University of Nebraska - Lincoln

DigitalCommons@University of Nebraska - Lincoln

Computer Science and Engineering: Theses, Dissertations, and Student Research

Computer Science and Engineering, Department of

5-2018

Consensus Ensemble Approaches Improve De Novo Transcriptome Assemblies

Adam Voshall

University of Nebraska-Lincoln, avoshal1@unl.edu

Follow this and additional works at: https://digitalcommons.unl.edu/computerscidiss

Part of the Computer Engineering Commons, and the Computer Sciences Commons

CONSENSUS ENSEMBLE APPROACHES IMPROVE *DE NOVO*

TRANSCRIPTOME ASSEMBLIES

by

Adam Voshall

A THESIS

Presented to the Faculty of

The Graduate College at the University of Nebraska

In Partial Fulfillment of Requirements

For the Degree of Master of Science

Major: Computer Science

Under the Supervision of Professor Jitender S. Deogun

Lincoln, Nebraska

May, 2018

CONSENSUS ENSEMBLE APPROACHES IMPROVE *DE NOVO*

TRANSCRIPTOME ASSEMBLIES

Adam Voshall, M.S.

University of Nebraska, 2018

Advisor: Jitender S. Deogun

Accurate and comprehensive transcriptome assemblies lay the foundation for a range of analyses, such as differential gene expression analysis, metabolic pathway reconstruction, novel gene discovery, or metabolic flux analysis. With the arrival of next-generation sequencing technologies it has become possible to acquire the whole transcriptome data rapidly even from non-model organisms. However, the problem of accurately assembling the transcriptome for any given sample remains extremely challenging, especially in species with a high prevalence of recent gene or genome duplications, those with alternative splicing of transcripts, or those whose genomes are not well studied. This thesis provides a detailed overview of the strategies used for transcriptome assembly, including a review of the different statistics available for measuring the quality of transcriptome assemblies with the emphasis on the types of errors each statistic does and does not detect and simulation protocols to computationally generate RNAseq data that present biologically realistic problems such as gene expression bias and alternative splicing. Using such simulated RNAseq data, a comparison of the accuracy, strengths, and weaknesses of seven representative assemblers including *de novo*, genome-guided methods shows that all of the assemblers individually struggle to accurately reconstruct the expressed transcriptome, especially for alternative splice forms. Using a consensus of several *de novo* assemblers can overcome many of the weaknesses of individual

assemblers, generating an ensemble assembly with higher accuracy than any individual

assembler.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

Transcriptome assembly from high-throughput sequencing of mRNA (RNAseq) is a

powerful tool for detecting variations in gene expression and sequences between

conditions, tissues, or strains/species for both model and non-model organisms (1, 2).

However, the ability to accurately perform such analyses is crucially dependent on the

quality of the underlying assembly (3). Especially for the detection of sequence

variations, but also for isoform detection and transcript quantification, mis-assembly of

genes of interest can increase both the false positive and false negative rates, depending

on the nature of the mis-assembly (4). These problems are exacerbated in non-model

organisms where genomic sequences that can be used as the references, if available at all,

are sufficiently different than those from the individuals sequenced (5).

Transcripts can be mis-assembled in several ways (6). Two of the most drastic

assembly errors are fragmentation, where a single transcript is assembled as one or more

smaller contigs, and chimeras, where a contig is assembled using part or all of more than

one transcript. Fragmentation errors tend to result from fluctuations in the read coverage

along a transcript, with the breaks in the transcript sequence occurring in regions that

have lower coverage. By contrast, chimera errors often occur because of ambiguous

overlaps within the reads, coupled with algorithms that choose the longest possible contig

represented by the data, or by adjacent genes on the genome being merged. Both of these

types of errors can have major impacts on transcriptome assemblies for gene

identification. Small (single or few) nucleotide alterations to the contig sequence also

happen as mis-assemblies. Sequence mistakes are often the result of mis-sequenced

reads, but can also result from ambiguity for highly similar reads, both from

heterozygous genes and from duplicated genes. In some cases, these errors can shift the reading frame for the contig, which can have significant impacts on the translated protein sequence. Finally, transcripts can be mis-assembled when real alternative transcripts are collapsed into a single contig (6).

The following sections will first review strategies used for transcriptome assembly as well as how their performance can be assessed. Chapter 3 presents an actual performance analysis of representative methods using a simulated human transcriptome and RNAseq.

1.1 Transcriptome assembly strategies

De novo assemblers

De novo assemblers generate contigs based solely on the RNAseq data (7-13). Most of the *de novo* assemblers rely on de Bruijn graphs generated from kmer decompositions of the reads in the RNAseq data (14). The reads are subdivided into shorter sequences of length k (the kmers) of a given length, and the original sequence is reconstructed by the overlap of these kmer sequences. One major limitation of the de Bruijn graphs is the need for a kmer to start at every position along the original sequence in order for the graph to cover the full sequence (13). This limitation creates a tradeoff in regard to the length of the kmers. Shorter kmers are more likely to fully cover the original sequence, but are more likely to be ambiguous, with a single kmer corresponding to multiple reads from multiple transcripts. While by using longer kmers such ambiguity can be avoided, those kmers may not cover the entire sequence of some transcripts causing e.g. fragmented assembly. Consequently, each transcript, with its unique combination of expression level (corresponding to the number of reads in the RNAseq data generated from that transcript) and sequence uniqueness, will have a different best kmer length for its assembly (15). As

a result, even using the same *de novo* assembly algorithm, performing two assemblies with different kmer lengths will generate a different set of contigs, and will inevitably have variations in which of the original transcripts were correctly assembled (16).

Examples of popularly used *de novo* assemblers include idba-Tran (9), SOAPdenovo-Trans (8), rnaSPAdes (12), and Trinity (7). Idba-Tran is unique among these *de novo* assemblers, as it runs individual assemblies across a range of kmer lengths and merges the results to form the final prediction. The remaining assemblers use only the results of a single kmer length. For SOAPdenovo-Trans and Trinity, a kmer length needs to be chosen (default kmer: 23 and 25, respectively), while rnaSPAdes dynamically determines the kmer length to be used based on the read data. While all of these tools use the same fundamental strategies to construct, revise, and parse the de Bruijn graph for the assemblies, each method uses different thresholds and different assumptions to make decisions. These differences lead to different subsets of transcripts being correctly assembled by each method. An example of how these tools produce different sets of contigs is shown in Section 3.1.

Genome-guided assemblers

Genome-guided assemblers avoid the ambiguity of kmer decompositions used in de Bruijn graphs by instead mapping the RNAseq data to the reference genome. In order to account of introns, mapping of the reads for genome-guided assembly needs to allow them to be split, where the first part of the read maps to one location (an exon), and the other half maps to a downstream location (another exon). This mapping is done by split-read mappers such as TopHat (17), STAR (18), HISAT (19), or HPG-aligner (20). Each

of these methods map the reads slightly differently, which may impact the quality of subsequent assembly.

This read mapping greatly reduces the complexity of transcript assembly by clustering the reads based on genomic location rather than relying solely on overlapping sequences within the reads themselves (3). However, this approach still has some major drawbacks. The most obvious drawback is that genome-guided assemblers require a reference genome, which is not available for all organisms. The quality of the reference genome, if it is available, also impacts the quality of the read mapping and, by extension, the analysis. This impact is particularly noteworthy when genes of interest contain gaps in the genome assembly, preventing the reads necessary to assemble those genes from mapping to part or all of the transcript sequence. Ambiguity occurs also when reads map to multiple places within a genome. How the specific algorithm handles choosing which potential location a read should map to can have a large impact on the final transcripts predicted (6). This problem is expounded when working with organisms different from the reference, where not all of reads map to the reference without gaps or mismatches.

