<table>
<thead>
<tr>
<th>Title</th>
<th>IDBA-tran: a more robust de novo de Bruijn graph assembler for transcriptomes with uneven expression levels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author(s)</td>
<td>Peng, Y; Leung, HCM; Yiu, SM; Lv, MJ; Zhu, XG; Chin, FYL</td>
</tr>
<tr>
<td>Citation</td>
<td>Bioinformatics, 2013, v. 29 n. 13, p. i326-i334</td>
</tr>
<tr>
<td>Issued Date</td>
<td>2013</td>
</tr>
<tr>
<td>URL</td>
<td><a href="http://hdl.handle.net/10722/187134">http://hdl.handle.net/10722/187134</a></td>
</tr>
<tr>
<td>Rights</td>
<td>This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.</td>
</tr>
</tbody>
</table>
IDBA-tran: a more robust de novo de Bruijn graph assembler for transcriptomes with uneven expression levels

Yu Peng1, Henry C. M. Leung1, Siu-Ming Yiu1, Ming-Ju Lv2, Xin-Guang Zhu2 and Francis Y. L. Chin1,*

1Department of Computer Science, The University of Hong Kong, Hong Kong and 2CAS-MPG Partner Institute for Computational Biology, Shanghai Institutes for Biological Sciences, Chinese Academy Sciences, Shanghai 200031, China

ABSTRACT

Motivation: RNA sequencing based on next-generation sequencing technology is effective for analyzing transcriptomes. Like de novo genome assembly, de novo transcriptome assembly does not rely on any reference genome or additional annotation information, but is more difficult. In particular, isoforms can have very uneven expression levels (e.g. 1:100), which make it very difficult to identify low-expressed isoforms. One challenge is to remove erroneous vertices/edges with high multiplicity (produced by high-expressed isoforms) in the de Bruijn graph without removing correct ones with not-so-high multiplicity from low-expressed isoforms. Failing to do so will result in the loss of low-expressed isoforms or having complicated subgraphs with transcripts of different genes mixed together due to erroneous vertices/edges.

Contributions: Unlike existing tools, which remove erroneous vertices/edges with multiplicities lower than a global threshold, we use a probabilistic progressive approach to iteratively remove them with local thresholds. This enables us to decompose the graph into disconnected components, each containing a few genes, if not a single gene, while retaining many correct vertices/edges of low-expressed isoforms. Combined with existing techniques, IDBA-Tran is able to assemble both high-expressed and low-expressed transcripts and outperform existing assemblers in terms of sensitivity and specificity for both simulated and real data.


Contact: chin@cs.hku.hk

Supplementary information: Supplementary data are available at Bioinformatics online.

1 INTRODUCTION

Recent development of massively parallel cDNA sequencing (RNA-Seq) provides a more powerful and cost-effective way to analyze transcriptome data. RNA-Seq has been used successfully to identify novel genes, refine 5’ and 3’ ends of genes, study gene functions (Graveley, 2008), locate exon/intron boundaries (Nagalakshmi et al., 2008; Trapnell et al., 2009) and estimate expression levels of isoforms (Jiang and Wong, 2009).

However, transcriptome reconstruction (the reconstruction of all expressed transcripts) from RNA-seq data remains a challenging unresolved problem when there is splicing, i.e. when different combinations of regions (exons) of a single gene are decoded to multiple transcripts (isoforms) (Trapnell et al., 2010). Currently, there are two computational approaches to solve this problem. Alignment-based methods, such as Cufflinks (Trapnell et al., 2010) and Scripture (Guttman et al., 2010), first align reads to reference genomes using splice junction mappers, such as TopHat (Trapnell et al., 2009), to identify exon-intron boundary and then build a graph in which exons are the nodes and two exons are connected if reads connect them. Cufflinks (Trapnell et al., 2010) attaches weights to edges and models the isoform reconstruction problem as a minimum path cover problem, while Scripture (Guttman et al., 2010) creates a statistical model to identify significant segments as isoforms.

In contrast, de novo assembly methods, such as Trinity (Grabherr et al., 2011), Oases (Schulz et al., 2012), Trans-Abyss (Robertson et al., 2010) and T-IDBA (Peng et al., 2011), assemble transcripts directly from reads.

Alignment-based transcriptome assembly methods, which rely on reference genomes and additional annotation information, may suffer from missing/erroneous information. Also, the quality of these methods depends heavily on the accuracy of the alignment tools (Trapnell et al., 2009), which is also complicated by splicing and sequencing errors. As RNA-Seq technology becomes more mature, there will be an increasing need to reconstruct unknown transcriptomes without reference genome information, and de novo transcriptome assembly will become increasingly more important.

