STAR : Ultrafast universal RNA-seq aligner

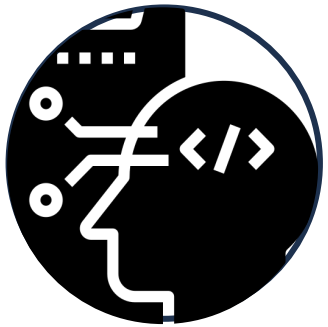
Suyeon Kim

2021-11-17

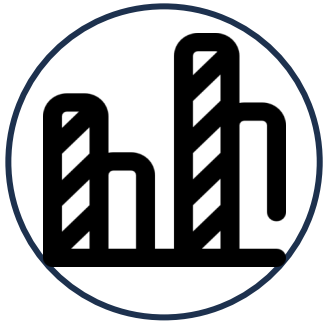
Advanced Bioinformatics 1

0 INDEX

1. RNA-Seq analysis



2. STAR Algorithm



3. Further analysis

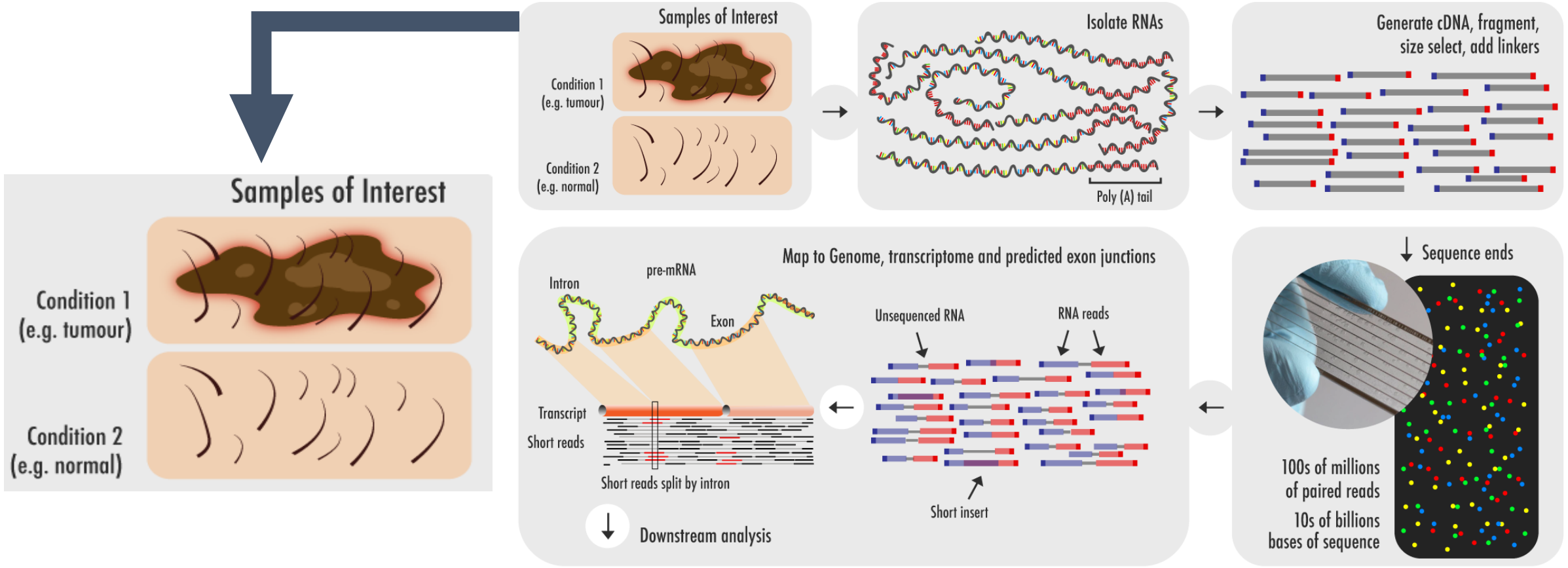
4. Google Colab

https://www.flaticon.com/

1. RNA-seq analysis

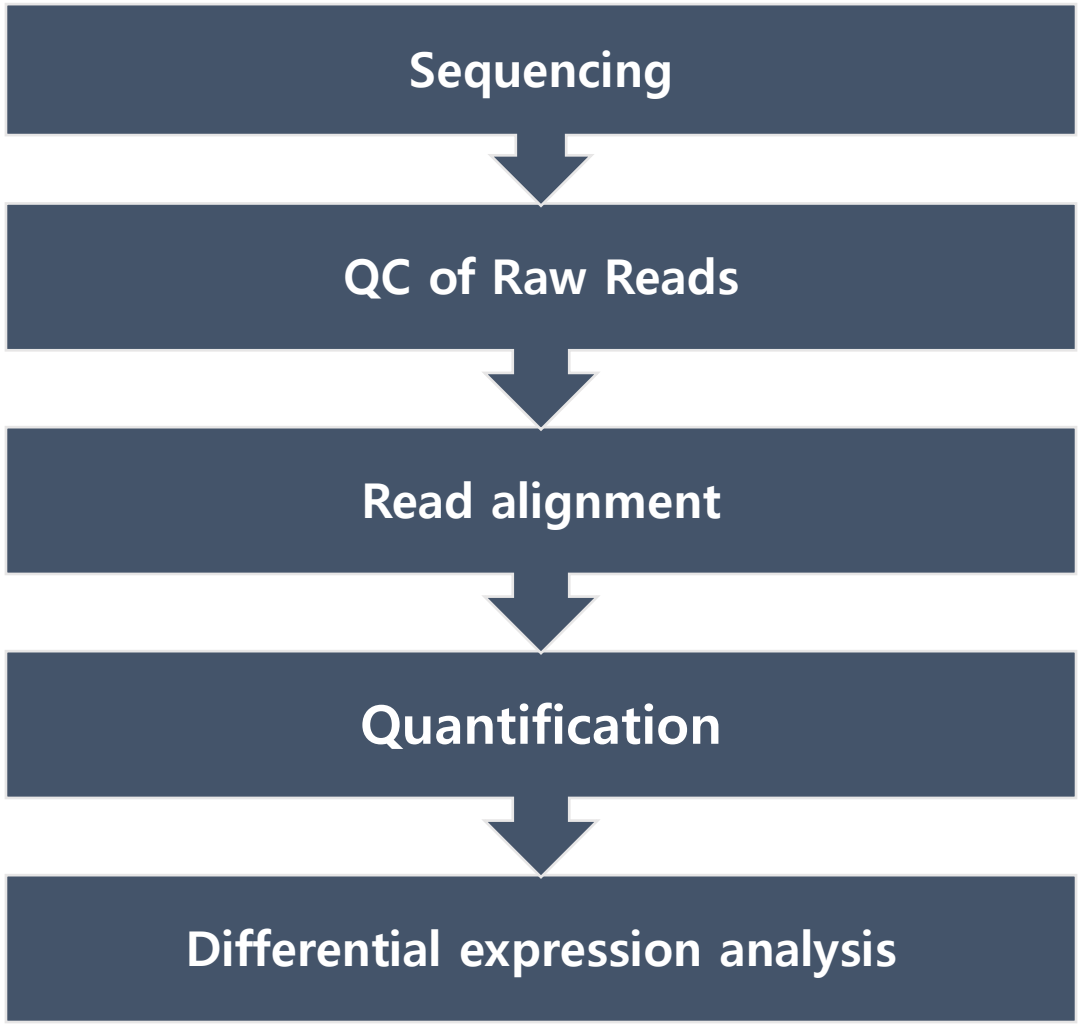
What is RNA-seq?