Examples of popularly used genome-guided assemblers include Bayesembler (21), Cufflinks (22), and StringTie (23). While each of these methods uses the mapped reads to create a graph representing the splice junctions of the transcripts, how they select which splice junctions are real differs fundamentally. Cufflinks constructs transcripts based on using the fewest number of transcripts to cover the highest percentage of mapped reads. StringTie uses the number of reads that span each splice junction to construct a flow graph, constructing the transcripts based in order of the highest flow. Bayesembler constructs all viable transcripts for each splice junction and uses a Bayesian likelihood

estimation based on the read coverage of each potential transcript to determine which combination of transcripts is most likely. Due to these fundamentally different approaches, each of these tools produces different sets of transcripts from the same set of reads. An example of assemblies produced by these methods and how the assembled contigs differ is described in Section 3.2.

Ensemble approach

While a core set of transcripts are expected to be assembled correctly by many different assemblers, many transcripts will be missed by any individual tool (24) (also see Section 4). Through combining the assemblies produced by multiple methods, ensemble assemblers such as EvidentialGene (25) and Concatenation (26) attempt to address the limitations of individual assemblers, ideally keeping contigs that are more likely to be correctly assembled and discarding the rest. Both of EvidentialGene and Concatenation filter the contigs obtained from multiple assemblers (usually *de novo*) by clustering the contigs based on their sequences, predicting the coding region of the contig, and using features of the overall contig and the coding region to determine the representative sequence for each cluster. EvidentialGene recommends using several different tools across a wide range of kmer lengths. It uses the redundancy from multiple tools generating nearly identical sequences and clusters them, scores the sequences in each cluster based of the features of the sequence (e.g. lengths of the 5′ and 3′ untranslated regions), and returns one representative sequence from each cluster (keeping also some alternative sequences). In contrast, Concatenation recommends using only three assemblers, with one kmer length each. This method merges nucleotide sequences that are identical or perfect subsets, only filters contigs with no predicted coding region.

These approaches greatly reduce the number of contigs present by removing redundant and highly similar sequences. However, there is no guarantee that the correct representative sequence is kept for a given cluster or that each cluster represents one unique gene. Because they require multiple assemblies to merge, they also come at a far greater computational cost. An example of how these ensemble assembly strategies perform compared to individual *de novo* and genome-guided methods is shown in Section 3.3.

Third generation sequencing

All of the above methods primarily use short but highly accurate reads from Illumina sequencing for assembly, with or without a reference. With the rise of third-generation sequencing technologies from Pacific Biosciences (PacBio SMRT) and Oxford Nanopore Technologies (ONT MinION), it is becoming possible to sequence entire mRNA molecules as one very long read, though with a high error rate (27). The ability to sequence the entire mRNA molecule is especially beneficial for detecting alternative splice forms, which remain a challenge for short-read only assembly, and potentially for more accurate transcript quantification if there is no bias in the mRNA molecules sequenced.

While many tools exist to perform genome assemblies using either these long reads alone or by combining long reads and Illumina reads, at present no short read transcriptome assemblers take advantage of long-reads in transcriptome assembly. If these long reads can be sufficiently error-corrected (e.g. 28, 29), they can be used for a snapshot of the expressed transcriptome, without requiring assembly or external references (30, 31). Alternatively, after an independent *de novo* assembly of short reads,

the long reads can be used to confirm alternative splice forms present in the assembly (32). The long reads can be also mapped to a reference genome similar to the split-read mapping methods used for genome-guided short-read assemblers discussed above (27, 33-35). With their accuracy increasing, in the future long reads can be used more to improve transcriptome assembly quality.

1.2 Performance metrics used for transcriptome assembly

In this section discusses commonly used metrics to assess the quality of transcriptome assemblies.

Metrics based on contig count and lengths

The most straightforward assembly metrics are those based on the number and lengths of the sequences produced (36). The number of sequences can be presented either or both of:

- the number of contigs
- the number of scaffolds

where for contigs no further joining of the sequences is performed after assembly, and for scaffold contigs that have some support for being from the same original sequence are combined together with a gap sequence between them.

Several different statistics are available for presenting the lengths of the sequences (either contigs or scaffolds). The most commonly reported metrics are:

- minimum length (bp): the length of the shortest sequence produced
- maximum length (bp): the length of the longest sequence produced
- mean length (bp): the average length of the sequences produced

- median length (bp): the length where half of the sequences are shorter, and half of the sequences are longer

- N50 (bp): a weighted median where the sum of the lengths of all sequences longer than the N50 is at least half of the total length of the assembly

- L50: the smallest number of sequences whose combined length is longer than the N50

Additional metrics similar to N50 (e.g. N90) based on different thresholds are also used.

For genome assemblies where the target number of sequences is known (one circular genome plus any smaller plasmids for prokaryotic organisms and the number of chromosomes for eukaryotic organisms), these metrics provide an estimate for the thoroughness of the assembly (36). For instance, in prokaryotic assemblies, the vast majority of the sequence is expected to be in one long sequence, and having many shorter sequences indicates fragmentation of the assembly (15). In this context, longer sequences (e.g. larger N50) tend to indicate higher quality assemblies. For transcriptome assemblies, however, the length of the assembled contigs varies depending on the lengths of the transcripts being assembled. For the human transcriptome, for example, while the longest transcript (for the gene coding the Titin protein) is over 100kb, the shortest is only 186bp, with a median length of 2,787bp (37). Emphasizing longer contigs also rewards assemblers that over-assemble sequences, either by including additional sequence incorrectly within a gene, or by joining multiple genes together to form chimeric contigs. Therefore, for transcriptome assembly, metrics based on contig lengths do not necessarily reflect its quality.

Metrics based on coded protein similarity

Rather than focusing on the number or length of the sequences produced by the assembly, performing similarity searches with the assembled sequences can provide an estimate of the quality of the contigs or scaffolds (24, 38). Typically, the process consists of either similarity searches against well annotated databases (such as the protein datasets of related genomes or targeted orthologs, the BLAST non-redundant protein database (39) or the UniProt/Swiss-Prot database (40)), conserved domain search within the contig sequence that determines the potential function of the gene (such as PFAM or Panther (41, 42)), or a search against a lineage specific conserved single-copy protein database (such as BUSCO (43)). These similarity searches are usually performed on the predicted protein sequences for the contigs (e.g. using GeneMarkS (44)), but can also be performed directly from the assembled nucleotide sequences using BLASTX where translated nucleotide sequences are used to search against a protein database (38). If the organism being sequenced is closely related to a model organism with a well-defined transcriptome, nearly all of the contigs that are not erroneously assembled and code proteins should have identifiable potential homologs in the database. If a large percentage of the contigs do not have similar proteins identified in the database, there is a high probability that the sequences are incorrectly assembled, regardless of the length of the sequences. By performing similarity searches, over assemblies can be also detected as large gaps in the alignment between the query and the hits or contigs that cover more than one gene. As protein sequence annotations are necessary for most downstream analyses, they also provide a convenient metric without the need for additional, otherwise unnecessary analyses.