Difficulties: At first glance, the de novo transcriptome assembly problem looks similar to the de novo genome assembly problem. In fact, many existing methods for de novo transcriptome assembly, like genome assembly, apply the de Bruijn graph approach with fragments of transcripts being simple paths in graph, in which a vertex is a k-mer and an edge exists between two vertices u and v if u and v appear consecutively in a read. However, two main aspects make the two assembly problems different.

1) Exons shared by multiple isoforms. In this paper, we focus on transcriptome assembly for eukaryotes with splicing since, without splicing, the problem is much easier. Consider the example (LOC_Os10g02220 from rice) in Figure 1. A to I represent different exons forming 5 isoforms (in red). Shared exons (e.g. D and H) look like repeats, and most genome assemblers try to resolve repeats at the branch level, i.e., each branch needs to be supported by paired-end reads. In our case, since all five isoforms are real, branches BD and CD as well as DE and DF will be supported. Some assemblers may stop at the junctions, reporting B, C, D, E and F as separate (short) contigs or falsely regard CDE as a transcript (provided both CD and DE have enough support). For example, running Velvet on the rice data...
Existing solutions: Oases (Schulz et al., 2012) and Trinity (Grabherr et al., 2011) are two popular de novo transcriptome assemblers for RNA-Seq data. In order to solve the splicing problem [Issue (1)], both apply a dynamic programming approach to identify potential paths in the graph, which are supported by many reads or paired-end reads. In other words, they try to identify isoforms more globally through a path-level analysis instead of a local branch-level analysis. The results are much better than those of genome assemblers. However, since the problem is NP-complete (proved in the Supplementary Appendix), the running time of the dynamic programming approach increases exponentially with the number of branches in the de Bruijn graph. Due to Issue (2), erroneous reads sampled from high-expressed transcripts introduce many branches (with more support than reads sampled from low-expressed transcripts) and thus dynamic programming takes a long time. In practice, these tools fall back on heuristic search instead of dynamic programming for large components.

To tackle Issue (2), T-IDBA (Peng et al., 2011) uses another approach to isolate components. Based on the observation that transcripts from different genes share less common vertices when \( k \) value is large, T-IDBA builds a de Bruijn graph from small \( k \) and iteratively updates the graph with larger \( k \) values. It then finds transcripts in the de Bruijn graph with large \( k \) value where transcripts from the same gene usually form a single component. However, it does not perform very well for low-expressed transcripts because there are more missing \( k \)-mers when \( k \) is large. There is no dedicated solution in T-IDBA that solves the issue of erroneous \( k \)-mers within a component and methods for isolating components are not sensitive to low-expressed isoforms.

To recover low-expressed transcripts, several post-processing methods (Robertson et al., 2010; Surget-Groba and Montoya-Burgos, 2010) were developed for Velvet (Zerbino and Birney, 2008) and Abyss (Simpson et al., 2009). They are all based on the observation that lower \( k \) values make the assembler more sensitive to low-expressed transcripts, while larger \( k \) values make it more specific to high-expressed transcripts. In order to combine the advantages of different values of \( k \), the resultant contigs, generated by different \( k \)-mer lengths independently, are merged together.

However, merging assembly results from different runs is not a straightforward task. Although output transcripts are clustered and duplicated transcripts are removed, many duplicates are difficult to detect and errors can accumulate in the cluster-remove step. As a result, multiple contigs with errors are generated for the same transcripts and the number of resulting contigs is much more than the number of expressed transcripts. Oases-M, an extension of Oases, makes use of multiple \( k \) to improve its assembly result and is now the best tool using this approach. However, since the fundamental problem of removing erroneous vertices from high-expressed isoforms while keeping correct vertices from low-expressed isoforms is not solved, there are still many false positives as well as duplicated transcripts (Section 3).

Some single-cell genome assemblers (Chitsaz et al., 2011; Peng et al., 2012) also have a problem with uneven multiplicities of correct \( k \)-mers. They resolve the problem based on the assumption that, although the multiplicities of these erroneous \( k \)-mers are high, their multiplicities should be lower than the nearby correct \( k \)-mers. Thus, they calculate a local threshold, based on
the multiplicities of nearby \( k \)-mers or contigs, for removing erroneous \( k \)-mers. However, as a \( k \)-mer representing the common exon of several expressed isoforms can have relatively higher multiplicity than nearby correct \( k \)-mers (Li and Jiang, 2012), calculating the local threshold from only one or two nearby \( k \)-mers or contigs may be misleading and the algorithms may remove many correct \( k \)-mers near these high multiplicity \( k \)-mers.

