Next Generation Sequencing Technologies: Data Analysis and Applications

Transcriptome Assembly Quantification

Dr. Riddhiman Dhar, Department of Biotechnology
Indian Institute of Technology Kharagpur

Good day, everyone. Welcome to the course on Next Generation Sequencing Technologies, Data Analysis, and Applications. In the last class, we talked about read mapping for transcriptome analysis, and we talked about some of the methods that we use. So we will continue that discussion, and we will talk about the next steps, which are transcriptome assembly and quantification. So these are the concepts that we will be covering today. So the first concept is transcriptome assembly, right?

We will talk about what assembly is. We will discuss some of the tools that are used for assembly and how you evaluate the quality of assembly. And the second concept that we will be covering today is the quantification of transcript abundance, right? So this is a very important step for any further analysis, right?

So we want to count, right, how many transcripts are there in the cell, ok? So this is what we do using this transcript abundance quantification. These are the keywords. One is assembly, and another is abundance. Now going back to the RNA sequencing data processing pipeline, right?

So we have seen this pipeline before. So we start with the read data, which is usually in FASTQ format if you are working with Illumina data especially. Then we do the quality control. This is a common step that we have also done for variant analysis, etc. Then we have the mapping step, which actually leads to this read mapping against a reference sequence.

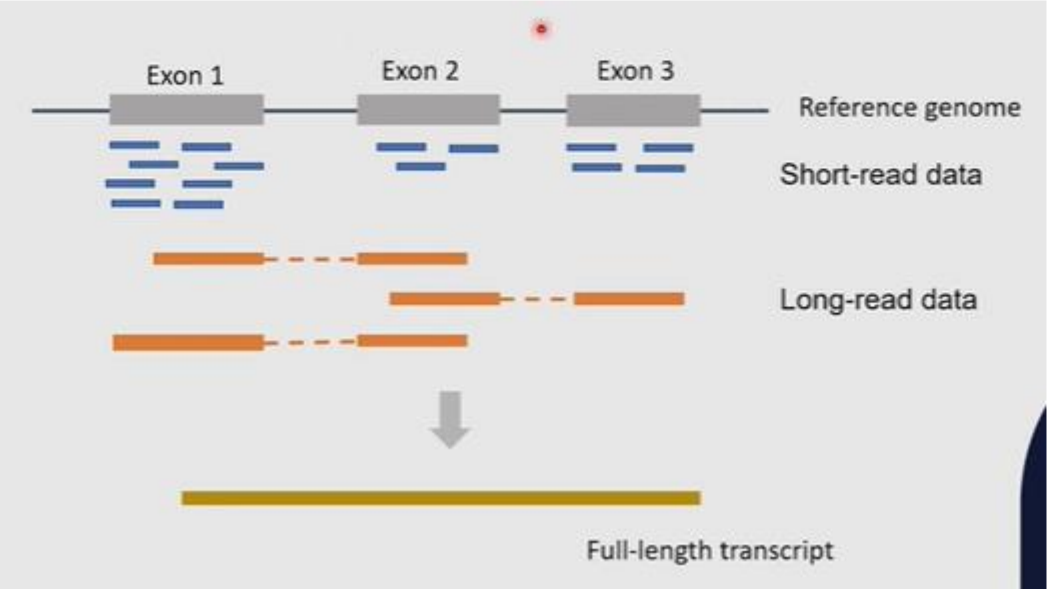
So we have talked about three methods. We have mentioned three methods, and we have discussed Star and HISAT2, and we have discussed HISAT2 in much more detail in the last class. And once we have done the mapping, this is done now. So we can now do something called assembly, ok? So we can do the assembly directly from the data itself, ok?

There are tools that will allow us to do that. So this is something we will discuss today in this class as well as in the next class, ok? And then, once you have done the assembly, we can do the quantification, ok? So let us start with the transcriptome assembly. What do we understand by assembly, ok? So assembly means the reconstruction of a complete set of full-length transcripts from the read data.

So we have mapped the reads against the reference sequence, but these are fragments, right? So you remember the steps, right? So we have this RNA isolated, we have cDNA synthesis, and we have fragmentation. So the reads come from fragments of transcripts, right? So in these cases, what we want to do is assemble the full-length transcript from the fragments.

