Supplementary Material for
"Accurate assembly of transcripts through phase-preserving graph decomposition"

Mingfu Shao and Carl Kingsford

Supplementary Note 1: Additional Related Work

Transcript assembly methods can be divided into reference-based methods and *de novo* methods, depending on whether a reference genome is available. Reference-based methods usually first use the read alignment produced by an RNA-seq aligner (e.g., TopHat2 [16], GSNAP [23], SpliceMap [24], STAR [17], and HISAT [18]) to create a splice graph for each gene locus, and then reconstruct the expressed transcripts by decomposing the graph. These methods mainly differ in the way they decompose the splice graph. For instance, StringTie [13] iteratively computes the heaviest path in the splice graph, collects that path, and updates the weights of the remaining graph via a max-flow formulation. TransComb [14] employs a bin-packing strategy to gradually reconstruct paths guided by the weighted junction graph. CLASS [9] uses a set-cover formulation to compute a smallest set of paths that collectively satisfy all contiguity constraints derived from spliced reads and paired-end reads. CLASS enumerates all possible *s*-*t* paths of the splice graph and uses a linear programming formulation to score each of them. (Scallop uses linear programming for a different purpose, i.e., to reassign weights for the newly created edges in the case of decomposing a single unsplittable vertex.) Unlike most of the reference-based methods, which use a splice graph, Cufflinks uses the overlap graph as the underlying data structure. This allows it to compute a minimum-sized set of paths that can explain all the reads in polynomial-time using Dilworth's theorem. The disadvantage of using an overlap graph is that the coverage of exons and junctions cannot be easily modeled. An earlier method, PASA [25], uses a dynamic programming algorithm to identify spliced variants from expressed sequence tags (ESTs) and cDNA sequences. PASA creates unique maximal assemblies by merging sets of compatible overlapping alignments.

De novo assembly methods (e.g., TransABySS [26], Trinity [27], SOAPdenovo-Trans [28], Oases [29], IDBA-Tran [30], Bridger [31], and Shannon [32]) usually use a de Bruijn graph to organize all the reads. The corresponding splice graph can be obtained by collapsing simple paths and resolving loops of the de Bruijn graph. The expressed transcripts can then be reconstructed by decomposing the splice graph. *De novo* methods are mainly used for non-model species and cancer samples, for which a reference genome is unavailable or significantly diverged. When a high-quality reference genome is available, reference-based methods usually obtain better accuracy, since they can tolerate much lower read coverage inside exons.

Transcript assembly is still an open and very challenging problem and remains an active area of research. In spite of the numerous developed methods, none of them is highly accurate when multiple splice forms, introns, and sequencing and alignment errors are present (see [15] for a detailed comparison of the existing transcript assemblers).

Supplementary Note 2: Experimental Setup and Evaluation

We use three RNA-seq aligners, TopHat2, STAR, and HISAT2 (Supplementary Table 2), to map the sequencing reads to the reference genome (GRCh38). Taking the read alignments as (the only) input, each method (Scallop, StringTie, and TransComb) predicts a set of expressed transcripts. (The current version of TransComb does not support HISAT2 alignments, so we only compare StringTie and Scallop when using HISAT2.) For the biological RNA-seq samples (the 10 RNA-seq samples and the 65 ENCODE RNA-seq samples) for which we do not have ground truth expressed transcripts, we evaluate the predicted transcripts by comparing them with the entire reference annotation database of known human transcripts (ENSEMBL release-87), as is commonly done [11, 12, 13, 14]. Usually for a given sample only a small subset of the transcripts in the reference annotation are expressed, and it is also likely that some predicted transcripts are novel and thus are not in the current reference annotation. For the 8 spike-in RNA-seq samples, we directly use the known ground truth expressed transcripts (i.e., sequins [33]) to evaluate the predicted transcripts.

Given a set of predicted transcripts and the ground truth, we use the standard tool gffcompare to compute accuracy, where gffcompare defines a predicted multi-exon transcript as correct if its intron chain can be exactly matched to a (multi-exon) transcript in the ground truth, while a predicted single-exon transcript is defined as correct if it overlaps at least 80% with a (single-exon) transcript in the ground truth. The gffcompare tool then reports sensitivity (the ratio between the number of correct transcripts and the total number of ground truth transcripts) and precision (the ratio between the number of correct transcripts and the total number of predicted transcripts) as measurements for the accuracy of the predicted transcripts.

All three methods (StringTie, TransComb, and Scallop) support specifying a parameter, the minimum coverage threshold, to control the minimum expression abundance of a predicted transcript. Since highly expressed transcripts are easier to assemble, this parameter essentially balances the sensitivity and precision. We run these methods on different thresholds and draw the precision-sensitivity curve to see their capability to balance sensitivity and precision (Figure 2A, Supplementary Figure 2A, Supplementary Figure 12).

To test these methods on assembling transcripts for different expression levels (results shown in Figure 2F and Supplementary Figure 2F), we use Salmon [19] to quantify each of the 10 RNA-seq samples using the human annotation (ENSEMBL release 87) as reference. For each sample, we collect all multi-exon transcripts with abundance larger than a threshold (TPM ≥ 0.1) and divide them into three equal subsets corresponding to low, middle and high expression levels. We then use gffcompare tool to compute the accuracy of the methods with each subset as ground truth. This gives the sensitivity and precision of each method for different expression levels.

Supplementary Note 3: Comparison on 65 ENCODE RNA-seq Samples

We collect all human RNA-seq paired-end samples from the ENCODE project (https://www.encodeproject.org, 2013–present) that provide pre-computed read alignments (for experiments with more than one such sample, we arbitrarily select one). This yielded 50 strand-specific and 15 non-strand-specific samples. Since the three methods use different parameters to balance precision and sensitivity, to compare them on equal footing, we compute an adjusted sensitivity and adjusted precision. Specifically, for each sample, we fix the precision ϕ of the method with highest precision, and for each of the other two methods with smaller precision, we discard its predicted transcripts from the lowest coverage until the precision equals ϕ; the adjusted sensitivity is the sensitivity at this precision ϕ. We compute the adjusted precision analogously, through filtering low-coverage transcripts of the two methods with higher sensitivity until all methods have the same sensitivity.

At default parameters (Supplementary Figure 8 and Supplementary Figure 9), Scallop shows higher adjusted sensitivity and adjusted precision than StringTie on 63 out of 65 samples, while higher than TransComb on 49 out of the 50 non-strand-specific samples. (TransComb fails on all 15 non-strand-specific samples). On average over these 50 strand-specific samples, Scallop obtains 18.0% and 19.9% more correct multi-exon transcripts (after adjustment to identical precision) than StringTie and TransComb, respectively. The average adjusted precision are 39.1%, 38.6%, and 47.7% for StringTie, TransComb, and Scallop, respectively. On average over the 15 non-strand-specific samples, Scallop obtains 15.4% more correct multi-exon transcripts (after adjustment to identical precision) than StringTie and obtains an average adjusted precision of 48.3%, while that of StringTie is 42.0%.

Scallop's advantage is even more pronounced when lowly expressed transcripts are included by setting the minimum coverage to 0 for all three methods (Supplementary Figure 10 and Supplementary Figure 11). This provides additional evidence that Scallop is particularly more sensitive to lowly expressed transcripts. Specifically, Scallop produces higher adjusted sensitivity and adjusted precision than StringTie on 64 out of the 65 samples, and than TransComb on all the 50 strand-specific samples. On average over the 50 strand-specific samples, Scallop obtains 25.5% and 19.4% more correct multi-exon transcripts (after adjustment) than StringTie and TransComb, respectively. The average adjusted precision is 30.8%, 34.8%, and 44.2% for StringTie, TransComb, and Scallop, respectively. Averaged over all the 15 non-strand-specific samples, Scallop finds 28.3% more correct multi-exon transcripts (after adjustment) than StringTie. The average adjusted precision for Scallop is 43.1%, significantly outperforming StringTie at 27.1%.

Supplementary Note 4: Comparison on Spike-in RNA-seq Samples

Spike-in control RNA-seq experiments provide known ground truth expressed transcripts through synthesizing and sequencing exogenous RNA molecules (possibly combined with biological RNA samples). We evaluate the three assemblers (StringTie, TransComb, and Scallop) using 8 spike-in RNA-seq datasets (Supplementary Table 3) based on a set of synthesized RNA standards termed sequins [33]. Among them, four samples (SQ1, . . . , SQ4) use the combination of sequins and RNA samples, which we call mixed samples, while the other four samples (SQ5, . . . , SQ8) use sequins only, which we call neat samples. We align each of these spike-in RNA-seq datasets to the synthesized chromosome (all reads from sequins are supposed to be able to be aligned to it while all reads that are from the biological RNA-seq samples are supposed to be not able to be aligned to it) with the three aligners (TopHat2, STAR, and HISAT2), run the three assemblers, and then evaluate them by comparing to the (known) ground truth of sequins using gffcompare.

The precision-sensitivity curve of Scallop is above that of StringTie and TransComb for all 8 datasets regardless of the aligners used (Supplementary Figure 12A). At default parameters, on average over the 4 mixed samples (Supplementary Figure 12B), Scallop produces 15.2%, 18.9%, and 33.2% more correct transcripts than StringTie when using TopHat2, STAR, and HISAT2 alignments, respectively; Scallop obtains roughly the same precision with StringTie for TopHat2 and STAR alignments, but improves (39.7% for StringTie and 48.8% for Scallop) for HISAT2 alignments. Similar results can be observed on the 4 neat samples (Supplementary Figure 12C), though the curves in those cases reflect the artificial nature of the sequin-only samples. Overall the performance of TransComb is worse than StringTie and Scallop in terms of both sensitivity and precision.

Supplementary Note 5: RNA-seq Quantification with Scallop

Although we report $f(p)$ as the predicted abundance of the corresponding transcript p (see Algorithm 1), this estimation is mainly for internal use on the way to gradually assembling transcripts and for filtering lowly expressed transcripts; we do not claim that this quantification estimate approaches the accuracy of what can be achieved by more advanced quantification methods such as Salmon [19] and kallisto [20]. We recommend performing quantification by combining Scallop and one of these state-of-the-art quantification methods. Specifically, one can assemble the expressed transcripts using Scallop and build an extended transcriptome for the sample by unioning the reference transcriptome and the novel assembled ones. The final quantification can be obtained using a quantification tool (e.g., Salmon or kallisto) against this extended transcriptome. We provide scripts for this pipeline at http://www.github.com/Kingsford-Group/scallop.

