OXFORD UNIVERSITY PRESS | Bioinformatics

iPAC: A genome-guided assembler of isoforms via phasing and combing paths

Journal:	*Bioinformatics*
Manuscript ID	BIOINF-2019-1908.R1
Category:	Original Paper
Date Submitted by the Author:	14-Dec-2019
Complete List of Authors:	Yu, Ting; Shandong University, School of Mathematics Liu, Juntao; Shandong University, School of Mathematics Gao, Xin; King Abdullah University of Science and Technology, Mathematical and Computer Sciences and Engineering; University of Waterloo, David R. Cheriton School of Computer Science Li, Guojun; Shandong University, School of Mathematics
Keywords:	Algorithms, Alternative splicing, phasing graph, transcriptome assembly, overlap graph

SCHOLARONE™
Manuscripts

Bioinformatics, YYYY, 0–0
doi: 10.1093/bioinformatics/xxxxx
Advance Access Publication Date: DD Month YYYY
Manuscript Category

Subject Section

iPAC: A genome-guided assembler of isoforms via phasing and combing paths

Ting Yu[†,1], Juntao Liu[†,1], Xin Gao[*,2], Guojun Li[*,1]

[1]School of Mathematics, Shandong University, Jinan 250100, China

[2]Computational Bioscience Research Center (CBRC), Computer, Electrical and Mathematical Sciences and Engineering (CEMSE) Division, King Abdullah University of Science and Technology (KAUST), Thuwal, 23955, Saudi Arabia

*To whom correspondence should be addressed.

[†]These authors contributed equally to this work.

For Peer Review

Abstract

Motivation: Full-length transcript reconstruction is very important and quite challenging for the widely used RNA-seq data analysis. Currently available RNA-seq assemblers generally suffered from serious limitations in practical applications, such as low assembly accuracy and incompatibility with latest alignment tools.

Results: We introduce iPAC, a new genome-guided assembler for reconstruction of isoforms, which revolutionizes the usage of paired-end and sequencing depth information via phasing and combing paths over a newly designed phasing graph. Tested on both simulated and real datasets, it is to some extent superior to all the salient assemblers of the same kind. Especially, iPAC is significantly powerful in recovery of lowly expressed transcripts while others are not.

Availability: iPAC is freely available at http://sourceforge.net/projects/transassembly/files.

Contact: guojunsdu@gmail.com

Supplementary information: Supplementary data are available at Bioinformatics online.

1 Introduction

High throughput RNA-seq technology makes it possible to identify and quantify all expressed RNAs from a sample, and therefore enhances the capability of exploration of complex transcriptomic landscapes. It has been widely used to elucidate all splicing events as well as isoforms incurred in the process of transcription of eukaryotes (Marguerat and Bähler, 2010; Marioni, et al., 2008; Wang, et al., 2009; Wilhelm and Landry, 2009). This clearly sheds light on the study of complex diseases related to abnormal splicing events or expression levels such as cancers. However, genes in eukaryotes would generally produce multiple transcripts due to the diversity of alternative splicing events. The different

expression levels of the diversified transcripts to be detected enables the possibility of recovering them computationally. The diversity of splicing events mainly include but are not limited to skipped exons, retained introns, mutually exclusive exons, and even some exons partially spliced (Black, 2003; Matlin, et al., 2005; Wang, et al., 2008). In addition, an RNA-seq run could generate over 200 million short reads (currently, 50-250bp length) with unknown sequencing errors (Canzar, et al., 2016; Metzker, 2009). Therefore, it poses a highly challenging task to computationally and accurately recover all the expressed transcripts along with their abundances via assembling the huge amount of short RNA-seq reads.

There have been quite a few algorithms publicly available for assembling short RNA-seq reads into full-length transcripts. Among them, some are genome-guided while the others are de novo (Sharon, et al., 2013). Generally, if a high quality reference genome is available for model species, such as human, traditional genome-guided assemblers such as Cufflinks (Trapnell, et al., 2010), StringTie (Pertea, et al., 2015), Scallop (Mingfu Shao, 2017), TransComb (Liu, et al., 2016), 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 the RNA-seq reads to the reference 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). Based on the mapping results, a splicing graph or overlap graph is built for each gene locus, and then different computational models were applied to extract transcript-representing paths via traversing the graph. De novo strategies, which are more challenging than genome-guided, directly assemble short RNA sequences into full-length transcripts without using a reference genome. Such approach is more useful in cases where the reference genome is fragmented, uncompleted, or even unavailable. The state-of-the-art de novo assemblers such as TransLiG (Liu, et al., 2019), BinPacker (Liu, et al., 2016), Bridger (Chang, et al., 2015), Trinity (Grabherr, et al., 2011), ABySS (Birol, et al., 2009), SOAPdenovo-Trans (Xie, et al., 2014), and IDBA-Tran (Peng, et al., 2013), usually show much lower accuracy than the genome-guided ones, which benefit a lot from the use of a high quality reference genome.