So this is what the assembly process does, ok? So we can use short and long-read data, right? So again, depending on the platform that we are using, if we are using Illumina, for example, or Ion Torrent, we will get short read data, but if we are using SMRT or Nanopore, we will get long read data. So depending on the platform, we can have different types of data. We can also have hybrid data, right?

We can mix this short and long read data, and we can do the assembly, ok? So the goal is to get the transcripts, but also to get the isoforms, right? So we have discussed isoforms when we are talking about transcriptomic analysis, and we discussed how many types of isoforms you can get and what are the processes that can give rise to isoforms, right? So this is something that is actually quite important to also get the isoforms, right, because we also want to know in many cases which isoforms are expressed, right, and maybe they have some specific function or specific role in the biological process, ok. So this is summarizing what we just said, right?



So we have, imagine we have a gene, right, with three exons. We have exon 1, exon 2, and exon 3 present in the reference genome, and we have let us say short read data, right, where reads are mapping to these regions within exons, but also some reads might also map in this junction, ok, but that will be quite rare in the case of short read data. But in the case of long read data, right, in many cases you will see these reads that will be spanning these junctions; right, these are splice junctions. So you will get this kind of read when you are working with long-read data, right? So one of the advantages of long-read data is that you can then identify which isoforms are expressed because of the presence of these junctions in your data.

So we can take this short read data and long read data and assemble the full-length transcript, ok? So how do you do that? We will not go into a lot of details, right, about the algorithm because the algorithm that is used is also common with the genome assembly process, and we will discuss that algorithm in the genome assembly part in much more detail. So I will just mention the algorithm and some of the tools that are used for this process, but we will not get into the algorithm itself, ok? So what you get in transcriptome assembly is that you will get many fragments that correspond to different transcripts. So you have, let us say, 10,000 genes; you expect 10,000 transcripts, right? So you will have these; after the assembly process, you should get around that number, right?

So that is quite a big number if you compare it against genome assembly, right? So if you have genome assembly, this is very similar. Right, transcriptome assembly is very similar to genome assembly. You will see when we discuss the genome assembly process. So in genome assembly, what we expect, right? Ideally, you expect multiple chromosomes, right? So if you have, let us say, 16 chromosomes, you expect 16 chromosomes—right, 16 fragments—assembled.

Whereas in the case of transcriptome assembly, you will expect maybe 10,000 fragments, again depending on the number of genes that are present in that organism. So this is a major difference between transcriptome assembly and genome assembly, right? So this is in contrast to the small number of large fragments, or what you call contigs in the case of genome assembly. Now, transcriptome assembly can be of two different types. The first one is reference-guided transcriptome assembly.

So as the name suggests, there is a reference genome against which you are doing this assembly. So we have the genome of the organism or a closed related species that is used as a guide, ok, for the assembly process. Or you can do genome assembly, right? So there is no reference genome, and you have to make sense of the whole thing somehow. And this is, as you can probably understand, this is much more challenging, ok? When you have a reference genome, it is actually slightly easier to do the assembly.

If you do not have any references, this genome assembly process is a much more challenging problem. We will come to this discussion later on when we talk about genome assembly, and I will show you, right, why this is a much more challenging process. And we will illustrate, right, the algorithms, and you will understand why this requires a lot more computational power and a lot more complex algorithm. So what are the major challenges for transcriptome assembly? So the first challenge is that there is non-uniform coverage due to differences in expression and other factors. This is something that you probably realize is a big difference compared to the genome assembly, right?

So in the genome, you have just one copy of the genome. Maybe some elements are repeated, right, or present in multiple copies in the genome, but they are not hugely variant. It is usually not the case that one region is present in a thousand copies and another region is present in five copies. It is usually not like that when you are going for genome assembly. In transcriptome assembly, some genes will be expressed at a very high level.

So maybe a large number of RNA molecules are present in the cell, whereas some genes are expressed at a very low level. Maybe you have just one or two mRNA molecules present in the cell. So given this difference, how do you actually assemble the whole thing? And this is even more challenging for genes that are expressed at very low levels because you have probably only a few reads mapping to those genes. So this is one of the major challenges for assembly. In addition, what you get is that you also have non-uniform coverage along the length of a transcript.

