Bioinformatics

Alignment algorithms

Short Read Alignment

- Differences to conventional alignment:
- millions of very short reads, rather than a few long ones, have to be mapped to the genome
- mismatches can be, frequently, errors in the sequencing process. In some cases they can be single point mutations/variations.
- base-call quality information are more important
- only small gaps are expected. A larger gap leads to a less biologically meaningful alignment.
- mate-paired reads require special handling

Alignment Tools

Type	Name	Link
General aligner	GMAP/GSNAP	http://research-pub.gene.com/gmap/
	BFAST	http://sourceforge.net/apps/mediawiki/bfast/index.php
	BOWTIE	http://bowtie-bio.sourceforge.net/index.shtml
	CloudBurst	http://sourceforge.net/apps/mediawiki/cloudburst-bio/index.php
	GNUmap	http://dna.cs.byu.edu/gnumap/index.shtml
	MAQ/BWA	http://maq.sourceforge.net/
	Perm	http://code.google.com/p/perm/
	RazerS	http://www.seqan.de/projects/razers.html
	Mrfast/mrsfast	http://mrfast.sourceforge.net/manual.html
	SOAP/SOAP2	http://soap.genomics.org.cn/
	SHRiMP	http://compbio.cs.toronto.edu/shrimp/
De Novo annotator	QPALMA/GenomeMapper/PALMapper	http://www.fml.tuebingen.mpg.de/raetsch/suppl/palmapper
	SpliceMap	http://www.stanford.edu/group/wonglab/SpliceMap/
	SOAPals	http://soap.genomics.org.cn/
	G-Mo. R-Se	http://www.genoscope.cns.fr/externe/gmorse/
	TopHat	http://tophat.cbcb.umd.edu/
	SplitSeek	http://solidsoftwaretools.com/gf/project/splitseek
De Novo transcript assembler	Oases	http://www.ebi.ac.uk/~zerbino/oases/
	MIRA	http://sourceforge.net/apps/mediawiki/mira-assembler/index.php

Examples of Alignment

```
groan      colo-r          theatre      theatre
|||:|      |||| |          |||||::      |||||×
grown      colour          theater      theater
```

```
elephant          vermiform     vermiform-----
|||: |||          ::||::::::          ||||
eleg-ant          formation     -----formation
```

```
disestablishment      disestablishment
|||         |  |||     |||           :|||
dis--------s--ent      dis----------sent
```

Relationship *alignment-biological events*

- *Query* (Qry): string that we want to align
- *Subject* (Sbj): reference string

(Qry)	A C D E F G	A C D E F G	A C D E F G	A C -- E F G
(Sbj)	A C D E F G	A C L E F G	A C -- E F G	A C D E F G
Biological event	**Conservation**	**Substitution**	**Insertion**	**Deletion**
Alignment represent	**Match**	**Mismatch**	**Gap**	**Gap**

Global Pair-wise alignment

Example of a Score function:

Points for a matching letter:	1
Points for a non-matching letter:	-1
Points for inserting a gap:	-2

ATCGATACG, ATGGATTACG

```
ATCGAT-ACG
|| | || | |
ATGGATTACG
```

Matches:	+1	+1		+1	+1	+1		+1	+1	+1	= +8
Mismatches:			-1								= -1
Gaps:							-2				= -2
											Total score = +5

Dynamic Programming and global alignment

- Dynamic programming is a method by which a larger problem may be solved by first solving smaller, partial versions of the problem.

- We demonstrate here how it may be applied to global sequence alignment, where at first we are interested only in the similarity of two sequences, and not the alignment that yields this score

- SEE slides 16 to 24 file “Detailed Alignment.pdf”

Comments on slides 20, 21 file "Detailed Alignment.pdf"

- If X is the *Query* (Qry) string and Y is the *Subject* (Sbj), and i identifies the letters of X and j the letters of Y,

- SIM (i,j) = SIM(i-1,j) + g implies a gap on X, the Qrj

- In fact, in order to update the score, we compare the letter (i-1)th of the Qry with letter jth of the Sbj. This is possible by adding a gap before the (i-1)th letter on Qry sequence so that the letter (i-1)th is in line with the jth letter on the Sbj (see next slide).