Despite these advantages, there are some limitations to using protein-similarity based metrics for assembler performance. First, the more divergent the organism being sequenced is from the sequences in the database searched and the more species-specific genes in the transcriptome, the lower the percentage of contigs with hits will be. This can result in some organisms appearing to have a lower quality assembly solely due to their divergence from those well represented in the databases. By extension, assemblies that recover more transcripts whose coded proteins have few similar sequences in the database will appear worse than assemblies that only recover conserved genes. This limitation can be somewhat mitigated by comparing only genes that are universally single-copy across different species, which are more likely to be conserved and similar enough to be identified. This is the strategy used in BUSCO (43). However, this comparison at best uses only a subset of the assembled contigs. Second, and more problematic, this metric rewards assemblies that artificially duplicate conserved genes with only small differences in the nucleotide sequence. In the extreme, this can result in several times as many contigs in the assembly than were present in the actual transcriptome, but with nearly all of the contigs coding conserved protein sequences. This is particularly an issue when the analysis depends on identifying the gene copy numbers in the assembly. It also has a large impact on the accuracy of contig quantification and differential expression analyses (45).

Assembly metrics based on benchmark transcriptomes

The only way to overcome the limitations of the metrics described in the previous sections is to compare the assembly output against a benchmark transcriptome where correct sequences of all transcripts are known. When an RNAseq data generated from a

well-established model organism is used for assembly, many of correctly assembled contigs can be identified. However, variability in the transcriptome among e.g. cell types limits the amount of information that can be gained for incorrectly assembled contigs. It is also not possible to determine whether sequences from the reference that are missing from the assembled transcriptome are due to assembly errors, or whether they were not expressed in the library sequenced. Transcriptome sequences may also vary between the individual under study and the reference. Such variations can mask assembly errors that affect the contig sequences. Although this limitation can be mitigated by sequencing an individual that is genetically identical to the reference, it severely limits the types of organisms that can be used for the benchmark.

To comprehensively assess all of the assembly errors, RNAseq data needs to be obtained from a transcriptome where all transcript sequences and expression patterns are known. Ideally, such a benchmark transcriptome would be synthetically produced and sequenced using standard protocols. However, currently no such synthetic mRNA library exists. An alternative approach is to simulate the sequencing of a given benchmark transcriptome. There are several tools that can generate simulated reads modelling short Illumina reads (46, 47) and/or long third-generation sequencing reads such as PacBio SMRT and ONT MinION (48, 49). These tools typically either focus on identifying the statistical distribution of reads across the sequences and errors within the reads, as is the case for RSEM (46), PBSIM (48), and Nanosim (49), or by attempting to reconstruct each step of the library preparation and sequencing pipeline, mimicking the errors and biases introduced at each step, as is the case for Flux Simulator (47).

Using simulated RNAseq data with a known transcriptome as a benchmark gives

the most detailed and close to true performance metric for assemblies. Specifically, this

strategy allows the quantification of each of the following categories:

correctly assembled sequences (true positives or TPs)

sequences that are assembled with errors (false positives or FPs)

sequences in the reference that are missing from the assembly (false negatives or FNs)

"Correctness" and "incorrectness" (or error) can be defined using varying degrees of

sequence similarities. Using the strictest threshold, a contig sequence is assembled

"correctly" only if the entire nucleotide or protein sequence is identical to a reference

transcript. All other contigs found in the assembly, including those whose sequences have

no similarity in the reference transcriptome (missing contigs), are considered to be

assembled "incorrectly" (FPs) regardless of the similarity against the reference sequences.

Note that true negatives (TNs) can be counted only if the assembly experiments are done

including reads that are derived from transcripts that are not part of the reference

transcriptome (negative transcripts). Using these categories, following assembly metrics

can be calculated:

- Accuracy $= \frac{TP+TN}{TP+FP+TN+FN}$

- Sensitivity (or Recall) $= \frac{TP}{TP+FN}$

- Specificity $= \frac{TN}{TN+FP}$

- Precision $= \frac{TP}{TP+FP}$

- F-measure (or F_1 score) $= \frac{2(TP)}{2(TP)+FP+FN}$

- False Discovery Rate (FDR) $= \frac{FP}{FP+TP}$

Often in an RNAseq simulation, negative transcripts are not included; hence TN cannot be counted. In such cases, the accuracy can instead be calculated using an alternative metric:

- Accuracy* $= \frac{TP}{TP+FP+FN}$

Despite the added benefits of simulation for measuring the performance of assemblers, these metrics assume that the simulation accurately reflects the nature of real RNAseq data. Differences in the distribution of reads or errors between the simulations and real data can impact the relative performance of the assemblers. Assemblers that perform well on simulated data may perform poorly on real data if those assumptions are not met. Consequently, great care must be taken to ensure that the simulated data captures the features of real data as accurately as possible to best characterize the performance of different assembly strategies.

1.3 Contribution of Thesis

This thesis contributes to the field of transcriptome assemblies using RNAseq data in three key ways. First, it presents the development of an RNAseq simulation pipeline that generates a realistic benchmark library to measure the performance of transcriptome assemblers. Second, it reports a comparative analysis of seven commonly used genome-guided and *de novo* assemblers using the benchmark libraries generated using this RNAseq simulation. Third, it introduces a consensus method for ensemble transcriptome assemblies to generate a more accurate *de novo* transcriptome assembly than any individual methods, without the need for an external reference

sequence. Taken together, these contributions show the current state of transcriptome

assemblies and highlight strategies to improve assembly accuracy.

Chapter 2: Materials and Methods

2.1 Benchmark transcriptome and simulated RNAseq

RNAseq data sets were generated by Flux Simulator (47) using the hg38 human genome (available at https://genome.ucsc.edu/cgi-bin/hgGateway?db=hg38) as the reference. The older hg19 human genome (available at http://genome.ucsc.edu/cgi-bin/hgGateway?db=hg19) was also used as an alternate reference genome to assess the impact of using a different reference with genome-guided assemblers. The gene expression profile was generated by Flux Simulator using the standard parameters from the hg38 reference genome and transcriptome model. Approximately 250 million pairs of reads were generated with the given expression model with no PolyA tail. The simulated library construction was fragmented uniformly at random, with an average fragment size of 500 ($\pm$ 180) nucleotides (nt). Because reads overlapping within read pairs can cause problems for some assemblers, fragments shorter than 150nt were removed. The simulated sequencing was performed using paired-end reads of read length of 76nt using the default error model based on the read quality of Illumina-HiSeq sequencers. Note that only reference transcripts with full coverage of RNAseq data were included in the benchmarking, as transcripts without full coverage cannot be correctly assembled as a single contig. This filtering removed 2,700 transcripts expressed in the benchmark transcriptome, leaving 14,040 unique sequences derived from 8,557 genes (5,309 have no alternative splicing, on average 1.64 transcripts per gene, ranging up to 13 isoforms per gene).

The read pairs generated by Flux Simulator were quality filtered using Erne-filter version 2.0 (50). The reads were filtered using ultra-sensitive settings with a minimum

average quality of q20 (representing a 99% probability that the nucleotide is correctly reported). The filtering was performed in paired-end mode to ensure that both reads of the pair were either kept or discarded concurrently to keep the pairs together. The remaining reads were normalized using Khmer (51) with a kmer size of 32 and an expected coverage of 50x. The normalization was also performed in paired-end mode to maintain pairs.

Chapter 3: Results

3.1 De novo assemblies

This section compares the performance among four *de novo* transcriptome assemblers:

idba-Tran version 1.1.1 (9), SOAPdenovo-Trans version 1.03 (8), rnaSPAdes version

3.11.0 (12), and Trinity version 2.5.1 (7), using the simulated human RNAseq data set as

described in the previous section. The results of the assemblies were compared against

the benchmark transcriptome. As shown in Table 3.1, all of the tools underestimated the

number of transcripts present, generating fewer contigs than the number of transcripts

expected (14,040). The best performing tool among the four compared was Trinity with

the most correct (5,782) and the highest correct/incorrect ratio (C/I = 0.8432). However,

even with Trinity, still only 41% (5,782/14,040) of transcripts in the benchmark were

correctly assembled; the remaining almost 60% of contigs either contained errors in the

sequence or were missed entirely. rnaSPAdes assembled the largest number of transcripts

(874 more unique transcripts compared to Trinity). The number of unique transcripts

generated, 13,513, is also the closest to the expected total number of transcripts (96% of

14,040). However, fewer of those sequences (36%) were correctly assembled than

Trinity, lowering the overall performance across all statistics than Trinity.

