Alignment Algorithms For RNA-Seq

Asa Thibodeau
Outline

- Overview of RNA-Seq
- Mapping Difficulties
- TopHat Algorithm
- MapSplice Algorithm
- Performance Review By Grant et al. (2011)
RNA-Seq

- Uses deep sequencing technologies to obtain reads from a population of RNA
- Reads are between 30 to 400 bp
- Has a very low background signal compared to microarray
- Requires small amounts of RNA compared to the other technologies (microarray or cDNA/EST Sequencing) which require high amounts of RNA
- Has a relatively low cost for mapping transcriptomes of large genomes

Wang, Gerstein, and Snyder (2009)
RNA-Seq Process

5'UTR → Exon → Exon → Exon → Exon → 3'UTR

5' UTR → Translated Region (Exons) → 3' UTR Poly(A) mRNA

AAAAAA
RNA-Seq Process

1. Obtain Reads With High Throughput Sequencing

   - GGAATTACCACTCTGGATCA
   - GGACGTAACCGGGAC
   - TTAAGTACCTCGGAACTATGA

2. Fragmentation

3. Attach Adaptors

   - 5' UTR
   - Translated Region (Exons)
   - 3' UTR
   - Poly(A)
   - mRNA

   - RNA or cDNA
Mapping Difficulties: Alternative Splicing

In the previous example, all exons in the gene were used to produce the mRNA, but sometimes exons are excluded to have different isoforms of the mRNA:

Mapping reads now becomes more difficult due to the possible combinations of the exons within the gene. Additionally exons between two different genes could be merged together, meaning that the problem cannot simply be localized to just one gene.
Mapping Difficulties

- Mapping to just the transcriptome misses unknown transcribed regions
- Additionally portions of the intron region could also be included, making it harder to map the reads to just the transcriptome
- Using the entire reference genome makes it more difficult to deal with alternative splicing junctions
TopHat

• Trapnell, Pachter, and Salzberg (2009) developed the TopHat pipeline to handle challenges of mapping RNA-Seq reads such as alternative splicing

• Maps using the entire reference genome in order to locate novel transcript sites and novel splicing junctions while not relying on known splicing junctions

• The pipeline is able to map nearly 2.2 million reads per CPU hour, faster than most other systems
TopHat Pipeline

Map reads to whole genome with Bowtie

Collect initially unmappable reads

Assemble consensus of covered regions

Generate possible splices between neighboring exons

Build seed table index from unmappable reads

Map reads to possible splices via seed-and-extend
TopHat Pipeline: Bowtie

- TopHat uses Bowtie in order to initially map all reads to the reference genome while collecting all the unmapped reads for further analysis.

- Here Bowtie is reporting reads with no more than a few mismatches (default: 2) within $s$ bp from the 5' end while the 3' end may have errors based on Phred-Quality-Weighted Hamming Distance.

- TopHat allows reads from bowtie that map up to 10 locations while discarding those that map to more than 10 locations and not including them into the collection of unmapped reads.
TopHat Pipeline: MAQ

- With the mapped reads, TopHat uses MAQ to obtain islands of contiguous sequences using the assemble subcommand.
- TopHat uses the reference genome to have MAQ produce a consensus file containing called bases.
- As most reads at the ends of exons will include splice junctions, the ends of exons will be covered with fewer reads, resulting in missing sequences at each end.
- Since there may be the inclusion of flanking introns (donor/acceptor sites), a portion of the reference genome is added to each side of the consensus sequence (45 bp).
TopHat Pipeline: Gaps

• TopHat defines a parameter that is used for when two exons should be merged

• To be conservative, TopHat uses a distance of 6 bp by default when determining whether or not an exon should be merged
TopHat Pipeline: Splice Junctions

• To find splice junctions, TopHat does the following:

  1. Enumerates the canonical donor/acceptor sites within the island sequences
  2. Considers all pairings between GT-AG introns between neighboring islands
  3. Each intron is checked against the unmapped reads to find splice junctions

• By default TopHat only checks introns between 70-20,000 bp
TopHat Pipeline: Multiple Transcripts

In some cases two transcripts are present but TopHat only recognizes one island sequence.
TopHat Pipeline: Multiple Transcripts

- To find these junctions, TopHat looks for introns within islands that are deeply sequenced.