"RNA-seq is a tool that make us know gene expression."



https://www.frontiersin.org/articles/10.3389/fgene.2020.00220/full
https://www.technologynetworks.com/genomics/articles/rna-seq-basics-applications-and-protocol-299461

Workflow

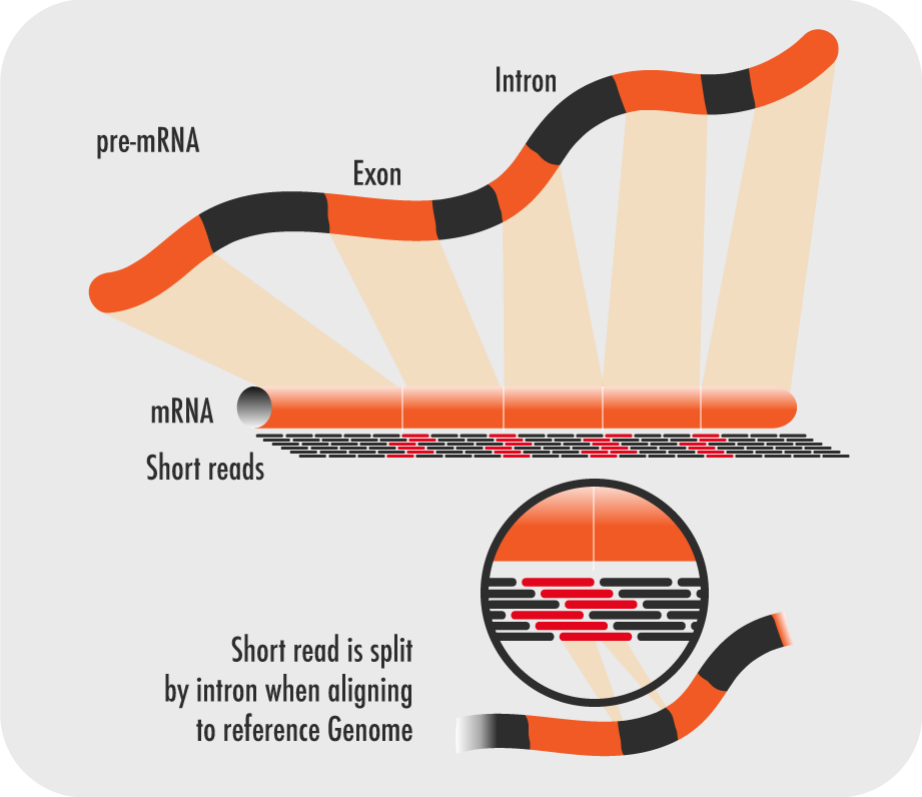


https://www.technologynetworks.com/genomics/articles/rna-seq-basics-applications-and-protocol-299461

Challenge

During RNA-seq alignment process

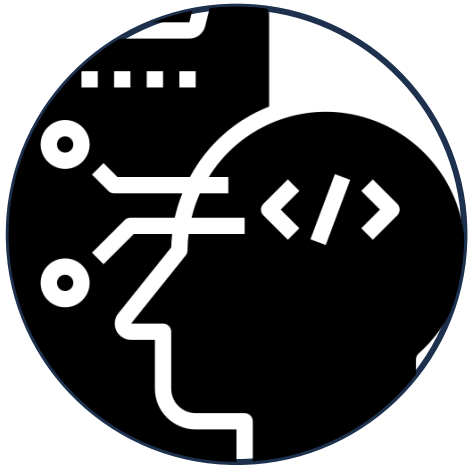
Two key tasks make the RNA-seq analysis computationally intensive.



① Mapping sequences from 'non-contiguous' genomic regions.

② Alignment of reads containing mismatches, indels caused by genomic variations and sequencing errors.

https://www.technologynetworks.com/genomics/articles/rna-seq-basics-applications-and-protocol-299461



2. STAR Algorithm

2 STAR Algorithm

What is STAR? And How?

- **STAR : Spliced Transcripts Alignment to a Reference.**
- **STAR algorithm consists of two major steps :**
  - **① Sequential seed searching**
  - **② Clustering/ Stitching/ Scoring**

2 STAR Algorithm

Comparisons

Of the way that deals with 'Non-contiguous structure'

The pre-existing RNA-seq aligners

① Align short reads to a DB of splice junctions.
② Align split-read portions contiguously to a reference genome.

- high mapping error rates, low mapping speed, and mapping biases

STAR

① Align the non-contiguous sequences directly to the reference genome.

