Downloaded from genome.cshlp.org on May 2, 2026 . Published by Cold Spring Harbor Laboratory Press

TransBorrow: genome-guided transcriptome assembly by borrowing assemblies from different assemblers

Ting Yu, Zengchao Mu, Zhaoyuan Fang, et al.

Genome Res. 2020 30: 1181-1190 originally published online August 17, 2020
Access the most recent version at doi:10.1101/gr.257766.119

References This article cites 48 articles, 4 of which can be accessed free at:
http://genome.cshlp.org/content/30/8/1181.full.html#ref-list-1

Open Access Freely available online through the *Genome Research* Open Access option.

Creative Commons License This article, published in *Genome Research*, is available under a Creative Commons License (Attribution 4.0 International), as described at http://creativecommons.org/licenses/by/4.0/.

Email Alerting Service Receive free email alerts when new articles cite this article - sign up in the box at the top right corner of the article or **click here.**

To subscribe to *Genome Research* go to:
https://genome.cshlp.org/subscriptions

© 2020 Yu et al.; Published by Cold Spring Harbor Laboratory Press

Downloaded from genome.cshlp.org on May 2, 2026 . Published by Cold Spring Harbor Laboratory Press

Method

TransBorrow: genome-guided transcriptome assembly by borrowing assemblies from different assemblers

Ting Yu,[1,4] Zengchao Mu,[1,4] Zhaoyuan Fang,[2] Xiaoping Liu,[1] Xin Gao,[3] and Juntao Liu[1]

[1] School of Mathematics and Statistics, Shandong University (Weihai), Weihai 264209, China; [2] Key Laboratory of Systems Biology, CAS Center for Excellence in Molecular Cell Science, Institute of Biochemistry and Cell Biology, Shanghai Institutes for Biological Sciences, Chinese Academy of Sciences, University of Chinese Academy of Sciences, Shanghai 200031, China; [3] Computational Bioscience Research Center (CBRC), Computer, Electrical and Mathematical Sciences and Engineering Division, King Abdullah University of Science and Technology (KAUST), Thuwal 23955, Saudi Arabia

RNA-seq technology is widely used in various transcriptomic studies and provides great opportunities to reveal the complex structures of transcriptomes. To effectively analyze RNA-seq data, we introduce a novel transcriptome assembler, TransBorrow, which borrows the assemblies from different assemblers to search for reliable subsequences by building a colored graph from those borrowed assemblies. Then, by seeding reliable subsequences, a newly designed path extension strategy accurately searches for a transcript-representing path cover over each splicing graph. TransBorrow was tested on both simulated and real data sets and showed great superiority over all the compared leading assemblers.

[Supplemental material is available for this article.]

RNA-seq technology is still widely used throughout the world to explore and study the very complex transcriptomic structures of eukaryotes (Marioni et al. 2008; Wang et al. 2009; Wilhelm and Landry 2009; Marguerat and Bähler 2010) because of its high throughput, high accuracy, and low cost. It is a powerful technology that identifies expressed transcripts and measures isoform expression levels at the whole-transcriptome level with unprecedented accuracy (Wang et al. 2009; Wilhelm and Landry 2009; Marguerat and Bähler 2010; Ozsolak and Milos 2011).

Most eukaryotic genes generally produce multiple isoforms because of alternative splicing in eukaryotes. Therefore, one of the most important tasks is to accurately identify all the expressed transcripts for subsequent biological studies. However, transcripts from the same locus can share exons owing to alternative splicing, and different isoforms from the same gene may have highly variable expression abundances, making the transcriptome assembly problem quite challenging. Moreover, RNA-seq runs generate hundreds of millions of short reads (usually 50–300 bp in length) with ~2% sequencing errors (Metzker 2010; Canzar et al. 2016). Therefore, computationally identifying all the expressed transcripts from the large amounts of short sequencing reads with unknown sequencing errors poses a great challenge.

Currently available transcriptome assemblers are usually categorized into two strategies: genome-guided (or reference-based) and de novo. In general, if a high-quality genome is available for some species, such as humans, genome-guided assemblers such as Scallop (Shao and Kingsford 2017), TransComb (Liu et al. 2016b), StringTie (Pertea et al. 2015), StringTie2 (Kovaka et al. 2019), Cufflinks (Trapnell et al. 2010), Class2 (Song et al. 2016), Scripture (Guttman et al. 2010), IsoInfer (Feng et al. 2011), IsoLasso (Li et al. 2011), iReckon (Mezlini et al. 2013), CEM (Li and Jiang 2012), Traph (Tomescu et al. 2013), and Mitie (Behr et al. 2013) usually first map all RNA-seq reads to the genome using mapping tools such as HISAT (Kim et al. 2015), STAR (Dobin et al. 2013), TopHat (Trapnell et al. 2009), TopHat2 (Kim et al. 2013), SpliceMap (Au et al. 2010), or GSNAP (Wu and Nacu 2010). Then, a graph model, such as a splicing graph or overlap graph, is built based on the mappings, and different methods are implemented to search for transcript-representing paths in the constructed graphs. Different from the reference-based strategy, which benefits from high-quality genomes, the de novo strategy, with assemblers such as TransLiG (Liu et al. 2019), BinPacker (Liu et al. 2016a), Bridger (Chang et al. 2015), Trinity (Grabherr et al. 2011), ABySS (Simpson et al. 2009), SOAPdenovo-Trans (Xie et al. 2014), and IDBA-Tran (Peng et al. 2013), usually builds graph models and assembles all expressed transcripts directly from the RNA-seq reads. Therefore, the de novo strategy is more challenging than the genome-guided strategy, and its assembly accuracy is generally much lower. However, the de novo strategy plays an important role in studying species for which genome assemblies of high quality are not available at the moment.

For genome assembly, a variety of assembly tools are available, but it is not always obvious which tool to use for a specific genome. Therefore, a compelling approach is to merge multiple assemblies to produce a higher-quality consensus assembly (Alhakami et al. 2017). Several tools for merging multiple assemblies have been developed, such as CISA (Lin and Liao 2013), GAA (Yao et al. 2012), GAM_NGS (Vicedomini et al. 2013), GARM (Soto-Jimenez et al. 2014), Metassembler (Wences and Schatz 2015), and MIX (Soueidan et al. 2013) via different merging strategies. For example, the CISA tool first selects representative contigs and discards contained contigs. It then extends representative contigs and detects misassembly in the representative contigs by aligning them to query contigs. Finally, the resulting contigs are iteratively merged. Another tool, MIX, uses an extension graph to determine a set of nonoverlapping maximal independent longest paths to merge contigs. Contigs not included in any path are

[4] **These authors contributed equally to this work.**
Corresponding authors: xin.gao@kaust.edu.sa, juntao@sdu.edu.cn
Article published online before print. Article, supplemental material, and publication date are at http://www.genome.org/cgi/doi/10.1101/gr.257766.119. Freely available online through the *Genome Research* Open Access option.

© 2020 Yu et al. This article, published in *Genome Research*, is available under a Creative Commons License (Attribution 4.0 International), as described at http://creativecommons.org/licenses/by/4.0/.

Downloaded from genome.cshlp.org on May 2, 2026 . Published by Cold Spring Harbor Laboratory Press

examined for duplications. Contigs that are contained or nearly contained are removed, and the rest are added to the assembly.

For transcriptome assembly, no assembler consistently generates the most accurate assemblies when tested across different RNA-seq data sets, and it is difficult to determine which assembler to use for a specific RNA-seq data set (Smith-Unna et al. 2016; Voshall and Moriyama 2018). Therefore, it is imperative to develop approaches to merge assemblies from different assemblers (Haas et al. 2003; Venturini et al. 2018). The de novo assemblers EvidentialGene (Gilbert 2013) and Concatenation (Cerveau and Jackson 2016) try to address the limitations of individual assemblers, ideally keeping the correctly assembled transcripts. Both tools process the transcripts obtained from multiple assemblers by clustering the transcripts and predicting coding regions to determine the representative sequence for each cluster. The de novo assembler Trans-ABySS (Robertson et al. 2010) attempts to solve this problem by merging assemblies from ABySS (Simpson et al. 2009) using different *k*-mer lengths. The genome-guided assembler Mikado (Venturini et al. 2018) generates a coherent transcript annotation by integrating multiple RNA-seq assemblies from multiple samples. It defines loci, scores transcripts, determines a representative transcript for each locus, and finally returns a set of gene models. In addition, some other assembling tools for reconstructing a consensus transcriptome from multiple RNA-seq samples have also been developed, such as TACO (Niknafs et al. 2017) and StringTie-merge (Pertea et al. 2015).