Performance statistics for each assembler is given in Table 3.2. Precision is a

measure of how likely an assembled contig is to be correct, and recall is a measure of

how likely the assembler is to correctly assemble a contig. In these terms, for assemblers

with high precision, the contigs produced are more likely to be correct, but the assembly

may miss a large number of sequences present in the sample. Conversely, assemblers

with a high recall correctly assemble more of the sequences present in the sample, but

may do so at the cost of accumulating a large number of incorrectly assembled contigs. In these statistics, both the modified accuracy score (Accuracy*; see Section 3.3) and the F_1 score are a measure of the number of correctly assembled contigs relative to the number of missing and incorrectly assembled contigs. FDR is the proportion of assembled reads that are incorrect. Based on these statistics, Trinity is the best performing *de novo* assembler with the highest precision, recall, accuracy* and F_1 score, and the lowest FDR, followed by rnaSPAdes then SOAPdenovo-Trans. Despite idba-Tran running multiple kmers and merging the results, it performed worst across every metric.

In Table 3.1, the result from pooling (taking the union of) the outputs of multiple runs of each assembler across a range of kmer lengths are also shown. With these pooled assemblies, the proportion of correctly assembled transcripts in the benchmark for Trinity increased from 41% to 46%, and for rnaSPAdes from 36% to 47%. However, the pooling process also accumulated several times more unique incorrect sequences than additional correct sequences recovered. For Trinity, the C/I decreased from 0.8432 to 0.3470, and for rnaSPAdes this ratio decreased from 0.5900 to 0.0621.

Although the four *de novo* assembly methods uses the same core approach, each assembler assembled a different set of sequences correctly (Figure 3.1A). Only a set of 5,331 contigs were correctly assembled by all of the four *de novo* assemblers with at least one kmer length. Additional 813, 567, and 670 contigs were correctly assembled by at least three, at least two or only one of the assemblers, respectively. In contrast, the vast majority of the incorrectly, assembled contigs were produced by only one assembler (Figure 3.1B). For these contigs, 3,764 were produced by all four assemblers, while an

additional 2,692, 7,977 and 166,720 were produced by at least three, at least two or only one of the assemblers, respectively.

3.2 Genome-guided assemblies

This section compares the transcriptome assembly performance among three genome-guided assemblers: Bayesembler version 1.2.0 (21), Cufflinks version 2.2.1 (22), and StringTie version 1.0.4 (23). To demonstrate the impact of using different reference genomes on genome-guided transcriptome assemblies, using both of the hg38 as well as hg19 genomes as the references. Assembly assessment was done against the hg38 benchmark transcriptome.

Table 3.3 shows the performance of each of these tools in the two scenarios (RNAseq data and the reference genome were derived from the same or different individuals or strains). As observed with *de novo* methods, all of these genome-guided methods underestimated the number of transcripts present, even more severely than *de novo* methods. In terms of the number of contigs correctly assembled, StringTie performed slightly better than other two methods. All three methods had comparable percent correct (36-41%) and C/I (0.87-0.88). While none of the genome-guided assemblers produced as many correctly assembled contigs as the best performing *de novo* assembler (Trinity), proportions of correctly assembled contigs were higher with genome-guided methods (C/I = 0.87-0.88) than with the four *de novo* methods (C/I = 0.41-0.84). When the performance metrics are compared between the best performing *de novo* assembler (Trinity) and genome-guided assembler (StringTie) (Table 3.4), while both methods showed similar accuracy, StringTie (when using the same reference)

showed slightly higher precision, accuracy* and F_1 and lower FDR compared to Trinity, but a slightly lower recall. It reflects fewer FPs and FNs produced by StringTie.

As with the *de novo* assemblers, each of these tools correctly assembled a different set of transcripts (Figure 3.2A and C). When the assemblies were performed using the same reference as the simulation, all of the genome-guided tools correctly assembled a core set of 4,013 transcripts (Figure 3.2A). There were nearly a quarter as many (936) that were unique to only one genome-guided tool. When a different reference was used, the number of sequences correctly assembled by all of the tools dropped to 2,546 (Figure 3.2C). Similar to the *de novo* assemblers, most of the incorrectly assembled contigs produced by each of the genome-guided assemblers were produced by only one assembler regardless of the reference genome used (Figure 3.2B and D). For assemblies using the same reference genome, 2,013 incorrectly assembled contigs were produced by all of the tools, while an additional 2,382 and 7,546 were produced by any two or only one tool, respectively (Figure 3.2B). For assemblies using a different reference genome, 1,420 incorrectly assembled contigs were produced by all of the tools, while an additional 1,667 and 4,772 were produced by any two or only one tool, respectively (Figure 3.2D).

3.3 Comparison of de novo and genome-guided assemblers

While the overall statistics are comparable between the best *de novo* assemblies and the genome-guided assemblies using the same reference genome, these tools produced different sets of contigs. The overlap of correctly assembled contigs between the assemblers from *de novo* with pooled kmers lengths and the three genome-guided assemblers are shown in Figure 3.3A. All of the *de novo* assemblers and at least one genome-guided assembler correctly assembled 4,605 contigs. An additional 629 were

assembled by at least three *de novo* and at least one genome-guided assembler and 427

assembled by at least two *de novo* and at least one genome-guided assembler.

Conversely, 3,861 contigs were correctly assembled by all of the three genome-guided

assemblers and at least one *de novo* assembler, with 1,338 assembled by at least two

genome-guided assemblers and at least one *de novo* assembler (Figure 3.3B).

Additionally, these tools produced only 602 correctly assembled contigs that were not

predicted by any *de novo* assembly, while 1,514 sequences were correctly assembled by

at least one *de novo* assembly, but no genome-guided assemblies.

As with the individual assemblies, fewer incorrectly assembled contigs were

produced by all of the tools, and most are assembler specific (Figure 3.3C and D). In

particular, only 1,387 incorrectly assembled contigs were produced by all of the *de novo*

assemblers and at least one genome-guided assembler (Figure 3.3C), and only 1,593

contigs were produced all of the genome-guided assemblers and at least one *de novo*

assembler (Figure 3.3D). In contrast, 4,823 incorrectly assemblers were produced by at

least one genome-guided assembler but no *de novo* assemblers, and 176,397 incorrectly

assembled contigs were produced by at least one *de novo* assembler but no genome-

guided assemblers.

Overall, these results suggest that genome-guided assemblies provide relatively few

correctly assembled contigs relative to performing multiple *de novo* assemblies, even

when using the same reference genome. However, they produce far fewer incorrectly

assembled contigs than the pooled *de novo* assemblies. If the correctly assembled contigs

produced by each of the *de novo* assemblies can be retained while filtering out the

incorrectly assembled contigs, *de novo* assemblies can outperform all of the genome-

guided assemblies. This result forms the motivation of ensemble assembly strategies, discussed in the next section.