We quantified 10 RNA-seq samples using the above pipeline, combining Scallop and Salmon (see Supplementary Figure 13). For each sample and each aligner, we identify the strongly expressed transcripts (defined as TPM $\geq$ 10) in the extended transcriptome; the average number of them over the 10 samples are 8185, 9140, and 8436 when using TopHat2, STAR, and HISAT2 alignments, respectively. Among these highly expressed transcripts, we compute the percentage that are the result of an assembly by Scallop but that are not in the reference transcriptome. Across the 10 RNA-seq experiments, the average percentage of the strongly expressed transcripts that were only present due to the inclusion of the Scallop assemblies was 19.2%, 19.5%, and 17.6% when using the three aligners, respectively. This suggests that a large portion of highly expressed transcripts may be missing in the reference transcriptome and indicates the necessity of combining assembled transcripts into quantification.

Supplementary Note 6: Building the Splice Graph

Given the alignment of sequencing reads to a reference genome, we first cluster together those reads that overlap on their aligned coordinates. These clusters correspond to different gene loci and are assembled independently. For each gene locus, we collect all splice positions (i.e., coordinates of junctions) from spliced reads. We then infer the starting and ending positions of expressed transcripts, which include the starting and ending coordinates of the whole gene locus, and also the coordinates within each intron region where reads emerge or disappear (see Supplementary Figure 1). The splice positions together with all starting and ending positions partition the gene locus into exons (or partial exons) and introns.

We build the splice graph, denoted as $G = (V, E)$, for each gene locus. For each (partial) exon, we add a vertex v to V. If there exist reads spanning two exons u and v (where u occurs before v in the reference genome), which could be spliced reads that connect two distant exons or continuously aligned reads that span two adjacent (partial) exons, we add a directed edge $e = (u, v)$ to E, and set the weight of e, denoted as $w(e)$, to the number of such reads that span u and v. We also add a source vertex s, and for each vertex $u \in V \setminus \{s\}$ with in-degree of 0 (i.e., those exons adjacent to starting positions), we add a directed edge (s, u) with weight $w(s, u) = \sum_{(u,v)\in E} w(u, v)$. Similarly, we add a sink vertex t, and for each vertex $v \in V \setminus \{s, t\}$ with out-degree of 0 (i.e., those exons adjacent to ending positions), we add a directed edge (v, t) to E with weight $w(v, t) = \sum_{(u,v)\in E} w(u, v)$. See Figure 1 for an example.

Many reads (including paired mates) can span more than two exons, providing phasing information to reconstruct the expressed transcripts. We collect such phasing information as a set of phasing paths of G, denoted as H, as follows. If a read spans vertices $v_{i_1}, v_{i_2}, \cdots, v_{i_m}$ of G (i.e., this read sequentially aligns to these corresponding exons), $m \geq 3$, we then add a phasing path $(v_{i_1}, v_{i_2}, \cdots, v_{i_m})$ to H. For the case of paired-end reads, if we have that one read spans vertices $v_{i_1}, v_{i_2}, \cdots, v_{i_m}$, and its mate read spans vertices $v_{j_1}, v_{j_2}, \cdots, v_{j_n}$, and there exists a unique path $(v_{i_m}, v_{k_1}, v_{k_2}, \cdots, v_{k_l}, v_{j_1})$ from v_{i_m} to v_{j_1} in G, and that $m + n + l \geq 3$, we then add a phasing path $(v_{i_1}, \cdots, v_{i_m}, v_{k_1}, \cdots, v_{k_l}, v_{j_1}, \cdots, v_{j_n})$ to H. In the following, we shall equivalently represent each phasing path with k vertices as a consecutive list of $(k-1)$ edges. Different reads or paired-end reads might produce the same phasing path. For each phasing path $h \in H$, we use $g(h)$ to record the number of such reads or paired-end reads that produce h.

There are false positive edges in the splice graph due to sequencing errors and alignment errors. We use the following rules to filter such edges. Let u and v be two vertices corresponding to two adjacent exons, i.e., the right coordinate of exon u equals the left coordinate of exon v. Let $e = (u, w)$ be an edge corresponding to a junction connecting the right coordinate of exon u and the left coordinate of another exon w. If the coverage of both exons u and v is considerably larger than the weight of edge e, then edge e is very likely a false positive edge. Specifically, we remove edge e if we have $cov(u), cov(v) \geq 2 \cdot w(e) \cdot w(e) + 18$, where $cov(u)$ is the average coverage of exon u. (This rule is a heuristic and the parameters were tuned using the 5 training samples.) Once an edge has been identified as a false positive edge, all phasing paths containing this edge will also be identified as false positive phasing paths and thus be removed from H.

Supplementary Note 7: Proof of the Termination of Algorithm 1

We prove that Algorithm 1 always terminates. We define a potential function on the splice graph, and then prove that after decomposing a nontrivial vertex, the potential decreases at least by 1; while after decomposing a trivial vertex, the potential does not change.

We first define the potential function. Let $I(v) := \{u \in V \mid (u,v) \in E\}$ be the set of predecessors of v, and let $O(v) := \{w \in V \mid (v,w) \in E\}$ be the set of successors of v. Thus, the in-degree and out-degree of v are $|I(v)|$ and $|O(v)|$, respectively. We define the super-degree of v, denoted as $d(v)$, through the recursion $d(v) := K(|O(v)| \geq 2) + \sum_{w \in O(v)} d(w)$, where $K(\cdot)$ is the indicator function which equals 1 if $|O(v)| \geq 2$ and 0 otherwise. For each vertex $v \in V \setminus \{s,t\}$, we define its potential $\phi(v) := (|I(v)| - 1) \cdot d(v)$, and the overall potential of splice graph G is defined as $\Phi(G) := \sum_{v \in V \setminus \{s,t\}} \phi(v)$.

We denote by G' the splice graph after decomposing a certain type of vertex of G. We use $I'(v)$, $O'(v)$, $d'(v)$, and $\phi'(v)$ to denote the corresponding functions *w.r.t.* G'. They have the same definition with those for G; we distinguish them because we might use the same symbol for a vertex that appears both in G and G'.

We now prove that decomposing an unsplittable vertex reduces the potential at least by 1 (Supplementary Figure 19). Let $v \in V$ be an unsplittable vertex. Notice that v will be replaced by a subgraph (a bipartite graph) in G', denoted as $G_v = (V_v, E_v)$. According to our algorithm, V_v consists of a copy of $I(v)$ and $O(v)$; we denote them as I_v and O_v, respectively, i.e., we have $V_v = I_v \cup O_v$. For each vertex $u \in I(v) \cup O(v)$, we use $u' \in I_v \cup O_v$ to denote the corresponding vertex of u in G'. Without loss of generality, we assume the worst case that G_v is fully connected, i.e., $E_v = \{(u,v) \mid u \in I_v, v \in O_v\}$. Consequently, for any vertex $u' \in I_v$, we have that $|O'(u')| = |O_v| = |O(v)|$, and symmetrically, for any vertex $w' \in O_v$, we have that $|I'(w')| = |I_v| = |I(v)|$.

It is clear that for each vertex $w \in O(v)$, we have that $d'(w) = d(w)$. We now prove that for each vertex in $u \in I(v)$ we also have that $d'(u) = d(u)$. In fact, we can write $d'(u) - d(u) = d'(u') - d(v) = (1 + \sum_{w' \in O_v} d'(w')) - d(v) = (1 + \sum_{w \in O(v)} d'(w)) - d(v) = (1 + \sum_{w \in O(v)} d(w)) - d(v) = 0$.

Hence, we only need to compare the potential contributed by v in G with that contributed by $I_v \cup O_v$ in G'. Notice that the in-degree of each vertex in I_v is 0, thus they contribute 0 to the overall potential of G'. We have $\Phi(G') - \Phi(G) = \sum_{w' \in O_v} \phi'(w') - \phi(v) = \sum_{w' \in O_v} (I'(w') - 1) \cdot d'(w') - (I(v) - 1) \cdot d(v) = \sum_{w \in O(v)} (I(v) - 1) \cdot d(w) - (I(v) - 1) \cdot (1 + \sum_{w \in O(v)} d(w)) = 1 - I(v) \leq -1$. This concludes the case for decomposing unsplittable vertices.

We then prove that decomposing a splittable vertex also decreases the potential at least by 1. We consider the case that, after decomposition, v is replaced by two new vertices v_1 and v_2 in G', and both v_1 and v_2 are nontrivial vertices (Supplementary Figure 20).

It is clear that for each vertex $u \in I(v)$, we have $\phi'(u) \leq \phi(u)$. Thus, we have that

$$\begin{aligned} & \Phi(G') - \Phi(G) \leq \phi'(v_1) + \phi'(v_2) - \phi(v) \\ = \; & (I'(v_1) - 1) \cdot d'(v_1) + (I'(v_2) - 1) \cdot d'(v_2) - (I(v) - 1) \cdot d(v) \\ = \; & (I'(v_1) - 1) \cdot (1 + \textstyle\sum_{w \in O(v_1)} d'(w)) + (I'(v_2) - 1) \cdot (1 + \textstyle\sum_{w \in O(v_2)} d'(w)) - (I(v) - 1) \cdot (1 + \textstyle\sum_{w \in O(v)} d(w)) \end{aligned}$$

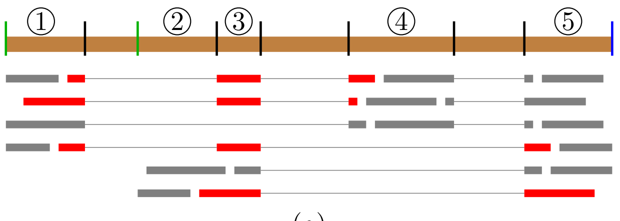
By applying $I'(v_1) + I'(v_2) = I(v)$ and $\sum_{w \in O(v_1)} d'(w) + \sum_{w \in O(v_2)} d'(w) = \sum_{w \in O(v)} d(w)$ to simplify the above equation, we can obtain that $\Phi(G') - \Phi(G) \leq -1$.