In this study, we developed a new genome-guided assembler TransBorrow, which assembles transcripts by first building splicing graphs based on the mapped reads and extracting reliable paired subpaths from splicing graphs. It then borrows reliable subsequences from different assemblies by building a so-called colored graph. Then, those reliable subsequences and paired subpaths are mapped to the splicing graphs as reliable subpaths for guiding the correct assemblies of expressed transcripts. Finally, a newly designed path extension method is applied to search for a transcript-representing path cover over each splicing graph by seeding those reliable subpaths (for the flowchart of TransBorrow, see Fig. 1). Below, we describe the algorithmic approaches used in TransBorrow in detail and benchmark it against the state-of-the-art transcriptome assemblers StringTie2, Scallop, and Cufflinks and two merging-based assemblers StringTie-merge and TACO on both simulated and real RNA-seq data sets.

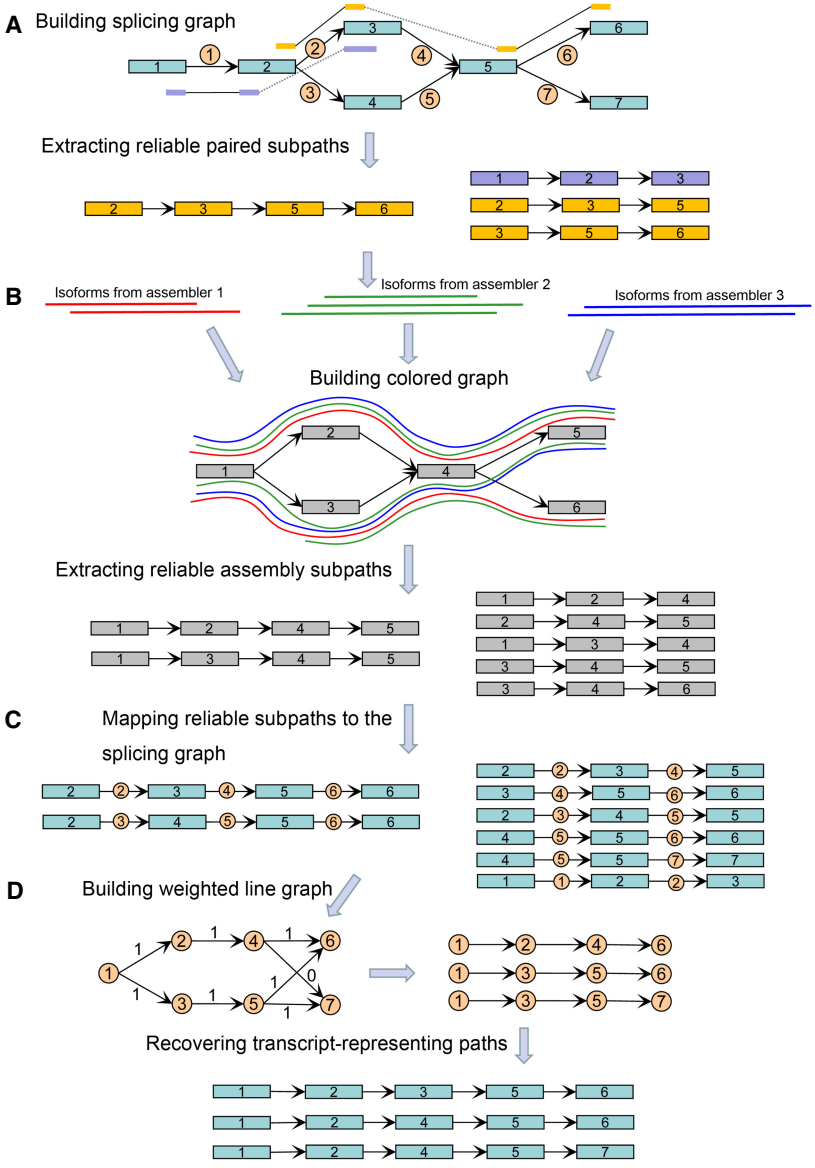

Figure 1. Flowchart of the TransBorrow algorithm. (*A*) Building splicing graph and extracting reliable paired subpaths; (*B*) building colored graph and extracting reliable assembly subpaths; (*C*) mapping reliable subpaths to the splicing graph; (*D*) recovering transcript-representing paths.

Results

TransBorrow is a transcriptome assembler that takes advantage of assemblies from different assembly tools by searching for reliable assembly subpaths from different assemblies and then seeding these subpaths for transcript-representing path extensions subsequently in each splicing graph. To evaluate its performance, we first assembled the RNA-seq reads by using different assemblers, and then the algorithm TransBorrow was run by merging these different assemblies. The results showed that TransBorrow effectively takes advantage of the assemblies from different tools and that TransBorrow has enhanced performance compared with that of other assembly tools.

In this study, the latest alignment tools HISAT2 and STAR were used for mapping the RNA-seq reads to a reference genome, and then the leading assemblers Scallop, StringTie2, and Cufflinks were applied to assemble all the expressed transcripts of both simulated and

Downloaded from genome.cshlp.org on May 2, 2026 . Published by Cold Spring Harbor Laboratory Press

real data. Then, we ran TransBorrow by merging the assemblies from the three assemblers and comparing its performance with each of them. In addition, two merging-based assemblers, StringTie-merge and TACO, were also tested, which combined assemblies from different samples (rather than assemblies from different assemblers) to improve assembly accuracy. In this study, we also compared with StringTie-merge and TACO by taking the assemblies from StringTie2, Scallop, and Cufflinks (as those are from different samples). The detailed commands for running all the mappers and assemblers and the versions of all the tools are described in the Supplemental Material. The common comparison criteria used in this study were that a reference transcript is considered to be correctly detected if and only if its intron chain is exactly matched with an assembled transcript. The human genome GRCh37/hg19 and all the reference transcripts downloaded from the UCSC hg19 gene annotation were used as a reference genome and transcriptome, respectively.

Performance of TransBorrow on simulated data

The FLUX simulator (Griebel et al. 2012) was used to generate the simulation data (150-bp length, approximately 73 million paired-end reads), on which we tested the performance of TransBorrow, Scallop, StringTie2, and Cufflinks by commonly used criteria such as assembly accuracy (recall and precision) at both the transcript and gene levels and the identification of transcripts with different expression levels (low, medium, and high). The parameters and running commands for generating the simulated data set are described in the Supplemental Material.

Comparison of assembly accuracy at the transcript and gene levels

We first ran Scallop, StringTie2, and Cufflinks on the simulated data by using the mapping results from both HISAT2 and STAR. Then TransBorrow was run by merging the assemblies from the three assemblers on the mapping results from HISAT2 and STAR. The accuracy was evaluated by recall (the fraction of correctly detected expressed transcripts out of all the expressed transcripts) and precision (the percentage of assembled transcripts that exactly matched an expressed transcript).

Based on both HISAT2 and STAR mapping, TransBorrow consistently achieved the highest recall and precision among all the compared assemblers on the simulated data (see Fig. 2A; for details, see Supplemental Table S1). For the correctly assembled transcripts based on HISAT2 and STAR mapping, TransBorrow correctly detected 5.64% and 1.29% more expressed transcripts than StringTie2, 35.58% and 7.53% more than Scallop, 52.29% and 38.55% more than Cufflinks, 37.96% and 8.3% more than StringTie-merge, and 30.13% and 8.44% more than TACO (see Fig. 2B; for details, see Supplemental Table S1). Therefore, TransBorrow performed better than all the other compared assemblers on the simulated data at the transcript level.