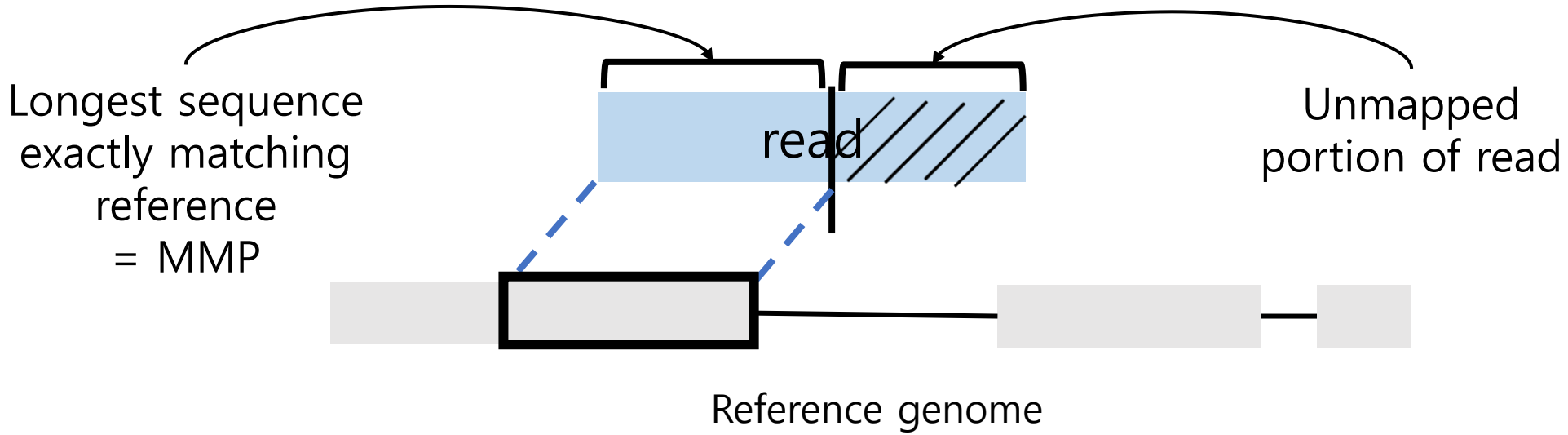
- improving sensitivity and precision, high mapping speed

2 STAR Algorithm

① Sequential seed search

For detecting splice junctions

- For every reads, STAR will search for the longest sequence that exactly matches one or more locations on the reference genome.
- These longest matching sequences are called the Maximal Mappable Prefixes (MMPs)



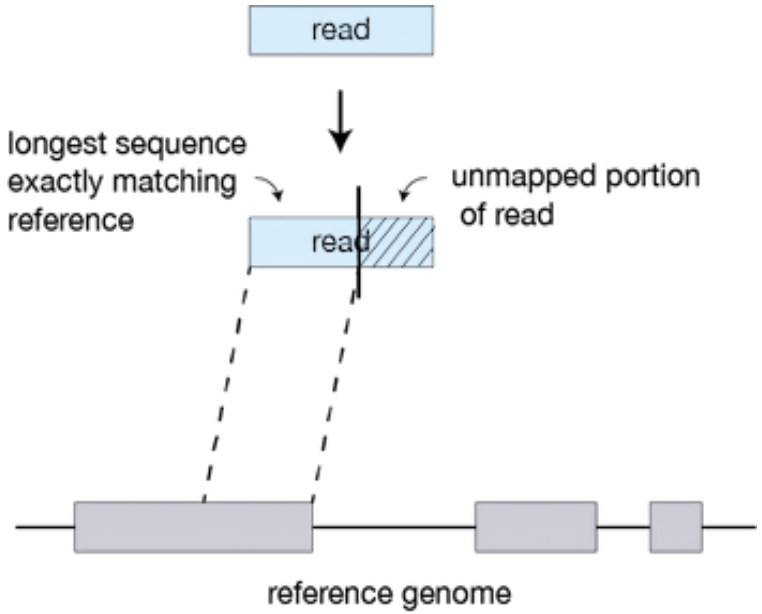
https://hbctraining.github.io/Intro-to-rnaseq-hpc-O2/lessons/03_alignment.html