The explosive growth of RNA-seq data has been driving the development of algorithms for assembling short RNA-seq reads into full-length transcripts. However, this problem is still remains to be solved as of today. As claimed in a recent study (Steijger, et al., 2013), even if all the exact exons of a gene are given, the tools are often unable to assemble the exons into correct isoforms. In this study, we challenge the problem by introducing a new assembler iPAC developed by revolutionizing the traditional use of paired-end and sequence depth information via phasing and combing paths over a newly designed phasing graph (see Figure 1 and the Methods section for details).

Tested on both simulated and real datasets, iPAC demonstrated a significant superiority over all compared assemblers, including Scallop, StringTie and Cufflinks under each of Hisat2, Star or Tophat2 mapping tools. For example, on one tested human dataset, iPAC correctly recovered 10.5%, 22.8% and 75.8% more transcripts than Scallop, StringTie and Cufflinks, respectively, with fewer false positives at the same time. In particular, iPAC is able to recover lowly expressed transcripts with accuracy significantly higher than others. For example, iPAC accurately recovered 34% more lowly expressed transcripts than Scallop, 40% more than StringTie, and even 60% more than Cufflinks on the first simulated dataset S1.

2 Methods

Starting from the traditional splicing graph, iPAC build a new graph model, the overlap graph for effective uses of the paired-end sequencing information. And then another new graph model, the phasing graph is introduced for solving the ambiguity of the connections of the in- and out-splicing junctions at each exon by effectively integrating the paired-end and sequence depth information. Finally, a newly designed path extension strategy was applied for recovering all the expressed transcripts by searching for an optimal path cover over the phasing graph (see Figure 1 for an example illustrating the entire process of the iPAC algorithm).

2.1 Construction of splicing graphs

The splicing graph plays a fundamental role in the design of iPAC, which is constructed from the mapping of RNA-seq reads to a reference genome using the mapping tools such as Hisat2, Star and Tophat2. Based on the mapping results, reads are usually clustered into expressed gene loci, and a splicing graph is generally built for each gene locus. Identification of exon-intron boundaries and exon-exon junctions is done from junction reads (i.e. those reads spanning two or more exons). Generally speaking,

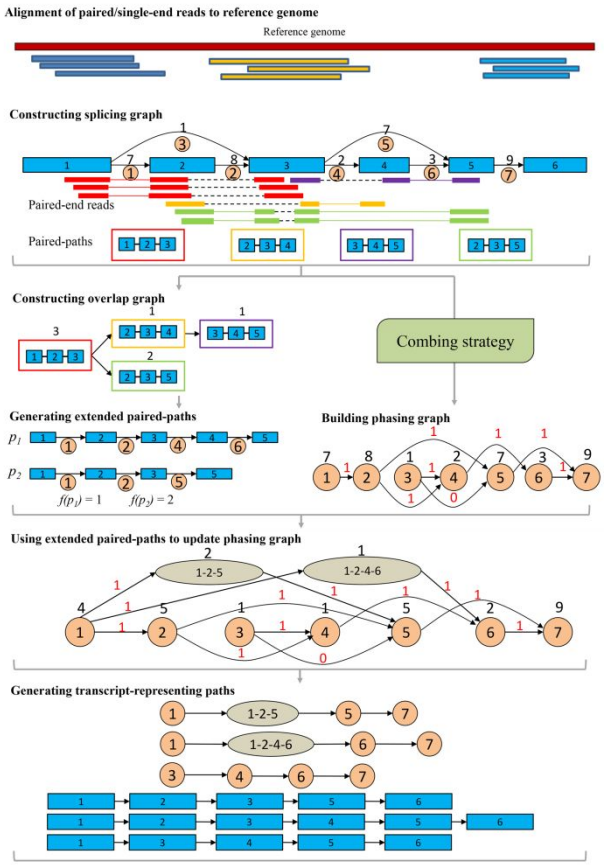

Figure 1 Flowchart of the iPAC algorithm.

each node in the splicing graph represents an exon in the corresponding gene, and a directed edge between two nodes means that there exists a junction read spanning the two exons. After graph construction, we assign a weight to each edge as the number of reads spanning it. Theoretically, edges in the splicing graphs could capture most splicing events in the expressed transcripts, and also, the sequencing depth information is appropriately integrated into the graph as the edge weight.

2.2 Construction of overlap graphs and generation of extended paired-paths

To make full use of the paired-end information, we construct an overlap graph of paired-end paths for each splicing graph and then extract the extended paired-paths from the overlap graph as follows.