We further compared the performance of the assemblers in identifying expressed genes. A gene is considered to be correctly detected if at least one of its isoforms is correctly assembled. Similarly, recall (at the gene level) is defined as the fraction of correctly detected genes out of the expressed genes, and precision (at the gene level) is defined as the fraction of correctly detected genes out of all assembled genes.

After running the assemblers based on both HISAT2 and STAR mapping, the recall and precision of TransBorrow again achieved the highest among all the compared assemblers (see Fig. 2C; for details, see Supplemental Table S1). Regarding the correctly detected genes, TransBorrow correctly detected 4.96% more genes than StringTie2 based on HISAT2 mapping, 33.33% and 5.48% more than Scallop based on HISAT2 and STAR mapping, 39.79% and 26.65% more than Cufflinks, 31.83% and 3.26% more than

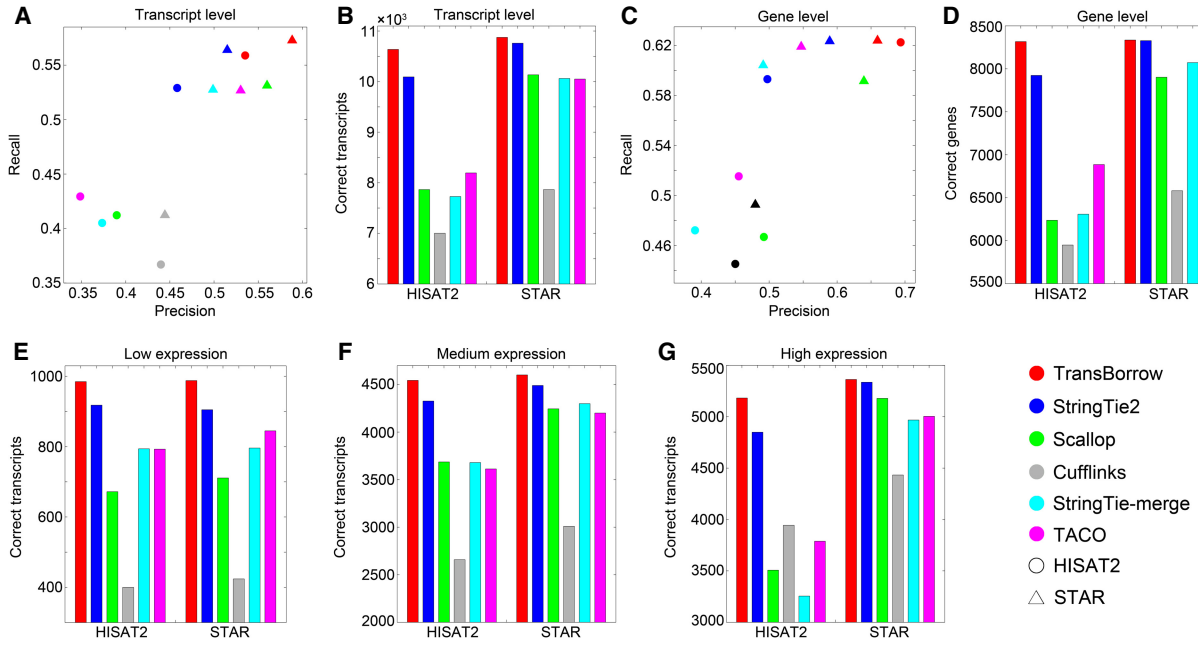

Figure 2. Performance comparisons of the assemblers on the simulated data. (*A*) Comparisons of assembly accuracy of the assemblers at the transcript level. (*B*) The number of correctly assembled transcripts by the assemblers. (*C*) Comparisons of assembly accuracy of the assemblers at the gene level. (*D*) The number of correctly detected genes by the assemblers. (*E–G*) Comparisons of detected transcripts with low, medium, or high expression levels on the simulated data.

Downloaded from genome.cshlp.org on May 2, 2026 . Published by Cold Spring Harbor Laboratory Press

StringTie-merge, and 20.78% and 0.8% more than TACO (see Fig. 2D; for details, see Supplemental Table S1). Therefore, TransBorrow reached the best performance among all the compared assemblers under both HISAT2 and STAR mappings at the gene level.

Comparison of detected transcripts with different expression levels

In theory, splicing isoforms with relatively lower expression levels are more difficult to correctly assemble than those with higher expression levels (Canzar et al. 2016; Shao and Kingsford 2017). To compare the performance of the assemblers in identifying transcripts with different expression levels, especially those with relatively lower expression levels, as previously described (Shao and Kingsford 2017), the expressed transcripts of the simulated data were first equally divided into three parts according to their expression levels, which corresponded to lowly, moderately, and highly expressed transcripts.

After comparison, the results showed that TransBorrow correctly detected the largest number of expressed transcripts, regardless of expression levels, based on both HISAT2 and STAR mappings (for details, see Fig. 2E–G). More importantly, TransBorrow correctly detected 7.3% and 9.17% more lowly expressed transcripts than StringTie2, 46.58% and 38.96% more than Scallop, 146.25% and 133.02% more than Cufflinks, 24.06% and 24.12% more than StringTie-merge, and 24.21% and 16.92% more than TACO, which clearly showed that TransBorrow performed the best in identifying transcripts with different expression levels, especially those with low expression levels.

Performance of TransBorrow on real data

The advantage of simulated data is that its ground truth is known; however, it is not able to capture all the features of real RNA-seq data. Therefore, performance evaluations of the assemblers should also be implemented on real data. As the ground truth of real data is difficult to know, we set all the reference transcripts downloaded from the UCSC gene annotation hg19 as the ground truth in this study. Four real data sets—including R1, K562 cells (replicate 1); R2, K562 cells (replicate 2); R3, H1 cells (replicate 1); and R4, H1 cells (replicate 2)—were collected from the NCBI Sequence Read Archive (SRA) with the accession codes SRR387661, SRR387662, SRR307911, and SRR30 7912, respectively. The four data sets, R1, R2, R3, and R4, contain approximately 125 million, 88 million, 41 million, and 37 million paired-end reads, respectively. Then, the performance of the assemblers was evaluated on the four real data sets by using the same criteria as that used for the simulated data.

Comparison of assembly accuracy at the transcript level

After running all the assemblers on the four real data sets, the results showed that TransBorrow achieved the highest recall on all four data sets based on both HISAT2 and STAR mappings (see Fig. 3A–D; for details, see Supplemental Table S2). In terms of precision, TransBorrow reached the highest precision among all compared assemblers on the data sets R2, R3, and R4 based on both HISAT2 and STAR mappings (see Fig. 3A–D; for details, see Supplemental Table S2). On the first data set R1, the precision of TransBorrow is slightly lower than that of Scallop based on HISAT2 mapping and slightly lower than that of StringTie2 based on STAR mapping. However, the *F*-score of TransBorrow is the highest (for details, see Supplemental Fig. S9), which indicates that the overall performance of TransBorrow was better than those of both StringTie2 and Scallop with their default settings. By default, assemblers filtered their assembled transcripts with low estimated expression levels after transcriptome assembly. Therefore, after filtering, the recall will generally decline, whereas the precision will increase. This flexible filtering parameter corresponds to a trade-off between recall and precision. If the users adjust the parameter, TransBorrow could reach both higher recall and precision than StringTie2 and Scallop. For example, if we filtered the assembled transcripts of TransBorrow by using parameters 2.2 and 1.7 under HISAT2 and STAR mappings to make its precision slightly higher than that of Scallop and StringTie2 on data set R1, and we found that the recall and precision of TransBorrow were 18.78% and 28.66% under HISAT2 mapping, and 17.76% and 27.74% under STAR mapping, which indicates that TransBorrow showed both higher recall and precision than Scallop and StringTie2 (see Fig. 3A; for details, see Supplemental Table S2). In addition, the recall/precision curves give more comprehensive comparison between different methods (for details, see Supplemental Fig. S14).