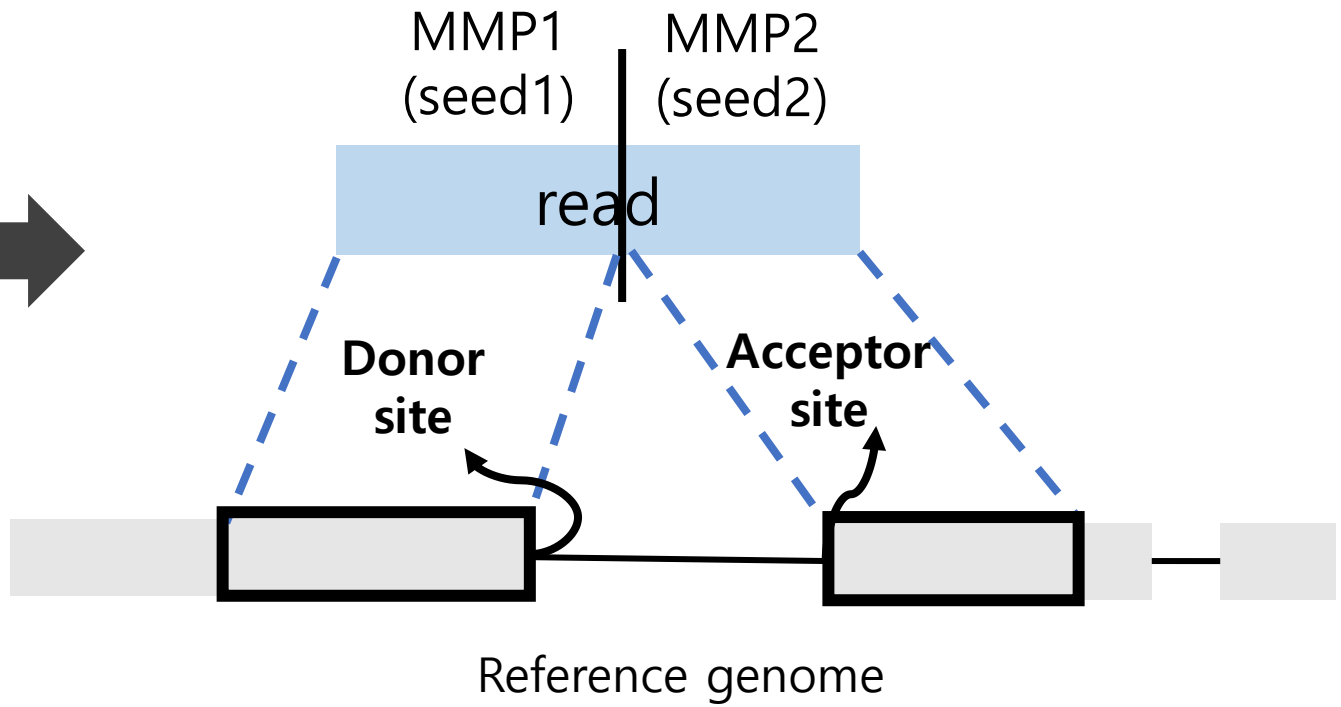
2 STAR Algorithm

① Sequential seed search

For detecting splice junctions

- STAR will then search again for only the unmapped portion of the read to find the next longest sequence that exactly matches the reference genome.





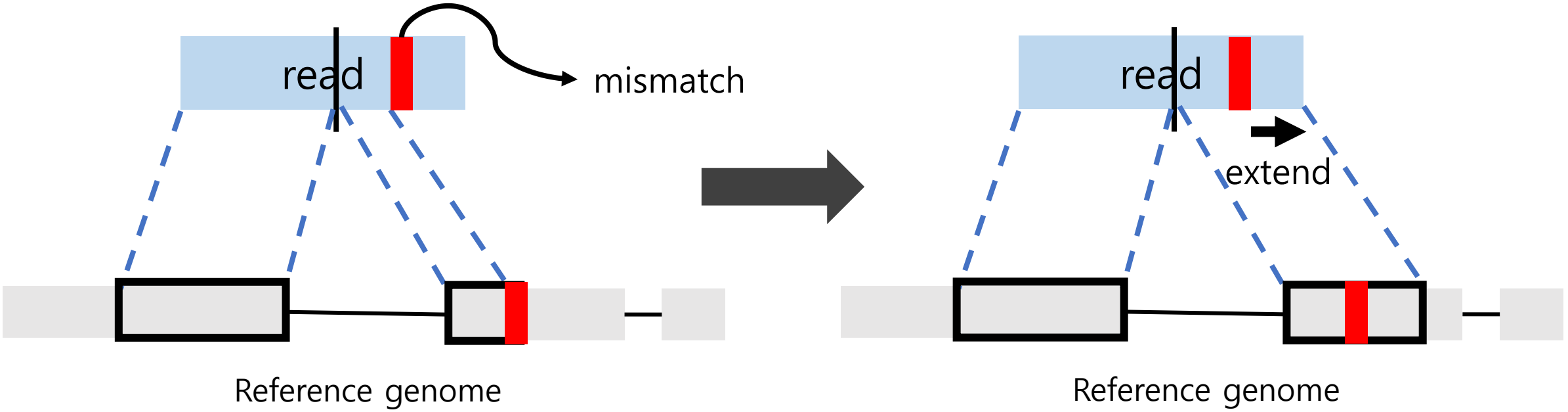
https://hbctraining.github.io/Intro-to-rnaseq-hpc-O2/lessons/03_alignment.html

2 STAR Algorithm

① Sequential seed search

If the MMP search does not reach the end of a read because of the presence of one or more mismatches (for detecting mismatches)

- The MMPs will serve as anchors in the genome that can be extended, allowing for alignments with mismatches.



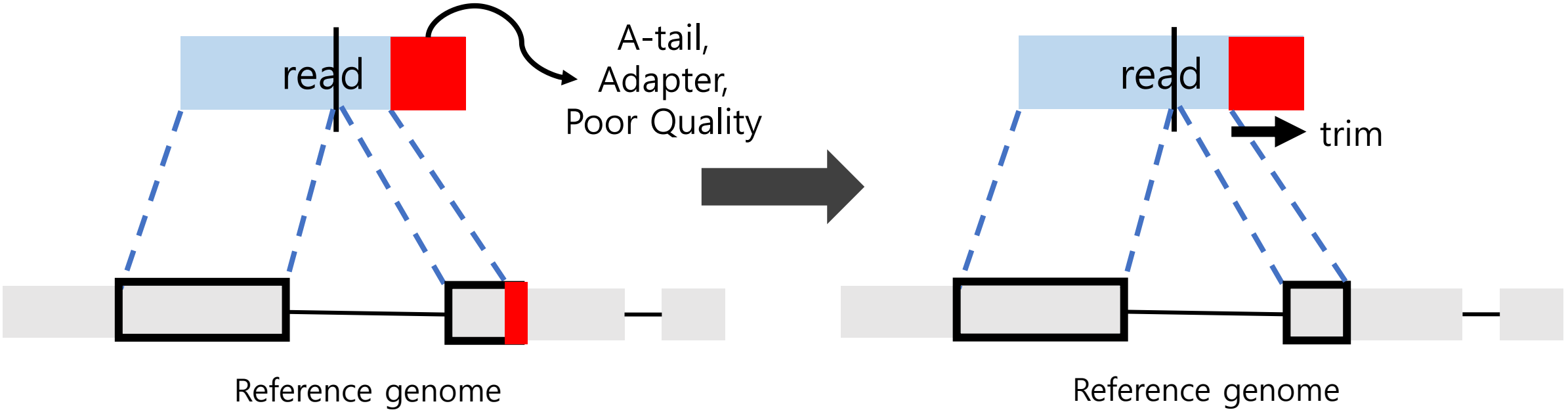
https://hbctraining.github.io/Intro-to-rnaseq-hpc-O2/lessons/03_alignment.html

2 STAR Algorithm

① Sequential seed search

If extension does not give a good alignment (for detecting poly-A tails, adapter, poor sequencing quality)

- Poly A tail, adapter sequence, poor quality (or other contaminating sequence) will be soft clipped.



https://hbctraining.github.io/Intro-to-rnaseq-hpc-O2/lessons/03_alignment.html

2 STAR Algorithm

① Sequential seed search

What is the advantage?