Comments on slides 20, 21 file “Detailed Alignment.pdf” (2)

(i-1)th (i)th ……

A C G T C Qry

(j-1)th (j)th ……

G A C T A Sbj

GAP insertion (thus deletion on the Qry)

(i-1)th (i)th ……

-- A C G T C Qry

(j-1)th (j)th ……

G A C T A Sbj

Comments on slides 20, 21 file "Detailed Alignment.pdf" (3)

- First case (without a gap): the score is compute by comparing (i-1)th letter on Qry with (j-1)th letter on Sbj

- Second case (adding a gap): the score is computed by comparing (i-1)th letter on Qry with (j)th letter on Sbj. This implies a penalty due to the insertion of the gap.

BLAST - Basic Local Alignment Search Tool

- Primarily designed to identify homologous sequences
  - Blast is a hashed seed-extend algorithm
  - Functional conservation
    - Only some parts of a sequence are under positive selection
    - From a functional perspective, these are the interesting parts

- Finding seeds significantly increases the speed of BLAST compared to doing a full local alignment over a whole sequence

- BLAST first finds highly conserved or identical sequences which are then extended with a local alignment.

BLAST - seed and extend

Example:

Seed size = 4,
No mismatches in seed

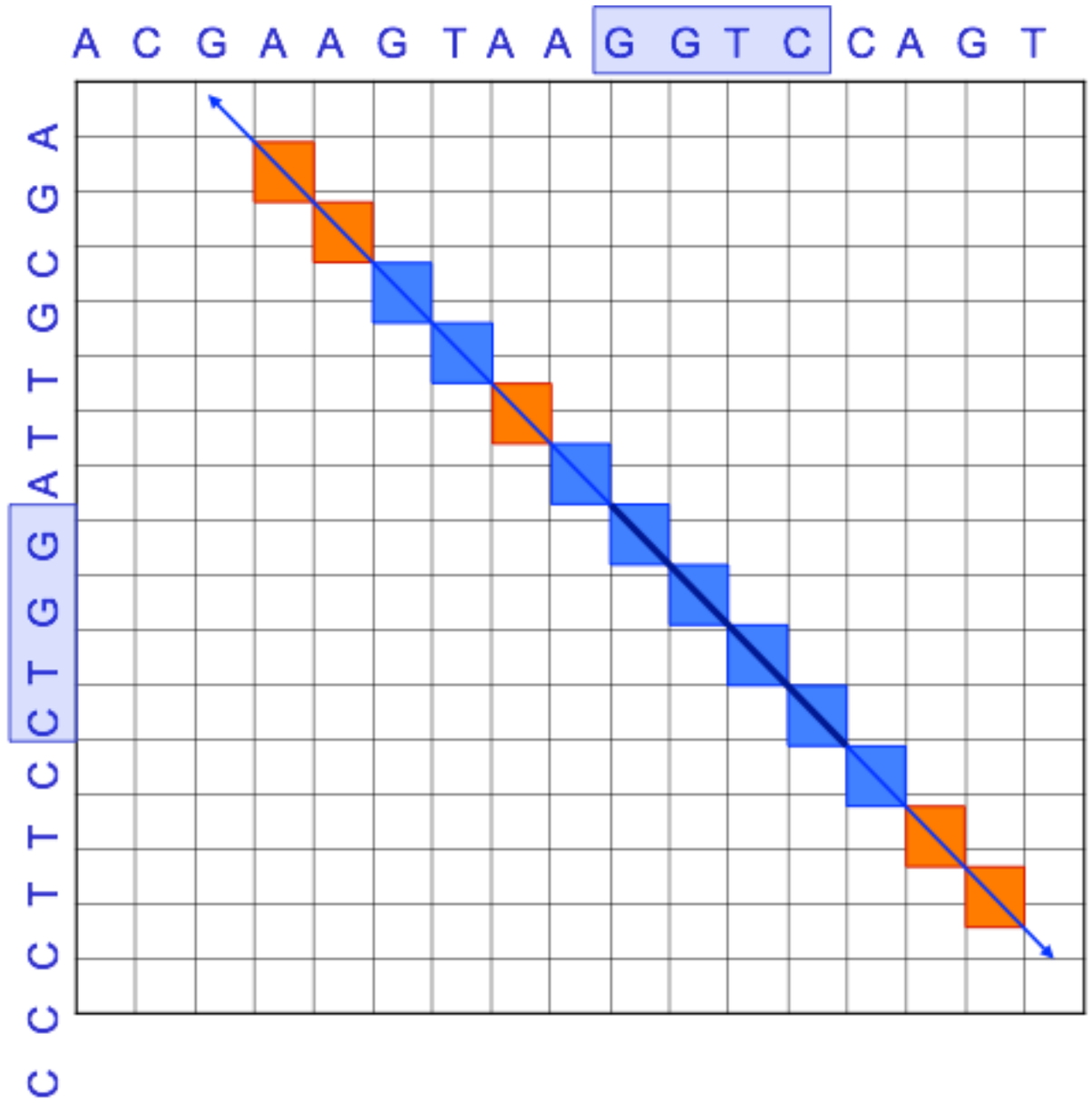
The matching word GGTC
initiates an alignment