1) Generating the set of paired-paths. Given the splicing graph G of a gene, we first extract all the sub-paths in graph G (named as **paired-paths**), each of which is supported by two paired-end reads. In detail, for two paired-end reads r_1 and r_2, if r_1 spans a path $P_1 = n_{i1}\rightarrow n_{i2}\rightarrow\ldots\rightarrow n_{ip}$ in graph G, while r_2 spans $P_2 = n_{j1}\rightarrow n_{j2}\rightarrow\ldots\rightarrow n_{jq}$, we will search for all the paths from n_{ip} to n_{j1} in graph G. If there exists one and only one path P_{in} between n_{ip} and n_{j1} satisfying the fragment insert size (e.g. $P_{in} = n_{ip}\rightarrow n_{m1}\rightarrow n_{m2}\rightarrow\ldots\rightarrow n_{ms}\rightarrow n_{j1}$) and $p + s + q \geq 3$, then reads r_1 and r_2 are connected by the path P_{in} and the corresponding paired-path should be $P = P_1\rightarrow P_{in}\rightarrow P_2$. After all the paired-end reads in graph G are processed, we obtain a set S_p of all the paired-paths. Different paired-end reads may generate the same paired-path. Therefore, each paired-path P is assigned a coverage $cov(P)$ as the number of paired-end reads that could generate the path P.

2) Constructing overlap graphs of paired-paths. Generating all paired-paths, we build an overlap graph G_p of all the paired-paths for each splicing graph G. To do so, we first define the compatibility between two paired-paths. The paired-paths P_1 and P_2 are said to be compatible if they share the same sub-path at the right (left) terminal of P_1 and the left (right) terminal of P_2, and the shared sub-path contains at least one edge of the splicing graph (see Figure S1 for an example). We then build the overlap graph G_p with nodes representing paired-paths in S_p, and two nodes being connected by a directed edge if and only if the corresponding paired-paths are compatible. Simultaneously, each node is weighted by the coverage of the corresponding paired-path, and each edge is weighted by the length of the shared sub-path between the two paired-paths. To simplify the structure of the overlap graph G_p, we contract the consecutive nodes in a linear path of graph G_p into a single node weighted by the minimum weight among those of the contracted nodes (see Figure S2 for an example).

3) Generating extended paired-paths. From the definition of overlap graphs, it is obvious that paired-paths could be extended for further approaching full-length transcripts. iPAC generates the extended paired-paths as follows.

(1) Choose an edge $e_{max} = (n_L, n_R)$ in graph G_p with the largest edge weight as the seed, which is not included in any extended paired-path. The node n_L (n_R) is then extended to one of its left (right) neighbor nodes n_L' (n_R') with the largest edge weight. If there are multiple choices, then the neighbor of the largest node weight will be selected for extension. Keep extending towards left (right) until encountering a node without in-degree (out-degree) and an extended paired-path p is extracted with its abundance $f(p)$ estimated as the minimum node weight in this path.

(2) For each node n in the extracted path p, we update its weight $cov(n)$ to be $cov(n)$-$f(p)$ in graph G_p. Suppose $cov_0(n)$ is the original weight of node n, and if $(cov(n)\text{-}f(p))/cov_0(n)$ is smaller than a threshold ε, it is considered that node n should not be included in any further extended paired-path, and thus this node along with its incident edges are removed from G_p. After each node in path p is processed, the overlap graph G_p is updated accordingly.

Repeat the above procedure until the graph G_p is empty, and then we obtain a set P_G of all the extended paired-paths for each splicing graph G. Intuitively, each path $p \in P_G$ should be included in an expressed transcript (see Figure S2 for an example).

2.3 Assembly of full-length transcripts

In this study, each extended paired-path with a high probability corresponds to a segment of an expressed transcript, and therefore should be covered by at least one predicted transcript. To achieve it, we first create a new graph model named as the phasing graph, then we update the phasing graph by contracting each extended paired-path into a new node in the phasing graph, and finally a transcript-representing path cover on the updated phasing graph is obtained by a newly designed path extension technique.

1) Building phasing graphs. Given a splicing graph G, each node may have multiple in-coming and out-going edges. To accurately connect the in-coming and out-going edges for each node in the splicing graph G, we first build a so-called phasing graph $L(G)$ of the splicing graph with nodes representing the edges in G and an edge representing two incident edges in G. The constructed phasing graph is actually a so-called line graph. The readers are referred to Figure S3 for an example. Each node in the phasing graph $L(G)$ is weighted by the coverage of its corresponding edge in the splicing graph G, while each edge is weighted by value 1 meaning that this edge possibly comes from an expressed transcript, and by value 0 otherwise (see Figure S3 for details). Therefore, the value 1 and 0 with a high probability dscriminate the correct connections between the in-coming and out-going edges for each node in splicing graph G. And the value 1 and 0 could be computed by updating a quadratic program obtained in TransLiG (Liu, et al., 2019) (see supplementary methods for details).