So when mRNA is produced, there is a cDNA synthesis step; there is amplification if you are going for Illumina, etc. So all these processes lead to some sort of bias, right? So we will talk about these biases a bit later in subsequent presentations, where we will have a detailed discussion of these biases, but it will be enough to know now that there is non-uniform coverage along the length of a transcript. So if you consider a gene A, right, and you see the transcript A, you will not get reads across the whole transcript, right? You will see some bias; you will see some parts of the transcript, especially, for example, the ends of the transcript; they will show more reads compared to the middle of the transcript.

So there are a lot of different types of biases that we will discuss in the subsequent classes, ok? So this is a major challenge again, right? Given that some regions of the transcript have a very high number of reads and other regions have a low number of reads, how do you actually assemble the full transcript? And the bigger question is: how do you actually identify the isoforms? And this is the next point, right? So isoforms are present for a transcript, right, and this could be present for many genes depending on the organism that you are working with.

And what you have to do is identify which isoforms are being expressed, ok? This is a challenge given the short-read data because you will get very few reads that map across splice junctions, right? So why am I focusing on these reads that map across splice junctions? Because these reads will tell us which exons are being connected, right? So splice junctions will tell us whether some exons are connected or whether some exons are skipped. So these reads that map to the splice junction are very important for the identification of isoforms.

Whereas in long-read data, this is slightly less challenging, right? You can get many reads that map across the splice junctions, and hence you can identify which isoforms are being expressed in the cell or in certain conditions, etcetera. On top of that, you have sequencing errors, right? So again, we have already talked about these sequencing errors and what types of errors you get on different platforms, and this will be there. And this is something that we will have to take into account when you are doing the transcriptome assembly, ok?

So once you understand, right, the different challenges that are present when you want to do the transcriptome assembly, we will now discuss some of the assembly tools and algorithms. So when I discuss the tools and algorithms, I will just mention the tools and briefly mention the algorithms. We will not go into the details of these algorithms because it will not be possible to discuss all of them in a short period of time. If you are interested, of course, have a look. I will give all the references in the presentation.

So please have a look into the algorithms, and I will just mostly highlight the algorithms and the tools that are used for these processes. So let's start discussing these tools. So many of these assembly algorithms or assemblers—right, these are the tools—use something called the deep brain graph assembly approach, ok? And this is an approach that is also used for genome assembly, ok? So I will not go into this deep brain graph assembly approach because we will discuss this when we talk about genome assembly approaches, right?

So there are different approaches for genome assembly, but the predominant one that we use today is the deep brain graph assembly approach. We will talk about these algorithms, these approaches, their advantages, their limitations, etc. when we talk about the genome assembly process, ok? So it will be enough to know that they use this deep brain graph approach, and this is a very efficient approach that works very well for genome assembly. So this is for de novo assembly, and here are some tools that are available.

Of course, there are many other tools available as well. So one is Trinity, then it is a very popular one, then you have rnaSPAdes, then you have SOAPdenovo-Trans, ok? So again, as I mentioned, I am just mentioning some of them; there are other tools out there. And as I mentioned, this is a really challenging process—the assembly of the transcriptome—because you have many fragments involved as well as the non-uniformity in coverage across the transcriptome and across transcripts. So this makes the whole process very challenging, and this is something you need to keep in mind, ok?

So I just mentioned the tools, and as I mentioned, this used deep brain graph assembly. So here is the paper describing the tool Trinity, ok? So it talks about the process of this assembly, etc., and you can also get the code in this link and the tool in this link, and if you want to use it, you can go there, download, install, and use this tool, ok?

So have a look. We are not going to go into the algorithm, as I mentioned, right? The algorithm is deep brain graph assembly, which we will discuss a little later. Here is the next tool, which is rnaSPAdes. So this is again a de novo transcriptome assembler, and this is used for RNA-Seq data transcriptome assembly. There is an equivalent tool called SPADES, right? SPADES without the RNA part in the first, and that is actually used for genome assembly.