- **This sequential searching of only the unmapped portions of reads underlies the efficiency of the STAR algorithm.**

- This approach represents a natural way of finding precise locations of splice junctions in a read and is advantageous over an arbitrary splitting of read sequences.

⇒ Facilitates finding anchors for reads with errors near the ends and improves mapping sensitivity for high sequencing error rate conditions.

https://hbctraining.github.io/Intro-to-rnaseq-hpc-O2/lessons/03_alignment.html

2 STAR Algorithm

① Sequential seed search

What is the advantage?

- **STAR also uses an Suffix Array (SA) to efficiently search for MMPs, this allows for quick searching against even the largest reference genomes.**

(Other slower aligners use algorithms that often search for the entire read sequence before splitting reads and performing iterative rounds of mapping)

https://hbctraining.github.io/Intro-to-rnaseq-hpc-O2/lessons/03_alignment.html

Suffix Arrays (SA)

Suffix Array of the whole genome is utilized to find the Maximum Mappable Prefixes (MMP)

- **Suffix Array : a sorted array of all suffixes of a given string.**

Genome sequence = GATAGACA

i (start point)	Suffix
0	GATAGACA
1	ATAGACA
2	TAGACA
3	AGACA
4	GACA
5	ACA
6	CA
7	A

Sort the suffixes alphabetically

Sa[i]	Suffix
7	A
5	ACA
3	AGACA
1	ATAGACA
6	CA
4	GACA
0	GATAGACA
2	TAGACA

https://www.youtube.com/watch?v=zqKlL3ZpTqs
https://www.geeksforgeeks.org/suffix-array-set-1-introduction/
http://www.secmem.org/blog/2021/07/18/suffix-array-and-lcp/

Suffix Arrays (SA)

Suffix Array of the whole genome is utilized to find the Maximum Mappable Prefixes (MMP)

Genome sequence = GATAGACA

Sa[i]	Suffix
7	A
5	ACA
3	AGACA
1	ATAGACA
6	CA
4	GACA
0	GATAGACA
2	TAGACA

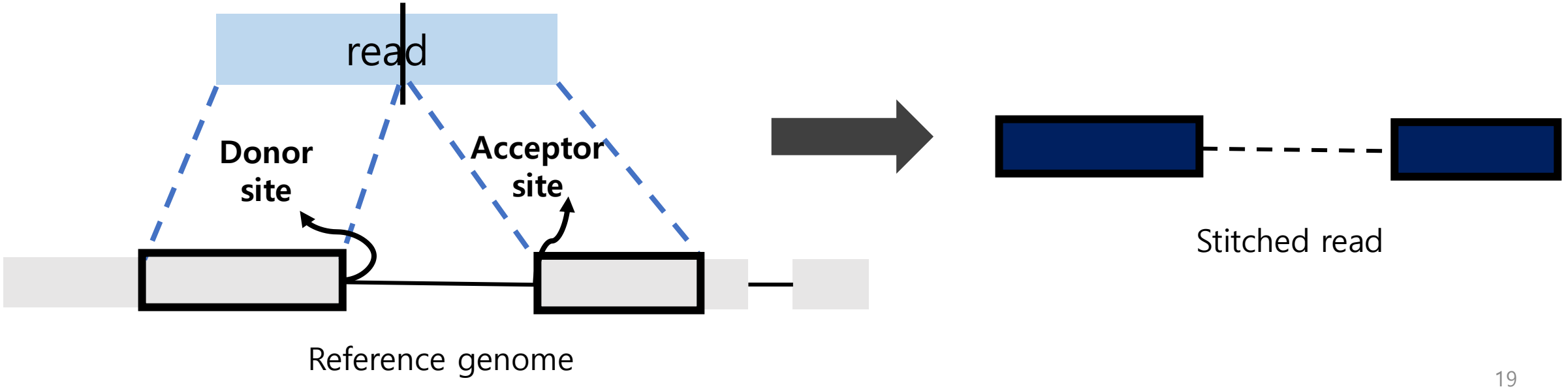
Suffix Array of Genome sequence = [7,5,3,1,6,4,0,2]

- Before the alignment begins, Suffix Array and genome sequence are loaded into RAM and stored, allowing access from multiple processes (threads).

② Clustering, stitching and scoring

Overview

- **To create a complete read**
  - ① clustering the seeds together based on proximity to a set of 'anchor' seeds, or seeds that are not multi-mapping.
  - ② Then the seeds are stitched together based on the best alignment for the read (scoring based on mismatches, indels, gaps, etc.).