Our contributions: If Issue (2) can be resolved, Issue (1) can be tackled by existing path-level analysis as the components will be simple enough. Thus, our core contribution is handling Issue (2). As mentioned before, the traditional filtering method of using one single global threshold for multiplicity cannot separate correct \( k \)-mers sampled from low-expressed transcripts from erroneous \( k \)-mers sampled from high-expressed transcripts, and single-cell genome assemblers calculating local thresholds from nearby \( k \)-mers may remove many correct \( k \)-mers. Thus, we propose a probabilistic progressive approach to solve this problem. Our proposed assembler IDBA-Tran calculates the probability that a \( k \)-mer or short simple path (contigs) contains error using not only the multiplicity of the \( k \)-mer or contig (or their neighboring \( k \)-mers or contigs) but also uses a multi-normal distribution to model the multiplicities of all \( k \)-mers in the whole connected component. Based on the multi-normal distribution and the contig length (as a short simple path is more likely to have error than a long one), IDBA-Tran calculates a local threshold for determining whether a \( k \)-mer or contig has error. By progressively removing erroneous \( k \)-mers, connected components representing isoforms from a single gene are identified. Since we successfully remove many erroneous \( k \)-mers, the size of each component is small. We can employ a path-level analysis (similar to Oases and Trinity) to identify transcripts from each component [Issue (1)]. Thus IDBA-Tran can perform better than Oases and Trinity, producing more contigs, particularly for low-expressed transcripts. Results show that IDBA-Tran outperforms other de novo transcriptome assembly approaches in terms of both sensitivity and specificity for both simulated and real data. IDBA-Tran also makes use of other techniques used in genome assemblers, such as tips pruning, path merging and error correction.

2 METHODS

Similar to Oases-M, IDBA-Tran also adopts the idea of multiple \( k \) to handle transcripts with different expression levels. However, instead of generating a de Bruijn graph and finding transcripts for each \( k \) value, an accumulated de Bruijn graph is built to capture all information from both high-expressed and low-expressed transcripts. During each iteration, an accumulated de Bruijn graph \( H_k \) for a fixed \( k \) is constructed from the input reads and the contigs constructed in previous iterations, i.e. those contigs constructed in \( H_{k-1} \) are treated as input reads for the construction of \( H_k \). The depth information is used to separate de Bruijn graph into components. Ideally, transcripts from different genes are decomposed into different components. In each component, alternative splicing can be detected and transcripts can be reconstructed. To accumulate information, all reconstructed transcripts are used as input reads for the next iteration.

Figure 3 shows the workflow of IDBA-Tran for assembling a set of paired-end reads. In the first iteration when \( k = k_{\min} \), \( H_k \) is equivalent to a de Bruijn graph for vertices whose corresponding \( k \)-mers have multiplicity of at least \( m \) (2 by default) times in all reads. During all subsequent iterations, sequencing errors are first removed according to the topological structure of \( H_k \) in a slightly different way to other assemblers (Section 2.1). The tips (dangling paths in \( H_k \) of length shorter than 2\( k \)) are likely to be false positives (Li et al., 2010; Simpson et al., 2009; Zerbino and Birney, 2008). Similar paths (bubbles) representing very similar contigs with the same starting vertex and ending vertex are likely to be caused by errors or SNPs and they should be merged (Li et al., 2010; Simpson et al., 2009; Zerbino and Birney, 2008). Then, the depth information for contigs and components is used to decompose the graph into components (Section 2.2). Paths with high support for the paired-end reads are reconstructed as transcripts in each component (Section 2.3). Errors in the assembled contigs are corrected by aligning reads to the contigs (Section 2.4). When constructing \( H_{k+1} \) from \( H_k \), each length \( s + 1 \) path in \( H_k \) is converted into a vertex (\( (k + s) \)-mer) and there is an edge between two vertices if the corresponding \( (k + s + 1) \)-mer appears \( f \) (1 by default) times in reads or once in contigs in \( C_k \cup LC_k \cup T_k \), where \( C_k \) represents the set of contigs, \( LC_k \) is the set of contigs constructed by local assembly using paired-end information (Section 2.5), and \( T_k \) is the set of transcripts when considering \( H_k \). In the following subsections, we describe each step of IDBA-Tran in detail.

