Similar Threads

Thread	Thread Starter	Forum	Replies	Last Post
merged.gtf or combined .gtf	harshinamdar	Bioinformatics	0	09-29-2011 10:10 PM
cuffcompare output combined.gtf question	arrchi	Bioinformatics	0	09-01-2011 08:15 AM
cuffcompare only generates .gtf.tmap and .gtf.refmap?	julio514	Bioinformatics	4	07-15-2011 04:18 AM
^@ in .combined.gtf from cuffcompare v0.8.4	glacierbird	Bioinformatics	2	01-14-2011 12:42 PM
how does cuffcompare choose which transcript to put in combined.gtf file?	d f	Bioinformatics	0	11-09-2010 11:30 AM

turnersd
Member

Location: Charlottesville, VA

Join Date: May 2011
Posts: 61

cuffdiff: use merged.gtf from cuffmerge or combined.gtf from cuffcompare?

Dear community,

A similar question was asked here previously and no answers were posted. I'll expand that question.

I'm trying to better understand the cufflinks --> cuffdiff workflow. Once I run cufflinks on each of my .bam files (from tophat), I have a separate .gtf assembly for each sample. To run cuffdiff I need a single unified .gtf file of my assembled transcripts.

*So should I use the **merged.gtf** file produced by cuffmerge or the **combined.gtf** file produced by cuffcompare?* How are these two files different, and what would be the downstream effect of using one or the other for differential expression in cuffdiff?

EDIT: Or would a better workflow be to forego cuffmerge/cuffcompare altogether in favor of running cufflinks on a merge of all the .bam files to generate a single assembly that maximizes assembly accuracy, and use this as the "reference" for cuffdiff? (E.g. samtools merge)

More info:

When running cuffmerge I run into a problem described here previously but not fully resolved ("/lib64/libz.so.1: no version information available" and "File ./merged_asm/tmp/mergeSam_fileBpOwTS doesn't appear to be a valid BAM file"). Cuffmerge still produced the merged.gtf file, but I'm concerned about continuing with cuffdiff without knowing what these cuffmerge errors are about. I have no problem running cuffcompare to get a combined.gtf file.

From the cuffcompare documentation:

> Quote:
>
> Cuffcompare clusters/tracks transfrags across samples, and writes a GTF file <outprefix>.combined.gtf containing a nonredundant set of transcripts across all input files (with a single representative transfrag chosen for each clique of matching transfrags across samples).

From the cuffmerge documentation:

> Quote:
>
> cuffmerge takes two or more Cufflinks GTF files and merges them into a single unified transcript catalog. Optionally, you can provide the script

with a reference GTF, and the script will use it to attach gene names and other metadata to the merged catalog.

Last edited by turnersd; 12-22-2011 at 07:15 AM. Reason: added third option about single cufflinks run

Quote

12-22-2011, 06:47 AM #2

mehc
Junior Member

Location: Montreal

Join Date: Aug 2011
Posts: 9

Might be wrong but combined might be all the transcripts that are found for any sample
Merged is then the intersection of that, so that each sample has the transcript, needed for differential expression

Quote

12-26-2011, 09:50 PM #3

marcowanger
Senior Member

Location: Hong Kong

Join Date: Dec 2008
Posts: 324

Quote:

> Originally Posted by **turnersd**
> *Dear community,*
>
> *A similar question was asked here previously and no answers were posted. I'll expand that question.*
>
> *I'm trying to better understand the cufflinks --> cuffdiff workflow. Once I run cufflinks on each of my .bam files (from tophat), I have a separate .gtf assembly for each sample. To run cuffdiff I need a single unified .gtf file of my assembled transcripts.*
>
> *So should I use the **merged.gtf** file produced by cuffmerge or the **combined.gtf** file produced by cuffcompare? How are these two files different, and what would be the downstream effect of using one or the other for differential expression in cuffdiff?*
>
> ***EDIT**: Or would a better workflow be to forego cuffmerge/cuffcompare altogether in favor of running cufflinks on a merge of all the .bam files to generate a single assembly that maximizes assembly accuracy, and use this as the "reference" for cuffdiff? (E.g. samtools merge)*
>
> *More info:*
>
> *When running cuffmerge I run into a problem described here previously but not fully resolved ("/lib64/libz.so.1: no version information available" and "File ./merged_asm/tmp/mergeSam_fileBpOwTS doesn't appear to be a valid BAM file"). Cuffmerge still produced the merged.gtf file, but I'm concerned about continuing with cuffdiff without knowing what these cuffmerge errors are about. I have no problem running cuffcompare to get a combined.gtf file.*
>
> *From the cuffcompare documentation:*
>
> *From the cuffmerge documentation:*

Back in old days (< v.1.0), you have to use Cuffdiff, cuffcompare is the way to use. Cuffmerge looks more like a housekeeping work.