Extension to the left and right
with no gaps until alignment
score falls below 50%

Output:

GTAAGGTCC

GTTAGGTCC



Speed of BLAST

- Typically BLAST will take approximately 0.1 – 1 second to search 1 sequence against a database
- Depends on size of database, e-value cutoff and number of hits to report selected
- 60 million reads equates to 70 CPU days!
- Even on multi-core systems this is too long!
- Especially if you have multiple samples!
- This is still true of FPGA and SIMD (vectorised) implementations of BLAST

When NOT to use BLAST

- A typical situation: you have a piece of DNA sequence and want to extend it or find what gene it belongs to.
- In other words, you want an **exact** or **near-exact** match to a sequence that is part of an **assembled genome**.
- Short reads require very fast algorithms for finding near-exact matches in genomic sequences:
  - BLAT
    - Highly recommended: the BLAT paper (Kent WJ (2003) *Genome Res* 12:656-64) - it is famous for its unorthodox writing style
  - SOAP
  - Bowtie/Bowtie 2
  - MAQ
  - BWA
  - Shrimp2

Adapted hashed seed-extend algorithms to work with shorted reads

- Improve seed matching sensitivity
  - Allow mismatches within seed
    - BLAST
  - Allow mismatches + Adopt spaced-seed approach
    - ELAND, SOAP, MAQ, RMAP, ZOOM
  - Allow mismatches + Spaced-seeds + Multi-seeds
    - SSAHA2, BLAT, ELAND2
- Above and/or Improve speed of local alignment for seed extension
  - Single Instruction Multiple Data
    - Shrimp2, CLCBio
  - Reduce search space to region around seed

Suffix-Prefix Trie

- A family of methods which uses a Trie structure to search a reference sequence
  - Bowtie
  - BWA
  - SOAP version 2
  - Trie – data structure which stores the suffixes (i.e. ends of a sequence)
- Key advantage over hashed algorithms:
  - Alignment of multiple copies of an identical sequence in the reference only needs to be done once
  - Use of an FM-Index to store Trie can drastically reduce memory requirements (e.g. Human genome can be stored in 2Gb of RAM)
  - Burrows Wheeler Transform to perform fast lookups

Suffix Array and BWT

Constructing suffix array and BWT string for X=googol$. String X is circulated to generate seven strings, which are then lexicographically sorted.

After sorting, the positions of the first symbols form the suffix array (6, 3, 0, 5, 2, 4, 1) and the concatenation of the last symbols of the circulated strings gives the BWT string lo$oogg.