For the correctly assembled transcripts based on HISAT2 mappings, TransBorrow correctly detected 15.19%–20.82% more transcripts than StringTie2, 14.61%–20.52% more than Scallop, 76.81%–114.93% more than Cufflinks, 22.2%–40.84% more than StringTie-merge, and 22.12%–34.95% more than TACO on the four real data sets (see Fig. 3E; for details, see Supplemental Table S2). Based on STAR mappings, TransBorrow correctly

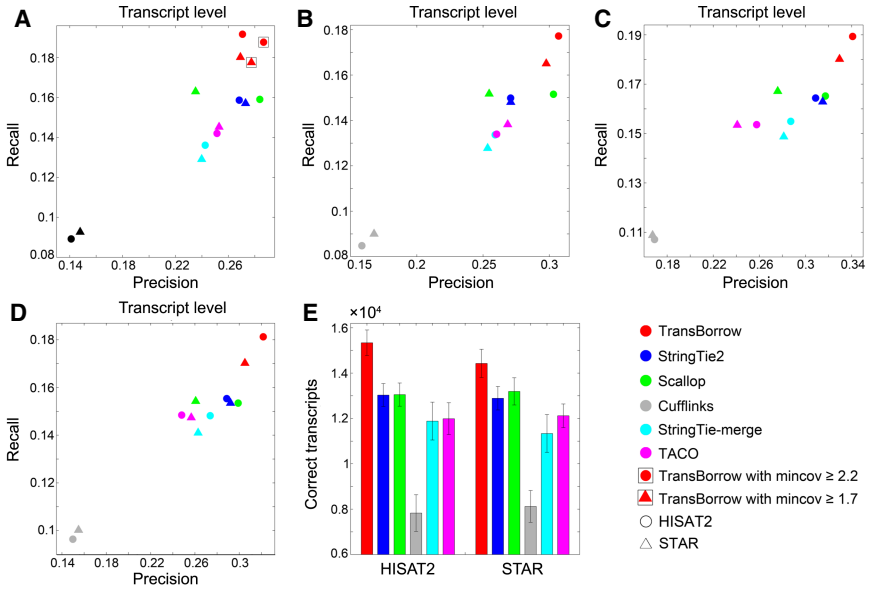

Figure 3. Accuracy comparisons of the assemblers on the four real data sets at the transcript level. (*A–D*) Comparisons of assembly accuracy of the assemblers on data sets R1, R2, R3, and R4, respectively. (*E*) The average number of correctly assembled transcripts by the assemblers on data sets R1, R2, R3, and R4.

Downloaded from genome.cshlp.org on May 2, 2026 . Published by Cold Spring Harbor Laboratory Press

detected 10.59%–14.7% more transcripts than StringTie2, 7.78%–10.54% more than Scallop, 65.62%–94.52% more than Cufflinks, 20.73%–39.58% more than StringTie-merge, and 15.5%–24.05% more than TACO on the four real data sets (see Fig. 3E; for details, see Supplemental Table S2). Therefore, TransBorrow performed better than all the other compared assemblers on the four real data sets at the transcript level.

Comparison of assembly accuracy at the gene level

We then compared the accuracy of the assemblers at the gene level using the four real data sets. After running the assemblers based on both HISAT2 and STAR mappings, TransBorrow reached much higher recall and precision than any of the compared assemblers on all four real data sets (see Fig. 4A–D; for details, see Supplemental Table S3).

Regarding the correctly detected genes based on the HISAT2 mappings, TransBorrow correctly detected 5.35%–5.94% more genes than StringTie2, 6%–9.74% more than Scallop, 47.76%–63.49% more than Cufflinks, 7.04%–15.57% more than StringTie-merge, and 7.97%–13.2% more than TACO (see Fig. 4E; for details, see Supplemental Table S3). Based on STAR mappings, TransBorrow correctly detected 0.52%–2.12% more genes than StringTie2, 5.11%–8.26% more than Scallop, 37.53%–48.65% more than Cufflinks, 5.94%–13.96% more than StringTie-merge, and 4.37%–7.5% more than TACO on the four real data sets (see Fig. 4E; for details, see Supplemental Table S3). Therefore, TransBorrow performed better than all the other compared assemblers on the four real data sets at the gene level.

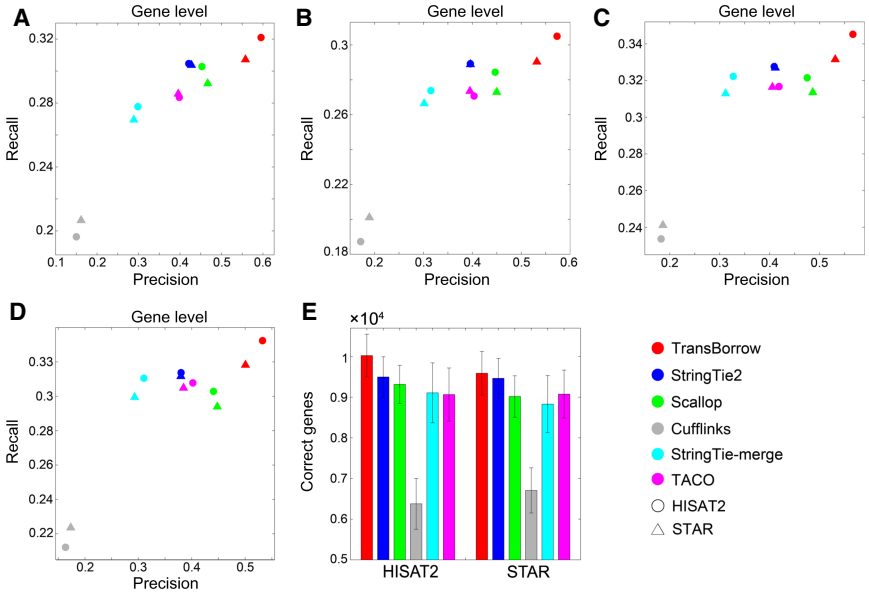

Figure 4. Accuracy comparisons of the assemblers on the four real data sets at the gene level. (*A–D*) Comparisons of assembly accuracy of the assemblers on data sets R1, R2, R3, and R4, respectively. (*E*) The average number of correctly detected genes by the assemblers on data sets R1, R2, R3, and R4.

Comparison of identifying transcripts with different expression levels

For real data, the exact expression abundances of the transcripts were unknown to us. To make relatively fair comparisons of the assemblers in identifying transcripts with different expression levels on real data, we first estimated the expression levels (TPM values) of the whole-reference transcripts using the well-known abundance estimator kallisto (Bray et al. 2016), based on which the reference transcripts could be equally divided into three parts corresponding to the transcripts with low, medium, and high expression levels as we did on the simulated data.

In comparison with any of the other assemblers, TransBorrow correctly detected more reference transcripts, regardless of expression levels, for all four real data sets based on both HISAT2 and STAR mappings (for details, see Fig. 5A–C). It is worth mentioning that based on HISAT2 mappings, TransBorrow correctly detected 44.19%–54.53% more lowly expressed transcripts from the four data sets than StringTie2, 51.66%–79.37% more than Scallop, 191.72%–361.22% more than Cufflinks, 52.69%–93.16% more than StringTie-merge, and 21.18%–59.15% more than TACO. For the STAR mappings, TransBorrow correctly detected 31.91%–36.22% more lowly expressed transcripts from the four data sets than StringTie2, 21.81%–37.8% more than Scallop, 161.78%–312.8% more than Cufflinks, 47.39%–84.97% more than StringTie-merge, and 11%–28.71% more than TACO. Therefore,

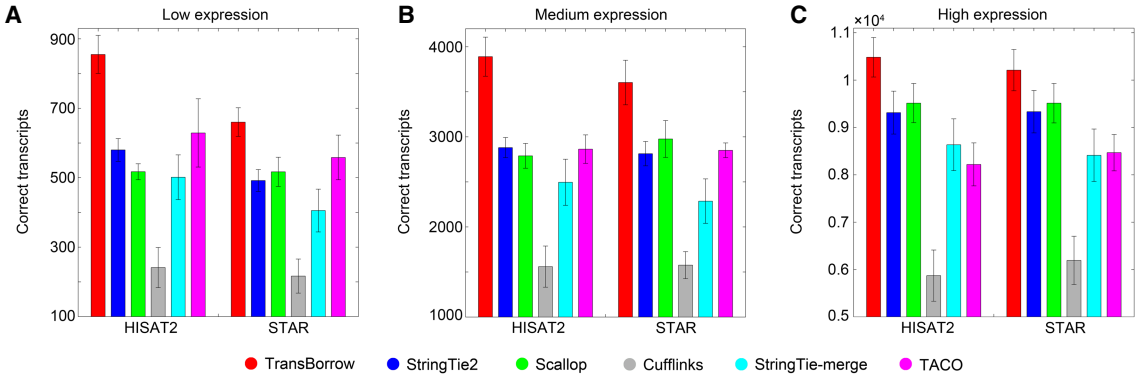

Figure 5. Performance comparisons of the assemblers in identifying transcripts with different expression levels on the real data. (*A–C*) The average number of correctly assembled transcripts with different expression levels by the assemblers on data sets R1, R2, R3, and R4.