For other cases that either v_1 or v_2 or both are trivial vertices, we can use the same approach to verify that the potential decrease is still at least 1. This concludes the case for decomposing splittable vertices.

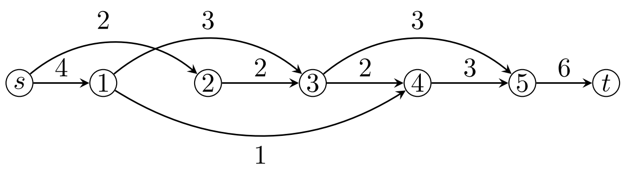
Finally, we prove that decomposing a trivial vertex keeps the potential. Assume that the in-degree of v is 1, and let u be the unique predecessor of v. According to Algorithm 1, u must be a trivial vertex, i.e., either the out-degree of u is 1 (Supplementary Figure 21(**a,b**)), or the in-degree of u is 1 (Supplementary Figure 21(**c,d**)). We can easily verify that, in either case, the potential does not change after decomposing v.

Since decomposing a trivial vertex requires removing an edge, we have that Algorithm 1 cannot continue to decompose trivial vertices without termination. In other words, after finite iterations of decomposing trivial vertices, Algorithm 1 either terminates, or it changes to decompose a nontrivial vertex, in which case the potential decreases. This implies that Algorithm 1 always terminates.

Supplementary Figure 1



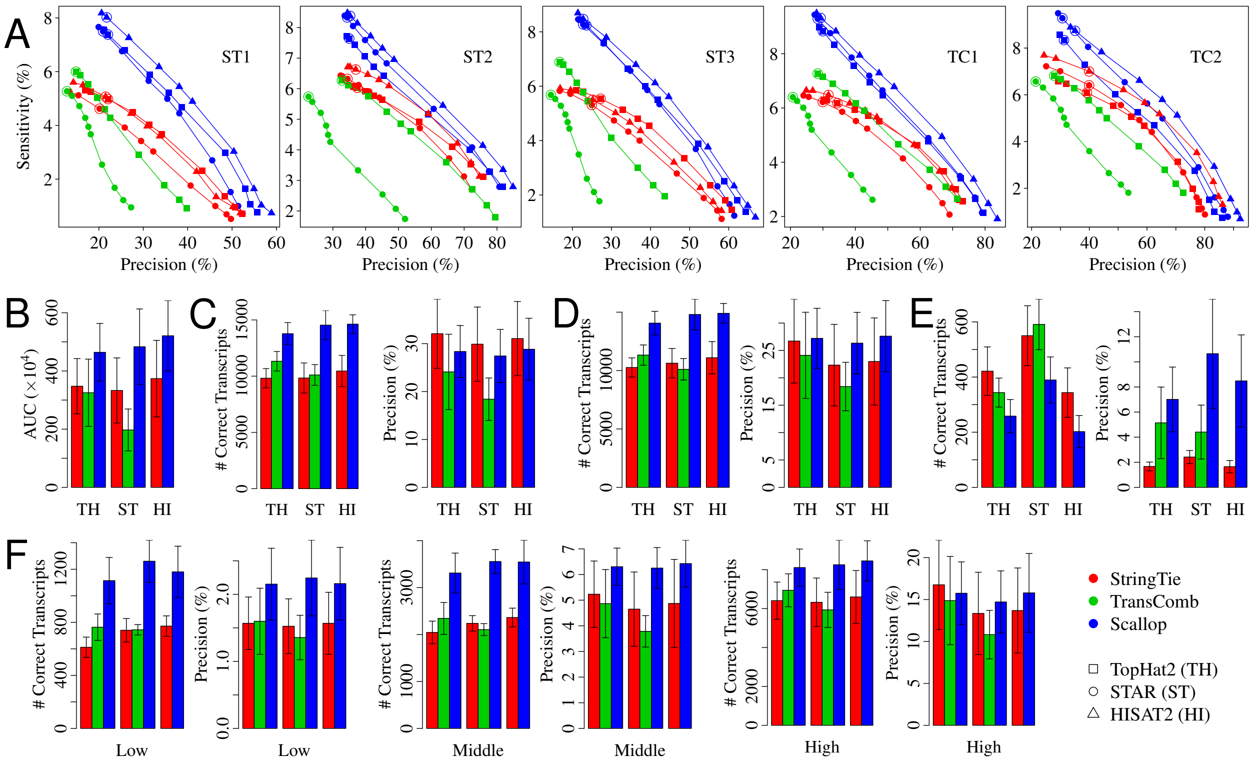
(a)



(b)

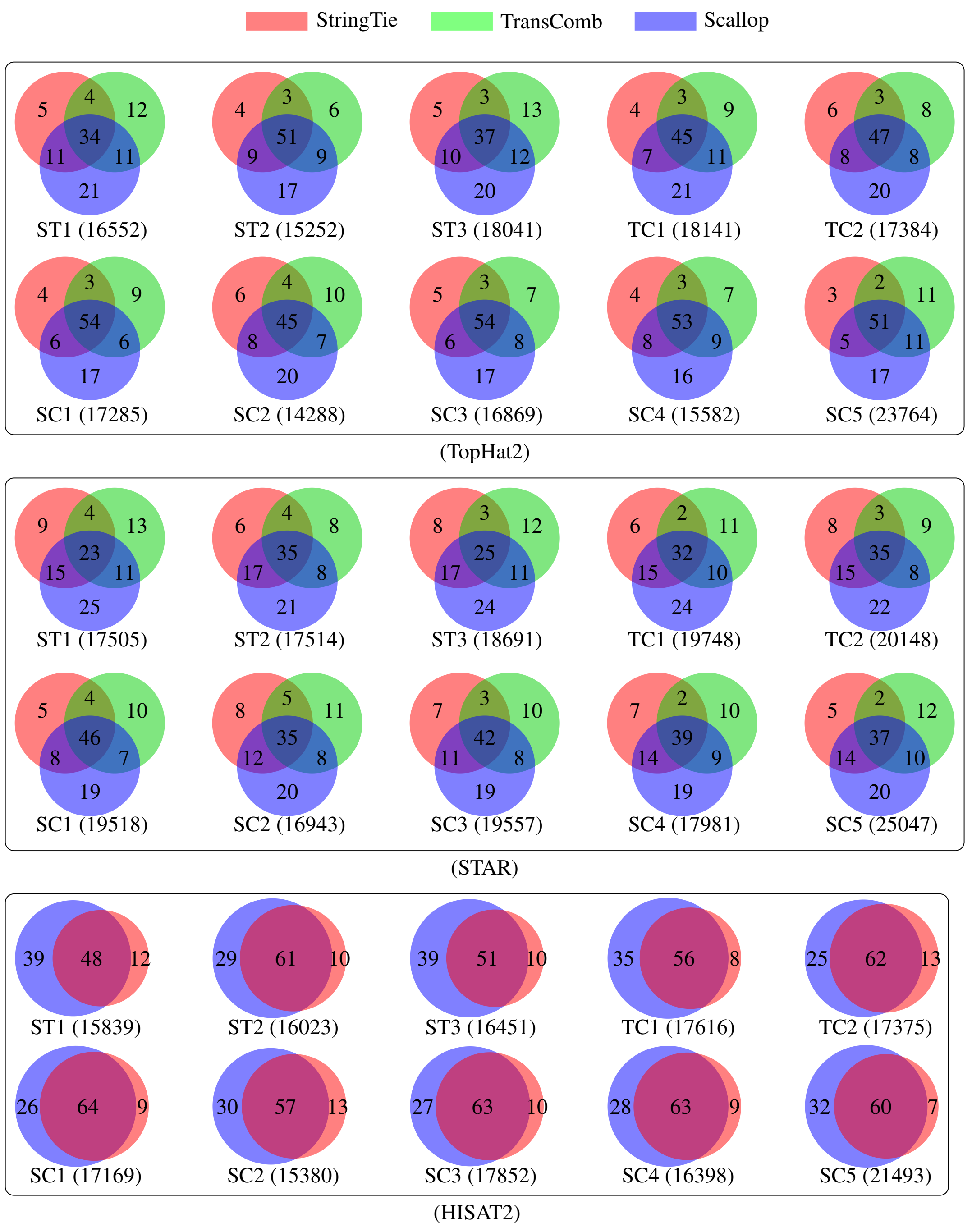
Supplementary Figure 1: Example of building splice graph and phasing paths. **(a)** Alignment of reads to the reference genome. Inferred splice positions are marked with black bars; inferred starting and ending positions are marked with green and blue bars, respectively. Exons and partial exons are labeled with numbers above the reference genome. Reads that span more than two exons are colored red, from which we can get the set of phasing paths as $\{(1,3,4),(2,3,5),(1,3,5)\}$. The abundance of these phasing paths are $g(1,3,4)=2$, $g(2,3,5)=1$, and $g(1,3,5)=1$. **(b)** The corresponding splice graph and weights for all edges.