Getting the weighted phasing graph $L(G)$, we update it by contracting each extended paired-path into a new node which is called a contracted node so as to distinguishing from the original ones. Concretely, for an extended paired-path $p = n_1\rightarrow n_2\rightarrow\ldots\rightarrow n_m$, where each n_i represents an original node in the phasing graph $L(G)$, we contract the path p into a new node n_p which is weighted by $f(p)$ estimated above in Section 2, while the two new edges (n_1, n_p) and (n_p, n_m) are also added into the phasing graph and weighed by 1. Then we update the node weight $w(n)$ for each original node n of the phasing graph to be $w(n)$-$f(p)$ if node n is included in the extended paired-path p, and we obtain the final weighted phasing graph $L(G)$ (see Figure S3 for details).

2) Recovering transcripts by a novel path extension strategy. Based on the final phasing graph $L(G)$, a newly designed path extension strategy is applied to assemble all the expressed transcripts by searching for an optimal path cover over the phasing graph. In details, iPAC reconstructs the expressed transcripts by the following steps.

(1) Path extension. Choose an unused contracted node n_p (or an unused original node if all the contracted nodes have been included in the

For Peer Review

extended paths) with the largest node weight as the seed and extend it to one of its left (right) neighbor, with edges of weight 1. If there are multiple choices, the neighbor with the largest node weight is selected for extension. Keep extending until encountering a node without in-going (out-going) edges and a transcript-representing path p_t is predicted.

(2) Graph update. Defining f_{min} as the minimum node weight in the extended path p_t, we update the weight $w(n)$ to be $w(n)$-f_{min} for each node n in p_t (see Figure S3 for details).

Repeat the path extension procedure until all the nodes in graph $L(G)$ have been covered by the predicted transcripts and then a transcript-representing path cover is obtained, where all the extended paired-paths have been covered by the assembled transcripts. After transcriptome assembly, the abundance estimator kallisto (Nicolas, et al., 2016) is applied for estimating the expression levels of the assembled transcripts.

3 Results

We focused the mapping task on the latest mapping tools Hisat2, Star and Tophat2 in this study. Upon the mapping results of RNA-seq reads to a reference genome obtained by employing Hisat2, Star, and Tophat2 we ran iPAC and the salient assemblers, Scallop, StringTie and Cufflinks. For Tophat2 mappings, we further ran TransComb and IsoLASSO which are not compatible with Hisat2 or Star. Then we tested them on both simulated and real RNA-seq datasets in terms of the commonly used comparison criteria. In particular, a reference transcript is considered to be correctly recovered if and only if its intron chain is exactly matched with an assembled transcript, and this assembled transcript is defined as correctly assembled. The human reference genome GRCh37/hg19 and all the reference transcripts from UCSC hg19 gene annotation were used to build the mapping indexes for Hisat2, Star and Tophat2.

3.1 Evaluation on simulated data

FLUX simulator (Griebel, et al., 2012) explicitly simulates RNA-seq experimental steps including reverse transcription, PCR amplification, fragmentation, size selection and sequencing. In this study, we used FLUX simulator to generate five types of simulation datasets S1 (75bp-length, ~96 million paired-end reads), S2 (100bp-length, ~90 million paired-end reads), S3 (125bp-length, ~82 million paired-end reads), S4 (100bp-length, ~91 million paired-end reads) and S5 (125bp-length, ~122 million paired-end reads), the average fragment lengths of S1, S2, S3 and S5 are approximately 170, and the average fragment length of dataset S4 is 200. On these five simulated datasets, we tested iPAC, Scallop, StringTie and Cufflinks under Hisat2, Star and Tophat2 mappings, but for the assemblers TransComb and IsoLASSO, which are not compatible with Hisat2 or Star, we only tested them under Tophat2 mappings. Their performance was evaluated in terms of the assembly accuracy (sensitivity and precision) and the recovery of transcripts with different expression levels (low, middle, and high).

1) Comparison of assembly accuracy

We ran iPAC, Scallop, StringTie and Cufflinks on the five simulated datasets (S1, S2, S3, S4 and S5) by using the mapping results from Hisat2, Star and Tophat2, and for Tophat2 mappings, we also ran TransComb and IsoLASSO. The accuracy was evaluated in terms of sensitivity (the fraction of correctly recovered reference transcripts in the experiment) and precision (the percentage of correctly assembled transcripts out of all the output transcripts).