2.1 Pruning short tips and merge similar path

Many de novo assemblers remove tips (short simple paths leading to dead ends) in the de Bruijn graph as erroneous contigs. It would not be advisable to remove such tips in transcriptome assembly, because transcripts are very short (could be several hundred bases) when compared to genomes. Removing one hundred bases from the end of a genome may not be a problem, but removing one hundred bases from the end of a transcript may lose much important information. When constructing the accumulated de Bruijn graph in IDBA-Tran, the tip removal process will take place at each iteration. Instead of removing all tips and producing shorter transcripts, IDBA-Tran keeps the longest tip (with highest probability of being a correct path) and removes all other short tips. For
Recall that T-IDBA (Peng et al., 2012) decomposes the graph by iterating depth. Assume branching at \( \exists \) for which the erroneous base is changed to each other nucleotide. Since there are three possible tide changed to a particular nucleotide, say 'A', is usually, the existence probability of erroneous \( k \)-mers is sampled as

\[
\frac{p_{\text{err}}}{3} = (1 - e^{-k})^{-1}.
\]

As \( v \) can be sampled with error as \( v' \), i.e. at least \( m \) of the samples have the same error at the last nucleotide. Since there are three possible \( v' \), the existence probability of \( v' \) (probability of branching at \( u \)) is

\[
P(m, x, p_{\text{err}}) = 3P(\geq 1 v' \text{ exist}) - 3P(\geq 2 v' \text{ exist}) + P(3 v' \text{ exist})
\]

\[
= 3 \sum_{i=0}^{m} P(\geq 1 v' \text{ exist}, i \text{ k-mer } v \text{ exist})P(i \text{ k-mer } v \text{ exist})
\]

\[
- 3 \sum_{i=0}^{2m} P(\geq 2 v' \text{ exist}, i \text{ k-mer } v \text{ exist})P(i \text{ k-mer } v \text{ exist})
\]

\[
+ \sum_{i=0}^{3m} P(3 v' \text{ exist}, i \text{ k-mer } v \text{ exist})P(i \text{ k-mer } v \text{ exist})
\]

\[
= 3 \sum_{i=0}^{m} C_{k-1}^{2}e^{-m-i}2^{n-i-1}(1 - p_{\text{err}})^{m-i} - 3 \sum_{i=0}^{2m} C_{k-1}^{2}e^{-2m-i}2^{n-i-1}(1 - p_{\text{err}})^{m-i} + \sum_{i=0}^{3m} C_{k-1}^{2}e^{-3m-i}2^{n-i-1}(1 - p_{\text{err}})^{m-i}.
\]
In order to estimate the value of depth $x$, we use a multi-normal distribution to model the depth distribution of a component as there can be multiple isoforms, say $t$, in a component. Given a set of $k$-mers with different multiplicities in the same component, we assume the multiplicities of the $k$-mers are sampled from $t$ normal distributions. Although the mean and standard deviation of each normal distribution can be estimated by expectation-maximization algorithm (Tanaseichuk et al., 2012), the time is too long because there are many $k$-mers and components. Thus IDBA-Tran applies an approximation by clustering the $k$-mers based on their multiplicities (the distance between two $k$-mers equals their difference in multiplicities) using $K$-means clustering method. The mean and standard deviation can then be calculated for each cluster. We set $t=3$ in the experiments based on the assumption that there are at most 3 transcripts in each component (at the final step).

Let $N(\mu, \sigma)$ be a normal distribution of depth with minimum mean depth value $\mu$. The probability that we wrongly remove a correct contig with average depth $\leq \beta$ is at most

$$P(\text{false positive}) = \frac{1}{\sqrt{2\pi}\sigma^2} \int_{-\infty}^{\beta} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \cdot 2P(m, x, p_{\text{err}})dx$$

Note that for an exon of length at least $I$ of sequencing depth $x$, the probability of branching is $2P(m, x, p_{\text{err}})$.

The value of $l$ should be selected based on the length of exons. If a very large $l$ is selected, true positive $k$-mers and paths are removed. If a very small $l$ is selected, most true negative $k$-mers and paths cannot be removed. We should select different values for $l$ depending on the properties of the data (we use $l=2k$ in the experiments). Once $l$ is selected, we can calculate the largest $\beta$ such that $P(\text{false positive})$ is lower than some value, say 1%, so as to remove most erroneous contigs without too many false positives.