Supplementary Figure 2



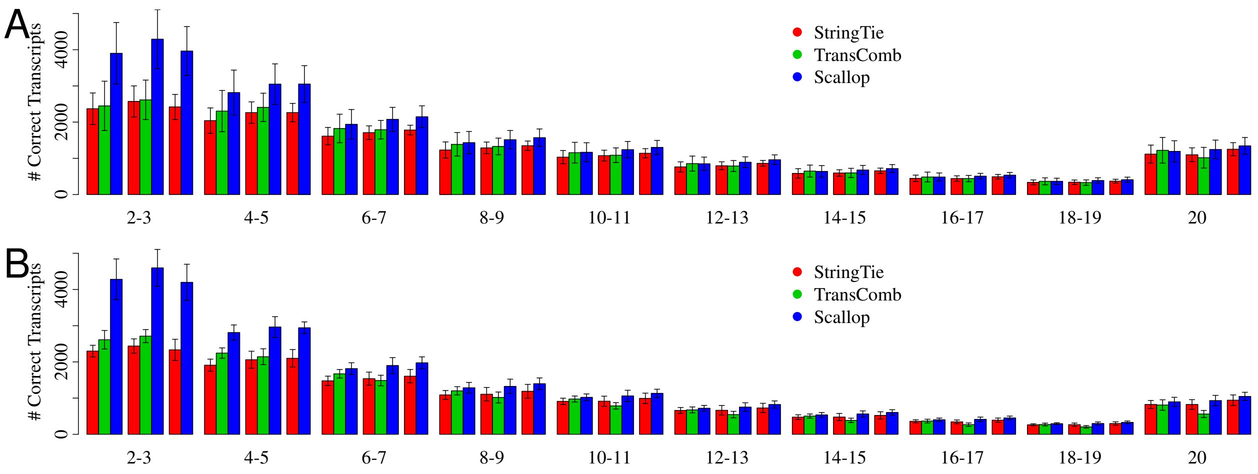
Supplementary Figure 2: Comparison of the three methods (StringTie, TransComb, and Scallop) over the 5 training samples. (A) The precision-sensitivity curves for multi-exon transcripts. Each curve connects 10 points, corresponding to the 10 different minimum coverage thresholds $\{0, 1, 2.5, 5, 7.5, 10, 25, 50, 75, 100\}$; the default value is circled. (B) The average AUC (area under the precision-sensitivity curve). The error bars show the standard deviation over the 5 samples (the same for other panels). (C) The average sensitivity and precision of multi-exon transcripts running at default parameters. (D) The average sensitivity and precision of multi-exon transcripts running with minimum coverage set to 0. (E) The average sensitivity and precision of single-exon transcripts running at default parameters. (F) The average sensitivity and precision of multi-exon transcripts with each subset of transcripts (corresponding to low, middle, and high expression level) as ground truth running with minimum coverage set to 0.

Supplementary Figure 3



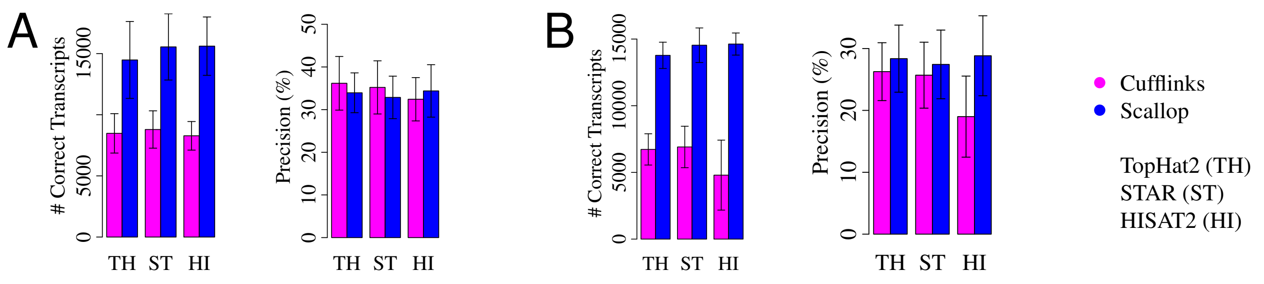
Supplementary Figure 3: Concordance among different assemblers. For each Venn diagram, the number in the parenthesis below gives the number of correct transcripts in the union of all three assemblers. The numbers inside the Venn diagram gives the percentage of the correct transcripts in the corresponding subset *w.r.t.* the union. The three assemblers are very diverse from each other. Specifically, the ratio between the number of correct transcripts shared by all the three assemblers and that in the union of them is 47.6% and 35.6% for TopHat2 and STAR alignments, respectively. With HISAT2 alignments, the ratio between the number of correct transcripts shared by StringTie and Scallop and that in the union of them is 59.1%.

Supplementary Figure 4



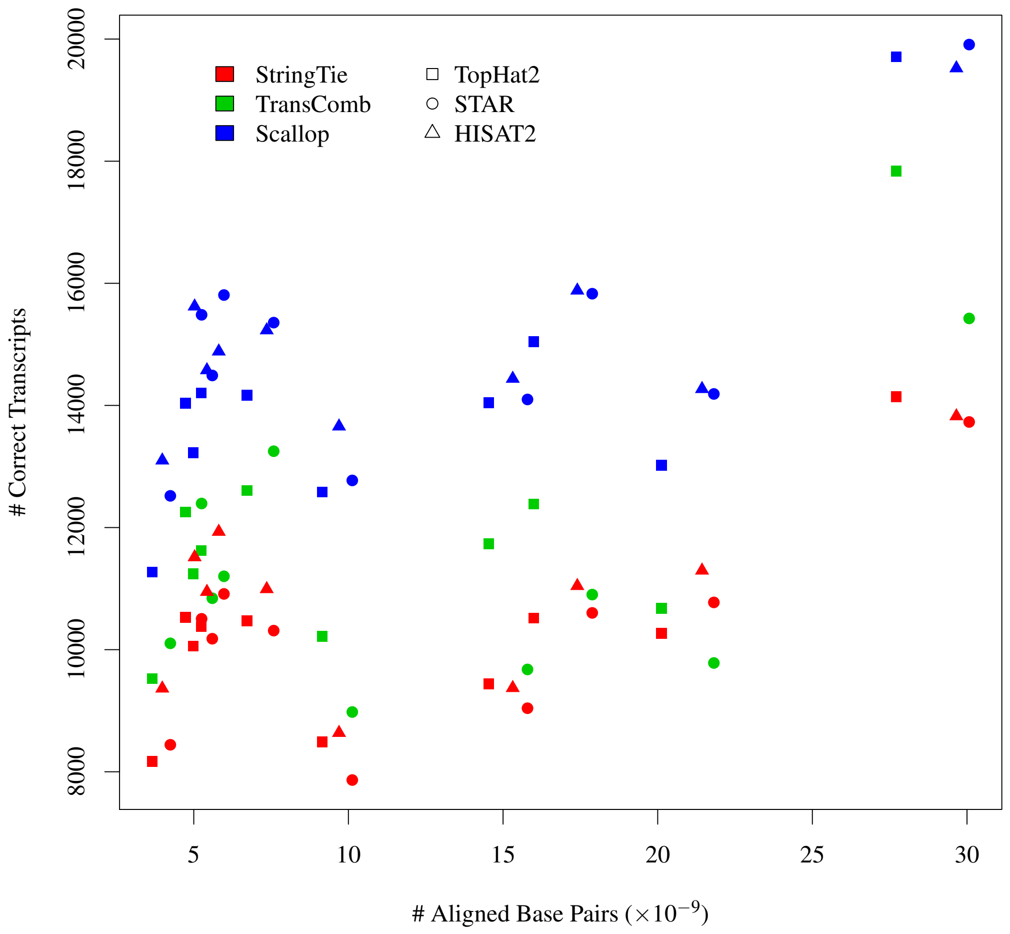
Supplementary Figure 4: The average number of correct transcripts with different number of exons over the 5 testing samples (panel A) and the 5 training samples (panel B) for methods running with minimum coverage set to 0. The error bars show the standard deviation over the 5 testing samples (panel A) and the 5 training samples (panel B). While Scallop identifies more correct transcripts for almost all number of exons, its advantage is mostly pronounced on transcripts with 2–9 exons, which consists of the majority of the correct transcripts identified by the three methods. For example, with TopHat2 alignments, averaged over the 5 testing samples, Scallop obtains 64.6% and 59.3% more correct transcripts with 2 or 3 exons than StringTie and TransComb, respectively.

Supplementary Figure 5



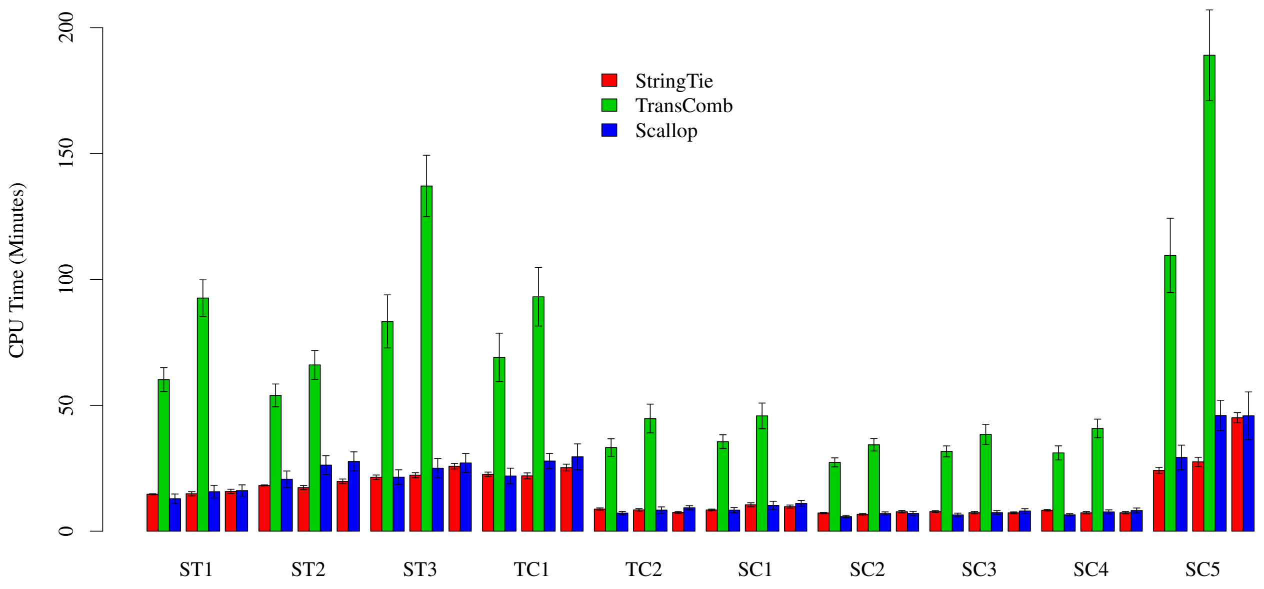
Supplementary Figure 5: Comparison of accuracy (in terms of correctly assembled transcripts and precision) of Cufflinks and Scallop at reconstructing multi-exon transcripts over the 5 testing samples (panel A) and the 5 training samples (panel B) when both methods are run at default parameters. The error bars show the corresponding standard deviation over the 5 testing samples (panel A) and the 5 training samples (panel B). Scallop obtains similar precision with Cufflinks but drastically outperforms in terms of sensitivity. For example, over the 5 testing samples, Scallop assembles 70.8%, 76.8%, and 88.5% more correct transcripts than Cufflinks when using TopHat2, STAR, and HISAT2 alignments, respectively.