Based on Hisat2 mappings, both the sensitivity and precision obtained by iPAC achieved the highest among all the compared assemblers. On the simulated datasets S1, S2, S3, S4 and S5, the sensitivity of iPAC respectively reached 53.08%, 54.39%, 54.07%, 49.79% and 54.64%, versus 51.22%, 51.36%, 44.58%, 47.69% and 45.90% by Scallop, 49.22% 49.98%, 48.41%, 45.73% and 47.55% by StringTie, and 40.78, 41.14%, 39.66%, 27.61% and 34.23% by Cufflinks. Evaluated in precision, iPAC achieved 60.36%, 60.67%, 60.33%, 58.33% and 59.13% on the five datasets, respectively, versus 58.41%, 56.18%, 46.35%, 48.79% and 42.45% by Scallop, 47.54%, 46.00%, 43.34%, 39.13% and 40.31% by StringTie, and 46.16%, 48.33%, 48.73%, 36.29% and 37.63% by Cufflinks.

Based on Star mappings, iPAC again showed the best performance, its sensitivity reached 52.80%, 53.89%, 54.16%, 49.57% and 55.01% respectively on the five datasets, versus 50.39%, 51.65%, 52.40%, 47.30% and 53.87% by Scallop, 49.54%, 50.85%, 51.00%, 45.79% and 50.96% by StringTie, and 43.79%, 43.80%, 42.85%, 37.93% and 41.58% by Cufflinks. When evaluated in terms of precision, iPAC achieved 62.58%, 62.23%, 62.03%, 57.73% and 61.05% on the five datasets, respectively, versus 60.23%, 59.01%, 58.35%, 48.93% and 52.25% by Scallop, 52.07%, 52.59%, 52.11% 43.13% and 48.86% by StringTie, and 46.92%, 47.92%, 48.18%, 37.71% and 42.43% by Cufflinks.

Based on Tophat2 mappings, iPAC also achieved the highest accuracy in terms of both sensitivity and precision. On the data S1-S5, the sensitivity of iPAC were 48.97%, 50.41%, 50.44%, 45.63% and 50.34% respectively, versus 47.50%, 49.01%, 49.01%, 45.10% and 49.90% by Scallop, 47.21%, 48.80%, 48.78%, 44.08% and 47.96% by StringTie, and 47.74%, 48.95%, 48.53%, 44.32% and 48.35% by TransComb, while the other assemblers showed much worse performance. In terms of precision, iPAC achieved 56.50%, 58.27%, 58.88%, 57.02% and 58.59% respectively on the five datasets, versus 56.11%, 57.27%, 56.54%, 49.35% and 53.45% by Scallop, 45.80%, 47.24%, 48.47%, 39.99% and 45.98% by StringTie, and 51.14%, 53.39%, 54.07%, 53.26% and 54.06% by TransComb. (see Figure 2 a-e and Table S2-S6)

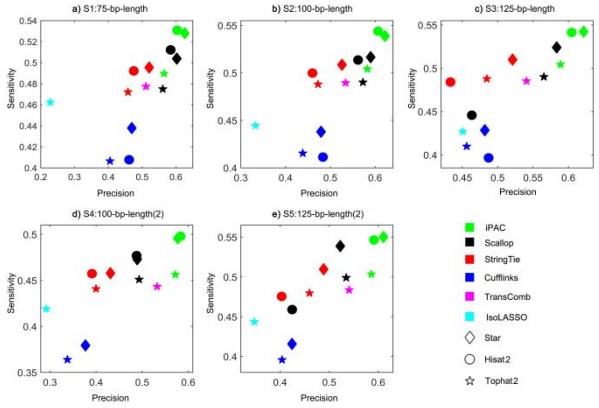

Figure 2 Comparisons of assembly accuracy on the five simulated datasets S1-S5.

2) Comparison of recovering transcripts with different expression levels

In general, transcripts with lower expression levels are more difficult to be recovered, while lowly expressed ones may play important roles in organisms. To evaluate the assembly tools in recovering transcripts with different expression levels, we sorted the expressed transcripts according

For Peer Review

to their expression abundances and then divided them into three equal subsets corresponding to low, middle and high expression levels following the same way in Scallop (Mingfu Shao, 2017). Comparing among these assemblers, we found that iPAC consistently achieved the highest sensitivity on different expression levels no matter using Hisat2 or Star mappings with an exception that a little lower than Scallop for middle expression levels where Tophat2 was used (see Figure 3). Moreover, iPAC demonstrated a significant superiority over all the others in recovering transcripts of low expression levels. It recovered 11%-55% more than Scallop, 17%-46% more than StringTie, 46%-110% more than Cufflinks, 3%-10% more than TransComb and 22%-152% more than IsoLASSO (see Table S11-S15). All the comparison results demonstrated that iPAC significantly outperforms Scallop, StringTie, Cufflinks, TransComb and IsoLASSO on all the simulated datasets of different sequencing lengths, different read numbers and different fragment lengths based on the mapping tools no matter Hisat2, Star or Tophat2.