Algorithm 1 shows the pseudocode for the decomposing step. According to (Peng et al., 2011), when the size of the component is small (with $\leq y = 50$ contigs), the component is likely to represent isoforms from a single gene and we can use a very low depth threshold $\beta = 0.1 \times T(c)$, where $T(c)$ is the average depth of connected component $c$, to prevent removing correct contigs. The filtering depth cutoff threshold $\tau$ is increased by a factor of $a$ progressively (10% by default). In each iteration, short contig $c$ is removed if its depth $T(c)$ is lower than the minimum of cutoff threshold $\tau$ and the depth threshold $\beta$.

### 2.3 Finding transcripts

**Algorithm 1.** Progressive-Component-Depth($G, k$)

$t \leftarrow 1$

repeat until $t > \max_{c \in G} T(c)$

for each component $c$ in $G$

if $\text{size}(c) > y$, then calculate $\beta$, else $\beta = 0.1 \times T(c)$

for each contig $c$ in $comp$

if $\text{len}(c) < 2k$ and $T(c) < \min(t, \beta)$

remove $c$ from $G$

$t \leftarrow t \times (1 + a)$

For each connected component in the de Bruijn graph, IDBA-Tran discovers those paths starting from a vertex with zero in-degree to a vertex with zero out-degree with the highest support from paired-end reads. A path is supported by paired-end reads if the paired-end reads can be aligned to the path with the distance between the aligned positions matching the insert distance of the paired-end reads. The problem definition can be simplified as follows [Transcripts Discovering (TD) Problem]: given a de Bruijn graph $G(V, E)$ with a set of vertices $V$ and edges $E$, a set of paired-end reads $P = \{(v_i, v_j)\}$, $v_i, v_j \in V$, an insert distance $d$ and error $s$, find $t$ paths in $G$ with the maximum number of supporting paired-end reads $P \subseteq P$. A path $p$ has a supporting paired-end read $(v_i, v_j)$ if $p$ contains vertices $v_i$ and $v_j$ and the distance between $v_i$ and $v_j$ in $p$ is between $d - s$ and $d + s$.

Since the TD problem is a NP-hard problem (see Supplementary Appendix), IDBA-Tran performs a heuristic depth-first search to find paths from a zero in-degree vertex to a zero out-degree vertex with maximum supporting paired-end reads. At each branch, the path with many supporting paired-end reads will be considered before other paths. In practice, IDBA-Tran reports at most $t_{\text{max}}$ (default 3) potential transcripts for each zero in-degree vertex in each connected component. IDBA-Tran applies a seed and extend method for aligning reads to contigs (paths in the de Bruijn graph). $k$-mers in a read appearing in the de Bruijn graph is considering as potential aligned position and IDBA-Tran will try to extend both ends of alignment considering substitution error only. Note that insertion and deletion error can be implemented in IDBA-Tran easily. However, as the number of substitution errors appears much more than the insertion/deletion errors, IDBA-Tran considers substitution error only for speeding up the alignment process.

### 2.4 Error correction

The error correction step is performed on reads and assembled contigs during the assembling process. At first, reads are aligned to each contig. The consensus of the aligned reads will replace the original contig, i.e. positions of the contig inconsistent with the majority of aligned reads will be corrected. Then aligned reads are corrected according to the aligned position in contigs, i.e. positions in the reads with nucleotides inconsistent with the consensus will be corrected. This error correction step can reduce the number of erroneous reads and branches in the de Bruijn graph.

### 2.5 Local assembly

Let $C$ be the set of contigs. We extract the beginning and end of each contig $c$ in $C$ to form a set of contigs $\tilde{C}$. Assume the insert distances of paired-end reads satisfy the normal distribution $N(d, \delta)$. IDBA-Tran performs local assembly (Peng et al., 2012) on the last $d + 3\delta$ bases of each end of the contig and the paired-end read with one end aligned to it. Since those reads which are far away from contig $c$ will not mix with reads with one end aligned to $c$, some missing $k$-mers can be reconstructed and the contigs can be extended longer.

### 2.6 Estimating expression levels

Since IDBA-Tran is designed for assembling reads to reconstruct expressed transcripts, sophisticated algorithms can be then applied to estimate the expression levels of each transcript. IDBA-Tran also provides an estimated expression level for each transcript by aligning reads to the transcript. RPKM (Reads Per Kilobase per Million mapped reads) is estimated by dividing the total length of reads uniquely aligned to a transcript by the total length of regions of transcript uniquely aligned by reads.