_____
Marco

Quote

12-27-2011, 08:50 AM #4

jbrwn
Member

Location: Denver, CO

Join Date: Mar 2011
Posts: 34

Quote:

> Originally Posted by **turnersd**
> *"/lib64/libz.so.1: no version information available"*
>
> *"File ./merged_asm/tmp/mergeSam_fileBpOwTS doesn't appear to be a valid BAM file"*

No need to worry about either of these. The second line concerning the bam would make more sense if you posted the next line of the output which is something like, "Trying SAM", which does work. That's a samtools thing and not a cufflinks/cuffmerge thing.

I use the merged.gtf ("a single unified transcript catalog.").

Here's my command line:

Code:

```
cuffmerge --ref-gtf %s --num-threads 8 --ref-sequence %s %s
```

(gene annotation, reference sequence, transcript_manifest)

Quote

12-27-2011, 01:03 PM #5

polyatail
Member

Location: New York, NY

Join Date: Dec 2010
Posts: 25

Quote:

> EDIT: Or would a better workflow be to forego cuffmerge/cuffcompare altogether in favor of running cufflinks on a merge of all the .bam files to generate a single assembly that maximizes assembly accuracy, and use this as the "reference" for cuffdiff? (E.g. samtools merge)

I'm curious about this too. Cuffmerge (apparently) merges GTFs by converting them to SAM and running Cufflinks on the resulting fake alignments to produce new transcripts. In what ways is this different from pooling your original alignments and running Cufflinks on those (besides memory issues)?

Quote

12-27-2011, 01:10 PM #6

turnersd
Member

Location: Charlottesville, VA

Join Date: May 2011
Posts: 61

Thanks all for the response. In addition to polyatail's question, I'm really more curious about how the interpretation of the results differs when you use one of three workflows:

1. Cufflinks on each bam file, cuffcompare on all the bam files, cuffdiff using combined.gtf

2. Cufflinks on each bam file, cuffmerge on all the bam files, cuffdiff using merged.gtf

3. samtools merge on all the bam files, cufflinks on that bam file, cuffdiff using accepted_hits.bam from that cufflinks run.

From an answer on biostar:

Quote:

> Cuffcompare tries to match transcripts across the samples to each other in order to do the differential expression analysis. Unsure what the algorithm they use to determine whether two transcripts are comparable.
>
> Cuffmerge will actually convert your .gtf assembles into .sam and run cufflinks on the .sam file to produce a merged assembly.

From a collaborator of mine:

Quote:

> My preferred strategy is to run cufflinks on a merge of all the .bam files, to generate one assembly that maximizes assembly accuracy, and to use that assembly as the "reference" for cuffdiff (which will then use the individual .bam files).
>
> I've never understood what cuffcompare's .combined GTF is useful for, as it's a hodgepodge of "examplars" from each transcript/isoform grouping. And the cuffmerge just smashes them all together.

Quote

12-27-2011, 01:50 PM #7

polyatail
Member

Location: New York, NY

Join Date: Dec 2010
Posts: 25

Check out this paper's supplemental:

Cabili MN, Trapnell C, Goff L, Koziol M, Tazon-Vega B, Regev A, Rinn JL. Integrative annotation of human large intergenic noncoding RNAs reveals global properties and specific subclasses. Genes Dev. 2011 Sep 15;25(18):1915-27.

http://www.ncbi.nlm.nih.gov/pubmed/21890647

To combine output from Scripture and Cufflinks across multiple samples, they ran everything individually then pooled the results using some interesting tricks. They don't mention how all the resulting assemblies were combined--maybe one of the authors can comment if it was cuffcompare- or cuffmerge-style?