2 STAR Algorithm

② Clustering, stitching and scoring

Clustering

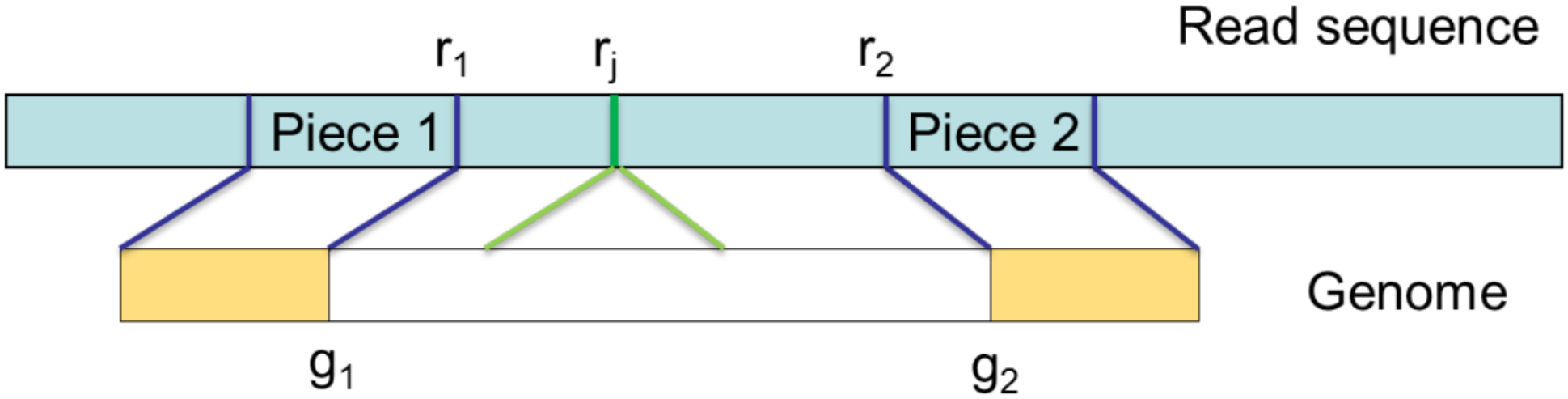
▶ All the alignments (anchor and non-anchor) located within an alignment window will be stitched to each other in an attempt to find the best linear align.

- Anchors : All the alignments that map less than a user defined value (20-50)
- Alignment windows : genomic regions selected around the anchors.

② Clustering, stitching and scoring

Stitching

- **The mapped seeds within the windows selected in clustering are stitched together into "transcripts" assuming a linear transcription model.**
  - Different blocks of the alignment do not overlap.
  - Blocks that follow each other in the read have to also follow each other in genome.



② Clustering, stitching and scoring

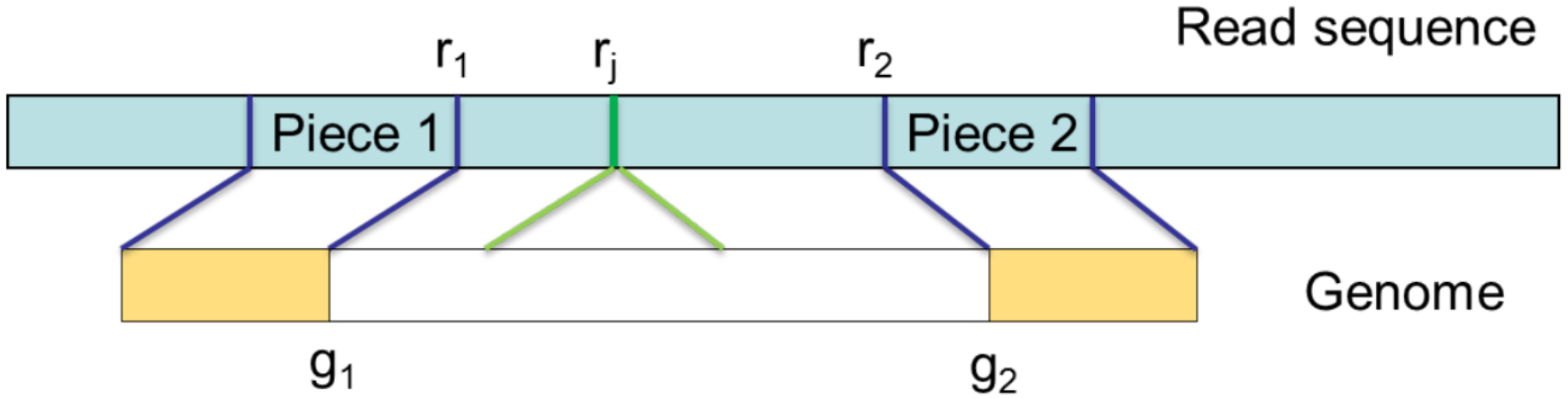
Stitching

- **The algorithm searches for the junction position in read r_j that yields the maximum score by finding the maximum of the following quantity.**

$$max\left\{\sum_{r=1}^{r_j-r_1}\begin{bmatrix}1 & if\ R(r_1+r)=G(g_1+r)\ \&R(r_1+r)\neq G(g_1+r+\Delta)\\ -1 & if\ R(r_1+r)\neq G(g_1+r)\ \&R(r_1+r)=G(g_1+r+\Delta)\\ 0 & otherwise\end{bmatrix}-P_{gap}(r_j)\right\}$$

$$R,G: read\ and\ genome\ sequence, r_1, r_2, g_1, g_2: coordinates\ defined\ as\ below$$

$$\Delta\equiv (g_2-g_1)-(r_2-r_1)$$



② Clustering, stitching and scoring

Scoring scheme

Total score for each alignment is calculated as below.

$$S = +\sum_{match} P_m - \sum_{mismatch} P_{mm} - \sum_{insertion} P_{ins} - \sum_{deletion} P_{del} - \sum_{gap} P_{gap}$$

- In the present version of STAR, match & mismatch : +/- 1
- For short deletions and all insertions, the penalty is calculated as below.

$$P_{ins/del} = P_{ins/del}^{open} + P_{ins/del}^{extend} \cdot L_{ins/del}$$

- Deletions that are longer than user-defined minimum intron size are considered splice junction(gaps), and their penalties consist of a constant gap opening penalty and a penalty which depends logarithmically on the gap length.

② Clustering, stitching and scoring

Extension and selection

Extension

- If necessary, the alignments are extended towards unmapped 5′ and 3′ end of the reads, using a simple algorithm.
- And stop the extension when the score reaches the maximum or there are too many mismatches.

Selection

- **Alignments from all windows are collected and sorted by their score.**
- **The stitched combination with the highest score is chosen as the best alignment of a read.**

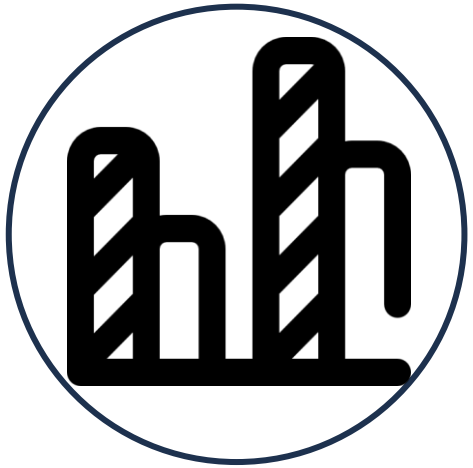
② Clustering, stitching and scoring

What is the advantage?