### 3 RESULTS

To evaluate the performance of IDBA-Tran, experiments were carried out on both simulated and real data. We compared IDBA-Tran with the latest transcriptome assemblers Trinity (Grabherr et al., 2011) and Oases (Schulz et al., 2012). We also compared IDBA-Tran with the single-cell genome assembler IDBA-UD (Peng et al., 2012) and Velvet-SC (Chitsaz et al., 2011), which apply multiple depths when assembling genomes. IDBA-Tran and IDBA-UD were run with $k$ ranging from 20 to 50 with step size 5. For Oases and Velvet-SC, $k$ values ranging from 20 to 50 with step size 5 were used, and the best result was selected as output. As the $k$ value of Trinity was fixed to 25, the default parameters were used to run it.
For transcriptome assembly, the most important indicator of assembly quality is the number of correct transcripts an assembler can reconstruct. In the experiments, known transcript references were used for benchmarking. A known transcript is reconstructed successfully if a certain portion, say 80% (referred as completeness), of its sequence is covered by a contig with 95% similarity. Similarly, the contig is considered correct if it can be aligned to at least 80% of a transcript with 95% similarity. The alignment of contigs to transcripts was performed by BLAT (Kent, 2002) without considering long gaps representing introns (as we aligned contigs to transcripts instead of genome). The sensitivity and specificity were calculated to measure performance. Sensitivity is the percentage of reconstructed transcripts over all expressed transcripts. Specificity is the percentage of correct contigs over all reported contigs.

We also compared the performance of IDBA-Tran and CEM (Li and Jiang, 2012) on estimating expression levels of reconstructed transcripts. CEM requires the genome sequence as additional information. By aligning reads to the reference genome, CEM can predict the expressed transcript sequences and estimate the expression level of each transcript based on a statistical model (quasi-multinomial model). Since some transcripts may align to multiple contigs and some contigs may align to multiple transcripts, we considered only those transcripts and contigs with one-to-one correspondence. The Pearson’s correlation between the predicted expression levels and the exact expression levels was calculated. As suggested in Li and Jiang, 2012, we also calculated the Pearson’s correlation between the log of predicted expression levels and the log of exact expression levels.

### 3.1 Simulated data

In order to simulate more realistic data, we aligned all reads in a real RNA-Seq data of *Oryza sativa* Transcriptome to known transcripts of *Oryza sativa* in the database. Based on the alignment results, we identified the set of expressed transcripts and estimated their expression levels. Note that, since the transcript sequences were known, no long gap (representing an intron) alignments were allowed. Reads aligned to multiple transcripts were not considered for estimating the depth of the transcript. We used all transcripts with at least 80% of the region aligned by reads and with depth at least 0.5× to generate the simulated data, i.e., there were 0.5 reads covering each nucleotide on average. We sampled paired-end reads from the transcripts according to their expression levels. The read length was 100 bp, error rate was 1% and the insert distance followed a normal distribution with mean 250 bp and standard deviation 25 bp.

Figure 4 shows the quality of the assembly results of the assemblers for different levels of completeness. Velvet-SC reconstructed the least number of transcripts in all completeness settings because it is designed for assembling genomic data and cannot handle multiple isoforms of the same gene well. IDBA-UD, which is designed for assembling genomic and metagenomic data, handled some simple cases of multiple isoforms of the same gene and had better performance than Velvet-SC. Trinity and Oases found more transcripts than Velvet-SC and IDBA-UD because they are designed for assembling transcriptome data. Oases had its best performance when k was set to 25, the same k value for Trinity. IDBA-Tran reconstructed the most transcripts and consistently reported more correct contigs. Table 1 shows the detailed figures when completeness was 0.8. The number of correct contigs and reconstructed transcripts for IDBA-Tran were the highest among the tools. IDBA-Tran also had the highest sensitivity and specificity.

Table 2 shows the expression level distribution of reconstructed transcripts of the assemblers when completeness was 0.8. For low-expressed transcripts with sequencing depth <5, IDBA-Tran reconstructed 664 more transcripts (33% more) than the second best tool (Trinity). This is probably due to the fact that IDBA-Tran can separate transcripts from different genes into components efficiently while preserving the low-expressed transcripts. We also checked the quality of decomposition and found that over 80% of the transcripts were still inside the same component after the decomposition step (referred to as unbroken transcripts after decomposition). Except transcripts with very low sequencing depth (<5), most transcripts were not broken after the decomposition and could be constructed successfully. Similar results were found for transcripts with sequencing depth between 5 and 10. For transcripts with higher sequencing depth, in general, IDBA-Tran also performed better (even though all assemblers had better performance for high-expressed transcripts).