IMHO, combining BAMs might result in lowly-expressed isoforms being diluted out by one of the coverage-based cutoffs (i.e. -F, -j). It sort of seems like a way of filtering out potentially low-confidence or low-coverage transcripts except without being able to specify the parameters and without knowing exactly what you're losing. Perhaps using cuffmerge puts the transcripts on a sort of level playing field since the underlying reads are artificially (and fairly) reconstructed.

Quote

01-04-2012, 09:01 AM #8

RogerAB
Junior Member

Location: Tucson, AZ

Join Date: Jan 2010
Posts: 2

More best guesses.

I have been asking myself the same question for several days, because it isn't obvious. But I think if you look at the manual for Cufflinks, the answer is more or less there. They both do the same thing. Cuffmerge uses Cuffcompare to combine the transcript.gtf files. The difference probably comes down to convenience. Cuffmerge allows you to use a list in text form if you have a lot of transcript.gtf files to work with (as some labs do). Cuffmerge uses Cuffcompare to do the combining. Cuffcompare offers a few more options (-R,-C,-V). Cuffmerge allows multithreading, again important if you are running a lot of data. Key sections from the manual (http://cufflinks.cbcb.umd.edu/manual.html):
"It [Cuffmerge] handles also handles running Cuffcompare for you"; and
"The main purpose of this script [Cuffmerge] is to make it easier to make an assembly GTF file suitable for use with Cuffdiff".
So I think Cuffmerge is the tool for large projects. Cuffcompare is the original script that may offer a few extra options.

Quote

01-20-2012, 05:23 AM #9

Cole Trapnell
Senior Member

Location: Boston, MA

Join Date: Nov 2008
Posts: 212

I can shed some light on this. We have an upcoming protocol paper that describes our recommended workflow for TopHat and Cufflinks that discusses some of these issues.

As turnersd outlined, there are three strategies:

1) merge bams and assemble in a single run of Cufflinks
2) assemble each bam and cuffcompare them to get a combined.gtf
3) assemble each bam and cuffmerge them to get a merged.gtf

All three options work a little differently depending on whether you're also trying to integrate reference transcripts from UCSC or another annotation source.

#1 is quite different from #2 and #3, so I'll discuss its pros and cons first. The advantage here is simplicity of workflow. It's one Cufflinks run, so no need to worry about the details of the other programs. As turnersd mentions, you might also think this maximizes the accuracy of the resulting assembly, and that might be the case, but it also might not (for technical reasons that I don't want to get into right now). The disadvantage of this approach is that your computer might not be powerful enough to run it. More data and more isoforms means substantially more memory and running time. I haven't actually tried this on something like the human body map, but I would be very impressed and surprised if Cufflinks can deal with all of that on a machine owned by mere mortals.

#2 and #3 are very similar - both are designed to gracefully merge full-length and partial transcript assemblies without ever merging transfrags that disagree on splicing structure. Consider two transfrags, A and B, each with a couple exons. If A and B overlap, and they don't disagree on splicing structure, we can (and according to Cufflinks' assembly philosophy, we should) merge them. The difference between Cuffcompare and Cuffmerge is that Cuffcompare will only merge them if A is "contained" in B, or vice versa. That is, only if one of the transfrags is essentially redundant. Otherwise, they both get included. Cuffmerge on the other hand, will merge them if they overlap, and agree on splicing, and are in the same orientiation. As turnersd noted, this is done by converting the transfrags into SAM alignments and running Cufflinks on them.

The other thing that distinguishes these two options is how they deal with a reference annotation. You can read on our website how the Cufflinks Reference Annotation Based Transcript assembler (RABT) works. Cuffcompare doesn't do any RABT assembly, it just includes the reference annotation in the combined.gtf and discards partial transfrags that are contained and compatible with the reference. Cuffmerge actually runs RABT when you provide a reference, and this happens during the step where transfrags are converted into SAM alignments and assembled. We do this to improve quantification accuracy and reduce errors downstream. I should also say that Cuffmerge runs cuffcompare in order annotate the merged assembly with certain helpful features for use later on.

So we recommend #3 for a number of reasons, because it is the closest in spirit to #1 while still being reasonably fast. For reasons that I don't want to get into here (pretty arcane details about the Cufflinks assembler) I also feel that option #3 is actually the most accurate in most experimental settings.