So as you see, these are actually adaptations of genome assembly algorithms into transcriptome assembly, and we have another tool, as I mentioned, right? This is again a de novo transcriptome assembly with short reads. It is called So de Novo Trans, and again, you can go and have a look. So we will discuss some of the tools that also do assembly and can quantify transcript abundance.

So one such tool is a stringTie, right? So this also helps in improving the reconstruction of transcriptomes, right? But compared to the previous ones, this is a tool that does assembly based on a reference, ok? So you get reference-guided assembly. So you have to have the reference genome present, and in addition to the assembly of the transcripts, this tool can also quantify the abundance of transcripts present in the cell. So what you will see is that many of these tools that we discussed can do a lot of these different things, and there might be some overlap. So it is not strictly defined that this tool will do just quantification; this tool will do just assembly.

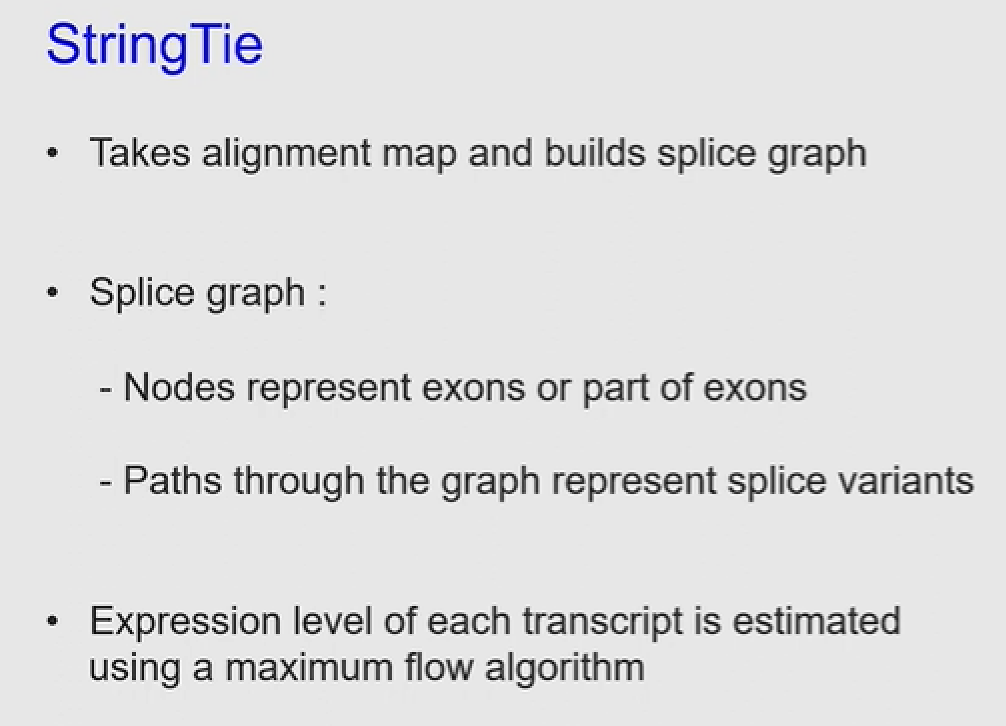
There are also tools that can actually combine two tasks or three tasks together, as we will see as we go into this discussion. So how does this tool work? So I will mention this very briefly. I have mentioned that this is a reference-guided assembly. So you have the reference genome, ok? So when you have a reference genome, you can have the alignment, right? The reads will be aligned against the reference genome, and we know the alignment map, right?

So in the form of a BAM file or SAM file, we know the alignment map. So stringTie takes this alignment map and builds something called a splice graph, ok? So, what is this splice graph? So in the splice graph, the idea is actually very simple. So you have a graph, which means you have nodes and edges, right? So if you are familiar with graph theory, you will have edges and nodes.

So in the case of a, we will have these nodes like this, right, and we have these edges between nodes, ok? So these circles are actually nodes, and then you have these edges. Now these edges can be directed or undirected, ok? So in the case of directed graphs, this is a directed graph.