Downloaded from genome.cshlp.org on May 2, 2026 . Published by Cold Spring Harbor Laboratory Press

the comparison showed that TransBorrow consistently maintained considerably superior performance in identifying lowly expressed transcripts not only on simulated data but also on real data sets.

Comparison of running time and memory usage

To compare the running time and memory usage of the assemblers, all the assemblers were run on the same server with 96 GB of memory and a 12-core CPU. The results showed that Scallop and StringTie2 ran the fastest on all the four real data sets. TransBorrow ran slightly slower than Scallop and StringTie2, but several times faster than Cufflinks. For example, on data sets R2 (88 million reads) and R3 (41 million reads) under STAR mappings, the running time of StringTie2 was 10 and 6 min, respectively. The running time of Scallop was 24 and 9 min, respectively, and the running time of TransBorrow was 45 and 33 min, respectively. However, the running time of Cufflinks was 148 and 97 min, respectively. For memory usage, the memory cost by StringTie2 was ~1 GB on all the four data sets, and all the other assemblers showed a similar trend, with maximum memory usage of no more than 10 GB in most cases (for the running time and memory usage of all the four real data sets, see Supplemental Table S5). Overall, although TransBorrow is not the most efficient in running time and memory usage, it is quite acceptable for practical use. The merging-based approaches, StringTie-merge and TACO, are different from the above assemblers because they only take the assemblies from the other assemblers as their inputs. Their running time was much faster, and their memory usage was also less than that of all the above assemblers (for details, see Supplemental Table S5).

Additional evaluations

In addition to the evaluations presented above, the Supplemental Material also includes evaluating the assembly accuracy of all assemblers on an additional 101 RNA-seq samples from the species *Homo sapiens*, *Saccharomyces cerevisiae*, *Drosophila melanogaster*, *Caenorhabditis elegans*, *Mus musculus*, *Arabidopsis thaliana*, and *Zea mays* (see Supplemental Figs. S1–S4). The assembly accuracy of the assemblers by using spike-in and single-cell RNA-seq data sets (Supplemental Figs. S5–S7), the performance of the assemblers at identifying long noncoding transcripts (see Supplemental Fig. S8) and the *F*-score of the assemblers on all the data sets (see Supplemental Figs. S9–S12) were also evaluated in the Supplemental Material. Moreover, some other evaluations were also performed, including the comparison of the performance of TransBorrow (by using the assemblies from StringTie, Scallop, and Cufflinks) with that of StringTie2 (see Supplemental Fig. S13), the recall/precision curves of each assembler on all the data sets (see Supplemental Figs. S14–S17), the comparison of the performance of TransBorrow with an approach that simply combined the assembled transcripts from different assemblers (see Supplemental Fig. S18), and the assembled transcripts of a gene with complicated splice junction patterns in the form of a Genome Browser snapshot (for details, see Supplemental Fig. S19).

Discussion

In this study, we present a novel genome-guided assembler, TransBorrow, for transcriptome assembly using short RNA-seq reads. Compared with three leading assemblers of the same kind on both simulated and real data sets, TransBorrow consistently performs the best under commonly used criteria. The superiority of TransBorrow may be attributed to the following.

First, TransBorrow attempts to identify all expressed transcripts by taking advantage of different assemblies from other assemblers. The reliable subsequences generated in this step serve as seeds and effectively guide the subsequent assembly process. Second, TransBorrow develops a new graph model, the colored graph, which was built by merging different assemblies. Based on colored graphs, reliable subsequences could be accurately and efficiently extracted from merged assemblies. Third, TransBorrow constructs a weighted line graph for each splicing graph, whose edge weight exactly indicates the correct connections between the incoming and outgoing edges for each node of the splicing graph. Fourth, TransBorrow implements a newly designed path extension strategy for searching for a transcript-representing path cover over each weighted line graph by seeding the extracted reliable subpaths and iteratively choosing the best neighbor for extension.

Although we have seen great advantages of TransBorrow, further improvements could still be made for TransBorrow in the future. For example, the current version of TransBorrow is not compatible with long-read RNA-seq data sets (e.g., Pacific Biosciences [PacBio] or Oxford Nanopore Technologies [ONT]). Similar to other assemblers, StringTie2, Scallop, and Cufflinks, the current version of TransBorrow performs transcriptome assembly in each individual gene locus without considering the resolution of chimeric transcripts. And the current version of TransBorrow is a genome-guided assembler, which is not compatible with de novo assemblies. The future version of TransBorrow will attempt to solve these problems and make further improvements. For the development of TransBorrow, difficulties existed when we were building it. For example, different assemblers used by TransBorrow may generate quite different exon–intron boundaries for some genes, which makes the colored graphs very complicated. In terms of the efficiency of TransBorrow, it depends in part on the performance of the borrowed assemblies, and we tried to optimize the performance of the assemblers that were used upstream of TransBorrow by adjusting their parameters, such as minimum isoform abundance, minimum transcript length, and minimum junction coverage, and the performance of TransBorrow was only slightly affected by these operations. Therefore, running the assemblers upstream of TransBorrow by using their default parameters would be good. In addition, we tried to replace Cufflinks with two other assemblers, CLASS2 and Strawberry, and tested the performance of TransBorrow. We found that the performance of TransBorrow was not improved when replacing Cufflinks with CLASS2 or Strawberry. For example, on the simulated data set, the recall and precision of TransBorrow by using StringTie2, Scallop, and Cufflinks were 57.12% and 60.32% versus 57.52% and 55.07% when replacing Cufflinks with CLASS2 and 57.14% and 55.57% when replacing Cufflinks with Strawberry. Therefore, we ran TransBorrow by using the assemblies from Cufflinks. Moreover, we also tested TransBorrow on four single-cell RNA-seq data sets, and it also showed appreciable improvements over the applied assemblers (see Supplemental Material; Supplemental Fig. S7).

Tools such as EvidentialGene, Concatenation, and Mikado also perform assemblies by combining assemblies from different assemblers, which is similar to TransBorrow. Different from the three tools, TransBorrow performs transcriptome assembly from the original read mapping results by building splicing graphs and searching for path covers over splicing graphs, and those

Downloaded from genome.cshlp.org on May 2, 2026 . Published by Cold Spring Harbor Laboratory Press

combined assemblies from different assemblers effectively provide reliable subpaths for TransBorrow guiding its accurate assembly. However, the three tools EvidentialGene, Concatenation, and Mikado perform assemblies completely based on the results from other assemblers, which clearly limits their performance in identifying novel transcripts. In addition, the performance of Mikado also relied on additional reference information, such as the BLAST information, whereas the current version of TransBorrow does not make use of additional reference information. The two assemblers EvidentialGene and Concatenation were designed for processing combined assemblies in the FASTA format, different from TransBorrow and Mikado, which use assemblies in the GTF format.

To the best of our knowledge, TransBorrow is the first genome-guided transcriptome assembler that uses assemblies from different tools by searching for reliable assembly subpaths from different assemblies and then seeding these subpaths for transcript-representing path extensions in each splicing graph. The software has been developed to be user-friendly and is expected to play a crucial role in new discoveries of transcriptome studies using RNA-seq, especially in complicated human diseases related to abnormal splicing events and expression levels, such as cancers.

