# Reconstructing Isoform Graphs from RNA-Seq data

Stefano Beretta, Paola Bonizzoni, Raffaella Rizzi
DISCo
Univ. Milano-Bicocca
Milan, Italy
Email: {beretta,bonizzoni,rizzi}@disco.unimib.it

Gianluca Della Vedova
Dip. Statistica
Univ. Milano-Bicocca
Milan, Italy
Email: gianluca.dellavedova@unimib.it

***Abstract*—Next-generation sequencing (NGS) technologies allow new methodologies for alternative splicing (AS) analysis. Current computational methods for AS from NGS data are mainly focused on predicting splice site junctions or de novo assembly of full-length transcripts. These methods are computationally expensive and produce a huge number of full-length transcripts or splice junctions, spanning the whole genome of organisms. Thus summarizing such data into the different gene structures and AS events of the expressed genes is an hard task.**

**To face this issue in this paper we investigate the computational problem of reconstructing from NGS data, in absence of the genome, a gene structure for each gene that is represented by the *isoform graph*: we introduce such graph and we show that it uniquely summarizes the gene transcripts. We define the computational problem of reconstructing the isoform graph and provide some conditions that must be met to allow such reconstruction. Finally, we describe an efficient algorithmic approach to solve this problem, validating our approach with both a theoretical and an experimental analysis.**

***Index Terms*—alternative splicing.**

## I. Introduction

Next-generation sequencing (NGS) technologies allow massive and parallel sequencing of biological molecules (DNA and RNA), and have a huge impact on molecular biology and bioinformatics [1]. In particular, RNA-Seq is a recent technique to sequence expressed transcripts, characterizing both the type and the quantity of transcripts expressed in a cell (its transcriptome). Challenging tasks of transcriptome analysis via RNA-Seq data analysis [2]–[4] are reconstructing full-length transcripts (or isoforms) of genes and their expression levels. The annotation of alternative splicing variants and AS events, to differentiate and compare organisms, is part of the central goal in transcriptome analysis of identifying and quantifying all full-length transcripts. However, the extraction of splicing variants or significant AS events from the different transcripts produced by a set of genes requires to compare hundreds of thousands of full-length transcripts. Graph representations of splice variants, such as splicing graphs [5], have emerged as a convenient approach to summarize several transcripts from a gene into the putative gene structure they support. The current notions of splicing graphs rely on some sort of gene annotations, such as the annotation of full-length transcripts by their constitutive exons on the genome.

In this paper, we first define the notion of *isoform graph* which is a gene structure representation of genome annotated full-length transcripts of a gene. The isoform graph is a variant of the notion of splicing graph that has been originally introduced in [6] in a slightly different setting. When using only RNA-Seqs, the genome annotation cannot be given, and thus it is quite natural to investigate and characterize the notion of splicing graph which naturally arises when a reference genome is not known or available. Thus, in the paper we focus on the following main question: *under which conditions the reconstruction of a gene structure can be efficiently accomplished using only information provided by RNA-Seq data?*

In order to face this problem, we give some necessary or sufficient conditions to infer the isoform graph, we introduce an optimization problem that guides towards finding a good approximation of the isoform graph and finally we describe an efficient heuristic for our problem on data violating the conditions necessary to exactly infer the isoform graph.

The novelty of our approach relies on the fact that it allows the reconstruction of the splicing graph in absence of the genome, and thus it is applicable also to highly fragmented or altered data, as found in cancer genomes. Moreover we focus on methods that can effectively be used for a genome-wide analysis on a standard workstation. Our paper is not focused on full-length transcripts reconstruction [7], such as Cufflinks [2], and Scripture [8] or de novo assembly methods that build a de Brujin graph such as TransAbyss [9], Velvet [10], and Trinity [11]. These tools are computationally expensive and are able to find only the majority of the annotated isoforms while providing a large amount of non annotated full-length transcripts that would need to be experimentally validated.

In the paper, we aim to advance towards the understanding of the possibilities and limitations of computing the distinct gene structures from which genome-wide RNA-Seq or short reads data have been extracted. In this sense our approach aims to keep separate gene structures in the reconstruction from genome wide RNA-Seq even in absence of a reference.