- For each island spanning $i$ to $j$, a score is computed with the following:

$$D_{ij} = \frac{\sum_{m=i}^{j} d_m}{j - i} \cdot \frac{1}{\sum_{m=0}^{n} d_m}$$

- $d_m$ is the depth of the coverage at coordinate $m$.

- $n$ is the length of the reference.

- TopHat looks in junctions where $D \geq 300$. 
TopHat: Unmapped Reads

- For every splicing junction found, TopHat searches the unmapped reads to find ones that span the junction using a seed-and-extend strategy
- Finds reads that span any splice junctions by $k$ bases on each side
- The unmapped reads are indexed using a lookup table to speed up the process and use $2k$-mer keys
- The table also contains $(s-2k+1)$ possible positions where a splice may be located where $s$ is the length of the high quality region near the 5' end
- Each splice junction makes a $2k$-mer seed by concatenating $k$ bases downstream from the acceptor and $k$ bases upstream from the donor
- The table is queried to find all reads that match the seed and aligns portions with the left and right corresponding islands
MapSplice

- Wang et al. (2010) developed the MapSplice algorithm which claims to have:
  - No dependence on splicing junction features
  - High specificity and sensitivity to junction discovery
  - The ability to be used on both long and short RNA-Seq reads
- The main idea of the algorithm is to further partition the reads as a part of a two phase (six step) process
MapSplice: Step 1

(1) Segmentation of reads
MapSplice: Step 1

- The first step partitions the reads of size $m$ into $n$ segments of length $k$ where $k \leq m/2$
- $k$ is typically 20-25 bp for reads greater than or equal to 50 bp
MapSplice: Step 2

(1) Segmentation of reads

(2) Segment exonic alignment
MapSplice: Step 2

- Exonic alignments are then performed using alignment programs such as Bowtie or BWA with the reference genome
- When more than one exonic alignment is found, all alignments are considered
MapSplice: Step 3

(2) Segment exonic alignment

Contiguous

Missed alignment double anchored

Missed alignment single anchored

(3) Segment spliced alignment
MapSplice: Step 3

- If $t_{i-1}$ and $t_{i+1}$ both have exonic alignments a double anchored splice alignment over all combinations of alignments between $t_{i-1}$ and $t_{i+1}$ are performed.

- If only one neighbor of $t_j$ has an exon alignment, then a single anchored splice alignment is performed with the possible alignments for $t_j$. 
MapSplice: Double-anchored

- Only need to consider the $k+1$ position of the splice junction formed by $t_{i-1}$ and $t_{i+1}$
- The optimal alignment has the minimum hamming distance:

$$\text{spliced-align}(t[1:k], G[i:j]) = \min \arg_{1 \leq x < k} D_H(t[1:x], G[i:i+x-1]) + D_H(t[x+1:k], G[j-(k-x)+1:j])$$

- Multiple alignments are kept if multiple alignments have the same minimum score
MapSplice: Single-anchored

- In the single-anchored splice alignment case, the algorithm searches for the suffix $s_i$ of $t_i$.
- The search window is upstream of $t_{i-1}$ by a value given by parameter D (usually 50,000 bp).
- All single-anchored splice alignments are resolved using a single traversal of the genome and a window of size D.
MapSplice: Small Exons

- For exons shorter than $2k$ it is possible for a read to overlap more than two exons.
- To solve this issue, divide into a set of $h$-mers by scanning the reference genome between the two anchor locations and perform two double-anchored spliced alignments if a match is found.
MapSplice: Step 4