Quote

01-20-2012, 05:37 AM #10

turnersd
Member

Location: Charlottesville, VA

Join Date: May 2011
Posts: 61

Thanks, Cole. I'm very much looking forward to the protocol paper.

-Stephen

Quote

01-20-2012, 06:00 AM #11

marcowanger
Senior Member

Location: Hong Kong

Join Date: Dec 2008
Posts: 324

Thanks Cole. Thanks for the clarification on cuffcompare and cuffmerge. Looking forward for the paper.

> Quote:
> Originally Posted by **Cole Trapnell**
> *I can shed some light on this. We have an upcoming protocol paper that describes our recommended workflow for TopHat and Cufflinks that discusses some of these issues.*
>
> *As turnersd outlined, there are three strategies:*
>
> *1) merge bams and assemble in a single run of Cufflinks*
> *2) assemble each bam and cuffcompare them to get a combined.gtf*
> *3) assemble each bam and cuffmerge them to get a merged.gtf*
>
> *All three options work a little differently depending on whether you're also trying to integrate reference transcripts from UCSC or another annotation source.*
>
> *#1 is quite different from #2 and #3, so I'll discuss its pros and cons first. The advantage here is simplicity of workflow. It's one Cufflinks run, so no need to worry about the details of the other programs. As turnersd mentions, you might also think this maximizes the accuracy of the resulting assembly, and that might be the case, but it also might not (for technical reasons that I don't want to get into right now). The disadvantage of this approach is that your computer might not be powerful enough to run it. More data and more isoforms means substantially more memory and running time. I haven't actually tried this on something like the human body map, but I would be very impressed and surprised if Cufflinks can deal with all of that on a machine owned by mere mortals.*
>
> *#2 and #3 are very similar - both are designed to gracefully merge full-length and partial transcript assemblies without ever merging transfrags that disagree on splicing structure. Consider two transfrags, A and B, each with a couple exons. If A and B overlap, and they don't disagree on splicing structure, we can (and according to Cufflinks' assembly philosophy, we should) merge them. The difference between Cuffcompare and Cuffmerge is that Cuffcompare will only merge them if A is "contained" in B, or vice versa. That is, only if one of the transfrags is essentially redundant. Otherwise, they both get included. Cuffmerge on the other hand, will merge them if they overlap, and agree on splicing, and are in the same orientiation. As turnersd noted, this is done by converting the transfrags into SAM alignments and running Cufflinks on them.*
>
> *The other thing that distinguishes these two options is how they deal with a reference annotation. You can read on our website how the Cufflinks Reference Annotation Based Transcript assembler (RABT) works. Cuffcompare doesn't do any RABT assembly, it just includes the reference annotation in the combined.gtf and discards partial transfrags that are contained and compatible with the reference. Cuffmerge actually runs RABT when you provide a reference, and this happens during the step where transfrags are converted into SAM alignments and assembled. We do this to improve quantification accuracy and reduce errors downstream. I should also say that Cuffmerge runs cuffcompare in order annotate the merged assembly with certain helpful features for use later on.*
>
> *So we recommend #3 for a number of reasons, because it is the closest in spirit to #1 while still being reasonably fast. For reasons that I don't want to get into here (pretty arcane details about the Cufflinks assembler) I also feel that option #3 is actually the most accurate in most experimental settings.*

_____
Marco

Quote

01-20-2012, 11:42 AM #12

amackey
Junior Member

Location: 19129

Join Date: Mar 2009
Posts: 3

Thanks for this!

I've heard something like this before, i.e. that strategy #1 has some arcane bias introduced by assembling across conditions that might exhibit different isoform expression -- e.g. imagine a gene with two (complicatedly different) isoforms A and B, and some "perfect" situation wherein A is expressed only in condition 1, and B is

expressed only in condition 2. Assembling them independently guarantees that the structures of A and B are assembled with maximal accuracy (given sufficient read depth and isoform coverage). Assembling them together may lead to unresolvable structural uncertainties, affecting the accuracy of the final transcripts. I need to come up with a picture to show an example of this (or even better, a runnable "toy" example). However, I've never trusted (or I guess fully understood) what cuffcompare vs. cuffmerge do, and so I've been hesitant to follow one of the other strategies. I also don't believe that the "perfect" A vs. B situation happens very often (usually every sample already has a mixture of A and B, and assembling them independently just lowers accuracy by decreasing read depth; and still unclear to me that the accuracy is recovered by the cuffcompare/cuffmerge steps), but I'd be happy to be shown wrong.