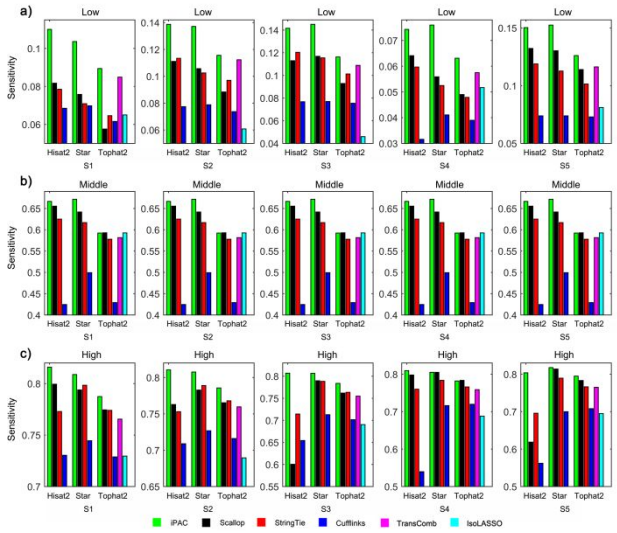

Figure 3 Comparisons of recovering transcripts with low, middle or high expression levels on the five simulated datasets S1-S5.

3.2 Evaluation on real data

Tests on simulation data are persuasive especially for the situation where the ground truth is unknown for real RNA-seq data. However, it is not realistic to capture the entire features of biological data by simulation alone, so it is necessary to implement an evaluation on real data to further verify the assembling performance in real applications. For fairness, the real RNA-seq reads were mapped to all the known reference transcripts from UCSC gene annotation hg19, and the ground truth is defined as the set of all these transcripts, whose entire junctions were covered by the mapped reads. We selected four human RNA-seq datasets, R1: K562 cells, R2: H1 cells (replicate 1), R3: H1 cells (replicate 2) and R4: VM-Cub1 cells on which the assemblers were tested. R1, R2 and R3 respectively contain 88 million, 44 million, and 37 million strand-specific paired-end reads, and R4 contains 28 million strand-specific single-end reads, which were downloaded from NCBI Sequence Read Archive (SRA) with accession codes: SRR387662, SRR307911, SRR307912 and ERR3639852, respectively. We then evaluated the assemblers on the four real datasets in terms of the same criteria as we did on simulation data, and results showed that iPAC consistently achieves the best performance on all the four real datasets using no matter Hisat2, Star or Tophat2 mappings.

1) Comparison of assembly accuracy

We therein ran the assemblers on the four real datasets based on Hisat2, Star and Tophat2 mappings respectively. When Hisat2 was employed, the sensitivity of iPAC on the four real datasets R1-R4 reached 39.60%, 37.89%, 37.90% and 41.87%, versus 36.39%, 35.15%, 34.21% and 41.49% by Scallop, 31.72%, 31.09%, 30.79% and 34.13% by StringTie, and 20.21%, 22.90%, 21.51% and 26.46% by Cufflinks. With Star, iPAC again obtained highest sensitivity, which were 42.18%, 40.50%, 39.76% and 43.85% respectively on all the four real datasets, versus 40.98%, 40.16%, 38.80% and 43.49% by Scallop, 33.22%, 32.84%, 32.19% and 36.19% by StringTie, and 23.62%, 25.70%, 24.68% and 30.60% by Cufflinks. Based on Tophat2 mapping, iPAC also showed the best performance (see Figure 4 and Table S7-S10). The comparison results showed that iPAC significantly improved on sensitivity, especially on R3 with Hisat2 mapping, where it correctly assembled 10.5% more reference transcripts than Scallop, 22.7% more than StringTie and 75.8% more than Cufflinks. With Star on R3, iPAC correctly assembled 2.3%, 23.3% and 60.8% more reference transcripts than those of Scallop, StringTie and Cufflinks, respectively.

The high sensitivity of iPAC was not at the cost of its precision. Evaluated in terms of precision, iPAC remains the best on datasets R2, R3 and R4 under all the three mappings tools (see Figure 4 and Table S7-S10). On dataset R1 under Hisat2 mapping, iPAC and Scallop showed similar precision of 31.49% and 31.22%, respectively, where filtering parameter of iPAC was set to 0.2, but they correctly recovered 13334 and 12488 reference transcripts respectively.

Moreover, we have evaluated the performance of the assemblers on additional 49 real datasets, and iPAC generally outperforms all the others on all the datasets (see supplementary Table S20 for details).

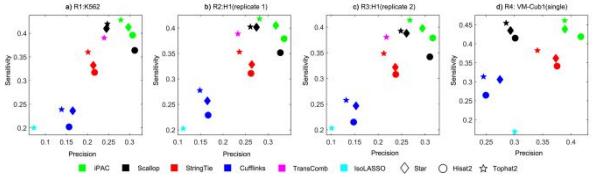

Figure 4 Comparisons of assembly accuracy on four real RNA-seq datasets R1-R4.