Methods

Building splicing graphs and extracting reliable paired subpaths

Assembly of expressed transcripts in this study is completed with the traditional graph model, the splicing graph. Therefore, we first build accurate splicing graphs and then collect all subpaths in the graphs supported by the paired-end sequencing reads.

Building splicing graphs

The splicing graph is constructed from mapping the RNA-seq reads to a reference genome using mapping tools such as HISAT2 or STAR. According to the mapping results, reads are usually clustered into corresponding gene loci, and a splicing graph is generally built for each gene. The exon–intron boundaries and exon–exon junctions are derived from those mapping reads spanning two or more exons. Generally, each node in a splicing graph represents an exon in the corresponding gene, and a directed edge between two nodes means a splicing junction between the two exons. For a splicing graph, we assign a weight to each edge depending on the number of reads spanning it.

Theoretically, edges in the splicing graphs can capture most splicing events in the expressed transcripts, and the sequencing depth information is appropriately integrated into the graph as the edge weight. Then the task of transcript assembly is to accurately search for an edge-path cover for the splicing graph, each path of which represents an expressed transcript.

Extracting reliable paired subpaths

To make full use of the paired-end information in the subsequent assembly procedure, we search for all the subpaths in each splicing graph supported by the paired-end reads, named paired subpaths.

In detail, for each two paired-end reads r_1 and r_2, if r_1 spans a path $P_1 = n_{i1} \to n_{i2} \to \ldots \to n_{ip}$ in a splicing graph G while r_2 spans $P_2 = n_{j1} \to n_{j2} \to \ldots \to n_{jq}$, we will search for all paths from n_{ip} to n_{j1} in graph G. If there exists one and only one path $P_{in} = n_{ip} \to n_{m1} \to n_{m2} \to \ldots \to n_{ms} \to n_{j1}$ between n_{ip} and n_{j1} satisfying $p + s + q \geq 3$, then reads r_1 and r_2 are connected by the path P_{in}, and the corresponding paired subpath is extracted as $P = P_1 \to P_{in} \to P_2$ (see Fig. 1A). After all the paired-end reads in graph G are processed, we obtain a set S_p of all paired subpaths. Different paired-end reads may generate the same paired path. Therefore, we record the number of paired-end reads that generate each paired subpath P, and this number is called the coverage of path P.

In fact, some paired subpaths may be erroneously extracted because of mapping or sequencing errors. These paired subpaths usually show very low coverage and need to be removed. Although a paired subpath P is erroneously extracted, the subpaths of P may be reliable and should not be removed. Therefore, we attempt to obtain all reliable paired subpaths by the following steps. Given a paired subpath P, we first extract all the subpaths of length three (called paired 3-subpaths) by decomposing the paired subpath P. For example, a paired subpath $P = n_1 \to n_2 \to n_3 \to n_4$ will be decomposed into two paired 3-subpaths, $P_1 = n_1 \to n_2 \to n_3$ and $P_2 = n_2 \to n_3 \to n_4$. Different paired subpaths usually have different coverage, whereas they may generate the same paired 3-subpath. Therefore, for each paired 3-subpath, we record the sum of coverage of all paired subpaths that generate the paired 3-subpath, and this number is called the coverage of the paired 3-subpath. A paired 3-subpath is defined as a reliable paired 3-subpath if its coverage is no less than two, and all reliable paired 3-subpaths can be sorted by their coverage. Similarly, we could extract all reliable paired 4-subpaths, paired 5-subpaths, …, and paired n-subpaths, where n is the length of the longest paired subpath (see Fig. 1A). Finally, we can obtain all reliable paired subpaths of different lengths, cluster them according to their lengths, and sort them in each cluster by their coverage.

Building colored graphs and extracting reliable assembly sequences

The main contribution of TransBorrow is to take advantage of the assemblies from different assemblers, which is achieved by extracting all reliable sequences of the assembled transcripts from different assembly tools. These extracted reliable sequences together with the above reliable paired subpaths then serve as key information guiding the subsequent assembly procedure.

Building colored graphs

To accurately search for all reliable subsequences, we first build a novel graph model, the colored graph, as follows. Given the merged transcripts assembled by two or more different assemblers, we first cluster all the transcripts into different gene loci. For each gene locus, a colored graph G_c is constructed with nodes and edges representing the exons and splice junctions appearing in the merged transcripts. Then each of the merged transcripts corresponds to a unique path in the colored graph, named an assembled-transcript-representing path. As each graph G_c is built from those assembled-transcript-representing paths and the paths belong to different assemblers, we call graph G_c a colored graph (see Fig. 1B). Based on colored graphs, reliable sequences of the merged transcripts can be effectively extracted as follows.

Extracting reliable assembly subpaths in colored graphs

The merged transcripts were predicted by different assemblers; therefore, the merged transcripts usually contain more true positives than the single assembly by any of the assemblers. However, the false positives would also be much more than each single assembly. To extract reliable sequences from the merged transcripts, we take the following steps. It is worth mentioning that a reliable sequence means a segment or the whole of an assembled transcript, which corresponds to a unique subpath of a colored graph.

Downloaded from genome.cshlp.org on May 2, 2026 . Published by Cold Spring Harbor Laboratory Press

In theory, if a subpath of a colored graph is covered by a transcript assembled by only one assembler, then this subpath has very low reliability. However, if a subpath is detected by two or more assemblers, then it should have relatively higher reliability. To obtain reliable assembly subpaths based on such considerations, we first extract all the subpaths of length three (called assembly 3-subpaths) by decomposing each assembled-transcript-representing path in the colored graph, similar to that in the subsection "Extracting reliable paired subpaths." Different assemblers may generate the same assembly 3-subpath; we record the number of assemblers that generate the assembly 3-subpath, and the number is named the depth of the assembly 3-subpath. An assembly 3-subpath is defined as a reliable assembly 3-subpath if its depth is no less than two, and all reliable assembly 3-subpaths can be sorted by their depth. Similarly, we could extract all the reliable assembly 4-subpaths, assembly 5-subpaths, …, and assembly m-subpaths, where m is the length of the longest assembled-transcript-representing path (see Fig. 1B). Finally, we can obtain all reliable assembly subpaths of different lengths, cluster them according to their lengths, and sort them in each cluster by their depth.

Mapping reliable subpaths to the splicing graphs

The assembly procedure is performed on splicing graphs, and all reliable paired subpaths and assembly subpaths actually guide the assembly process on splicing graphs. Therefore, we need to map all the reliable assembly subpaths to splicing graphs; then each reliable assembly subpath corresponds to a unique subpath of a splicing graph. To effectively map those assembly subpaths to the splicing graphs, we first build a hash table recording all splicing graphs. For each edge in a splicing graph, the key of the hash table records its corresponding splicing junction positions on a specific chromosome, and the value of the hash table records its graph and edge indexes. Then each assembly subpath could be effectively located to a splicing graph according to the splice junctions in the assembly subpath.

After mapping all the reliable assembly subpaths to splicing graphs, we combine the assembly subpaths and paired subpaths and remove the redundant subpaths (for the redundant subpaths appearing both in the set of assembly subpaths and paired subpaths, we keep only one copy), and the combined subpaths are called reliable subpaths (see Fig. 1C). These reliable subpaths will serve as the seeds and guide the subsequent transcript assembly.

Searching for transcript-representing paths by seeding reliable paths

Theoretically, each reliable subpath corresponds to a segment of an expressed transcript and therefore should be covered by at least one transcript to be assembled. To achieve this goal, we first create a weighted line graph for each splicing graph, and then a transcript-representing path cover over each line graph is obtained by a newly designed path extension strategy.

Building the weighted line graph

To accurately connect the incoming and outgoing edges of each node in a splicing graph G, we first build a line graph $L(G)$ of the splicing graph G with nodes representing the edges in G and an edge representing two incident edges in G. The weight of each node in $L(G)$ is defined by the coverage of the corresponding edge in G, whereas each edge is weighted by solving a quadratic program updated from our previous study (Liu et al. 2019). The quadratic program effectively integrates the information from transcript coverage differences and extracts reliable subpaths (for details, see Supplemental Material). Then, the edge coverage in $L(G)$ clearly indicates the correct connections between incoming and outgoing edges of each node in graph G, with an edge (e_i, e_j) weight in $L(G)$ of 1, meaning that this edge comes from an expressed transcript with a high probability and zero otherwise (see Fig. 1D).