3.4 Ensemble assemblies

This section compares the two ensemble transcriptome assembly methods, EvidentialGene version 2017.03.09 (25) and Concatenation version 1 (26) using the simulated RNAseq data. The strategies for these assemblies followed the recommendations by each method. For EvidentialGene, the pooled results from all of the four *de novo* assemblies performed across the full range of kmer lengths (described in Section 3.1) were used. For Concatenation, the results of a single assembly each from idba-Tran (using kmer length of 50), rnaSPAdes (with default kmer selection), and Trinity (with default kmer length). These assemblers were chosen to match the assemblies used in (26), substituting the commercial CLC Assembly Cell (https://www.qiagenbioinformatics.com/products/clc-assembly-cell/) with freely available rnaSPAdes.

In addition to the two ensemble methods, we also included three "consensus" approaches taking the consensus of the pooled *de novo* methods. These consensus assemblies involve keeping all of the unique protein sequences produced by any two, three and four tools (named Consensus 2, Consensus 3 and Consensus 4, respectively). Note that Consensus 4 is a subset of Consensus 3, and Consensus 3 is a subset of Consensus 2.

The performance of these ensemble strategies is shown in Table 3.5. Both of EvidentialGene and Concatenation resulted in an over-estimation in the number of transcripts present. Interestingly, while Concatenation produced a larger total number of

transcripts (19,767) than EvidentialGene (19,177), ~2,300 of those sequences were redundant, leading to fewer unique sequences (17,497 by Concatenation). Additionally, Concatenation both kept more of the correctly assembled contigs from the individual *de novo* assemblies, and removed more of the incorrectly assembled contigs than EvidentialGene. These differences lead Concatenation to outperform EvidentialGene across every statistic (Table 3.6). The performance of the consensus approach varied based on the number of assemblers required.

Consensus 2 produced the most correctly assembled contigs of any method (6,711), but at the cost of more incorrectly assembled contigs than Concatenation (14,433). However, both Consensus 3 and Consensus 4 kept the majority of the correctly assembled contigs while reducing the number of incorrectly assembled contigs by roughly half or three quarters, respectively. Consensus 4 had highest precision (0.5861) and lowest FDR (0.4139) of any method, but the additional reduction in the number of correctly assembled contigs lead to Consensus 3 having the highest accuracy* (0.2998) and F_1 score (0.4613).

In Figure 3.4 all individual methods (both *de novo* and genome-guided) as well as ensemble methods are compared. Concatenation performed more poorly than Trinity despite the Trinity assembly forming part of the ensemble. In contrast, Consensus 3 kept more correctly assembled contigs than any individual assembly, with fewer incorrectly assembled than any approach except Consensus 4. This test highlights the weakness of ensemble assembly strategies to retain the incorrect version of a transcript, even if the correct version of the transcript exists in the individual assemblies. More robust methods,

such as the consensus approaches we showed, are needed to reliably improve over individual assemblies.
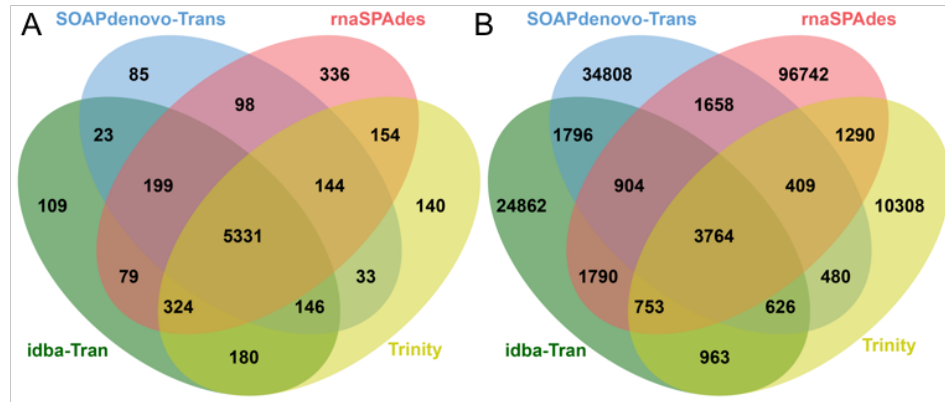
Figure 3.1: Venn diagrams showing the pooled sequences across all k-mers of each *de novo* assembler.

A) Correctly assembled sequences, where the protein sequence of the contig matches the protein sequence in the benchmark transcriptome. B) Incorrectly assembled sequences, where the protein sequence of the contig does not exactly match any protein sequence in the benchmark transcriptome.
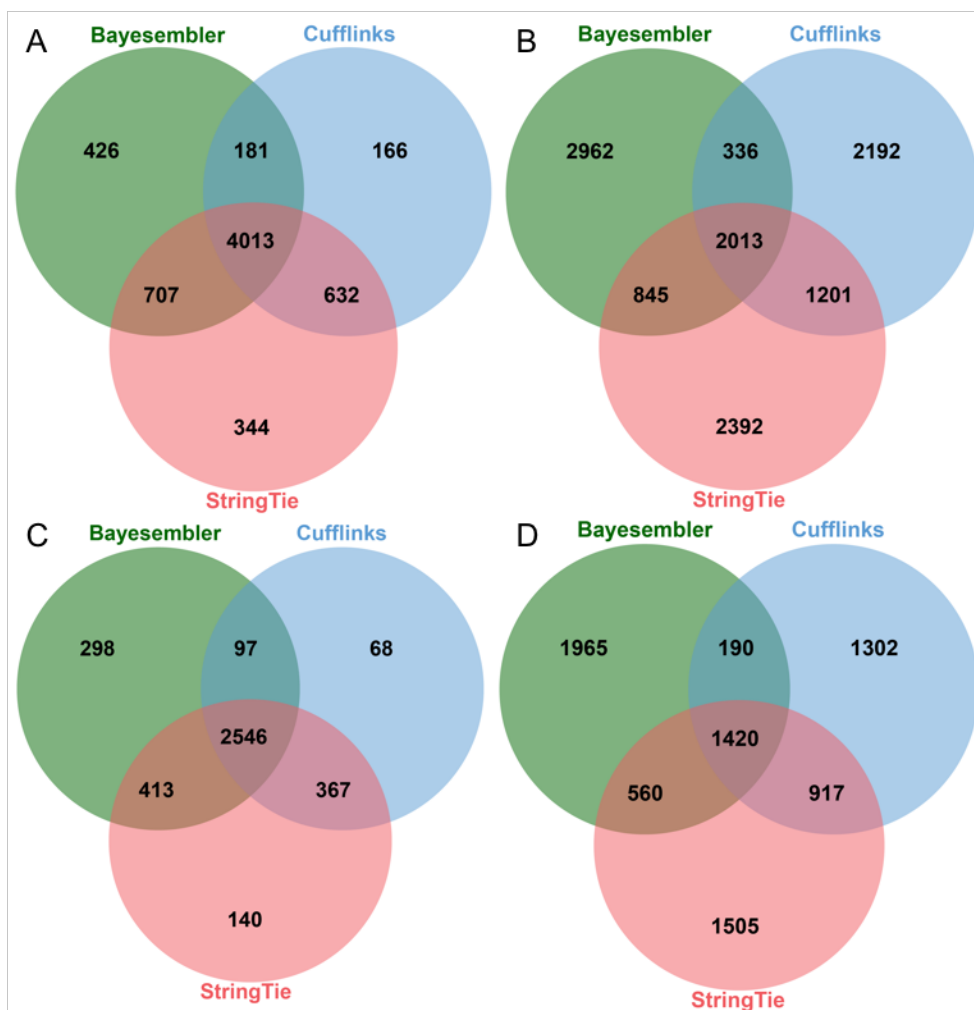
Figure 3.2: Venn diagrams showing the sequences from all of the genome-guided assemblers.