2) Comparison of consumptions of computing resources

To evaluate the computational efficiency of the assemblers, we compared the CPU time and memory usage of all the compared assemblers on the three real datasets. All the assemblers were run using 10 threads on the same server (96GB memory, 12-core CPU), and Scallop and StringTie consume the least CPU time under all the three mappings tools on all the four tested data. iPAC ran a little slower than Scallop and StringTie, but much faster than Cufflinks, TransComb and IsoLASSO. For memory usage, all the assemblers exhibited a similar trend, with maximum memory usage no more than 10 GB on all the tested data (see supplementary materials Table S19 for details). Overall, iPAC is not the most parsimonious, but it is acceptable for practical use.

4 Discussion

We present iPAC, a new genome-guided assembler for reconstruction of isoforms. Compared to the state-of-the-art assemblers including Scallop, StringTie, and Cufflinks, it demonstrated a significant superiority in performance on both simulated and real biological datasets using different mapping tools. The advantages may be attributed to 1) the overlap graph of paired paths, followed by a newly designed technique for iteratively extending paired-paths, leading to a quite effective use of paired-end information; 2) the phasing graph with the edges being weighted by 0 or 1, indicating whether or not an edge should be involved in a transcript. Resorting to the phasing graphs, the paired-end and sequencing depth are effectively integrated into the assembly procedure by solving a constrained quadratic program; 3) the newly developed technique for extracting all the transcript-representing paths over the phasing graphs that are guided by the edge weights on the phasing graphs. We then tested the effect of each of the above three techniques in the improvements of iPAC on the simulated datasets (see supplementary materials Section 2.1 for details), and found that the combination of these three techniques make the iPAC not only more sensitive, but also more precise in comparison with the other compared assemblers (see Figure S4 for an IGV screenshot of a specific gene).

The software has been developed to be user-friendly, and expected to play a crucial role in new discoveries of transcriptome study using RNA-seq, especially in complicated human diseases related to abnormal splicing events and expression levels, such as cancers.

Code and data availability

iPAC is implemented in C++ and is freely available as open source software at http://sourceforge.net/projects/transassembly/files. The real data K562 cells, H1 cells (replicate 1), and H1 cells (replicate 2) in this study can be downloaded from NCBI Sequence Read Archive (SRA) database with accession codes SRR387662, SRR307911, and SRR307912, respectively. The parameter setups of the Flux simulator and the three simulation datasets can be downloaded from http://sourceforge.net/projects/transassembly/files/Data. The reference transcripts and the running commands of the three mapping tools Hisat2, Star, and Tophat2, and each assembler have been detailed in Supplementary Materials.

Competing Financial Interests

The authors declare that they have no competing interests.

Authors' Contributions

Conceived and designed the experiments: GL. Performed the experiments: TY JL. Analyzed the data: JL TY. Contributed reagents/materials/analysis tools: TY JL. Wrote the paper: GL JL XG. Designed the software used in analysis: TY JL. Oversaw the project: GL.

Acknowledgments

This work was supported by the National Natural Science Foundation of China with codes 11931008, 61801265, 61877064, 31571354 and 61771009, and the Natural Science Foundation of Shandong Province with code ZR2018PA001, and by funding from King Abdullah University of Science and Technology (KAUST), under award number FCC/1/1976-18-01, FCC/1/1976-23-01, FCC/1/1976-25-01, FCC/1/1976-26-01, FCS/1/4102-02-01, 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.

References

Au, K.F., et al. (2010) Detection of splice junctions from paired-end RNA-seq data by SpliceMap, Nucleic acids research, 38, 4570-4578.

Behr, J., et al. (2013) MITIE: Simultaneous RNA-Seq-based transcript identification and quantification in multiple samples, Bioinformatics, 29, 2529-2538.

Birol, I., et al. (2009) De novo transcriptome assembly with ABySS, Bioinformatics, 25, 2872-2877.

Black, D.L. (2003) Mechanisms of alternative pre-messenger RNA splicing, Annu Rev Biochem, 72, 291-336.

Bray, N.L., et al. (2016) Near-optimal probabilistic RNA-seq quantification, Nat Biotechnol, 34, 525-527.

Canzar, S., et al. (2016) CIDANE: comprehensive isoform discovery and abundance estimation, Genome Biology, 17.

Chang, Z., et al. (2015) Bridger: a new framework for de novo transcriptome assembly using RNA-seq data, Genome Biol, 16.

Dobin, A., et al. (2013) STAR: ultrafast universal RNA-seq aligner, Bioinformatics, 29, 15-21.

Feng, J.X., Li, W. and Jiang, T. (2011) Inference of Isoforms from Short Sequence Reads, Journal of Computational Biology, 18, 305-321.