Cole, I'm curious whether your logic extends to assembling replicates -- i.e. if I have three biological replicates of the same condition, should I assemble these independently as well? And then for four experimental conditions, I'm cuffmerge'ing across 12 assemblies? Should the "best" strategy change with increasing experimental complexity?

Nowadays we are learning more towards using Trinity for the assembly (which, by the way, would likely have the same problem resolving a complicated mixture of A/B isoforms), with RSEM, but I'd love to see a "bakeoff".

Quote

03-20-2012, 12:12 AM #13

dadada4ever
Member

Location: Beijing

Join Date: Mar 2010
Posts: 12

Quote:

> Originally Posted by **turnersd**
> *Thanks all for the response. In addition to polyatail's question, I'm really more curious about how the interpretation of the results differs when you use one of three workflows:*
>
> *1. Cufflinks on each bam file, cuffcompare on all the bam files, cuffdiff using combined.gtf*
>
> *2. Cufflinks on each bam file, cuffmerge on all the bam files, cuffdiff using merged.gtf*
>
> *3. samtools merge on all the bam files, cufflinks on that bam file, cuffdiff using accepted_hits.bam from that cufflinks run.*
>
> *From an answer on biostar:*
>
> *From a collaborator of mine:*

same question for me. still confused about cuffcompare and cuffmerge

Quote

03-20-2012, 04:31 AM #14

turnersd
Member

Location: Charlottesville, VA

Join Date: May 2011
Posts: 61

The protocol paper that Cole mentioned is now out in Nature Protocols.

Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks.

The paper recommends using cuffmerge to merge all the transcripts into a single GTF file that you then use downstream (cuffdiff):

Quote:

> When you are working with several RNA-seq samples, it becomes necessary to pool the data and assemble it into a comprehensive set of transcripts before proceeding to differential analysis. A natural approach to this problem would be to simply pool aligned reads from all samples and run Cufflinks once on this combined set of alignments. However, we do not usually recommend this tactic for two reasons. First, because assembly becomes more computationally expensive as read depth increases, assembling the pooled alignments may not be feasible with the machines available in your laboratory. Second, with a pooled set of reads, Cufflinks will be faced with a more complex mixture of splice isoforms for many genes than would be seen when assembling the samples individually, and this increases the probability that it will assemble the transcripts incorrectly (associating the wrong outcomes of different splicing events in some transcripts). A better strategy is to assemble the samples individually and then merge the resulting assemblies together. We have recently developed a utility program, Cuffmerge, which handles this task using many of the same concepts and algorithms as Cufflinks does when assembling transcripts from individual reads.

If you actually look at what cuffmerge is doing, its converting all the GTFs to sam files, then running cufflinks with RABT on each of those sam files, then runs cuffcompare on the resulting GTF.

The paper also recommends using the RABT option differently than I had been doing. Previously I was running

cufflinks with the RABT option for each alignment, but in the protocols paper, they recommend running cufflinks without the reference annotation, then running cuffmerge with RABT.

> Quote:
> Cuffmerge is essentially a 'meta-assembler'—it treats the assembled transfrags the way Cufflinks treats reads, merging them together parsimoniously. Furthermore, when a reference genome annotation is available, Cuffmerge can integrate reference transcripts into the merged assembly. It performs a reference annotation-based transcript (RABT) assembly to merge reference transcripts with sample transfrags and produces a single annotation file for use in downstream differential analysis.

This makes sense, because if you pepper your reads with faux reads from the reference separately for each assembly, seems like you would get lots of stuff from the reference that isn't really represented in the sample, and this problem is compounded when doing separately for each assembly then merging.

The paper didn't give many details about why using cuffmerge is better than merging the alignments and running cufflinks there, except to say that "Cufflinks will be faced with a more complex mixture of splice isoforms for many genes than would be seen when assembling the samples individually, and this increases the probability that it will assemble the transcripts incorrectly."