Supplementary Figure 6



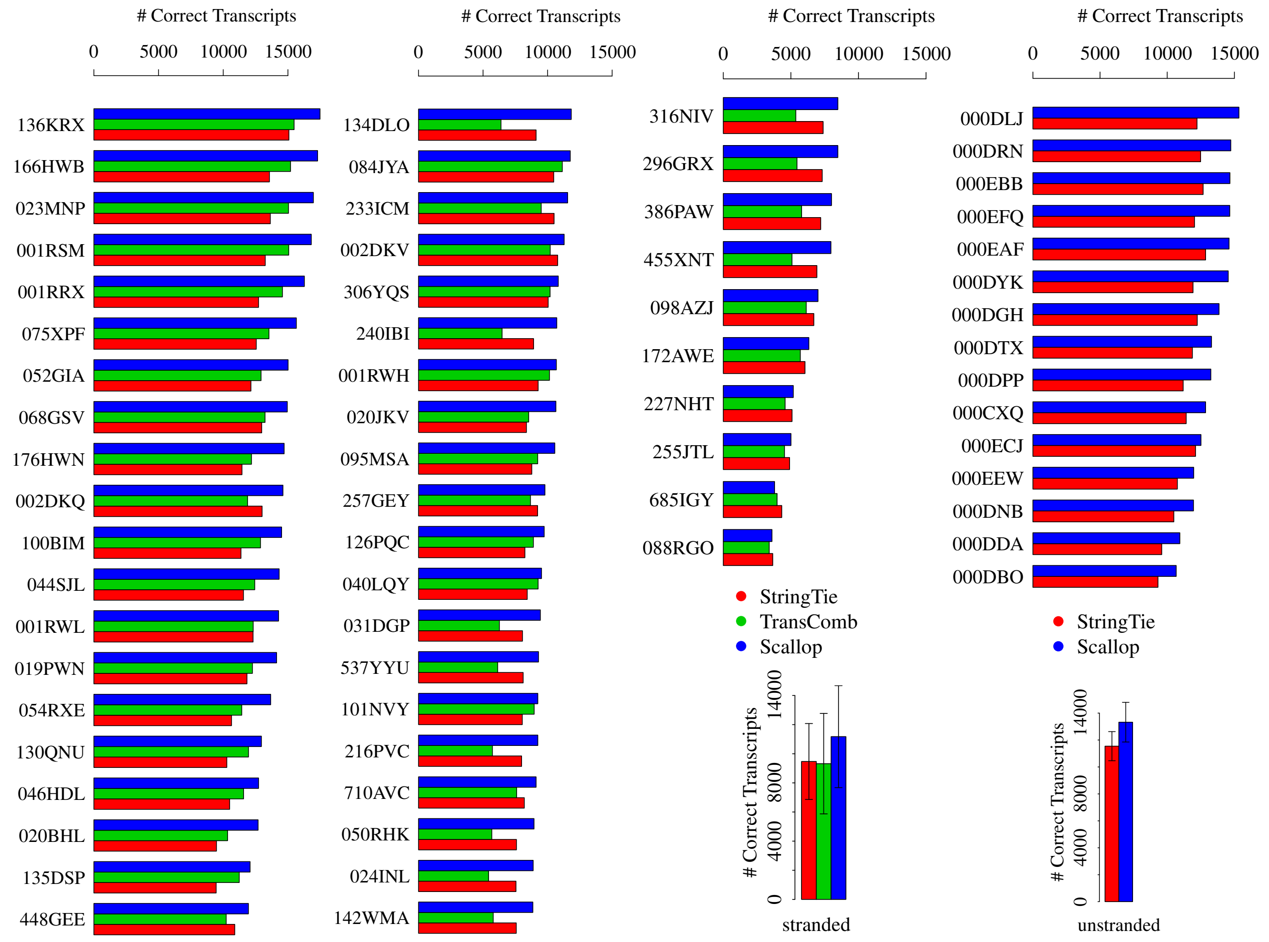
Supplementary Figure 6: Correlation between the number of aligned base pairs and the number of correct multi-exon transcripts obtained by the methods (StringTie, TransComb, and Scallop) running at default parameters. For each method (distinguished with different colors) and each aligner (distinguished with different shapes), there are 10 points, corresponding to the 10 RNA-seq samples listed in Supplementary Table 1.

Supplementary Figure 7



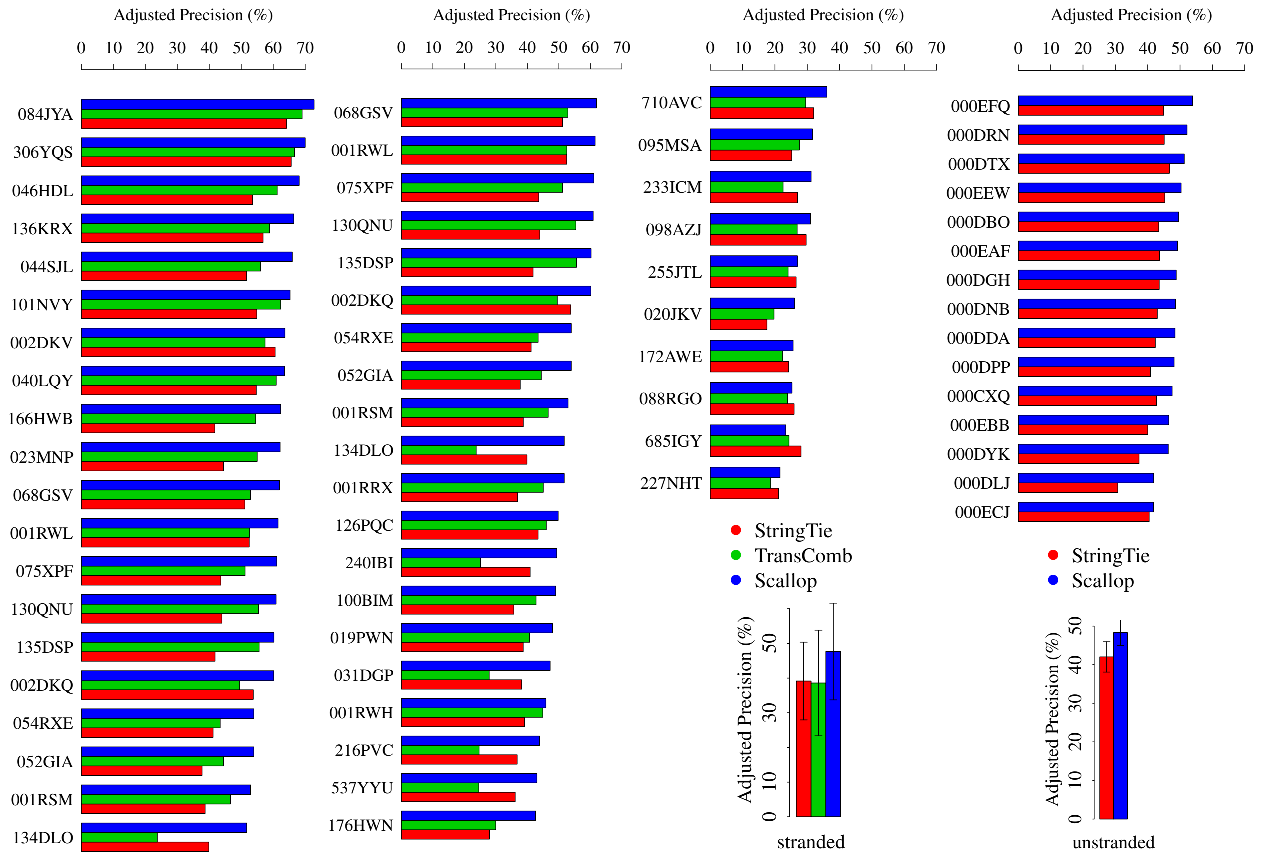
Supplementary Figure 7: Comparison of the running time (measured as CPU time) of the three assemblers (StringTie, TransComb, and Scallop) on the 10 RNA-seq samples listed in Supplementary Table 1. All programs are run with their single-thread mode on a machine with 48 cores and 40GB RAM. The error bars show the standard deviation over the 10 runs with different minimum coverage parameters $\{0, 1, 2.5, 5, 7.5, 10, 25, 50, 75, 100\}$. On average over the 10 RNA-seq samples and over the 10 thresholds, with TopHat2 alignments, Scallop and TransComb take $0.99\times$ and $3.78\times$ longer than StringTie, respectively. With STAR alignments, the ratios are $1.25\times$ and $5.42\times$. With HISAT2 alignments, Scallop takes $1.10\times$ longer than StringTie.