A) Correctly assembled sequences using the same reference genome, where the protein sequence of the contig matches the protein sequence in the benchmark transcriptome. B) Incorrectly assembled sequences using the same reference genome, where the protein sequence of the contig does not exactly match any protein sequence in the benchmark transcriptome. C) Correctly assembled sequences using a different reference genome, where the protein sequence of t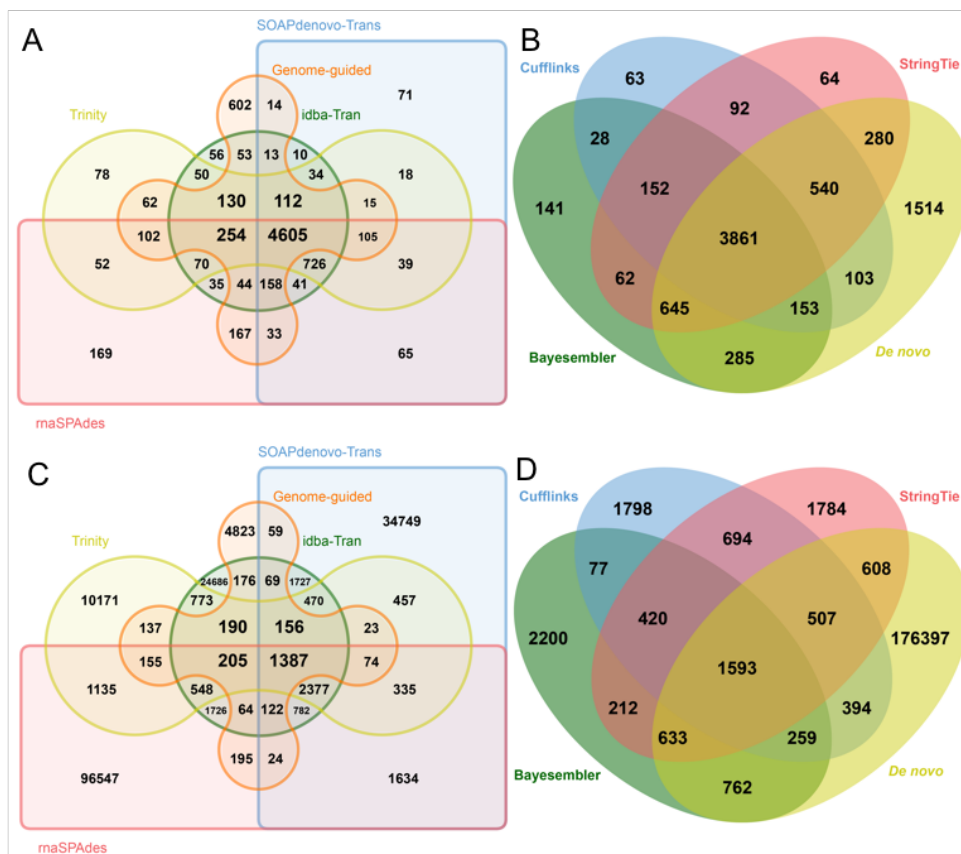he contig matches the protein sequence in the benchmark transcriptome. D) Incorrectly assembled sequences using a different reference genome, where the protein sequence of the contig does not exactly match any protein sequence in the benchmark transcriptome.

Figure 3.3: Venn diagrams showing the pooled sequences across all k-mers of each de novo assembler and the pooled sequences from all of the genome-guided assemblers.

A) Correctly assembled sequences for each de novo assembler and combined genome-guided assemblers. B) Correctly assembled sequences for each genome-guided assembler and combined de novo assemblers. C) Incorrectly assembled sequences for each de novo assembler and combined genome-guided assemblers. D) for each genome-guided assembler and combined de novo assemblers.
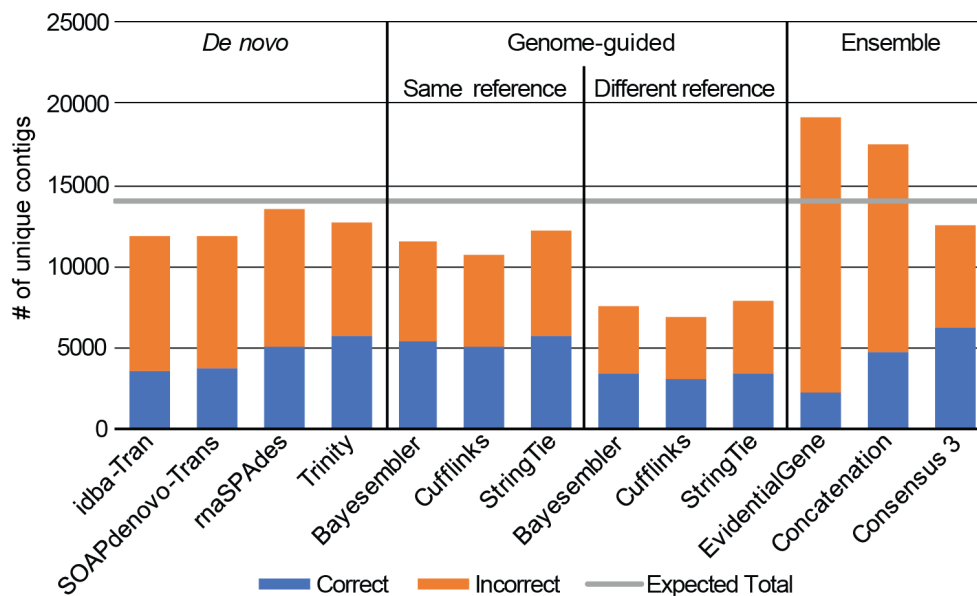
Figure 3.4: Performance comparison among all assemblers including de novo, genome-guided, and ensemble strategies.

Simulated RNAseq data were used for testing, and the default parameters were used for each assembler. See Tables 3.1, 3.3, and 3.5 for the actual numbers. The expected number of contigs is 14,040.

Table 3.1: Performance of individual *de novo* assemblers on simulated RNAseq library using default parameters or pooled across multiple kmer lengths.

	Total[a]	Unique[a]	Correct (%)[b]	Incorrect	CI ratio[c]
[Default]					
idba-Tran	11943	11941	3504 (24.96)	8437	0.4153
SOAPdenovo-Trans	12902	11830	3754 (26.74)	8076	0.4648
rnaSPAdes	15670	13513	5014 (35.71)	8499	0.5900
Trinity	14044	12639	5782 (41.18)	6857	0.8432
[Pooled][d]					
idba-Tran	170358	41849	6391 (45.52)	35458	0.1802
SOAPdenovo-Trans	297192	50504	6059 (43.16)	44445	0.1363
rnaSPAdes	765525	113975	6665 (47.47)	107310	0.0621
Trinity	89126	25045	6452 (45.95)	18593	0.3470

[a]Number of contigs assembled.

[b]Proportion (%) of transcripts in the benchmark that were correctly assembled.

[c](Number of correctly assembled contigs)/(number of incorrectly assembled contigs).

[d]Pooled results from using multiple kmers as follows: 15, 19, 23, 27, and 31 for Trinity;

15 kmer values ranging from 15 to 75 in increments of 4 for SOAPdenovo-Trans and

rnaSPAdes; 20, 30, 40, 50, and 60 for idba-Tran.

Table 3.2: Performance statistics of individual *de novo* assemblers using default parameters on simulated RNAseq library

	Precision	**Recall**	**Accuracy***	**F$_1$**	**FDR**
idba-Tran	0.2934	0.2496	0.1559	0.2697	0.7066
SOAPdenovo-Trans	0.3173	0.2674	0.1697	0.2902	0.6827
rnaSPAdes	0.3711	0.3571	0.2225	0.3640	0.6289
Trinity	0.4575	0.4118	0.2767	0.4334	0.5425

Table 3.3: Performance of individual genome-guided assemblers using default parameters on simulated RNAseq library with both the same and different references genome as the benchmark transcriptome.