(2) Segment exonic alignment

Contiguous

5' \[ t_j \quad t_{j+1} \quad t_{j+2} \quad \text{3'} \]

Missed alignment double anchored

? \[ t_{j+1} \]

\[ t_j \]

Missed alignment single anchored

? \[ t_{j+1} \]

(3) Segment spliced alignment

5' \[ t_j \quad ? \quad t_{j+1} \quad t_{j+2} \quad \text{3'} \]

(4) Segment assembly

5' \[ t_1 \quad t_2 \quad \ldots \quad t_j \quad t_{j+1} \quad \ldots \quad t_{n-1} \quad t_n \quad \text{3'} \]

T_1 \quad \ldots \quad \ldots \quad \ldots
MapSplice: Step 4

- Given that a segment may align to multiple locations, all combinations of the alignments must be searched for the best overall alignment.

- Two adjacent segments that are not contiguous are checked for a splice junction using the double-anchored approach.

- For each assembly that yields a candidate alignment, a mismatch score is computed using a modified hamming distance.

- A base quality score is used along with the mismatch score for bases x and y:

  $$ s(x,y) = \begin{cases} 
  (1 - p)/f_x, & x = y \\
  p/(1 - f_x), & x \neq y 
  \end{cases} $$

  where $f_x$ is of a base x in a background distribution of nucleotides and p is the probability that base x was called incorrectly.

- Given an alignment T and reference genome $G_T$ with probability $p_i$ that a nucleotide in T was called incorrectly the expected mismatch is:

  $$ E[mismatch(T,G_T)] = \sum_{j=1,m} s(b_j,g_{ij}) $$

  And kept if $E[mismatch(T,G_T)] \leq \epsilon_k$ having a quality of $\epsilon_k - E[mismatch(T,G_T)]$ where $\epsilon_k$ is the total number of allowed mismatches.
MapSplice: Step 5

(4) Segment assembly

(5) Junction inference

1. Alignment quality
2. Anchor significance
3. Entropy
MapSplice: Step 5

- Let $A(J)$ be the set of reads or tags that include a splice junction $J$
- Define two statistical measures:
  1. Anchor Significance: $s(A(J))$, scoring how long the anchors are for $J$ on each side
  2. Entropy: $h(A(J))$, measuring the diversity of the splice junction
- The anchor significance is of a junction $J$ is the greatest occurrence over all tags in $A(J)$ where the anchors are scored based on their minimum confidence with respect to the probability of not choosing the correct alignment anchor.
- A splice junction $J$ is expected to be uniformly distributed from $A(J)$ and so Shannon maximum entropy is applied where $p_i$ is the frequency of $J$ at position $i$:

  $$h(A(J)) = - \sum_{1 \leq i < m} p_i \log_2 p_i$$

- A combined score $p(J)$ derived from $s(A(J))$ and $h(A(J))$ to determine if a junction $J$ is true is obtained using Bayesian regression:

  $$p(J) = \alpha \cdot s(A(J)) + \beta \cdot h(A(J)) + \gamma \cdot q(A(J)) + \epsilon$$

where $\alpha$, $\beta$, and $\gamma$ are configuration parameters obtained by linear regression.
MapSplice: Step 6

(5) Junction inference

1. Alignment quality
2. Anchor significance
3. Entropy

(6) Identify best alignment for tags

OUTPUTS:
- Splices and splice coverage
- Tag alignments
MapSplice: Step 6

- For each tag, the alignment with the best score from combining the scores from steps 4 and 5 is chosen
Performance Review

- Grant et al. (2011) compared many different algorithms including TopHat and MapSlice using their own benchmark program.
- Required that the programs followed three criteria:
  1. Handling single reads across splice junctions
  2. Handling paired end reads
  3. Running in a reasonable amount of time
Accuracy Performance

**A**
Base Level Accuracy

- % of bases aligned correctly

**B**
Junctions Accuracy

- Junctions FP rate
- Junctions FN rate
Run Time Performance
Run Time Performance
References