Quote

03-22-2012, 02:01 AM #15

kenietz
Member

Location: Singapore

Join Date: Nov 2011
Posts: 63

Hi guys,
very interesting thread. I have one question though. What about the merging of genes when they are very close on the same strand?

------- from gene_exp.diff ---------
AT1G01225,AT1G01230 Chr1:95986-99240
----

when actually gene AT1G01225 is 95987-97407 and AT1G01230 is 97456-99240. But the merged.gtf says that AT1G01225 is contained in AT1G01230. Cant get that

I have WT and MUT and my flow for both is like that:
>tophat -p 4 -o tophat_wt -G TAIR_9.gff ~/Work_drive/Genomes/TAIR_9/TAIR_9_rg WT-1.fq.gz
>cufflinks -o cuffl_wt -p 4 -g TAIR_9.gff tophat_wt/accepted_hits.bam

Then:
>cuffmerge -p 4 -s TAIR_9_rg.fa -g TAIR_9.gff assemblies.txt
>cuffdiff -p 4 -o cuffdiff_out -L WT,MUT tophat_wt/accepted_hits.bam tophat_mut/accepted_hits.bam

Contents of 'assemblies.txt":
cuffl_wt/transcripts.gtf
cuffl_mut/transcripts.gtf

I saw somewhere on the forum that -g should be used with cufflinks in order to do RABT and solve the issue with merging of genes. But i see its not working so. So now is advised to run cufflink without any -g or -G option and then give the reference annotation to cuffmerge only?

Thank you for you help

Quote

07-05-2012, 01:55 PM #16

stanwish
Junior Member

Location: michigan

Join Date: Oct 2011
Posts: 2

Haha, same question here.
I followed the review protocol paper and used almost same command as kenietz, the gene_exp.diff combined several genes together but the isoform_exp.diff is normal.

Does anyone have any idea on it?

Thanks.

> Quote:
> Originally Posted by **kenietz**
> *Hi guys,*
> *very interesting thread. I have one question though. What about the merging of genes when they are very close on the same strand?*
>
> *------- from gene_exp.diff ---------*

AT1G01225,AT1G01230 Chr1:95986-99240
----

when actually gene AT1G01225 is 95987-97407 and AT1G01230 is 97456-99240. But the merged.gtf says that AT1G01225 is contained in AT1G01230. Cant get that

I have WT and MUT and my flow for both is like that:
>tophat -p 4 -o tophat_wt -G TAIR_9.gff ~/Work_drive/Genomes/TAIR_9/TAIR_9_rg WT-1.fq.gz
>cufflinks -o cuffl_wt -p 4 -g TAIR_9.gff tophat_wt/accepted_hits.bam

Then:
>cuffmerge -p 4 -s TAIR_9_rg.fa -g TAIR_9.gff assemblies.txt
>cuffdiff -p 4 -o cuffdiff_out -L WT,MUT tophat_wt/accepted_hits.bam tophat_mut/accepted_hits.bam

Contents of 'assemblies.txt":
cuffl_wt/transcripts.gtf
cuffl_mut/transcripts.gtf

I saw somewhere on the forum that -g should be used with cufflinks in order to do RABT and solve the issue with merging of genes. But i see its not working so. So now is advised to run cufflink without any -g or -G option and then give the reference annotation to cuffmerge only?

Thank you for you help

Quote

07-09-2012, 11:29 AM #17

billstevens
Member

Location: Baltimore

Join Date: Mar 2012
Posts: 98

Hi guys,

I have a question on read depth. Cole has mentioned that cuffdiff only tests when there are at least 10 fragments in one of the conditions. Which output file tests that? I was thinking it was genes.count_tracking. However, I checked this when I had only 100M reads versus 200M reads, and both times I ended up with the same number of genes over 10 fragments.

Anyone know how to check read depth?

Quote

Post Reply

Tags

cuffcompare, cuffdiff, cufflinks, cuffmerge, rna-seq

« Previous Thread | Next Thread »

Posting Rules

You **may not** post new threads
You **may not** post replies
You **may not** post attachments
You **may not** edit your posts

BB code is **On**
Smilies are **On**
[IMG] code is **On**
HTML code is **Off**

Forum Rules

All times are GMT -8. The time now is 12:08 AM.