	Total	Unique	Correct (%)	Incorrect	CI Ratio
[Same reference]					
Bayesembler	12989	11482	5327 (37.94)	6155	0.8655
Cufflinks	11257	10733	4992 (35.56)	5741	0.8695
StringTie	13218	12147	5696 (40.57)	6451	0.8830
[Different reference]					
Bayesembler	8536	7479	3345 (23.82)	4134	0.8091
Cufflinks	7234	6906	3078 (21.92)	3828	0.8041
StringTie	8608	7867	3466 (24.69)	4401	0.7875

Table 3.4: Performance statistics of individual genome-guided assemblers using default parameters on simulated RNAseq library with both the same and different references genome as the benchmark transcriptome.

	Precision	Recall	Accuracy*	F_1	FDR
[Same reference]					
Bayesembler	0.4639	0.3794	0.2638	0.4174	0.5361
Cufflinks	0.4651	0.3556	0.2524	0.4030	0.5349
StringTie	0.4689	0.4057	0.2780	0.4350	0.5311
[Different reference]					
Bayesembler	0.4473	0.2382	0.1841	0.3109	0.5527
Cufflinks	0.4457	0.2192	0.1723	0.2939	0.5543
StringTie	0.4406	0.2469	0.1880	0.3164	0.5594

Table 3.5: Performance of individual ensemble assembly strategies using the *de novo* assemblies.

	Total	Unique	Correct (%)	Incorrect	CI Ratio
EvidentialGene	19177	19175	2267 (16.15)	16908	0.1341
Concatenation	19767	17497	4697 (33.45)	12800	0.3670
Consensus 2	21444	21444	6711 (47.80)	14433	0.4650
Consensus 3	12600	12600	6144 (43.76)	6456	0.9517
Consensus 4	9095	9095	5331 (37.97)	3764	1.416

Table 3.6: Performance statistics of ensemble assembly strategies using *de novo* assemblies on simulated RNAseq library.

	Precision	Recall	Accuracy*	F_1	FDR
EvidentialGene	0.1182	0.1615	0.0733	0.1365	0.8818
Concatenation	0.2684	0.3345	0.1750	0.2979	0.7316
Consensus 2	0.3174	0.4780	0.2357	0.3815	0.6826
Consensus 3	0.4876	0.4376	0.2998	0.4613	0.5124
Consensus 4	0.5861	0.3797	0.2994	0.4609	0.4139

Chapter 4: Conclusions

Transcriptome assembly can be approached from multiple different strategies.

Historically, these approaches have revolved around assembling short but highly accurate

Illumina reads with or without an existing genome assembly as a reference, referred to as

genome-guided or *de novo* assemblies, respectively. All of the widely used *de novo*

assemblers decompose the short reads into smaller kmers and use de Bruijn graphs built

on these kmers to attempt to reconstruct the original transcripts. Due to the limitations of

the de Bruijn graphs, this approach presents a trade-off between the uniqueness of the

longer kmers and increased coverage of the shorter kmers. As a result, different kmer

lengths can produce drastically different graphs, leading to large differences in the final

assemblies.

Genome-guided assemblers avoid the limitations of the de Bruijn graphs by

mapping the reads to the reference genome. This mapping, however, introduces its own

limitations and trade-offs. Reads that are ambiguous between splice forms in the same

genomic locations or across multiple genomic locations create similar challenges to the

de Bruijn graphs. These ambiguities are compounded when the mapping must take into

account mismatches due to sequencing errors as well as biological variations.

The limitations of the individual tools can potentially be overcome by combining

multiple different assemblies in ensemble. As each tool and set of parameters results in a

different set of correctly assembled contigs, accurately selecting these correctly

assembled contigs without selecting any redundant incorrectly assembled contigs would

leverage the strengths of each methods without the weaknesses of any. However,

currently available ensemble strategies cannot guarantee that the correct sequence is

chosen, leading to ensemble assemblies that are less accurate than individual assemblies. As the selection criteria for ensemble methods improve, such as with the "Consensus" approach shown here, these methods can also leverage new assembly approaches that can better handle certain subsets of transcripts (e.g. alternative splice forms) that may have other weaknesses that prevent them from being competitive as a general transcript assembly tool.

Overall, as our results demonstrated, transcriptome assemblies can still be improved, regardless of the approach used. While the genome-guided assemblers generally perform best when the assembly is performed against the same reference sequence that the RNAseq data was generated from, this is not universally true. Furthermore, when these sequences differ, the genome-guided assemblers may have lower accuracy than the *de novo* assemblers. While ensemble assembly strategies can potentially improve on accuracy over individual assemblies, it is also possible that they instead reduce the accuracy. Improving the performance of these tools, whether individual assemblers, ensemble strategies, or combined with long-read sequencing, will improve the accuracy of the reconstructed transcriptome. These improvements will also increase the accuracy of downstream analyses, such as sequence annotation, quantification, and differential expression.

References:

1.   Wang Z, Gerstein M, Snyder M. RNA-Seq: a revolutionary tool for transcriptomics. Nat Rev Genet. 2009;10(1):57-63.

2.   Ozsolak F, Milos PM. RNA sequencing: advances, challenges and opportunities. Nat Rev Genet. 2011;12(2):87-98.

3.   Huang X, Chen XG, Armbruster PA. Comparative performance of transcriptome assembly methods for non-model organisms. BMC Genomics. 2016;17:523.

4.   Conesa A, Madrigal P, Tarazona S, Gomez-Cabrero D, Cervera A, McPherson A, et al. A survey of best practices for RNA-seq data analysis. Genome Biol. 2016;17:13.

5.   Simonis M, Atanur SS, Linsen S, Guryev V, Ruzius FP, Game L, et al. Genetic basis of transcriptome differences between the founder strains of the rat HXB/BXH recombinant inbred panel. Genome Biol. 2012;13(4):r31.

6.   Smith-Unna R, Boursnell C, Patro R, Hibberd JM, Kelly S. TransRate: reference-free quality assessment of de novo transcriptome assemblies. Genome Res. 2016;26(8):1134-44.

7.   Grabherr MG, Haas BJ, Yassour M, Levin JZ, Thompson DA, Amit I, et al. Full-length transcriptome assembly from RNA-Seq data without a reference genome. Nature biotechnology. 2011;29(7):644-52.

8.   Xie Y, Wu G, Tang J, Luo R, Patterson J, Liu S, et al. SOAPdenovo-Trans: de novo transcriptome assembly with short RNA-Seq reads. Bioinformatics. 2014;30(12):1660-6.

9.   Peng Y, Leung HC, Yiu SM, Lv MJ, Zhu XG, Chin FY. IDBA-tran: a more robust de novo de Bruijn graph assembler for transcriptomes with uneven expression levels. Bioinformatics. 2013;29(13):i326-34.

10.   Zerbino DR, Birney E. Velvet: algorithms for de novo short read assembly using de Bruijn graphs. Genome Res. 2008;18(5):821-9.

11.   Schulz MH, Zerbino DR, Vingron M, Birney E. Oases: robust de novo RNA-seq assembly across the dynamic range of expression levels. Bioinformatics. 2012;28(8):1086-92.

12.   Bankevich A, Nurk S, Antipov D, Gurevich AA, Dvorkin M, Kulikov AS, et al. SPAdes: a new genome assembly algorithm and its applications to single-cell sequencing. J Comput Biol. 2012;19(5):455-77.

