The overlap problem in read assembly

- Sequencing DNA means to determine the sequence of bases, the DNA consists of.
- This sequence is called *target sequence*.
- It typically has length between 30 000 and several million bases.
- The different sequencing technologies do not allow to determine the target sequence directly.
- Therefore, one applies the *shotgun sequencing technique*.

This technique sequences parts of the DNA (called *reads*) as follows:

- the sequencing starts at a random position in one of the two complementary strands of the DNA.
- a continuous stretch of bases is sequenced in canonical direction, from $5'$ to $3'$.

The overlap problem in read assembly

genome

The overlap problem in read assembly

genome

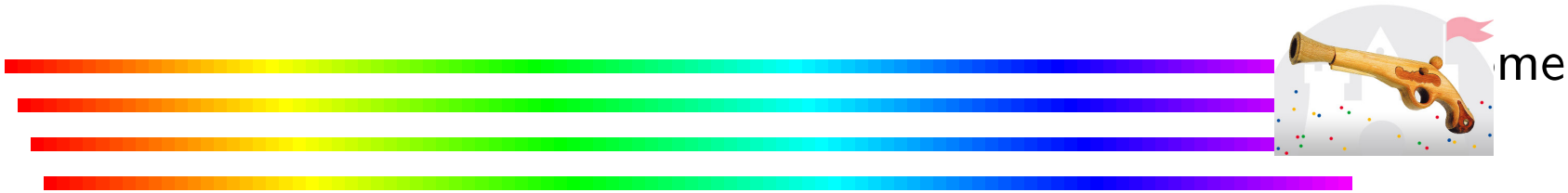
The overlap problem in read assembly

genome

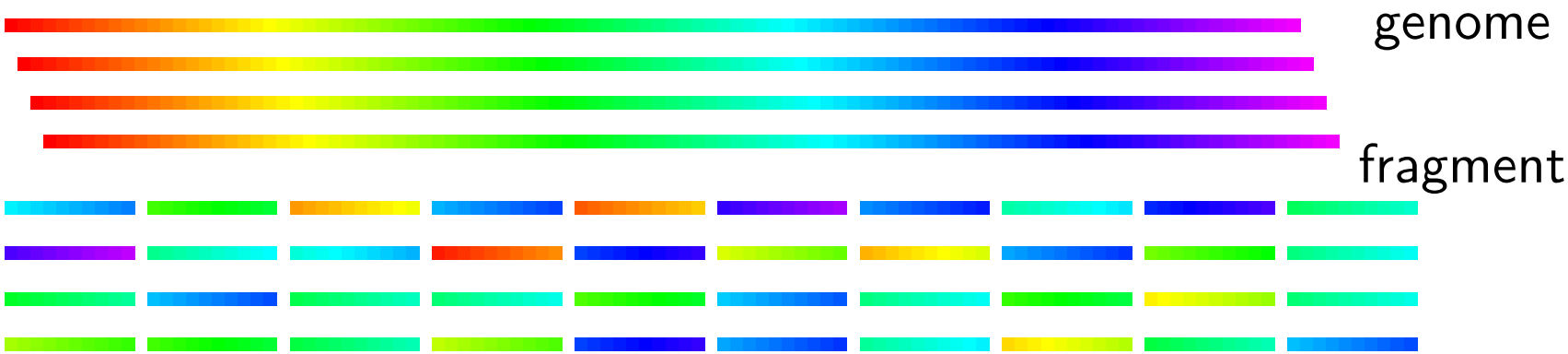