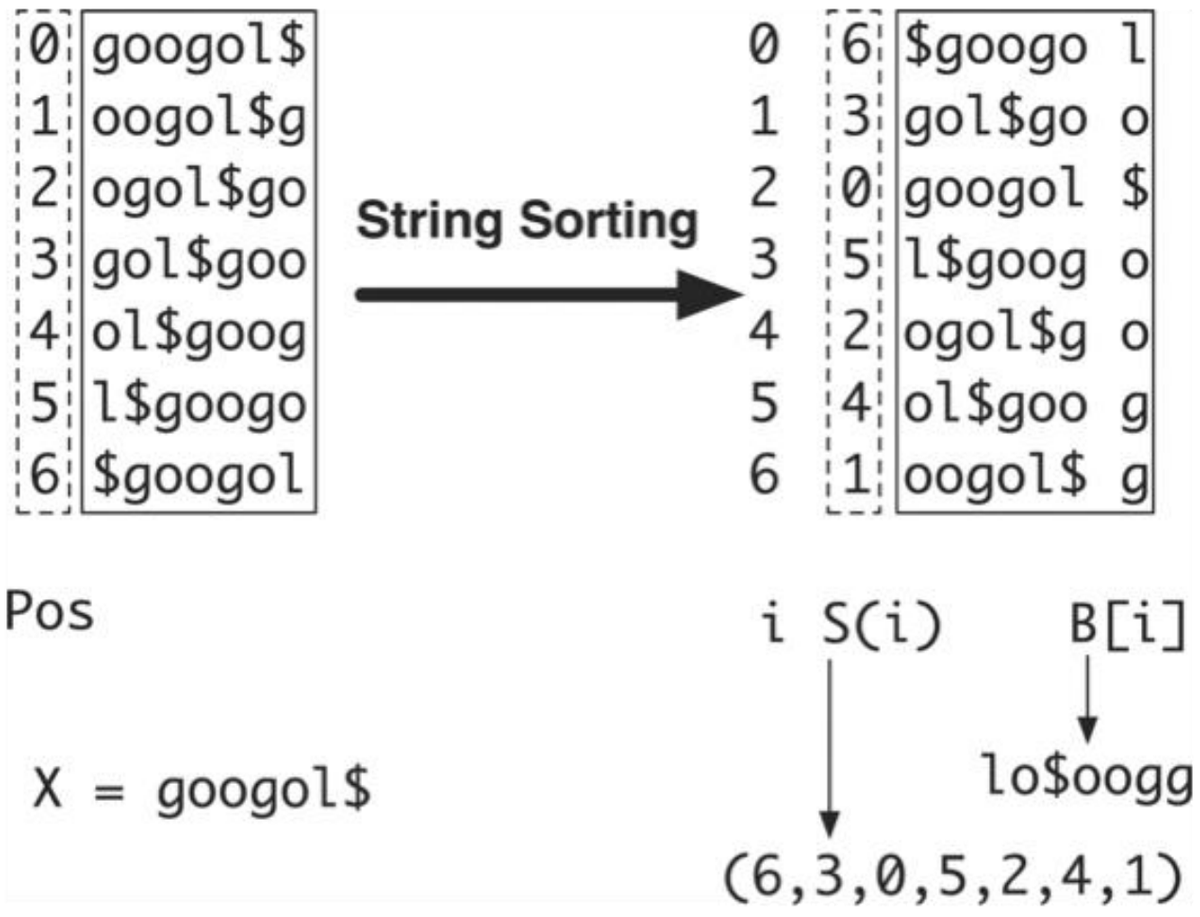
If string W is a substring of X, the position of each occurrence of W in X will occur in an interval in the suffix array.

This is because all the suffixes that have W as prefix are sorted together.

The SA interval of string 'go' is [1, 2]. The suffix array values in this interval are 3 and 0 which give the positions of all the occurrences of 'go'.

Knowing the intervals in suffix array we can get the positions. Therefore, sequence alignment is equivalent to searching for the SA intervals of substrings of X that match the query. For the exact matching problem, we can find only one such interval; for the inexact matching problem, there may be many.

Constructing suffix array and BWT string for X=googol$.



Li H , and Durbin R Bioinformatics 2009;25:1754-1760

Bioinformatics

Prefix Trie

Brute force search of string ‘LOL’ (query) with <2 mismatches:

1: G-mismatch, O-match, O-mismatch => stop (2 mismatches)