Searching for transcripts by a novel path extension technique

After assigning weights to both the nodes and edges of the weighted line graph $L(G)$, a newly designed path extension strategy was applied to assemble all the expressed transcripts by searching for an optimal transcript-representing path cover over the weighted line graph. Before processing path extension, all the reliable subpaths of the splicing graphs should correspond to the paths of the line graphs, and a reliable n-subpath in a splicing graph should correspond to a unique reliable $(n-1)$-subpath in the corresponding line graph. Then TransBorrow searches for all the expressed transcripts for a line graph $L(G)$ by the following steps.

Step 1. Choose the longest reliable subpath $p_r = p_{r1} \to p_{r2} \to \ldots \to p_{rm}$ that is not covered by any predicted path (or choose an unused node in $L(G)$ if all the reliable subpaths have been included in the predicted paths) as a seed and extend it to one of its right neighbors, with the edge weight being one. If there are multiple choices, we choose the neighbor n_i, which is supported by a reliable 2-subpath $p_{rm} \to n_i$. If there are still multiple choices, we choose the neighbor n_j, which is supported by a reliable 3-subpath $p_{r(m-1)} \to p_{rm} \to n_j$. The process is iterated until no neighbor is supported by any longer reliable subpath. If there are still multiple choices, then the neighbor with the largest node weight is selected for extension. The extension is continued until the last node of the current path has no outgoing edges. A similar process could be performed for left extensions, and then a transcript-representing path p_t is predicted.

Step 2. Define c_{min} as the minimum node weight in the extended path p_t, and then we update each node weight $c(n)$ of the line graph $L(G)$ to be $c(n)$-c_{min} if node n is included in the path p_t.

The above extension process is repeated until all the reliable subpaths and nodes in the line graph $L(G)$ have been covered by the predicted paths, and then a transcript-representing path cover over the line graph is obtained, each path of which corresponds to a unique path in the splicing graph G.

Software availability

The source code for the latest version of TransBorrow package is available at https://sourceforge.net/projects/transcriptomeassembly/files/TransBorrow/ and as Supplemental Code.

Data access

The simulated data set used in this study is available at https://sourceforge.net/projects/transcriptomeassembly/files/TransBorrow/Data/. All the real data sets were downloaded from NCBI SRA with the accession codes recorded in Supplemental Table S4. The reference genome and transcripts used for evaluating the performance of the assemblers are described in Supplemental Material.

Competing interest statement

The authors declare no competing interests.

Downloaded from genome.cshlp.org on May 2, 2026 . Published by Cold Spring Harbor Laboratory Press

Acknowledgments

This work was supported by the National Natural Science Foundation of China with code 61801265, and King Abdullah University of Science and Technology (KAUST) Office of Sponsored Research (OSR) under award numbers BAS/1/1624-01, FCC/1/1976-18-01, FCC/1/1976-23-01, FCC/1/1976-25-01, FCC/1/1976-26-01, REI/1/0018-01-01, REI/1/4216-01-01, and URF/1/4098-01-01. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Author contributions: J.L. conceived and designed the experiments. J.L., T.Y., and Z.M. performed the experiments. T.Y., Z.F., and X.L. analyzed the data. J.L., X.G., Z.F., and X.L. contributed reagents/materials/analysis tools. J.L., Z.M., and X.G. wrote the paper. J.L. and T.Y. designed the software used in analysis. J.L. and X.G. oversaw the project.

References

Alhakami H, Mirebrahim H, Lonardi S. 2017. A comparative evaluation of genome assembly reconciliation tools. *Genome Biol* **18:** 93. doi:10.1186/s13059-017-1213-3

Au KF, Jiang H, Lin L, Xing Y, Wong WH. 2010. Detection of splice junctions from paired-end RNA-seq data by SpliceMap. *Nucleic Acids Res* **38:** 4570–4578. doi:10.1093/nar/gkq211

Behr J, Kahles A, Zhong Y, Sreedharan VT, Drewe P, Rätsch G. 2013. MITIE: simultaneous RNA-Seq-based transcript identification and quantification in multiple samples. *Bioinformatics* **29:** 2529–2538. doi:10.1093/bioinformatics/btt442

Bray NL, Pimentel H, Melsted P, Pachter L. 2016. Near-optimal probabilistic RNA-seq quantification. *Nat Biotechnol* **34:** 525–527. doi:10.1038/nbt.3519

Canzar S, Andreotti S, Weese D, Reinert K, Klau GW. 2016. CIDANE: comprehensive isoform discovery and abundance estimation. *Genome Biol* **17:** 16. doi:10.1186/s13059-015-0865-0

Cerveau N, Jackson DJ. 2016. Combining independent *de novo* assemblies optimizes the coding transcriptome for nonconventional model eukaryotic organisms. *BMC Bioinformatics* **17:** 525. doi:10.1186/s12859-016-1406-x

Chang Z, Li GJ, Liu JT, Zhang Y, Ashby C, Liu DL, Cramer CL, Huang XZ. 2015. Bridger: a new framework for *de novo* transcriptome assembly using RNA-seq data. *Genome Biol* **16:** 30. doi:10.1186/s13059-015-0596-2

Dobin A, Davis CA, Schlesinger F, Drenkow J, Zaleski C, Jha S, Batut P, Chaisson M, Gingeras TR. 2013. STAR: ultrafast universal RNA-seq aligner. *Bioinformatics* **29:** 15–21. doi:10.1093/bioinformatics/bts635

Feng JX, Li W, Jiang T. 2011. Inference of isoforms from short sequence reads. *J Comput Biol* **18:** 305–321. doi:10.1089/cmb.2010.0243

Gilbert DG. 2013. Gene-omes built from mRNA-seq not genome DNA. *7th Annual Arthropod Genomics Symposium*, Notre Dame. *F1000Research* doi:10.7490/f1000research.1112594.1

Grabherr MG, Haas BJ, Yassour M, Levin JZ, Thompson DA, Amit I, Adiconis X, Fan L, Raychowdhury R, Zeng Q, et al. 2011. Full-length transcriptome assembly from RNA-Seq data without a reference genome. *Nat Biotechnol* **29:** 644–652. doi:10.1038/nbt.1883

Griebel T, Zacher B, Ribeca P, Raineri E, Lacroix V, Guigó R, Sammeth M. 2012. Modelling and simulating generic RNA-Seq experiments with the flux simulator. *Nucleic Acids Res* **40:** 10073–10083. doi:10.1093/nar/gks666

Guttman M, Garber M, Levin JZ, Donaghey J, Robinson J, Adiconis X, Fan L, Koziol MJ, Gnirke A, Nusbaum C, et al. 2010. *Ab initio* reconstruction of cell type–specific transcriptomes in mouse reveals the conserved multiexonic structure of lincRNAs. *Nat Biotechnol* **28:** 503–510. doi:10.1038/nbt.1633

Haas BJ, Delcher AL, Mount SM, Wortman JR, Smith RK, Hannick LI, Maiti R, Ronning CM, Rusch DB, Town CD, et al. 2003. Improving the *Arabidopsis* genome annotation using maximal transcript alignment assemblies. *Nucleic Acids Res* **31:** 5654–5666. doi:10.1093/nar/gkg770

Kim D, Pertea G, Trapnell C, Pimentel H, Kelley R, Salzberg SL. 2013. TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. *Genome Biol* **14:** R36. doi:10.1186/gb-2013-14-1-r4

Kim D, Langmead B, Salzberg SL. 2015. HISAT: a fast spliced aligner with low memory requirements. *Nat Methods* **12:** 357–360. doi:10.1038/nmeth.3317

Kovaka S, Zimin AV, Pertea GM, Razaghi R, Salzberg SL, Pertea M. 2019. Transcriptome assembly from long-read RNA-seq alignments with StringTie2. *Genome Biol* **20:** 278. doi:10.1186/s13059-019-1910-1