We describe a simple and efficient algorithm that is guaranteed to reconstruct the isoform graph under some conditions. Moreover we give an improved algorithm to handle instances where the aforementioned conditions do not hold due to, for example, errors in the reads or lower coverage that typically affect real data. We show the scalability of our implementation to huge quantity of data. In fact limiting the computational resources used by our algorithm, when compared to other tools of transcriptome analysis, is a main aim of ours. Our

978-1-4673-2560-8/12/$31.00 ©2012 IEEE

algorithmic approach works in time that is linear in the number of reads, while having space requirements bounded by the size of hashing tables used to memorize reads. Moreover, we are able to show that our method is able to distinguish among different gene structures though processing a set of reads from various genes, limiting the process of fusion of graphs structures from distinct genes due to the presence of repeated sequences.

## II. The Isoform Graph and the SGR Problem

A conceptual tool that has been used to investigate the reconstruction of full-length transcripts from ESTs (Expressed Sequence Tags) [5] or RNA-Seq data is the *splicing graph*. Since our main goal is the reconstruction of the splicing graph from the nucleotide sequences of a set of short reads without the knowledge of a genomic sequence, some definitions will be slightly different from [6] where the notion of abundance of reads spanning some splicing junction sites is central. Informally, a splicing graph is the graph representation of a gene structure, inferred from a set of RNA-Seq data, where isoforms correspond to paths of the splicing graphs, while splicing events correspond to specific subgraphs.

Let $s = s_1 s_2 \cdots s_{|s|}$ be a sequence of characters, that is a *string*. Then $s[i : j]$ denotes the substring $s_i s_{i+1} \cdots s_j$ of $s$, while $s[: i]$ and $s[j :]$ denote respectively the *prefix* of $s$ consisting of $i$ symbols and the *suffix* of $s$ starting with the $j$-th symbol of $s$. We denote with $\mathrm{pre}(s, i)$ and $\mathrm{suf}(s, i)$ respectively the prefix and the suffix of length $i$ of $s$. Among all prefixes and suffixes, we are especially interested into $\mathrm{LH}(s) = \mathrm{pre}(s, |s|/2)$ and $\mathrm{RH}(s) = \mathrm{suf}(s, |s|/2)$. Given two strings $s_1$ and $s_2$, the *overlap* $ov(s_1, s_2)$ is the length of the longest suffix of $s_1$ that is also a prefix of $s_2$. The *fusion* of $s_1$ and $s_2$, denoted by $\varphi(s_1, s_2)$, is the string $s_1[: |s_1| - ov(s_1, s_2)]s_2$ obtained by concatenating $s_1$ and $s_2$ after removing from $s_1$ its longest suffix that is also a prefix of $s_2$. We extend the notion of fusion to a sequence of strings $\langle s_1, \ldots, s_k \rangle$ as $\varphi(\langle s_1, \ldots, s_k \rangle) = \varphi(s_1, \varphi(\langle s_2, \ldots, s_k \rangle))$ if $k > 2$, and $\varphi(\langle s_1, s_2 \rangle) = \varphi(s_1, s_2)$. Alternative splicing regulates how different coding regions are included to produce different full-length isoforms or transcripts, which are modeled here as sequences of *blocks*. Formally, a *block* consists of a string, typically taken over the alphabet $\Sigma = \{a, c, g, t\}$, and an integer called rank, such that no two blocks share the same rank. Given a block $b$, we denote by $s(b)$ and $r(b)$ its string and rank respectively. In our framework a *gene coding region* is a sequence (that is, an ordered set) $B = \langle b_1, b_2, \cdots, b_n \rangle$ of blocks with $r(b_i) = i$ for each $i$. Then, the *string coding region for* $B$ is the string $s(b_1) \cdots s(b_n)$ obtained by orderly concatenating the strings of the blocks in $B$. We define a *block isoform* $f$ *compatible* with $B$, as a subsequence of $B$, that is $f = \langle b_{i_1}, \cdots, b_{i_k} \rangle$ where $i_j < i_{j+1}$. We distinguish between classical isoforms (defined on exons or genomic regions) and block isoforms (defined on blocks). Nonetheless, we will use interchangeably the terms isoforms and block isoforms whenever no ambiguity arises. By a slight abuse of language we define the string of $f$, denoted by $s(f)$, as the concatenation of the strings of the blocks of $f$, while an *expressed gene* is a pair $\langle B, F \rangle$, where $B$ is a gene coding region, $F$ is a set of block isoforms compatible with $B$ where (1) each block of $B$ appears in some isoform of $F$, and (2) for each pair $(b_i, b_j)$ of blocks of $B$, appearing consecutively in some isoform of $F$, there exists a isoform $f \in F$ s.t. exactly one of $b_i$ or $b_j$ appears in $f$. Observe that our definition of expressed gene implies that the set $B$ of blocks of a string coding region of an expressed region $\langle B, F \rangle$ is unique and is a minimum-cardinality set explaining all isoforms in $F$. Thus, the pair $\langle B, F \rangle$ describes a specific gene. The uniqueness of blocks of an expressed gene allows us to define the associated graph representation, or isoform graph. Given an expressed gene $\mathcal{G} = \langle B, F \rangle$, the *isoform graph* of $\mathcal{G}$ is a directed graph $G_I = \langle B, E \rangle$, where $\langle b_i, b_j \rangle$ is an arc of $E$ iff $b_i$ and $b_j$ are consecutive in some isoform of $F$. Notice that $G_I$ is a directed acyclic graph, since the sequence $B$ is also a topological order of $G_I$. Notice that isoforms correspond to paths in $G_I$.