Supplementary Figure 8



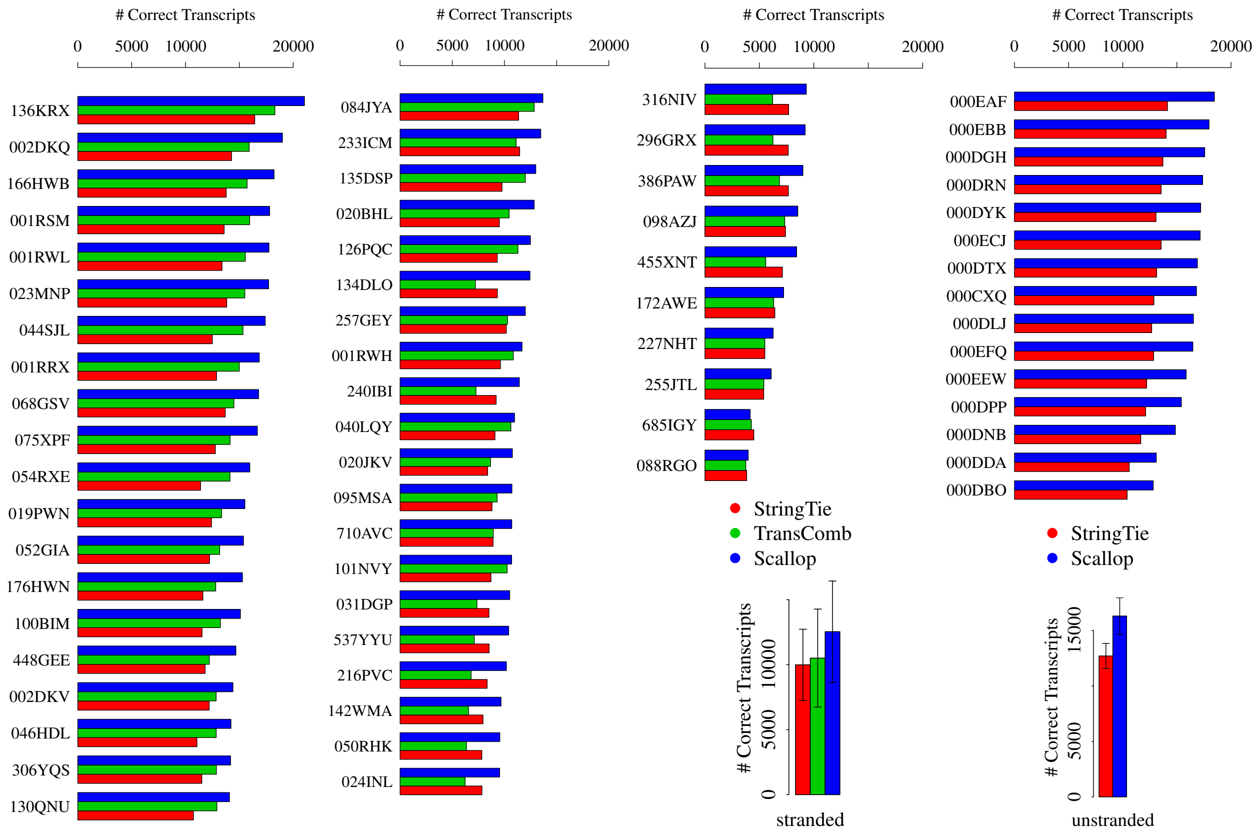
Supplementary Figure 8: Comparison of the adjusted sensitivity (shown as the number of correct transcripts) of multi-exon transcripts for methods (StringTie, TransComb, and Scallop) running with their default parameter settings. The experiment uses 50 strand-specific samples (leftmost three columns of this figure) and 15 non-strand-specific samples (the rightmost column of this figure). The two vertical bar-plots show the average performance and standard deviation over the 50 strand-specific samples (bottom of the third column) and the 15 non-strand-specific (bottom of the four column), respectively. Read alignments for these samples were downloaded from ENCODE project (2013–present). For each sample, we mark its (partial) ID in ENCODE on the left side. The complete ID adds the prefix "ENCFF". TransComb fails on the 15 non-strand-specific samples so for them we only compare the results given by Scallop and StringTie.

Supplementary Figure 9



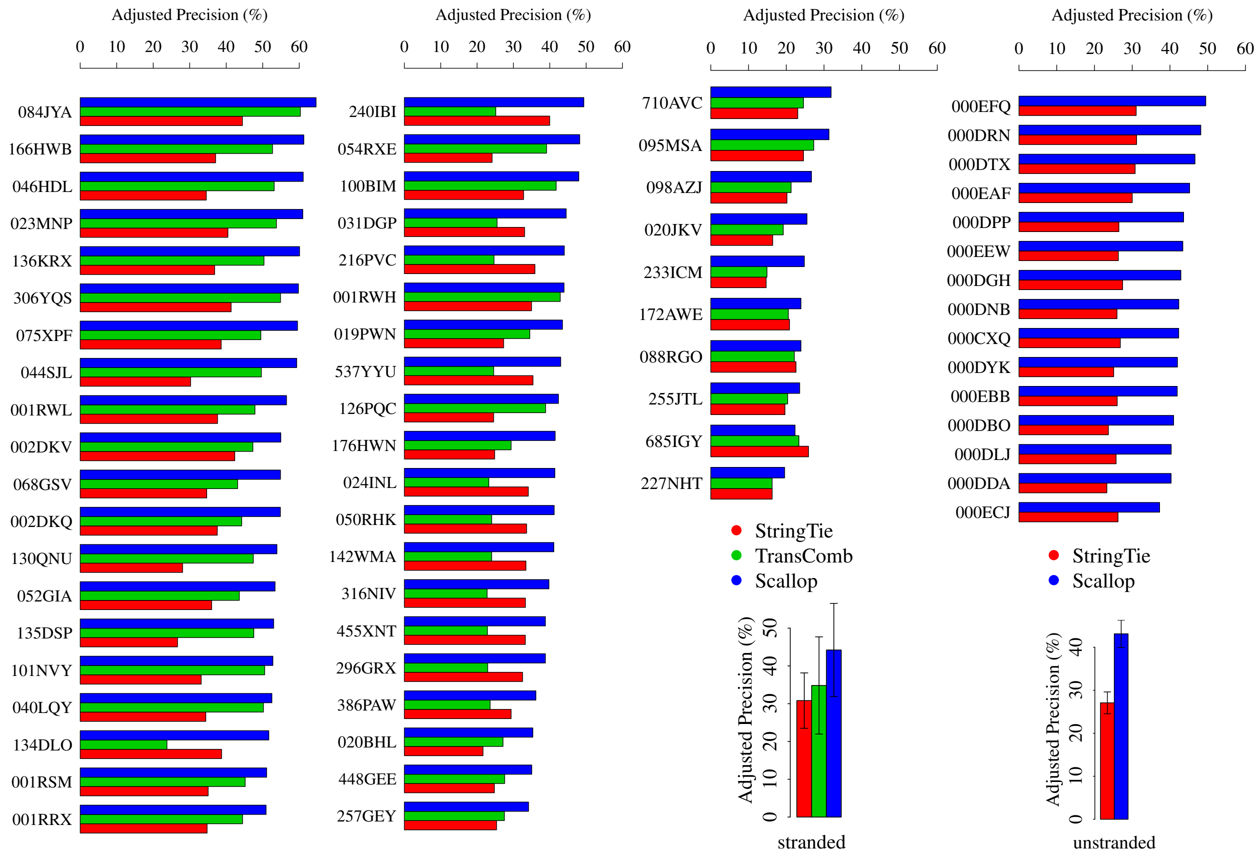
Supplementary Figure 9: Comparison of the adjusted precision of multi-exon transcripts for methods (StringTie, TransComb, and Scallop) running with their default parameter settings. The samples are identical to those in Supplementary Figure 8.

Supplementary Figure 10



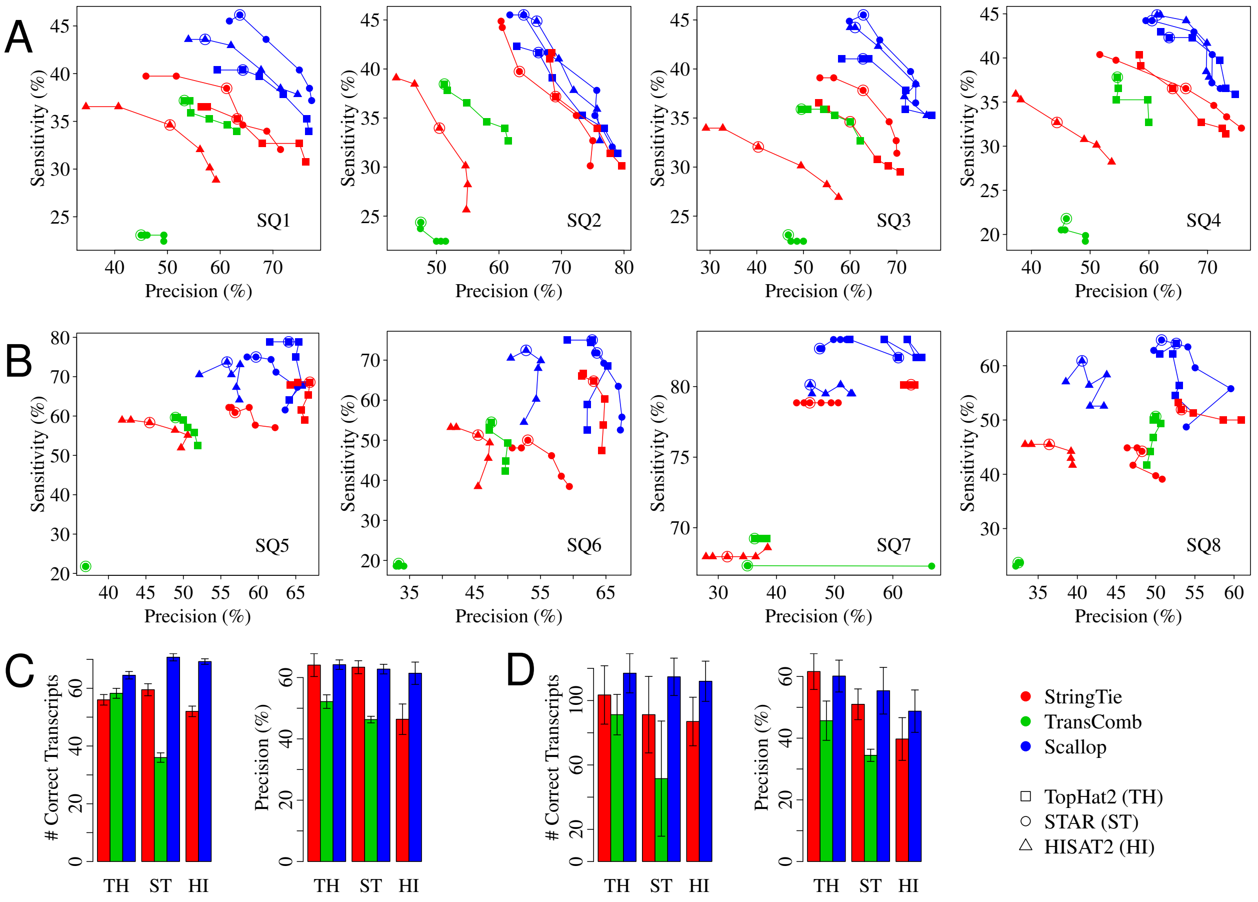
Supplementary Figure 10: Comparison of the adjusted sensitivity (shown as the number of correct transcripts) of multi-exon transcripts for methods (StringTie, TransComb, and Scallop) running with the minimum coverage threshold set to 0. The samples are identical to those in Supplementary Figure 8.