In the case of undirected graphs, this is an undirected graph, ok? So let us not go into a lot of details, but this will be enough, right? So for these nodes, there are nodes and edges, ok? So what these nodes represent in the splice graph is that they represent exons or parts of exons, ok? So you have seen, if you remember this image that I showed you, that you have these axon 1, axon 2, and axon 3, right? So if you have this, axon 1 is one node, axon 2 is one node, and axon 3 is one node, right?

So these nodes represent these exons or parts of exons, and then you have these edges between these nodes, ok? And if you find a read, right, so let us say it goes through a splice junction, you can have an edge between some of these nodes, right? So, for example, axon 1 to axon 3 Now if you have these edges between axon 1 and axon 3, that means this is a splice variant that is present in your data, ok? So if you see reads, right, that show that map to the splice junction between axon 1 and axon 3, you will add an edge to the graph, and this will mean that this splice variant will be present, right?



StringTie

- Takes alignment map and builds splice graph

- Splice graph :

  - Nodes represent exons or part of exons

  - Paths through the graph represent splice variants

- Expression level of each transcript is estimated using a maximum flow algorithm

So you can now imagine this as a much more complex example, right? Instead of just 3 exons, you can have 5 exons and 8 exons, and you have these connections, ok? And each connection represents a splice variant, ok? So maybe you can traverse through this splice graph, right, from axon 1 to axon 2, maybe to axon 5, then to axon 8, this would be a splice variant, right, so where you have this axon 1, axon 2, axon 5, and axon 8,

So this is again kind of an isoform, right? So you can actually use this splice graph to identify which variants are present in your sample or in your data. Once you have built this splice graph, this method actually uses something called a maximal flow algorithm, which we will not discuss again, and it actually estimates the abundance of these transcripts. For each of these transcripts, there would be an abundance value that is estimated from this splice graph using something called a maximal flow algorithm. Now, I can also quantify expression without doing any assembly, right? So this is something you can say, ok, we do not do any assembly, just quantify, right, based on the data that is present because we are giving the alignment map.

So you can simply say just quantify, and you will see there are tools that will not do assembly; they will simply go through mapping and then quantification, ok, without doing any assembly. So seriously, I can also consider annotation in GFF or GTF format. So we have talked about this annotation, right? When we are talking about variant annotation, So this is also important for transcriptome assembly because you want to know, right, which gene contains how many reads?

So you need this annotation of genes, coding regions, etc. And it can also compare with known annotations, generate statistics, etc. So you can explore, right, this tool and see what features are available in it. Again, it is impossible to cover everything in the class. So I will encourage you to go and check all the resources that are available—the options, the tools, etc.

How do you run them, etc., all these things? So the next step is once you have done the assembly, right? So what you want to do is do something called quality control, ok? This is something that is also done for genome assembly, and similarly, we also have to do this for transcriptome assembly. Now what do we check actually? What can you check that will tell us, okay, this is a good quality transcriptome assembly, similar to that we check in the case of genome assembly? So one of the things we check is something like sequence length and fragmentation pattern, ok? So in the case of genome assembly, let us say, in an organism, we have, let us say, 16 chromosomes.

So what you want is that these assembled fragments, right? They should be around 16 in the ideal scenario, so we have complete assembly; we get 16 fragments, 16 contigs, which represent 16 chromosomes, ok? So similarly, here we have an idea about the transcripts. We are working with an organism where we know probably around 10,000 genes are expressed, let us say, and if we see that there is a huge number—much more than 10,000 fragments—that are obtained after assembly, this might mean that the assembly process was not very good, or there was not enough data, right, for the assembler to actually do the complete assembly. So this is what we can assess, right? So if you have too many very short fragments, this could be an indication that the assembly was not very successful, ok? And this could be due to sequence quality or there could be some issue with the assembly process itself because of some peculiarity of the data. The assembly process did not work very well. And another measure that we can also check is the fraction of reads that are used in the assembly. So if the assembler used the most of the reads, that is actually a good sign, right? So the assembler could utilize most of the data for the assembly process, right? So that is why if you have a much higher utilization, that is a good sign that the assembly actually worked very well, ok?