13.   Chevreux B, Pfisterer T, Drescher B, Driesel AJ, Muller WEG, Wetter T, et al. Using the miraEST assembler for reliable and automated mRNA transcript assembly and SNP detection in sequenced ESTs. Genome Research. 2004;14(6):1147-59.

14.   Martin JA, Wang Z. Next-generation transcriptome assembly. Nat Rev Genet. 2011;12(10):671-82.

15.   Koren S, Treangen TJ, Hill CM, Pop M, Phillippy AM. Automated ensemble assembly and validation of microbial genomes. BMC Bioinformatics. 2014;15:126.

16.   Deng X, Naccache SN, Ng T, Federman S, Li L, Chiu CY, et al. An ensemble strategy that significantly improves de novo assembly of microbial genomes from metagenomic next-generation sequencing data. Nucleic Acids Res. 2015;43(7):e46.

17.   Trapnell C, Pachter L, Salzberg SL. TopHat: discovering splice junctions with RNA-Seq. Bioinformatics. 2009;25(9):1105-11.

18.   Dobin A, Davis CA, Schlesinger F, Drenkow J, Zaleski C, Jha S, et al. STAR: ultrafast universal RNA-seq aligner. Bioinformatics. 2013;29(1):15-21.

19.   Kim D, Langmead B, Salzberg SL. HISAT: a fast spliced aligner with low memory requirements. Nat Methods. 2015;12(4):357-60.

20.   Medina I, Tarraga J, Martinez H, Barrachina S, Castillo MI, Paschall J, et al. Highly sensitive and ultrafast read mapping for RNA-seq analysis. DNA Res. 2016;23(2):93-100.

21.   Maretty L, Sibbesen JA, Krogh A. Bayesian transcriptome assembly. Genome biology. 2014;15(10):501.

22.   Trapnell C, Williams BA, Pertea G, Mortazavi A, Kwan G, van Baren MJ, et al. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. Nature biotechnology. 2010;28(5):511-5.

23.   Pertea M, Pertea GM, Antonescu CM, Chang TC, Mendell JT, Salzberg SL. StringTie enables improved reconstruction of a transcriptome from RNA-seq reads. Nat Biotechnol. 2015;33(3):290-5.

24.   Nakasugi K, Crowhurst R, Bally J, Waterhouse P. Combining transcriptome assemblies from multiple de novo assemblers in the allo-tetraploid plant Nicotiana benthamiana. PLoS One. 2014;9(3):e91776.

25.   Gilbert D. Gene-omes built from mRNA seq not genome DNA. 7th annual arthropod genomics symposium Notre Dame. 2013.

26.   Cerveau N, Jackson DJ. Combining independent de novo assemblies optimizes the coding transcriptome for nonconventional model eukaryotic organisms. BMC Bioinformatics. 2016;17(1):525.

27.   Abdel-Ghany SE, Hamilton M, Jacobi JL, Ngam P, Devitt N, Schilkey F, et al. A survey of the sorghum transcriptome using single-molecule long reads. Nat Commun. 2016;7:11706.

28.   Salmela L, Walve R, Rivals E, Ukkonen E. Accurate self-correction of errors in long reads using de Bruijn graphs. Bioinformatics. 2017;33(6):799-806.

29.   Salmela L, Rivals E. LoRDEC: accurate and efficient long read error correction. Bioinformatics. 2014;30(24):3506-14.

30.   Hargreaves AD, Mulley JF. Assessing the utility of the Oxford Nanopore MinION for snake venom gland cDNA sequencing. PeerJ. 2015;3:e1441.

31.   Cheng B, Furtado A, Henry RJ. Long-read sequencing of the coffee bean transcriptome reveals the diversity of full-length transcripts. Gigascience. 2017;6(11):1-13.

32.   Mei W, Liu S, Schnable JC, Yeh CT, Springer NM, Schnable PS, et al. A Comprehensive Analysis of Alternative Splicing in Paleopolyploid Maize. Front Plant Sci. 2017;8:694.

33.   Sharon D, Tilgner H, Grubert F, Snyder M. A single-molecule long-read survey of the human transcriptome. Nat Biotechnol. 2013;31(11):1009-14.

34.   Minoche AE, Dohm JC, Schneider J, Holtgrawe D, Viehover P, Montfort M, et al. Exploiting single-molecule transcript sequencing for eukaryotic gene prediction. Genome biology. 2015;16:184.

35.   Zhang SJ, Wang C, Yan S, Fu A, Luan X, Li Y, et al. Isoform Evolution in Primates through Independent Combination of Alternative RNA Processing Events. Mol Biol Evol. 2017;34(10):2453-68.

36.   Earl D, Bradnam K, St John J, Darling A, Lin D, Fass J, et al. Assemblathon 1: a competitive assessment of de novo short read assembly methods. Genome Res. 2011;21(12):2224-41.

37.   Piovesan A, Caracausi M, Antonaros F, Pelleri MC, Vitale L. GeneBase 1.1: a tool to summarize data from NCBI gene datasets and its application to an update of human gene statistics. Database (Oxford). 2016;2016.

38.   O'Neil ST, Emrich SJ. Assessing De Novo transcriptome assembly metrics for consistency and utility. BMC Genomics. 2013;14:465.

39.   Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. Journal of molecular biology. 1990;215(3):403-10.

40.   The UniProt Consortium. UniProt: the universal protein knowledgebase. Nucleic Acids Res. 2017;45(D1):D158-D69.

41.   Finn RD, Bateman A, Clements J, Coggill P, Eberhardt RY, Eddy SR, et al. Pfam: the protein families database. Nucleic Acids Res. 2014;42(Database issue):D222-30.

42.   Thomas PD, Campbell MJ, Kejariwal A, Mi H, Karlak B, Daverman R, et al. PANTHER: a library of protein families and subfamilies indexed by function. Genome Res. 2003;13(9):2129-41.

43.   Waterhouse RM, Seppey M, Simao FA, Manni M, Ioannidis P, Klioutchnikov G, et al. BUSCO applications from quality assessments to gene prediction and phylogenomics. Mol Biol Evol. 2017.

44.   Besemer J, Lomsadze A, Borodovsky M. GeneMarkS: a self-training method for prediction of gene starts in microbial genomes. Implications for finding sequence motifs in regulatory regions. Nucleic Acids Res. 2001;29(12):2607-18.

45.   Wang S, Gribskov M. Comprehensive evaluation of de novo transcriptome assembly programs and their effects on differential gene expression analysis. Bioinformatics. 2017;33(3):327-33.

46.   Li B, Dewey CN. RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. BMC Bioinformatics. 2011;12:323.

47.   Griebel T, Zacher B, Ribeca P, Raineri E, Lacroix V, Guigo R, et al. Modelling and simulating generic RNA-Seq experiments with the flux simulator. Nucleic Acids Res. 2012;40(20):10073-83.

48.   Ono Y, Asai K, Hamada M. PBSIM: PacBio reads simulator--toward accurate genome assembly. Bioinformatics. 2013;29(1):119-21.

49.   Yang C, Chu J, Warren RL, Birol I. NanoSim: nanopore sequence read simulator based on statistical characterization. Gigascience. 2017;6(4):1-6.

50.   Del Fabbro C, Scalabrin S, Morgante M, Giorgi FM. An extensive evaluation of read trimming effects on Illumina NGS data analysis. PLoS One. 2013;8(12):e85024.

51.   Crusoe MR, Alameldin HF, Awad S, Boucher E, Caldwell A, Cartwright R, et al. The khmer software package: enabling efficient nucleotide sequence analysis. F1000Res. 2015;4:900.