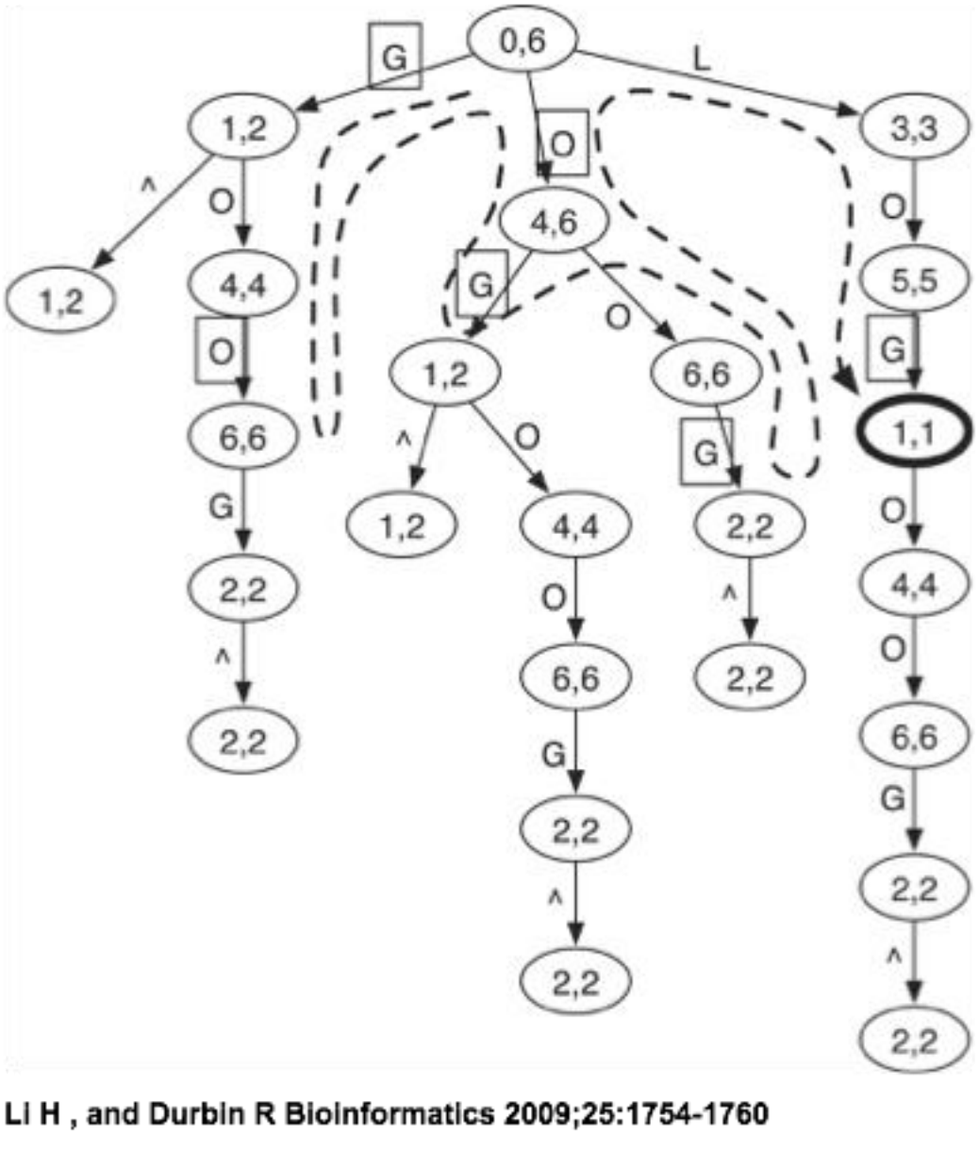
2: O-mismatch, G-mismatch => stop (2 mismatches)

3: O-mismatch, O-match, G-mismatch => stop (2 mismatches)

4: L-match, O-match, G-mismatch => OK (1 mismatch)

Interval: 1,1 => S(i) = 3 is the position in the original sequence

Prefix trie of string ‘GOOGOL’.



LI H , and Durbin R Bioinformatics 2009;25:1754-1760

Bioinf

Bowtie/Soap2 example

Reference

BWT(Reference)