Grabherr, M.G., et al. (2011) Full-length transcriptome assembly from RNA-Seq data without a reference genome, Nature biotechnology, 29, 644-652.

Griebel, T., et al. (2012) Modelling and simulating generic RNA-Seq experiments with the flux simulator, Nucleic Acids Research, 40, 10073-10083.

Guttman, M., et al. (2010) Ab initio reconstruction of cell type-specific transcriptomes in mouse reveals the conserved multi-exonic structure of lincRNAs, Nature biotechnology, 28, 503-510.

Kim, D., Langmead, B. and Salzberg, S.L. (2015) HISAT: a fast spliced aligner with low memory requirements, Nat Methods, 12, 357-360.

Kim, D., et al. (2013) TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions, Genome Biology, 14.

Li, W., Feng, J.X. and Jiang, T. (2011) IsoLasso: A LASSO Regression Approach to RNA-Seq Based Transcriptome Assembly, Journal of Computational Biology, 18, 1693-1707.

Li, W. and Jiang, T. (2012) Transcriptome assembly and isoform expression level estimation from biased RNA-Seq reads, Bioinformatics, 28, 2914-2921.

Liu, J., et al. (2016) BinPacker: Packing-Based De Novo Transcriptome Assembly from RNA-seq Data, PLoS Comput Biol, 12, e1004772.

Liu, J., et al. (2016) TransComb: genome-guided transcriptome assembly via combing junctions in splicing graphs, Genome Biol, 17, 213.

Liu, J., T. Yu, et al. (2019). TransLiG: a de novo transcriptome assembler that uses line graph iteration. Genome Biol 20(1): 81.

For Peer Review

Marguerat, S. and Bähler, J. (2010) RNA-seq: from technology to biology, Cellular and molecular life sciences, 67, 569-579.

Marioni, J.C., et al. (2008) RNA-seq: an assessment of technical reproducibility and comparison with gene expression arrays, Genome Res, 18, 1509-1517.

Matlin, A.J., Clark, F. and Smith, C.W. (2005) Understanding alternative splicing: towards a cellular code, Nat Rev Mol Cell Biol, 6, 386-398.

Metzker, M.L. (2009) Sequencing technologies—the next generation, Nature Reviews Genetics, 11, 31-46.

Mezlini, A.M., et al. (2013) iReckon: Simultaneous isoform discovery and abundance estimation from RNA-seq data, Genome Research, 23, 519-529.

Mingfu Shao, C.K. (2017) Accurate assembly of transcripts through phase-preserving graph decomposition, Nat Biotechnol, 35, 1167 - 1169.

Nicolas, et al. (2016) Near-optimal probabilistic RNA-seq quantification, Nat Biotechnol, 34, 525 - 527.

Peng, Y., et al. (2013) IDBA-tran: a more robust de novo de Bruijn graph assembler for transcriptomes with uneven expression levels, Bioinformatics, 29, i326-334.

Pertea, M., et al. (2015) StringTie enables improved reconstruction of a transcriptome from RNA-seq reads, Nat Biotechnol, 33, 290-295.

Sharon, D., et al. (2013) A single-molecule long-read survey of the human transcriptome, Nat Biotechnol, 31, 1009-1014.

Song, L., Sabunciyan, S. and Florea, L. (2016) CLASS2: accurate and efficient splice variant annotation from RNA-seq reads, Nucleic Acids Res, 44, e98.

Steijger, T., et al. (2013) Assessment of transcript reconstruction methods for RNA-seq, Nat Methods, 10, 1177-+.

Tomescu, A.I., et al. (2013) A novel min-cost flow method for estimating transcript expression with RNA-Seq, BMC Bioinformatics, 14 Suppl 5, S15.

Trapnell, C., Pachter, L. and Salzberg, S.L. (2009) TopHat: discovering splice junctions with RNA-Seq, Bioinformatics, 25, 1105-1111.

Trapnell, C., et al. (2010) Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation, Nature biotechnology, 28, 511-515.

Wang, E.T., et al. (2008) Alternative isoform regulation in human tissue transcriptomes, Nature, 456, 470-476.

Wang, Z., Gerstein, M. and Snyder, M. (2009) RNA-Seq: a revolutionary tool for transcriptomics, Nature Reviews Genetics, 10, 57-63.

Wilhelm, B.T. and Landry, J.-R. (2009) RNA-Seq—quantitative measurement of expression through massively parallel RNA-sequencing, Methods, 48, 249-257.

Wu, T.D. and Nacu, S. (2010) Fast and SNP-tolerant detection of complex variants and splicing in short reads, Bioinformatics, 26, 873-881.

Xie, Y.L., et al. (2014) SOAPdenovo-Trans: de novo transcriptome assembly with short RNA-Seq reads, Bioinformatics, 30, 1660-1666.

For Peer Review