Supplementary Figure 11



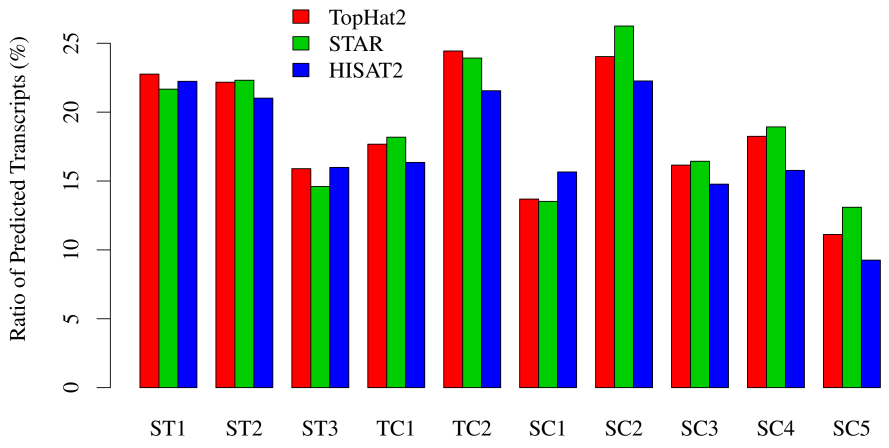
Supplementary Figure 11: Comparison of the adjusted precision of multi-exon transcripts for the three methods (StringTie, TransComb, and Scallop) running with the minimum coverage threshold set to 0. The samples are identical to those in Supplementary Figure 8.

Supplementary Figure 12



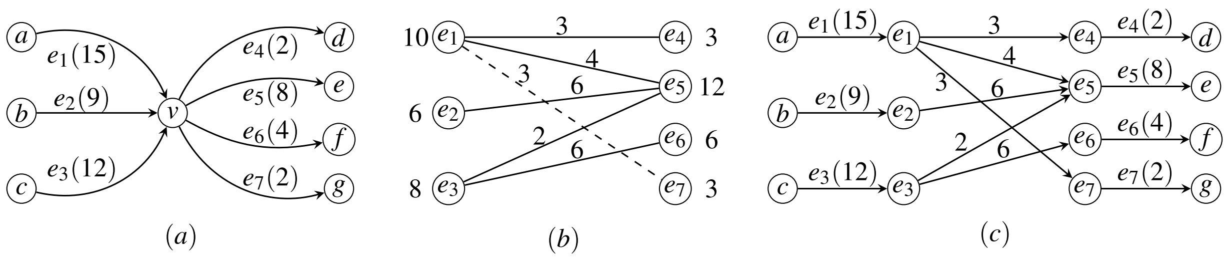
Supplementary Figure 12: Comparison of the three methods (StringTie, TransComb, and Scallop) over the 8 spike-in RNA-seq samples. (A) The precision-sensitivity curves of multi-exon transcripts over the 4 mixed samples. Each curve connects 6 points, corresponding to the 6 different minimum coverage thresholds $\{0, 1, 2.5, 5, 7.5, 10\}$; the default value of this parameter is circled. (B) The precision-sensitivity curves of multi-exon transcripts over the 4 neat samples. (C) The average sensitivity (shown as number of correct transcripts) and precision of multi-exon transcripts over the 4 mixed samples for methods running at default parameters. The error bars show the standard deviation over the 4 mixed samples. (D) The average sensitivity (shown as number of correct transcripts) and precision of multi-exon transcripts over the 4 neat samples for methods running at default parameters. The error bars show the standard deviation over the 4 neat samples.

Supplementary Figure 13

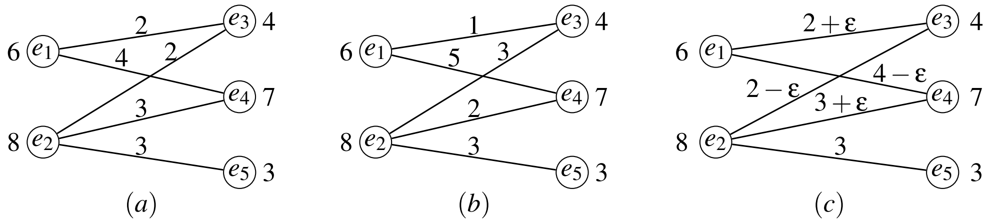


Supplementary Figure 13: Quantification results of the 10 RNA-seq samples listed in Supplementary Table 1. Each sample is aligned with three aligners and assembled with Scallop separately. For each sample and each aligner, the extended transcriptome was computed as the union of the reference transcriptome (ENSEMBL release-87) and the assembled transcripts. Salmon was applied to quantify with respect to the extended transcriptome, and the strongly expressed transcripts (measured as TPM $\geq$ 10) were identified. The percentage of these strongly expressed transcripts that were assembled by Scallop but that were not in the reference transcriptome is shown on the y-axis.

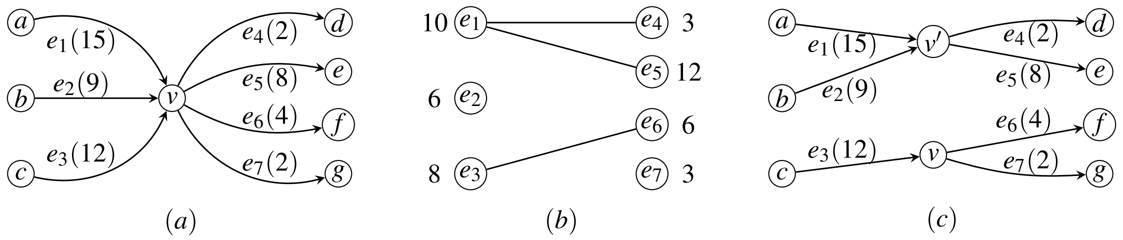
Supplementary Figures for Online Methods



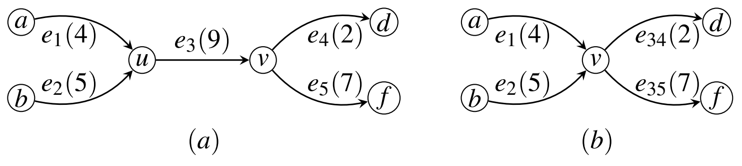
Supplementary Figure 14: Example of decomposing an unsplittable vertex. **(a)** Subgraph associated with vertex v. The weight of each edge is shown in the parenthesis. Assume that phasing paths contain (e_1,e_4), (e_1,e_5), (e_2,e_5), (e_3,e_5) and (e_3,e_6). **(b)** Bipartite graph G_v (without the dashed edge), and the extended bipartite graph $\overline{G_v}$ (with the dashed edge). The balanced weights are next to the vertices. The weights given by the optimal solution of the linear programming are next to edges. **(c)** Updated subgraph after decomposing v.



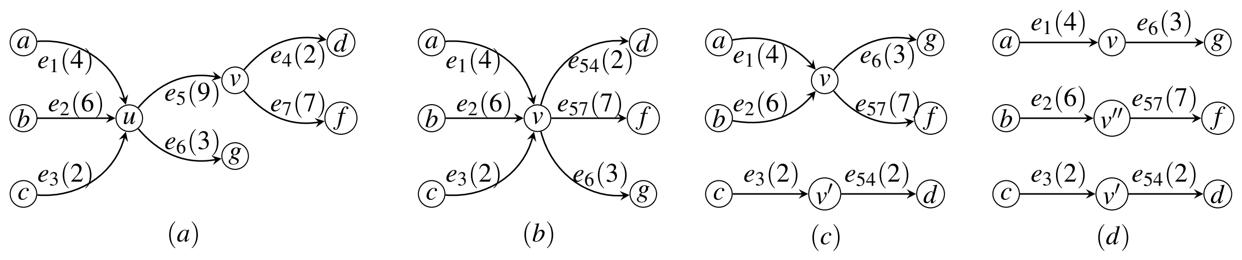
Supplementary Figure 15: Example with multiple optimal solutions when the extended bipartite graph contains cycles. The balanced weights, i.e., $\overline{w}(\cdot)$, are next to the vertices. **(a, b)** Two optimal solutions with deviation of 0 *w.r.t.* $\overline{w}(\cdot)$. **(c)** When $-2 < \varepsilon < 2$, the solution is always optimal.



Supplementary Figure 16: Example of decomposing a splittable vertex. **(a)** Subgraph associated with v. Assume that phasing paths contain (e_1,e_4), (e_1,e_5) and (e_3,e_6). **(b)** Bipartite graph G_v with balanced weights next to vertices. Optimal decomposition gives $S'^*_v = \{e_1,e_2\}$, $T'^*_v = \{e_4,e_5\}$. **(c)** Updated subgraph after decomposing v.

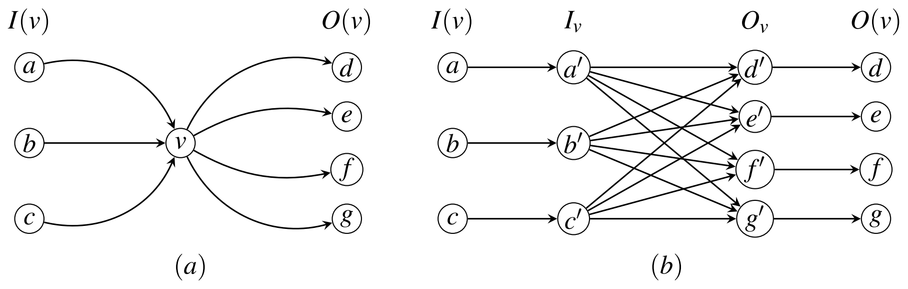


Supplementary Figure 17: Illustration of decomposing trivial vertex v. **(a)** Subgraph before decomposing v. **(b)** Subgraph after decomposing v. We record e_4 and e_5 are preceded by e_3 by labeling them as e_{34} and e_{35}.