Query:
AATGATACGGCGACCACCGAGATCTA

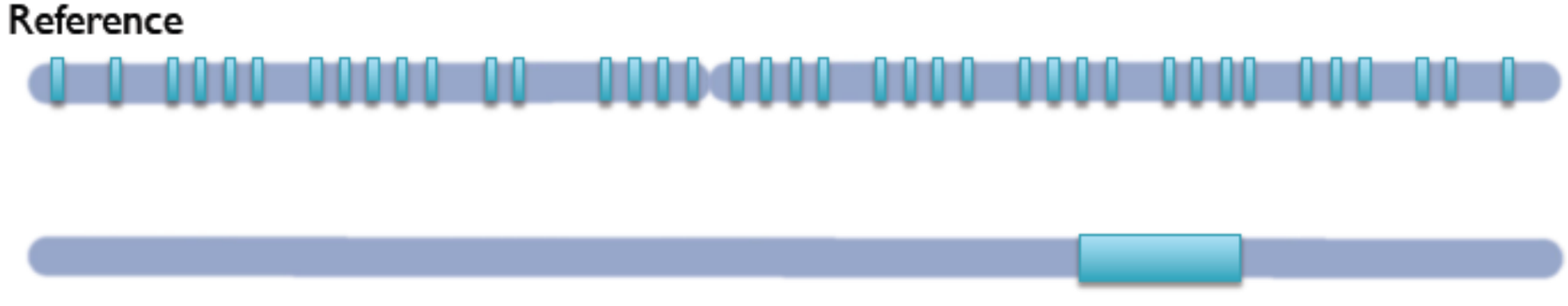
Bowtie/Soap2 example

Reference

BWT(Reference)