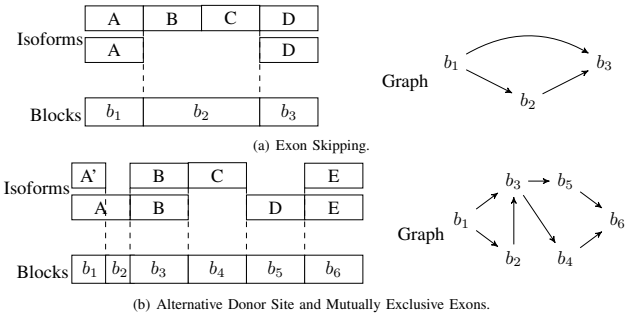


Fig. 1. Examples of Isoform Graphs. Capital letters correspond to exons. In (a) is represented a skipping of the two consecutive exons $B$ and $C$ of the second isoform w.r.t. the first one. In (b) is represented an alternative donor site variant between exons $A$ and $A'$ and two mutually exclusive exons $C$ and $D$. Notice that, in this figure, the isoforms represented are classical.

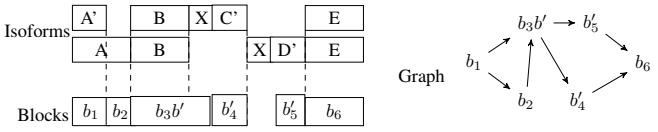


Fig. 2. Alternative splicing graph compatible with the reads of the expressed gene of Fig. 1(b)

Our first aim of the paper is to characterize when and how accurately the isoform graph of an expressed gene can be reconstructed from a set of substrings (i.e. RNA-Seq data) of the isoforms of the gene. More precisely, we want to investigate the following two main questions. *Question 1:* What are the conditions under which the isoform graph of a gene can be reconstructed from a sample of RNA-Seqs (without putting any bounds on the computational resources)? *Question 2:* Can we build efficiently such a graph or an approximation of it?

Notice that the isoform graph is the real gene structure that we would like to infer from data but, at the same time, we must understand that the transcript data might not be sufficient to determine the isoform graph, as we have no information on the genomic sequence and on the blocks in particular. Therefore

we aim at computing a slightly less informative kind of graph: the *splicing graph*, which is a directed graph where each vertex $v$ is labeled by a string $s(v)$.

Since we do not know the block position or rank on the genomic sequence, we introduce the notion of splicing graph *compatible* with a set of isoforms. Let $\langle B, F\rangle$ be an expressed gene, and let $G = \langle V, E\rangle$ be a splicing graph. Then $G$ is *compatible* with $F$ if, for each isoform $f \in F$, there exists a path $p = \langle w_1, \ldots, w_k\rangle$ of $G$ such that $s(f) = s(w_1) \cdots s(w_k)$.

We need some more definitions, as we investigate the problem of reconstructing a splicing graph compatible with RNA-Seqs obtained from the gene transcripts. Let $\langle B, F\rangle$ be an unknown expressed gene. Then, a *RNA-Seq read* (or simply *read*) extracted from $\langle B, F\rangle$, is a substring of the string $s(f)$ of some isoform $f \in F$. We need to introduce some criteria to rank all splicing graphs compatible with $R$. The parsimonious principle leads us to a natural objective function: to minimize the sum of the lengths of strings associated to the vertices (mimicking the search for the shortest string coding region).