Li W, Jiang T. 2012. Transcriptome assembly and isoform expression level estimation from biased RNA-Seq reads. *Bioinformatics* **28:** 2914–2921. doi:10.1093/bioinformatics/bts559

Li W, Feng JX, Jiang T. 2011. IsoLasso: a LASSO regression approach to RNA-Seq based transcriptome assembly. *J Comput Biol* **18:** 1693–1707. doi:10.1089/cmb.2011.0171

Lin SH, Liao YC. 2013. CISA: contig integrator for sequence assembly of bacterial genomes. *PLoS One* **8:** e60843. doi:10.1371/journal.pone.0060843

Liu J, Li G, Chang Z, Yu T, Liu B, McMullen R, Chen P, Huang X. 2016a. BinPacker: packing-based *de novo* transcriptome assembly from RNA-seq data. *PLoS Comput Biol* **12:** e1004772. doi:10.1371/journal.pcbi.1004772

Liu J, Yu T, Jiang T, Li G. 2016b. TransComb: genome-guided transcriptome assembly via combing junctions in splicing graphs. *Genome Biol* **17:** 213. doi:10.1186/s13059-016-1074-1

Liu J, Yu T, Mu Z, Li G. 2019. TransLiG: a de novo transcriptome assembler that uses line graph iteration. *Genome Biol* **20:** 81. doi:10.1186/s13059-019-1690-7

Marguerat S, Bähler J. 2010. RNA-seq: from technology to biology. *Cell Mol Life Sci* **67:** 569–579. doi:10.1007/s00018-009-0180-6

Marioni JC, Mason CE, Mane SM, Stephens M, Gilad Y. 2008. RNA-seq: an assessment of technical reproducibility and comparison with gene expression arrays. *Genome Res* **18:** 1509–1517. doi:10.1101/gr.079558.108

Metzker ML. 2010. Sequencing technologies: the next generation. *Nat Rev Genet* **11:** 31–46. doi:10.1038/nrg2626

Mezlini AM, Smith EJM, Fiume M, Buske O, Savich GL, Shah S, Aparicio S, Chiang DY, Goldenberg A, Brudno M. 2013. iReckon: simultaneous isoform discovery and abundance estimation from RNA-seq data. *Genome Res* **23:** 519–529. doi:10.1101/gr.142232.112

Niknafs YS, Pandian B, Iyer HK, Chinnaiyan AM, Iyer MK. 2017. TACO produces robust multisample transcriptome assemblies from RNA-seq. *Nat Methods* **14:** 68–70. doi:10.1038/nmeth.4078

Ozsolak F, Milos PM. 2011. RNA sequencing: advances, challenges and opportunities. *Nat Rev Genet* **12:** 87–98. doi:10.1038/nrg2934

Peng Y, Leung HC, Yiu SM, Lv MJ, Zhu XG, Chin FY. 2013. IDBA-tran: a more robust de novo de Bruijn graph assembler for transcriptomes with uneven expression levels. *Bioinformatics* **29:** i326–i334. doi:10.1093/bioinformatics/btt219

Pertea M, Pertea GM, Antonescu CM, Chang TC, Mendell JT, Salzberg SL. 2015. StringTie enables improved reconstruction of a transcriptome from RNA-seq reads. *Nat Biotechnol* **33:** 290–295. doi:10.1038/nbt.3122

Robertson G, Schein J, Chiu R, Corbett R, Field M, Jackman SD, Mungall K, Lee S, Okada HM, Qian JQ, et al. 2010. *De novo* assembly and analysis of RNA-seq data. *Nat Methods* **7:** 909–912. doi:10.1038/nmeth.1517

Shao M, Kingsford C. 2017. Accurate assembly of transcripts through phase-preserving graph decomposition. *Nat Biotechnol* **35:** 1167–1169. doi:10.1038/nbt.4020

Simpson JT, Wong K, Jackman SD, Schein JE, Jones SJ, Birol I. 2009. ABySS: a parallel assembler for short read sequence data. *Genome Res* **19:** 1117–1123. doi:10.1101/gr.089532.108

Smith-Unna R, Boursnell C, Patro R, Hibberd JM, Kelly S. 2016. TransRate: reference-free quality assessment of de novo transcriptome assemblies. *Genome Res* **26:** 1134–1144. doi:10.1101/gr.196469.115

Song L, Sabunciyan S, Florea L. 2016. CLASS2: accurate and efficient splice variant annotation from RNA-seq reads. *Nucleic Acids Res* **44:** e98. doi:10.1093/nar/gkw053

Soto-Jimenez LM, Estrada K, Sanchez-Flores A. 2014. GARM: genome assembly, reconciliation and merging pipeline. *Curr Top Med Chem* **14:** 418–424. doi:10.2174/1568026613666131204110628

Soueidan H, Maurier F, Groppi A, Sirand-Pugnet P, Tardy F, Citti C, Dupuy V, Nikolski M. 2013. Finishing bacterial genome assemblies with Mix. *BMC Bioinformatics* **14**(Suppl 15)**:** S16. doi:10.1186/1471-2105-14-S15-S16

Tomescu AI, Kuosmanen A, Rizzi R, Mäkinen V. 2013. A novel min-cost flow method for estimating transcript expression with RNA-Seq. *BMC Bioinformatics* **14**(Suppl 5)**:** S15. doi:10.1186/1471-2105-14-S5-S15

Trapnell C, Pachter L, Salzberg SL. 2009. TopHat: discovering splice junctions with RNA-Seq. *Bioinformatics* **25:** 1105–1111. doi:10.1093/bioinformatics/btp120

Trapnell C, Williams BA, Pertea G, Mortazavi A, Kwan G, van Baren MJ, Salzberg SL, Wold BJ, Pachter L. 2010. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nat Biotechnol* **28:** 511–515. doi:10.1038/nbt.1621

Venturini L, Caim S, Kaithakottil GG, Mapleson DL, Swarbreck D. 2018. Leveraging multiple transcriptome assembly methods for improved

Downloaded from genome.cshlp.org on May 2, 2026 . Published by Cold Spring Harbor Laboratory Press

gene structure annotation. *Gigascience* **7:** giy093. doi:10.1093/gigascience/giy093

Vicedomini R, Vezzi F, Scalabrin S, Arvestad L, Policriti A. 2013. GAM-NGS: genomic assemblies merger for next generation sequencing. *BMC Bioinformatics* **14**(Suppl 7)**:** S6. doi:10.1186/1471-2105-14-S7-S6

Voshall A, Moriyama EN. 2018. Next-generation transcriptome assembly: strategies and performance analysis. In *Bioinformatics in the era of post genomics and big data* (ed. Abdurakhmonov I), pp. 15–36. IntechOpen, London.

Wang Z, Gerstein M, Snyder M. 2009. RNA-Seq: a revolutionary tool for transcriptomics. *Nat Rev Genet* **10:** 57–63. doi:10.1038/nrg2484

Wences AH, Schatz MC. 2015. Metassembler: merging and optimizing de novo genome assemblies. *Genome Biol* **16:** 207. doi:10.1186/s13059-015-0764-4

Wilhelm BT, Landry J-R. 2009. RNA-Seq: quantitative measurement of expression through massively parallel RNA-sequencing. *Methods* **48:** 249–257. doi:10.1016/j.ymeth.2009.03.016

Wu TD, Nacu S. 2010. Fast and SNP-tolerant detection of complex variants and splicing in short reads. *Bioinformatics* **26:** 873–881. doi:10.1093/bioinformatics/btq057

Xie YL, Wu GX, Tang JB, Luo RB, Patterson J, Liu SL, Huang WH, He GZ, Gu SC, Li SK, et al. 2014. SOAPdenovo-Trans: *de novo* transcriptome assembly with short RNA-Seq reads. *Bioinformatics* **30:** 1660–1666. doi:10.1093/bioinformatics/btu077

Yao G, Ye L, Gao H, Minx P, Warren WC, Weinstock GM. 2012. Graph accordance of next-generation sequence assemblies. *Bioinformatics* **28:** 13–16. doi:10.1093/bioinformatics/btr588

Received September 30, 2019; accepted in revised form June 18, 2020.