For IDBA-Tran, we verified the effectiveness of its component separation algorithm and show the distribution of transcripts in Table 3. If a component contained a certain portion (80%) of a transcript, the component was deemed to have contained this transcript. There were 2865 components containing no transcripts and 10,611 components, each of which contained at most 5 transcripts, together containing most of the transcripts (16,663). Since very low-expressed transcripts cannot be
assembled by any assemblers, the total number of transcripts (18 180) in all components was less than the total number of expressed transcripts (22 402). A component containing a transcript does not guarantee that the transcript can be reconstructed. Thus, the number of reconstructed transcripts (17 243) is less than the total number of transcripts in all components (18 180). However, experiments showed that most transcripts decomposed correctly can be reconstructed successfully.

Table 1. Statistics of assembly result of each assembler for simulated data set (completeness = 0.8)

<table>
<thead>
<tr>
<th>Assembler</th>
<th>Contigs number</th>
<th>Average length (nt)</th>
<th>Total length (nt)</th>
<th>Reconstructed transcripts number</th>
<th>Correct contigs number</th>
<th>Sensitivity</th>
<th>Specificity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trinity</td>
<td>26 189</td>
<td>1941</td>
<td>41M</td>
<td>14 910</td>
<td>14 389</td>
<td>66.56%</td>
<td>54.94%</td>
</tr>
<tr>
<td>Oases</td>
<td>22 804</td>
<td>1963</td>
<td>39M</td>
<td>14 420</td>
<td>14 712</td>
<td>64.37%</td>
<td>64.51%</td>
</tr>
<tr>
<td>IDBA-UD</td>
<td>18 020</td>
<td>1322</td>
<td>24M</td>
<td>10 941</td>
<td>8 406</td>
<td>48.58%</td>
<td>46.65%</td>
</tr>
<tr>
<td>Velvet-SC</td>
<td>22 868</td>
<td>613</td>
<td>14M</td>
<td>389</td>
<td>357</td>
<td>1.74%</td>
<td>1.56%</td>
</tr>
<tr>
<td>IDBA-Tran</td>
<td>22 708</td>
<td>1933</td>
<td>39M</td>
<td>17 242</td>
<td>16 707</td>
<td>76.98%</td>
<td>73.57%</td>
</tr>
</tbody>
</table>

Table 2. Expression level distribution of reconstructed transcripts of each assembler for simulated data set (completeness = 0.8)

<table>
<thead>
<tr>
<th>Depth</th>
<th>0, 5</th>
<th>5, 10</th>
<th>10, 15</th>
<th>15, 20</th>
<th>≥20</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trinity</td>
<td>5943</td>
<td>5011</td>
<td>2943</td>
<td>1857</td>
<td>6646</td>
</tr>
<tr>
<td>Oases</td>
<td>1955</td>
<td>3251</td>
<td>2393</td>
<td>1527</td>
<td>5782</td>
</tr>
<tr>
<td>IDBA-UD</td>
<td>1648</td>
<td>3224</td>
<td>2461</td>
<td>1606</td>
<td>5481</td>
</tr>
<tr>
<td>Velvet-SC</td>
<td>1629</td>
<td>2563</td>
<td>1753</td>
<td>1107</td>
<td>3831</td>
</tr>
<tr>
<td>IDBA-Tran (unbroken transcripts after decomposition)</td>
<td>2619 (2700)</td>
<td>4177 (4337)</td>
<td>2746 (2824)</td>
<td>1723 (1811)</td>
<td>5977 (6505)</td>
</tr>
</tbody>
</table>

Table 3. Distribution of transcripts in IDBA-Tran components for simulated data set (completeness = 0.8)

<table>
<thead>
<tr>
<th>Transcripts in component</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>≥10</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of components</td>
<td>2865</td>
<td>6722</td>
<td>2407</td>
<td>954</td>
<td>370</td>
<td>158</td>
<td>71</td>
<td>28</td>
<td>24</td>
<td>4</td>
<td>22</td>
<td>13 625</td>
</tr>
<tr>
<td>Number of unbroken transcripts (after decomposition)</td>
<td>0</td>
<td>6720</td>
<td>4814</td>
<td>2859</td>
<td>1480</td>
<td>790</td>
<td>426</td>
<td>196</td>
<td>192</td>
<td>37</td>
<td>666</td>
<td>18 180</td>
</tr>
<tr>
<td>Number of reconstructed transcripts</td>
<td>0</td>
<td>6676</td>
<td>4682</td>
<td>2672</td>
<td>1324</td>
<td>667</td>
<td>349</td>
<td>164</td>
<td>141</td>
<td>33</td>
<td>535</td>
<td>17 243</td>
</tr>
</tbody>
</table>