## III. Methods

Here we propose a method for solving the SGR problem. Our approach to compute the isoform graph $G_S$ first identifies the vertex set $B_S$ and then the edge set $E_S$ of $G_S$. Moreover we focus on fast and simple methods that can possibly scale to genome-wide data. For ease of exposition, the discussion of the method assumes that reads have no errors, and $l = 64$. The basic idea of our method is that we can find two disjoint subsets $R_1$, and $R_2$ of the input set $R$ of reads, where reads of $R_1$, called *unspliced*, can be assembled to form the nodes in $B_S$, while reads of $R_2$, called *spliced*, are an evidence of a junction between two blocks (that is an arc of $G_S$).

The second main tenet of our algorithm is that each read is encoded by a 128-bit binary number, divided into a *left fingerprint* and a *right fingerprint* (respectively the leftmost and the rightmost 64 bits of the encoding). Then two reads $r_1$ and $r_2$ overlap for at least $l/2 = 32$ base pairs iff the right fingerprint of $r_1$ is a substring of the encoding of $r_2$ (a similar condition holds for the left fingerprint of $r_2$). Bit-level operations allows to look for such a substring very quickly. Let $r$ be a read of $R$. Then $r$ is *spliced* if there exists another $r' \in R$, $s(r) \neq s(r')$, such that $\mathrm{pre}(r, k) = \mathrm{pre}(r', k)$ or $\mathrm{suf}(r, k) = \mathrm{suf}(r', k)$, for $l/2 \leq k$. Moreover $r$ is *perfectly spliced* if there exists another $r' \in R$, $s(r) \neq s(r')$, such that the longest common prefix (or suffix) of $r$ and $r'$ is exactly of length $l/2$. A read that is not spliced is called *unspliced*.

In our framework, a junction site between two blocks $b_1$ and $b_3$, that appear consecutively within an isoform, is detected when we find a third block $b_2$ that, in some isoform, appears immediately after $b_1$ or immediately before $b_3$. For illustrative purposes, let us consider the case when $b_2$ appears immediately after $b_1$ in an isoform (Figure 1(a)). The strongest possible signal of such junction consists of two reads $r_1$ and $r_2$ such that $ov(s(b_1), r_1) = ov(r_1, s(b_3) = l/2$ and $ov(s(b_2), r_2) = ov(r_2, s(b_3)) = l/2$, that is $r_1$ is cut into halves by the junction separating $b_1$ and $b_3$, while $r_2$ is cut into halves by the junction separating $b_2$ and $b_3$ (i.e. $r_1$ and $r_2$ are perfectly spliced). In a less-than-ideal scenario, we still can find two reads $r_1$ and $r_2$ sharing a common prefix (or suffix) that is longer than $l/2$, in which case the two reads are spliced.

Notice that all reads extracted from the same block can be sorted so that any two consecutive reads have large overlap. We define a *chain* as a sequence $C = \langle r_1, \cdots, r_n\rangle$ of unspliced reads with $ov(r_i, r_{i+1}) = l/2$ for $1 \leq i < n$ (notice that the $\mathrm{RH}(r_i) = \mathrm{LH}(r_{i+1})$, which allows for a very fast computation). Given a chain $C$, the string of $C$ is the string $s(C) = \varphi(C)$, moreover $C$ is *maximal* if no supersequence of $C$ is also a chain. Under ideal conditions $s(C)$ is exactly the string of a block. Coherently with our reasoning, a perfectly spliced read $r$ is called a *link* for the pair of chains $(C, C')$, if $\mathrm{LH}(r) = \mathrm{suf}(s(C), l/2)$ and $\mathrm{RH}(r) = \mathrm{pre}(s(C'), l/2)$. Given a set $R$ of reads extracted from the isoforms $F$ of an unknown expressed region $\langle B, F\rangle$, we compute a likely isoform graph $G_R = (\mathcal{C}, E_R)$, where $\mathcal{C} = \{C_1, \cdots, C_n\}$ is a set of maximal chains that can be derived from $R$, and $(C_i, C_j) \in E_R$ iff there exists in $R$ a link for $(C_i, C_j)$.