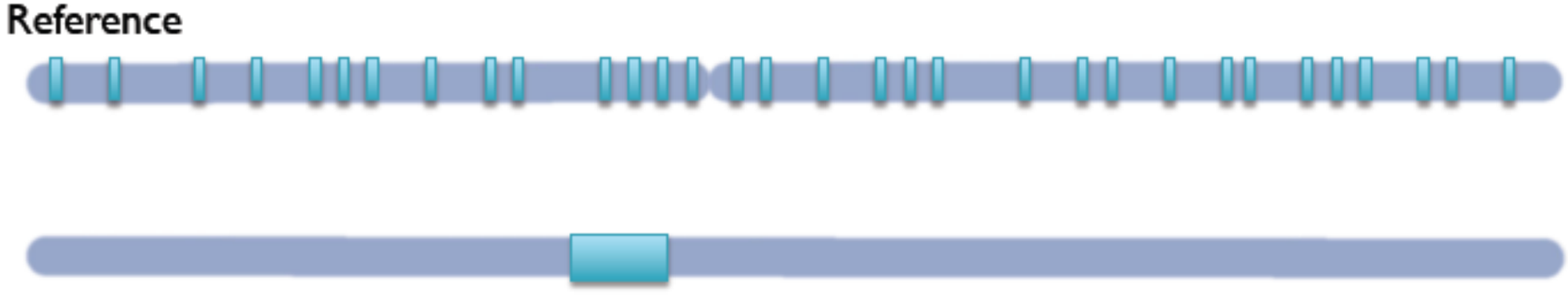
Query:
AATGATACGGCGACCACCGAGATCTA

Bowtie/Soap2 example



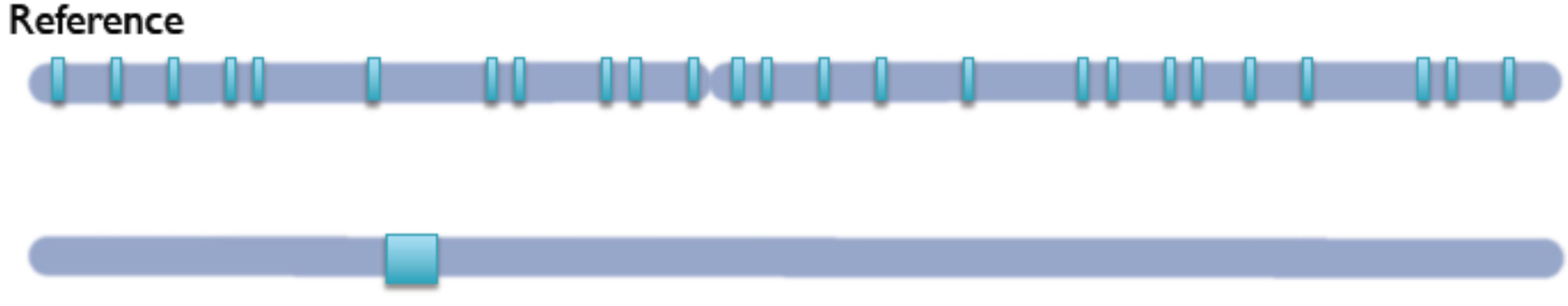
Query:
AATGATACGGCGACCACCGAGATCTA

Bowtie/Soap2 example



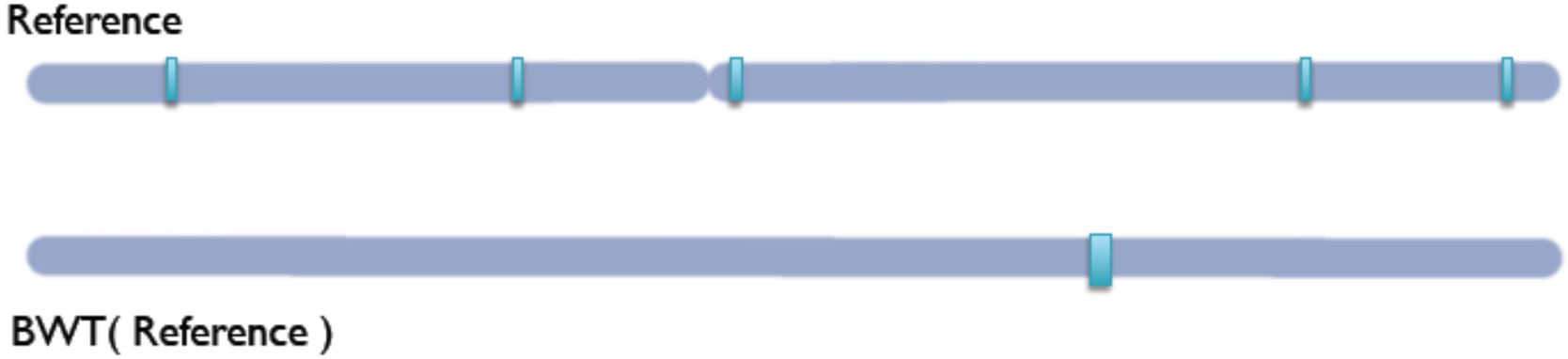
Query:
AATGATACGGCGACCACCGAGATCTA

Bowtie/Soap2 example



Query:
AATGATACGGCGACCACCGAGATCTA

Bowtie/Soap2 example



Query:
AATGATACGGCGACCACCGAGATCTA

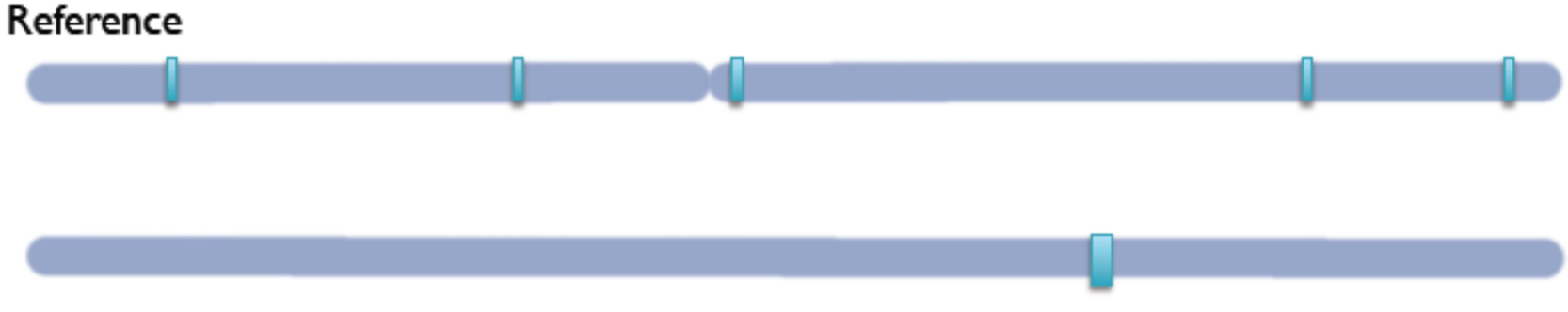
Bowtie/Soap2 example

Reference

BWT(Reference)

Query:
AATG 

Bowtie/Soap2 example



Query:
AATGTTACGGCGACCACCGAGATCTA

Bowtie/Soap2 example