Although the sequential MMP search only finds the seeds exactly matching the genome,

The subsequent stitching procedure is capable of aligning reads with a large number of mismatches, indels and splice junctions, scalable with the read length.

https://hbctraining.github.io/Intro-to-rnaseq-hpc-O2/lessons/03_alignment.html

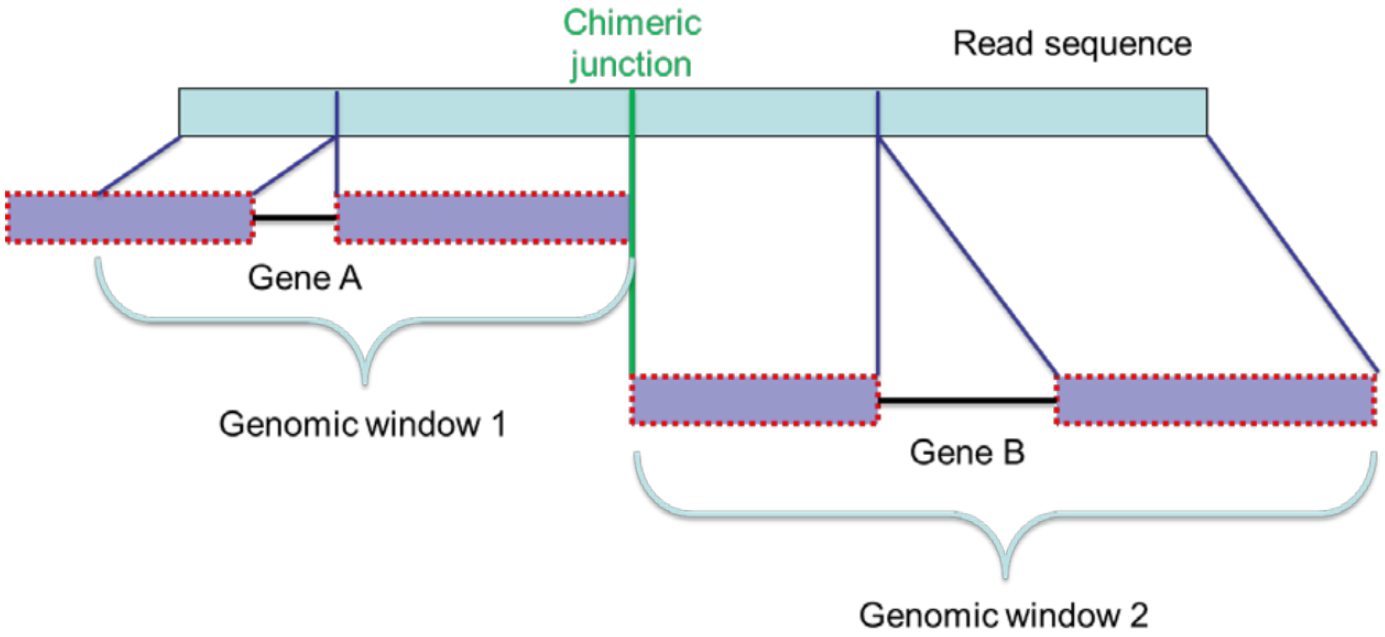


3. Further analysis

3 Further analysis

STAR can find chimeric alignments

If the best scoring alignment window (main window) does not cover the entire read