A main goal of our algorithm is to show how we compute such graph efficiently even under less-than-ideal conditions. The algorithm is organized into three steps. In the first step we build a data structure to store the reads in $R$. We use two hash tables which guarantee a fast access to the input reads. The second step creates the nodes of $G_R$ by constructing and merging the maximal chains of the unspliced reads of $R$. In the last step of the creation of $G_R$, the maximal chains obtained in the second step are linked using the perfectly spliced reads.

A practical concern is to deal with errors and insufficient coverage. An effect of the chain merging phase is that most errors are corrected and suboptimal coverage is not too relevant. In fact the typical effect of a single-base error in a read $r$ is the misclassification of $r$ as spliced instead of unspliced, shortening or splitting some chains. Anyway, as long as there are only a few errors, there exist some overlapping error-free unspliced reads that span the same block as the erroneous read. Those unspliced reads allow for the correction of the error and the construction of the correct chain spanning the block. We also point out that our method is able to cope with repeated sequences that are shorter than $l/2$ bases, which are usually a weakness of methods based on de Bruijn graphs.

## IV. Experimental Results

We have run our experimental analysis on simulated error-free RNA-Seq data obtained from the transcript isoforms annotated for a subset of 112 genes extracted from the 13 ENCODE regions used as training set in the EGASP competition [12]. We have chosen those genes because they are well annotated and, at the same time, are considered quite hard to be analyzed by tools aiming to compute a gene structure, mainly due to the presence of repeated regions. Moreover, the presence of similar genes makes this set very hard to be analyzed as a whole. Also, we decided to focus on a relatively small number of (hard to analyze) genes so that we could manually inspect the results to determine the causes of incorrect predictions.

A primary goal of our implementation is to use only a limited amount of memory, since this is the main problem with currently available programs [11]. Even on the largest dataset, our program has never used more 30 seconds or more than 250MB of memory. Our implementation is available under AGPLv3 at http://algolab.github.com/RNA-Seq-Graph/.

Now let us detail the experiments. For each of the 112 ENCODE genes, the set of the annotated full-length transcripts has been retrieved from NCBI GenBank. From those transcripts we have extracted two sets of 64bp substrings corresponding to our simulated reads. The first set consists of all possibile 64-mers and corresponds to the maximum possible coverage (*full coverage* dataset), while the second set consists of a random set of 64-mers simulating an 8x coverage (*low coverage* dataset). In the first experiment we have analyzed the behavior on the full coverage dataset where data originating from each gene have been elaborated separately and independently. The second experiment is identical to the first, but on the low coverage dataset. Finally, the third experiment has been run on the whole full coverage dataset, that is all reads have been elaborated together and without any indication of the gene they were originating from. The whole full coverage dataset consists of 1.4 million unique 64bp reads. For each input gene, the true isoform graph has been reconstructed from the annotation.

In all experiments, the accuracy of our method is evaluated by two standard measures, *sensitivity* (Sn) and *positive predictive value* (PPV) considered at vertex and arc level. Sensitivity is defined as the proportion of vertices (or arcs) of the isoform graph that have been correctly predicted by a vertex (or arc) of the computed splicing graph, while PPV is the proportion of the vertices (or arcs) of the splicing graph that correctly predict a vertex (or an arc) of the isoform graph.

In the first experiment, our implementation has perfectly reconstructed the isoform graph of 43 genes (out of 112), that is Sn and PPV are 1 both at vertex and arc level. Moreover we have obtained average Sn and PPV values that are 0.86 and 0.92 at vertex level, respectively, and 0.72 and 0.82 at arc level, respectively. The median values of Sn and PPV are 0.91 and 1 at vertex level, 0.86 and 0.98 at arc level, respectively.

The second experiment (separated gene, 8x coverage) is to study our method under a more realistic coverage. In this case, we have perfectly reconstructed the isoform graph of 39 genes (out of 112), and we have obtained average Sn and PPV values that are respectively 0.87, 0.91 at vertex level and 0.75, 0.81 at arc level. The median values of Sn and PPV are 0.93 and 1 at vertex level, 0.84 and 0.91 at arc level, respectively. In the third experiment the expected output is a large isoform graph $G_I$ with 1521 vertices and 1966 arcs, with a 1-to-1 mapping between connected components and input genes. Only 7 connected component have been mapped to more than one gene – 4 of them are genes with very similar sequence composition (i.e. CTAG1A, CTAG1B and CTAG2). Overall results are quite similar to those of the first experiment, as the number of correctly identified vertices goes from 1303 (first experiment) to 1274 (third experiment). Similarly, the number of correctly identified arcs goes from 1415 to 1396 – the quality of our predictions is only barely influenced by the fact that the program is run on data coming from 112 genes. The overall vertex Sn is 0.837, the vertex PPV is 0.778, while the arc Sn is 0.71 and the arc PPV is 0.679.