3.2 Real data
We verified IDBA-Tran and other assemblers on the real RNA-Seq data of *Oryza sativa* transcriptome. There were 24 855 142 paired-end length-90 reads in the data set. The insert distance was about 200. Previous simulated data used the expression level profile of this data set, so they had the same set of expressed transcripts and expression levels. The distribution of expressed transcripts is also included in Table 6 for comparison (with the distribution of expressed transcripts estimated as mentioned in Section 3.1). Note that since the expressed transcripts and expression levels were estimated from alignments, there may be some error due to the existence of unknown transcripts and transcripts with over 80% similarity. The sensitivities and specificities shown are approximation of the real sensitivities and specificities only.

Figure 5 shows the number of reconstructed transcripts and aligned (correct) contigs reported by each assembler under different completeness. The results were consistent with those for the simulated data. IDBA-Tran still performed the best for all levels of completeness. All assemblers had poorer performance for real data than for simulated data. Oases still had its best performance when k was set to 25, and had very similar performance in terms of reconstructed transcripts compared with Trinity in all completeness settings.
Detailed statistics of assembly results are shown in Table 5 when completeness is set to 0.8. IDBA-Tran, Oases and Trinity had about the same number of contigs and total contig bases. IDBA-UD assembled relatively fewer contigs than others. IDBA-Tran had the highest sensitivity (42.08%) and specificity (22.94%) while other assemblers had much lower sensitivity and specificity.

Table 6 shows the expression level distribution of reconstructed transcripts for different assemblers when completeness is set to 0.8. When comparing Tables 2 and 6, it is clear that the real data was more difficult to assemble than simulated data, especially for low-expressed transcripts. Only 732 transcripts (25%) with depth 5 were reconstructed by IDBA-Tran and worse for other assemblers. Similar to simulated data, IDBA-
Tran had better performance than other assemblers for all expression levels.

Table 7 shows the distribution of transcripts in components. Since the sampling depths of a single transcript may be uneven, many transcripts were broken into fragments. Moreover, there were some unknown transcripts in the data set. So, quite a number of components did not contain 80% of a transcript. However, for the other components, IDBA-Tran did a good job: 6353 components, each of which contained at most 5 transcripts, contained 9082 transcripts all together. Similar to simulated data, once a transcript was correctly assigned to a component, the transcript was reconstructed with high probability.

### 4 DISCUSSION

We have identified one key issue in transcriptome assembly, namely how to remove erroneous vertices/edges of high multiplicity (due to high-expressed isoforms) from the de Bruijn graph while keeping correct ones with relatively lower multiplicity (due to low-expressed isoforms). We developed a probabilistic progressive approach with local thresholds to solve the problem. We proposed IDBA-Tran, combined with other techniques, to assemble transcriptome sequencing data. Experiments on both simulated and real data confirm that IDBA-Tran can outperform existing de novo transcriptome assemblers in terms of both sensitivity and specificity. In particular, for low-expressed transcripts, the improvement of IDBA-Tran is substantial.

Recall that there is another approach to recover both low-expressed and high-expressed transcripts, namely: run the assembler for different k values and merge all contigs as output. Oases-M, which runs Oases several times with multiple k values, is a post-processing tool based on this approach. Oases-M can reconstruct many transcripts for both simulated and real data. However, since erroneous contigs cannot be merged, Oases-M produces many incorrect contigs and has a low specificity (see Supplementary Table SA8). Moreover, contigs representing some transcripts may appear multiple times (with small difference) in the output such that the number of correct contigs is double the number of reconstructed transcripts. The large number of erroneous contigs and redundant contigs may make analysis difficult, and it is very hard to distinguish the erroneous contigs from the correct ones. On the other hand, we found that Oases-M had slightly better performance than IDBA-Tran for high-expressed transcripts for real data (see Supplementary Table SA9).

### Funding

This research is partially supported by RGC HKU 7111/12E and HKU 71909E, the Shanghai Pujiang Plan (Y057C11501) and Bill & Melinda Gates Foundation Project (“C4 Rice”).

### Conflict of Interest

none declared.

### REFERENCES