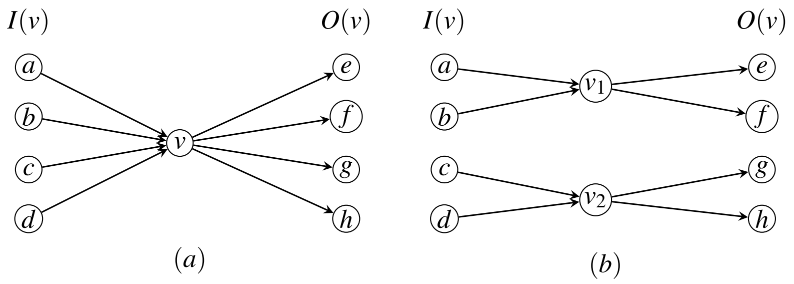


Supplementary Figure 18: Decomposing a trivial vertex may not preserve all phasing paths if the splice graph contain nontrivial vertices. **(a)** Splice graph G with trivial vertex v and nontrivial vertex u. Assume that we have a single phasing path of $H = \{(e_1, e_5)\}$. **(b)** Updated splice graph G' after decomposing trivial vertex v. Notice that now we have $H' = \{(e_1)\}$. Since a phasing path with a single edge is not informative, we actually have that $H' = \emptyset$. **(c,d)** The following decomposition of G' by applying the subroutine for splittable vertices. Notice that in the final three *s*-*t* paths, none of them covers (e_1, e_5).

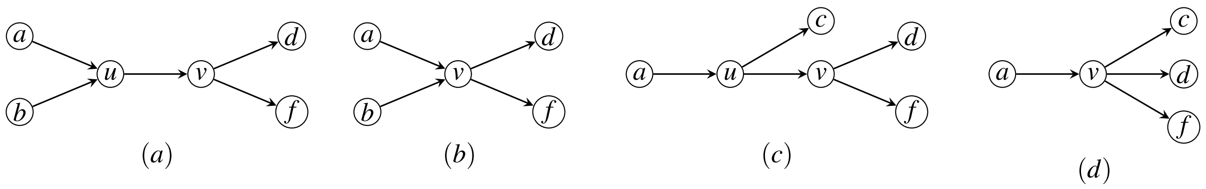
Supplementary Figures for Proof of Termination of Algorithm 1



Supplementary Figure 19: Example for proving that decomposing an unsplittable vertex decreases the potential at least by 1. **(a)** Splice graph G. **(b)** Splice graph G' after decomposing unsplittable vertex v.



Supplementary Figure 20: Example for proving that decomposing a splittable vertex decreases the potential at least by 1. **(a)** Splice graph G. **(b)** Splice graph G' after decomposing splittable vertex v.



Supplementary Figure 21: Example for proving that decomposing a trivial vertex keeps the potential. **(a)** Splice graph G with the out-degree of u being 1. **(b)** Splice graph G' after decomposing v in **(a)**. **(c)** Splice graph G with the in-degree of u being 1. **(d)** Splice graph G' after decomposing v in **(c)**.

Supplementary Table 1

ID	SRA Accession	GEO Accession	Chosen By	#Spots	Cell Line	Localization	Length
ST1	SRR534319	GSM981256	StringTie	25M	CD20+	cell	76
ST2	SRR534291	GSM981244	StringTie	114M	IMR90	cytosol	101
ST3	SRR545695	GSM984609	StringTie	40M	CD14+	cell	76
TC1	SRR387661	GSM840137	TransComb	125M	K562	cytosol	76
TC2	SRR307911	GSM758566	TransComb	41M	H1-hESC	cell	76
SC1	SRR545723	GSM984621	Scallop	147M	HMEpC	cell	101
SC2	SRR315323	GSM765399	Scallop	30M	NHEK	nucleus	76
SC3	SRR307903	GSM758562	Scallop	36M	BJ	cell	76
SC4	SRR315334	GSM765404	Scallop	39M	HeLa-S3	cytosol	76
SC5	SRR534307	GSM981252	Scallop	167M	MCF-7	cytosol	101

Supplementary Table 1: Summary of the 10 RNA-seq samples used in this paper. All these 10 samples are from human, and the sequencing employs paired-end and strand-specific protocols. Among them, ST1, ST2, and ST3 were evaluated in the StringTie paper, TC1 and TC2 were used in the TransComb paper, while the other five (SC1, SC2, . . . , SC5) were chosen by us from ENCODE project (https://genome.ucsc.edu/ENCODE, 2003–2012).

Supplementary Table 2

Software	Version	Command Line
StringTie	1.3.2d	`stringtie` *bam-file strandness* `-c` *min-coverage*
TransComb	v.1.0	`TransComb -b` *bam-file* `-s` *strandness* `-f` *min-coverage*
Scallop	v0.9.9 v0.10.2	`scallop -i` *bam-file* `-o` *gtf-file* `--library_type` *strandness* `--min_transcript_coverage` *min-coverage*
TopHat2	v2.1.1	`tophat2 -p 6` *index fastq1 fastq2*
STAR	2.5.2a	`STAR --outSAMstrandField` *intronMotif* `--chimSegmentMin 20` `--runThreadN 6 --genomeDir` *index* `--readFilesIn` *fastq1 fastq2*
HISAT2	2.0.4	`hisat2 -p 6 -x` *index* `-1` *fastq1* `-2` *fastq2*
gffcompare	v.0.9.9c	`gffcompare -r` *reference-gtf predicted-gtf*
Salmon	v.0.7.2	`salmon quant -i` *index* `-l` *ISR* `-1` *fastq1* `-2` *fastq2* `-p 6`

Supplementary Table 2: Programs and their versions and arguments used in this paper. Supplementary Figure 13 was produced with Scallop v0.10.2; other Figures were produced with Scallop v0.9.9. The changes from Scallop v0.9.9 to Scallop v0.10.2 are bug fixing and engineering, and there is no algorithmic change between them.

Supplementary Table 3

ID	SRA Accession	GEO Accession	Description	#Spots
SQ1	SRR3743144	GSM2225088	K562 spiked with Sequins Mix A rep1	38M
SQ2	SRR3743145	GSM2225089	K562 spiked with Sequins Mix A rep2	32M
SQ3	SRR3743146	GSM2225090	K562 spiked with Sequins Mix A rep3	37M
SQ4	SRR4011970	GSM2264866	GM12878 spiked with Sequins Mix B rep1	42M
SQ5	SRR3743147	GSM2225091	Neat RNA Sequins Mix A	22M
SQ6	SRR3743148	GSM2225092	Neat RNA Sequins Mix B	21M
SQ7	SRR3743149	GSM2225093	Neat RNA Sequins Flat Mix	15M
SQ8	SRR3743150	GSM2225094	Neat RNA Fusion Sequins Mix	45M

Supplementary Table 3: Summary of the 8 spike-in RNA-seq samples used in this paper. All samples are sequenced with strand-specific and paired-end protocals. The read length is 125 for all the samples.

Reference

[23] T.D. Wu and S. Nacu. Fast and SNP-tolerant detection of complex variants and splicing in short reads. *Bioinformatics*, 26(7):873–881, 2010.

[24] K.F. Au, H. Jiang, L. Lin, Y. Xing, and W.H. Wong. Detection of splice junctions from paired-end RNA-seq data by SpliceMap. *Nucleic Acids Res.*, 38(14):4570–4578, 2010.

[25] B.J. Haas, A.L. Delcher, S.M. Mount, J.R. Wortman, et al. Improving the arabidopsis genome annotation using maximal transcript alignment assemblies. *Nucleic Acids Res.*, 31(19):5654–5666, 2003.

[26] G. Robertson, J. Schein, R. Chiu, R. Corbett, M. Field, S.D. Jackman, K. Mungall, S. Lee, H.M. Okada, J.Q. Qian, et al. De novo assembly and analysis of RNA-seq data. *Nat. Methods*, 7(11): 909–912, 2010.

[27] M.G. Grabherr, B.J. Haas, M. Yassour, J.Z. Levin, D.A. Thompson, et al. Trinity: reconstructing a full-length transcriptome without a genome from RNA-Seq data. *Nat. Biotechnol.*, 29(7):644, 2011.

[28] Y. Xie, G. Wu, J. Tang, R. Luo, J. Patterson, S. Liu, et al. SOAPdenovo-Trans: de novo transcriptome assembly with short RNA-Seq reads. *Bioinformatics*, 30(12):1660–1666, 2014.

[29] M.H. Schulz, D.R. Zerbino, M. Vingron, and E. Birney. Oases: robust de novo RNA-seq assembly across the dynamic range of expression levels. *Bioinformatics*, 28(8):1086–1092, 2012.

[30] Y. Peng, H.C.M. Leung, S.-M. Yiu, M.-J. Lv, X.-G. Zhu, and F.Y.L. Chin. IDBA-tran: a more robust de novo de Bruijn graph assembler for transcriptomes with uneven expression levels. *Bioinformatics*, 29(13):i326–i334, 2013.

[31] Z. Chang, G. Li, J. Liu, Y. Zhang, C. Ashby, D. Liu, C.L. Cramer, and X. Huang. Bridger: a new framework for de novo transcriptome assembly using RNA-seq data. *Genome Biol.*, 16(1):30, 2015.

[32] S. Kannan, J. Hui, K. Mazooji, L. Pachter, and D. Tse. Shannon: an information-optimal de novo RNA-seq assembler. *bioRxiv*, page 039230, 2016. doi: 10.1101/039230.

[33] S.A. Hardwick, W.Y. Chen, T. Wong, I.W. Deveson, J. Blackburn, S.B. Andersen, L.K. Nielsen, J.S. Mattick, and T.R. Mercer. Spliced synthetic genes as internal controls in RNA sequencing experiments. *Nat. Methods*, 13:792–798, 2016.