So the next step, once you have done the assembly, you have kind of evaluated the assembly quality. The next step is to do the quantification of transcript abundance. So how do you actually quantify the transcript abundance? So this is the process where you actually convert this mapping data to abundance data, and this process can sometimes go through the transcript assembly process, as you have seen. What we get after this step is something called raw count data, ok, and this is a very important step in estimating and comparing gene expression levels, ok. This is a very important step, and if you look at the databases, you will see a lot of data is deposited in terms of raw count data because this is the starting point for a lot of downstream analysis, ok? So we will talk about some of the tools again, at least one tool that we use for this kind of analysis, and of course, there are many more that we will discuss in the next class.

So one of the tools that is very popular is called Cuff Links, and this is because, again, it can do transcript assembly, then it can do abundance, as well as something called differential gene expression analysis. So we will talk about differential gene expression analysis later. So as you can see, this can do a lot of these things together, ok? Again, we will not go into a lot of details. Here is the link to the paper and also the link to the tool, which you can actually explore and see all the details of what it can do.

So very briefly, what it does is something called build a something called an overlap graph from read data, and how do you build that overlap graph? Again, remember the graph theory: you have nodes and edges, and here it says that sequences that overlap are connected by edges, right? So the idea is that the reads that show overlap are probably coming from the same transcript, right? They are part of the same transcript because they are showing sequence overlap; right, at least part of them are identical in sequence, ok?

So that is why they are presented in graphs. Again, this is a very similar idea. Again, we will illustrate a bit more when we talk about genome assembly. Now what it does next is actually identify the minimum path that needs to be traversed to generate transcripts, ok? Again, it is a graph traversal algorithm, and then followed by that, there is something called a maximum likelihood abundance estimation from the graph itself. So again, we are not going into all the statistical details of what the maximum likelihood abundance estimate is.

I will just mention that this is what it does. In case you are interested, please have a look at how this maximum likelihood method works. This is a method that is widely used in statistics and in a lot of applications. So very briefly, here are some functions that you will find in couplings. For example, coupling cups are used for transcript assembly.

You also have the tool for differential expression analysis. You also have different visualization tools, etc. Again, I urge you to explore the website. Now what are the challenges for transcript abundance quantification? So we will end with some of these discussions, right? So one of the things is that many reads do not span over the splice junctions, right? So as we have seen in the case of short-read data, they are mostly mapping to the exon regions, right?

Most of them will map to the exon regions. So then using these reads, right, in the case of short read sequencing, how can we assign these two isoforms? How do you identify isoforms? This is a huge challenge for us. And how do you actually assign gene expression? Do you say isoform expression or overall gene expression, right? This is something that could be interesting depending on the research question that you have, okay? For example, you want to identify which isoforms are expressed or whether some isoforms show higher expression in certain conditions or certain treatments, etc. So in those cases, you want to do the isoform expression and not just the overall gene expression.

And also, again, as I said, you want to look at differences in isoform expression across experiments, okay? So I will just mention some of the tools that we used for transcript abundance. So Cufflinks is one such tool. Again, it does a lot of this assembly, then quantification of abundance, right?

This is again very similar to the stringTie that we just discussed. There are dedicated tools now that will actually do this transcript abundance estimation, right? And from the alignment map, we have something called htSeq-count, or feature counts. We will discuss them in the next class. Cufflinks we have just mentioned

And on top of that, you also have alignment-free methods, okay? Again, we will discuss in the next class what these methods are. So here are the references that we have used for this class. And to conclude, we have talked about transcriptome assembly, and this is a challenging problem due to several issues that we have mentioned. We have nonuniform coverage; we have isoforms; we also have sequence signals, etc.

We talked about two types of assembly. One is de novo assembly, where there is no reference sequence, and then we have the reference-guided assembly. And as we have seen, de novo assembly is much more challenging than reference-guided assembly, and we have discussed some of the tools that can do de novo assembly. We have not discussed algorithms, which we will discuss later when we talk about genome assembly. And then we have talked about transcript abundance quantification, and this is a very important step in gene expression analysis where we actually get the raw count data. As I mentioned, this is the data that we will see when you look at gene expression data across different studies, okay?

This is the data file that will be available for most datasets, okay? Because this is the starting point for all downstream analysis. And finally, we have talked about some of the tools for assembly and quantification, right? We briefly touched upon them, and I, of course, encourage you to explore more on these topics. Thank you.