight protocol for assembling up to a few hundred million mRNAseq reads, annotating the resulting assembly, and doing differential expression with RSEM https://escambron-protocols.readthedocs.io/en/latest/mrnaseq/.

INTRODUCTION

Crossing the streams: a framework for streaming analysis of short DNA sequencing reads

Is a paper written by Qingpeng Zhang, Sherine Awad, and C. Titus Brown that presents a semi-streaming algorithm for k-mer spectral analysis of DNA sequencing reads, together with a derivative approach that is fully streaming. It states that approaches derived from spectral analysis can be very effective since spectral error correction achieves high accuracy, and show that spectral k-mer trimming is considerably more effective at removing errors than quality score-based approaches. However, while the implementation is based on a memory-efficient software package for k-mer and graph exploration, they do not evaluate computational performance, their goal is algorithmic improvement, not implementation improvement.

Polyester

Polyester is an R package designed to simulate RNA sequencing experiments with differential transcript expression. Given a set of annotated transcripts, Polyester will simulate the steps of an RNA-seq experiment and produce files containing simulated RNA-seq reads. You can run an example by following the instructions in this link: http://bit.ly/vignettes-polyester.

The Eel Pond mRNAseq Protocol

This is a lightweight protocol for assembling up to a few hundred million mRNAseq reads, annotating the resulting assembly, and doing differential expression with RSEM. It is part of khmer-protocols; which are a set of protocols for doing genomic data analysis – specifically, de novo mRNAseq assembly and de novo metagenome assembly – in the cloud. You can access and see the process that was made in the following link: http://bit.ly/escambron-protocols.

Thoughts

Observations made of the paper Crossing the streams

Crossing the streams: a framework for streaming analysis of short DNA sequencing reads is a great paper for beginners, it presents commonly used algorithm inside the field in a simplified way, and compares the new algorithm with the old ones. I had trouble understanding the mathematical part of the paper, but it was because of my poor mathematical background which I've been working to improve.

The Eel Pond mRNAseq Protocol

The Eel Pond mRNAseq Protocol tutorial is well made and easy to execute. There is not much to criticize except some of the files used were moved, but Dr. Humberto Ortiz-Zuazaga provided them. Very nice hands on experience for someone with no prior knowledge to genome assembling.

Process

Using Polyester to simulate reads second example.

This example was made using a skeleton provided by Dr. Humberto Ortiz Zuazaga since the idea is to learn how everything works and expose me to R, bash, and a little bit of Julia code. The exercise mostly consisted of me reading code and modifying file names by using the example provided in the vintagettes polyester link above. Also a FASTA file called `chr22.fa` is provided with `polyester`. This file contains sequences for 918 transcripts on chromosome 22, as annotated in hg19.

First step is to make sure that the R packages that we are going to use are installed.

````
## Setup packages

Check to see if the `BiocManager` and `polyester` packages are loadable, or install them if not.

```{r install}
if (!requireNamespace("BiocManager"))
    install.packages("BiocManager")
if (!requireNamespace("polyester"))
  BiocManager::install("polyester")
if (!requireNamespace("edgeR"))
  BiocManager::install("edgeR")
```
````

We create a fold change matrix for each transcript and each group, specify a fold change. Polyester will generate baseline read numbers (assuming no differential expression), and will then multiply those mean numbers by the fold change you specify for the replicates in that group. The fold change matrix for this simple 2-group experiment looks like this:

````
```{r fcmat}
fold_changes = matrix(c(4,4,rep(1,18),1,1,4,4,rep(1,16)), nrow=20)
head(fold_changes)
```
````

The rest of the experiment can be simulated with code like the chunk below.

```
library(polyester)
library(Biostrings)
# FASTA annotation
fasta_file = system.file('extdata', 'chr22.fa', package='polyester')
fasta = readDNAStringSet(fasta_file)
# subset the FASTA file to first 20 transcripts
small_fasta = fasta[1:20]
writeXStringSet(small_fasta, 'chr22_small.fa')
# ~20x coverage ----> reads per transcript = transcriptlength/readlength * 20
# here all transcripts will have ~equal FPKM
readspertx = round(20 * width(small_fasta) / 100)

# set up transcript-by-timepoint matrix:
num_timepoints = 12
countmat = matrix(readspertx, nrow=length(small_fasta), ncol=num_timepoints)

# add spikes in expression at certain timepoints to certain transcripts:
up_early = c(1,2)
up_late = c(3,4)
countmat[up_early, 2] = 3*countmat[up_early, 2]
countmat[up_early, 3] = round(1.5*countmat[up_early, 3])
countmat[up_late, 10] = 6*countmat[up_late, 10]
countmat[up_late, 11] = round(1.2*countmat[up_late, 11])

# simulate reads:
simulate_experiment_countmat('chr22_small.fa', readmat=countmat,
    outdir='timecourse_reads')

```

Now we process the reads

After simulating the reads, the julia code builds a dictionary of counts per file for each kmer, and kmers with more than 1 read per file (on average) are stored in a table.

````
```{bash}
if [ ! -f hashcounts.tsv ]; then
  julia --project=@. diffhash.jl
  julia --project=@. showhash.jl > hashcounts.tsv
fi
```
````

Find kmers that are differentially expressed

````
```{r}
sim_rep_info <- read.delim("timecourse_reads/sim_rep_info.txt")
hashcounts <- read.delim("hashcounts.tsv", header=FALSE, row.names=1)
```
````

Build the design how the files will look

```r
design <- cbind(rep(1, 20), c(rep(0,10), rep(1,10)))
colnames(design) <- c("C", "CvsT")
```

```r
library(edgeR)
dge <- DGEList(counts=hashcounts)
logCPM <- cpm(dge, log=TRUE, prior.count=3)
fit <- lmFit(logCPM, design)
fit <- eBayes(fit, trend=TRUE)
topTable(fit, coef=ncol(design))
```

We can find which k-mers are significantly differentially expressed, using `fdr` to correct for multiple testing.

```r
testresults <- decideTests(fit[,2]$p.value, adjust.method = "fdr")
sum(testresults != 0)
```

Writing out the kmers to a file will allow us to filter the reads to find reads that contain these kmers.

```r
diffkmers <- rownames(fit)[testresults != 0]
write(diffkmers, "diffkmers1.txt")
```

RESULTS

Kmers that were differentially expressed. (89.3KB):

https://github.com/RLopez18/diffhash/blob/master/diffkmers1.txt

Kmer Hash table (2.6MB file):

https://github.com/RLopez18/diffhash/blob/master/hashcounts.tsv

FUTURE WORK

For future work and studies I'll keep learning about the field of bioinformatics and apply my newly acquired knowledge in the projects that I'll be assigned.

REFERENCES

1. Ortiz-Zuazaga, Humberto, diffhash(2018), diffhash, https://github.com/humberto-ortiz/diffhash
2. Zhang Q, Awad S, Brown CT. 2015. Crossing the streams: a framework for streaming analysis of short DNA sequencing reads. PeerJ PrePrints 3:e890v1 https://doi.org/10.7287/peerj.preprints.890v1