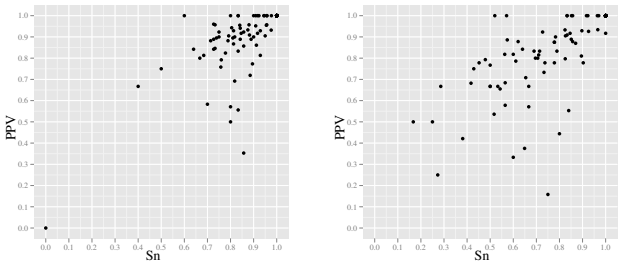


Fig. 3. Second experiment: Sn and PPV values at vertex level (on the left) and at arc level (on the right). Each point represents a gene.

The final part of our analysis is a comparison with Trinity [11] on the two full coverage datasets, i.e. first and third experiments. Since Trinity computes transcripts and not the splicing graph, we use the variation of number of predicted full-lengths transcripts as a proxy for the (in)stability of the method, as we would like to predict the same transcripts in both cases. We observed that, for the two datasets, Trinity has predicted 2689 and 1694 full-length transcripts, a variation much larger than that of our method.

## REFERENCES


[1] M. L. Metzker, "Sequencing technologies - the next generation." *Nature reviews. Genetics*, vol. 11, no. 1, pp. 31–46, 2010.

[2] C. Trapnell, B. A. Williams, G. Pertea, A. Mortazavi, et al. "Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation," *Nature Biotechnology*, vol. 28, no. 5, pp. 516–520, 2010.

[3] M. Nicolae, S. Mangul, I. I. Mandoiu, and A. Zelikovsky, "Estimation of alternative splicing isoform frequencies from RNA-Seq data." *Algorithms for molecular biology : AMB*, vol. 6, no. 1, p. 9, 2011.

[4] J. Feng, W. Li, and T. Jiang, "Inference of isoforms from short sequence reads," *J. Comp. Biol.*, vol. 18, no. 3, pp. 305–321, 2011.

[5] S. Heber, M. A. Alekseyev, S.-H. Sze, H. Tang, and P. A. Pevzner, "Splicing graphs and EST assembly problem," in *Proc. 10th Int. Conf. on Intelligent Systems for Mol. Biology (ISMB)*, vol. 18, pp. 181–188, 2002.

[6] V. Lacroix, M. Sammeth, R. Guigó, and A. Bergeron, "Exact transcriptome reconstruction from short sequence reads," in *Proc. 8th Int. Workshop on Algorithms in Bioinformatics*. 2008, pp. 50–63.

[7] M. Garber, M. Grabherr, M. Guttman, and C. Trapnell, "Computational methods for transcriptome annotation and quantification using RNA-Seq," *Nature Methods*, vol. 8, no. 6, pp. 469–77, 2011.

[8] M. Guttman, G. Manuel, J. Levin, J. Donaghey, et al. "From Ab initio reconstruction of cell type–specific transcriptomes in mouse reveals the conserved multi-exonic structure of lincRNAs," *Nature Biotechnology*, vol. 28, pp. 503–510, 2010.

[9] G. Robertson, J. Schein, R. Chiu, et al. "De novo assembly and analysis of RNA-seq data" *Nature Methods*, vol. 7, no. 5, pp. 909–912, 2010.

[10] D. Zerbino and E. Birney, "Velvet: Algorithms for de novo short read assembly using de Bruijn graphs," *Genome Research*, vol. 18, pp. 821–829, 2008.

[11] M. Grabherr, B. Haas, M. Yassour, J. Levin, et al. "Full-length transcriptome assembly from RNA-Seq data without a reference genome," *Nature Biotechnology*, vol. 29, no. 7, pp. 644–652, 2011.

[12] R. Guigó, P. Flicek, J. Abril, A. Reymond, et al. "EGASP: the human ENCODE Genome Annotation Assessment Project." *Genome biology*, vol. 7, no. Suppl 1, pp. S2.1–31, 2006.