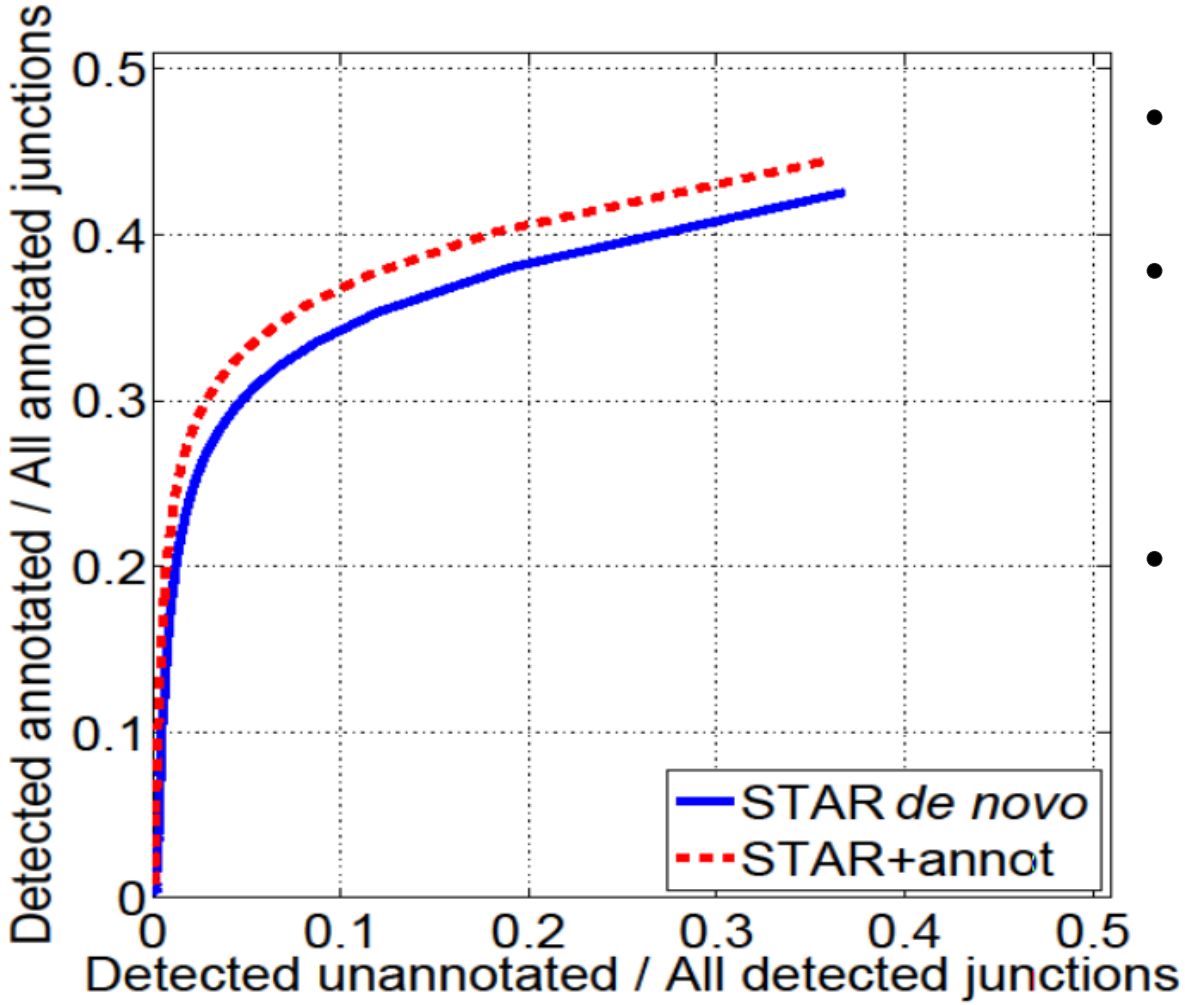


- It can be reported chimeric connections to the other windows that cover portions of the read not covered by the main window.
- These chimeric connections can span long distance on the same strand, or different strand on the same chromosome, or different chromosomes.

3 Further analysis

Running STAR with annotated junctions DB

STAR can utilize annotated splice junctions loci to improve sensitivity of the splice junction detection.



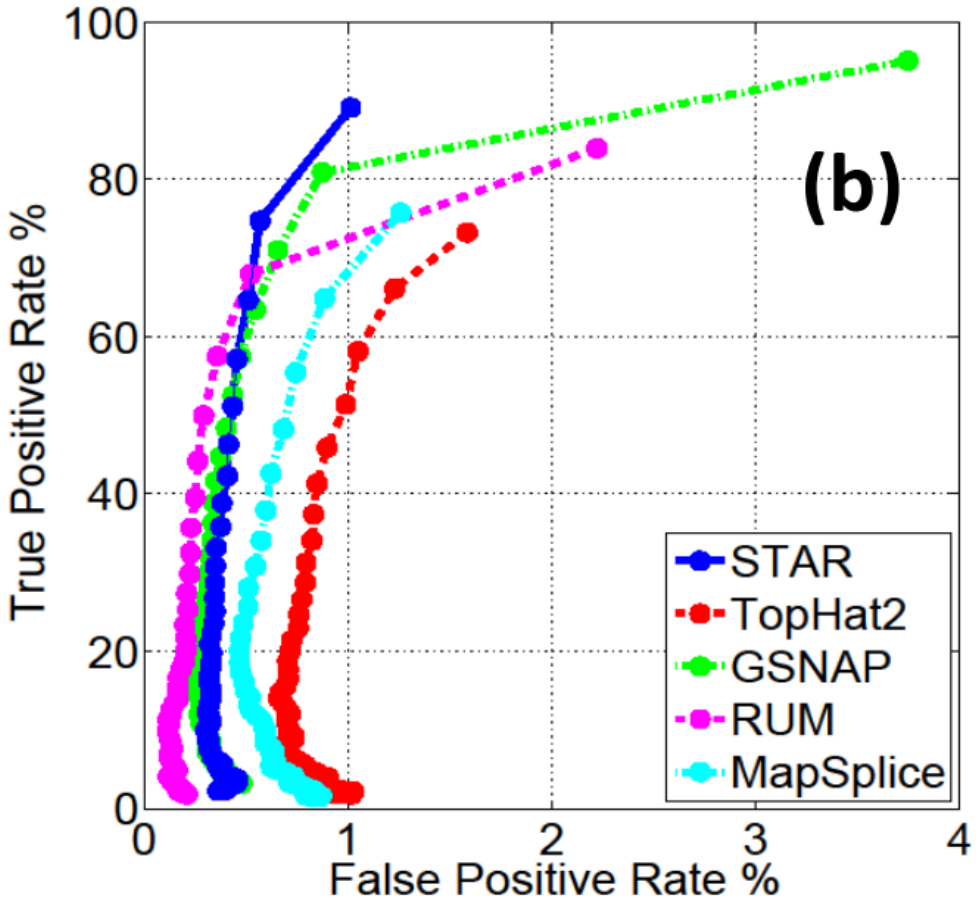
- STAR incorporates annotated junction sequences into the Suffix Array.
- Then, searches the seeds that cross the junctions simultaneously with the seeds that map contiguously to the genome.

- **This makes STAR more sensitive to splicing events that involve short sequence overhangs on either side of a junction.**

3 Further analysis

Performance comparison on simulated RNA-seq data

For each aligner, only junctions supported by at least N reads were selected for each point along ROC curves. N=1(lowest threshold)~100(highest threshold)



- Simulation data : Illumina-like read sequence with a reasonably high error rate were generated.
- All aligners exhibit desirable steep ROC curves at high values of detection threshold.
- **At the lowest detection threshold, STAR exhibits the lowest false-positive rate while achieving high sensitivity.**

4. Google Colab

4 Google Colab

Drawback of STAR

"STAR's high mapping speed is traded off against RAM usage"

- Uncompressed Suffix Arrays (used by STAR) demonstrate a significant speed advantage over the compressed Suffix Arrays implemented in many popular short read aligners.

▶ **This speed advantage is traded off against the increased memory usage by uncompressed.**

4 Google Colab

Comparison

Requirement of STAR and Google Colab

Requirement

- RAM : At least 10 * GenomeSize bytes. (ex) human genome of ~3 GigaBases will require ~30 GigaBytes of RAM. 32GB is recommended for human genome alignments.
- Disk Space : sufficient free disk space (>100 GigaBytes) for storing output files.

Colab

- Available RAM : 12GB
- Disk Space : 25GB