The overlap problem in read assembly

genome

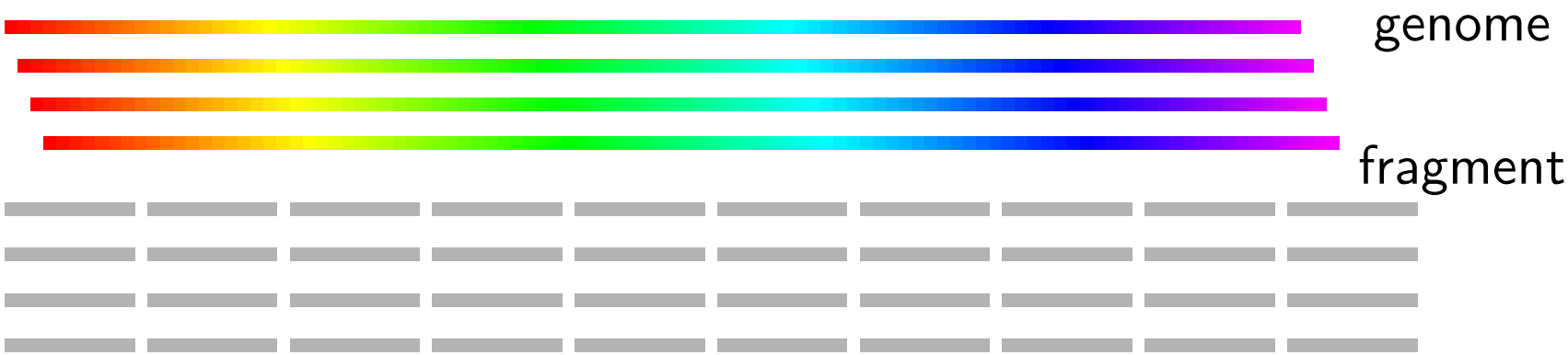
The overlap problem in read assembly



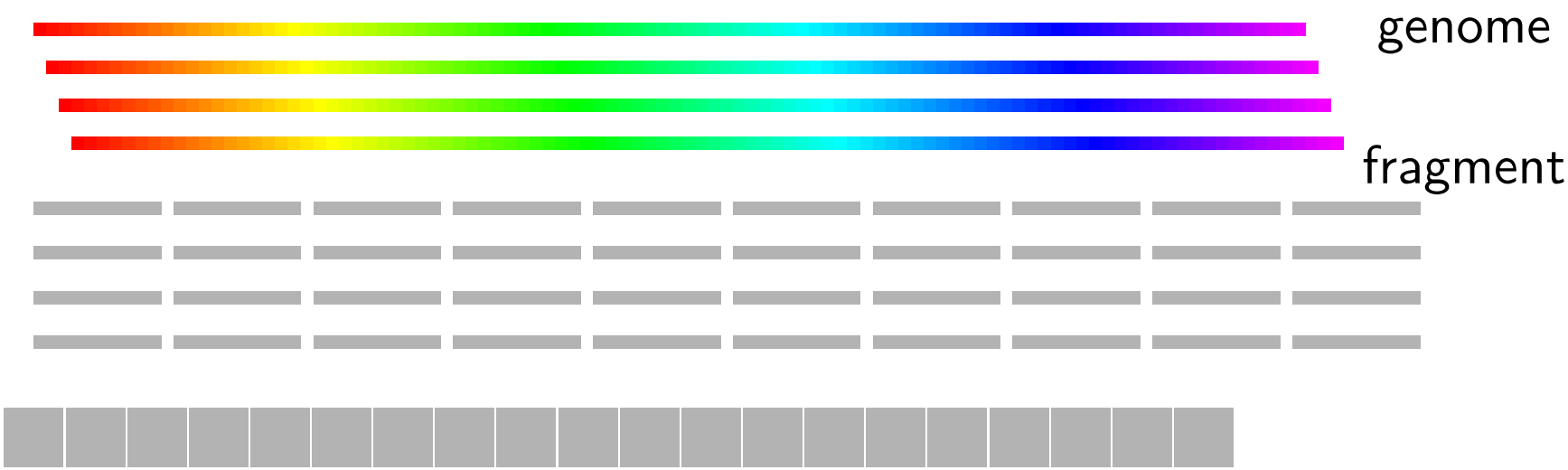
The overlap problem in read assembly



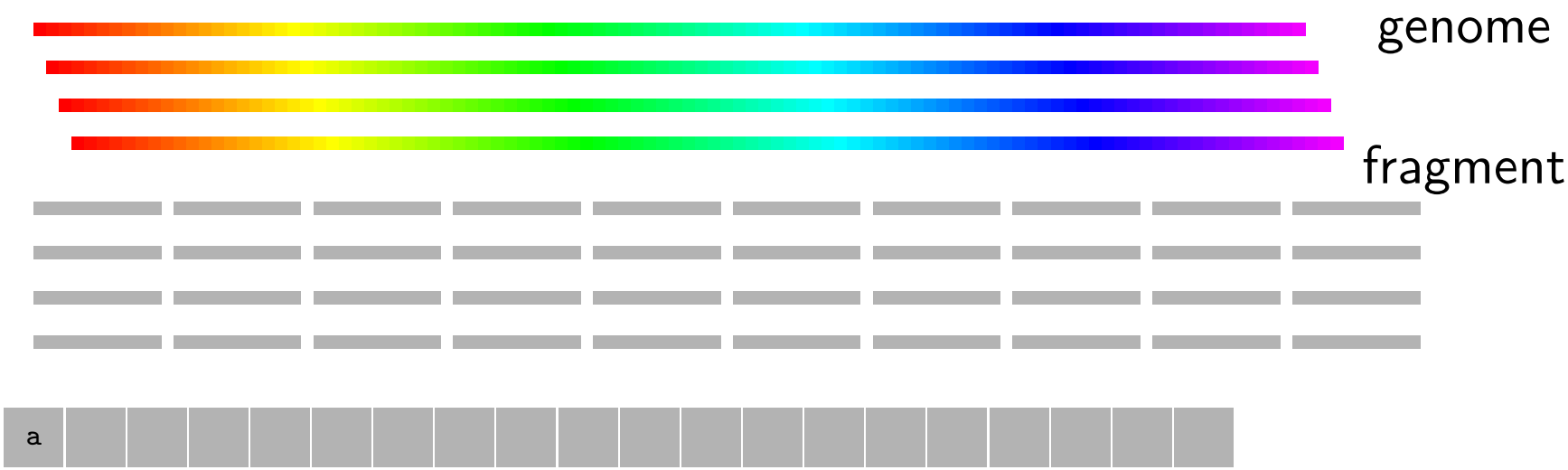
The overlap problem in read assembly



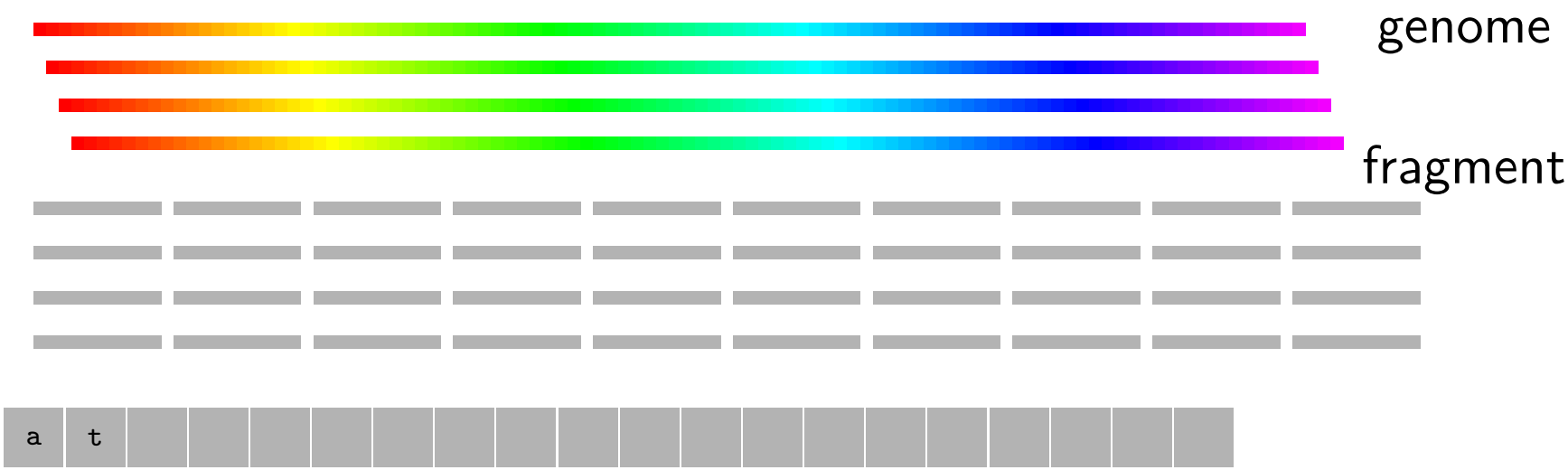
The overlap problem in read assembly



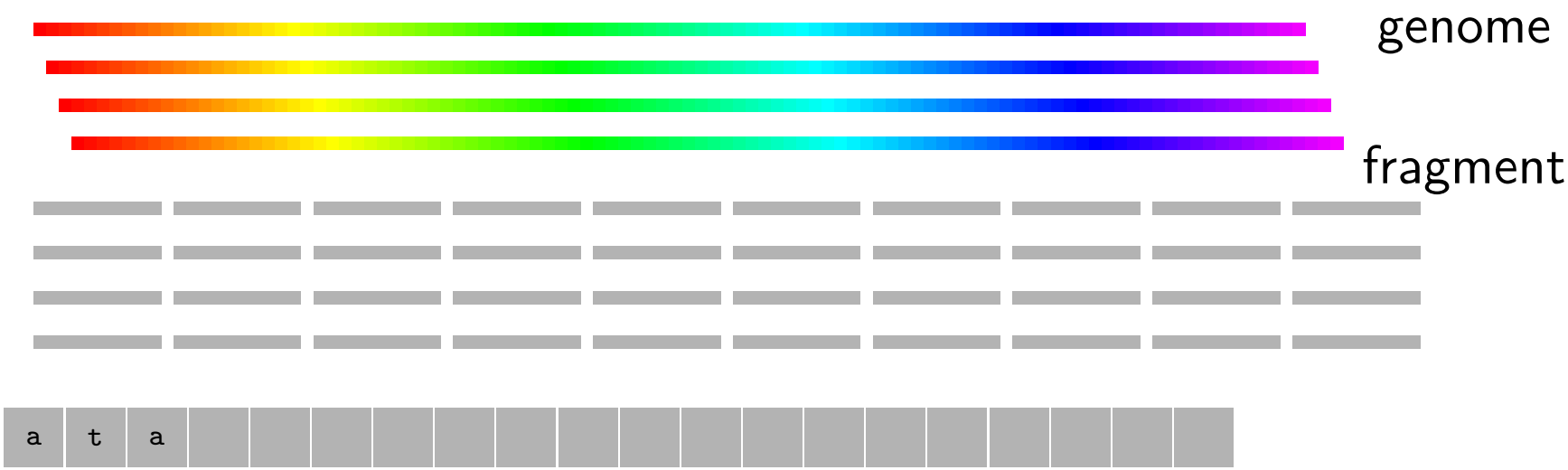
The overlap problem in read assembly



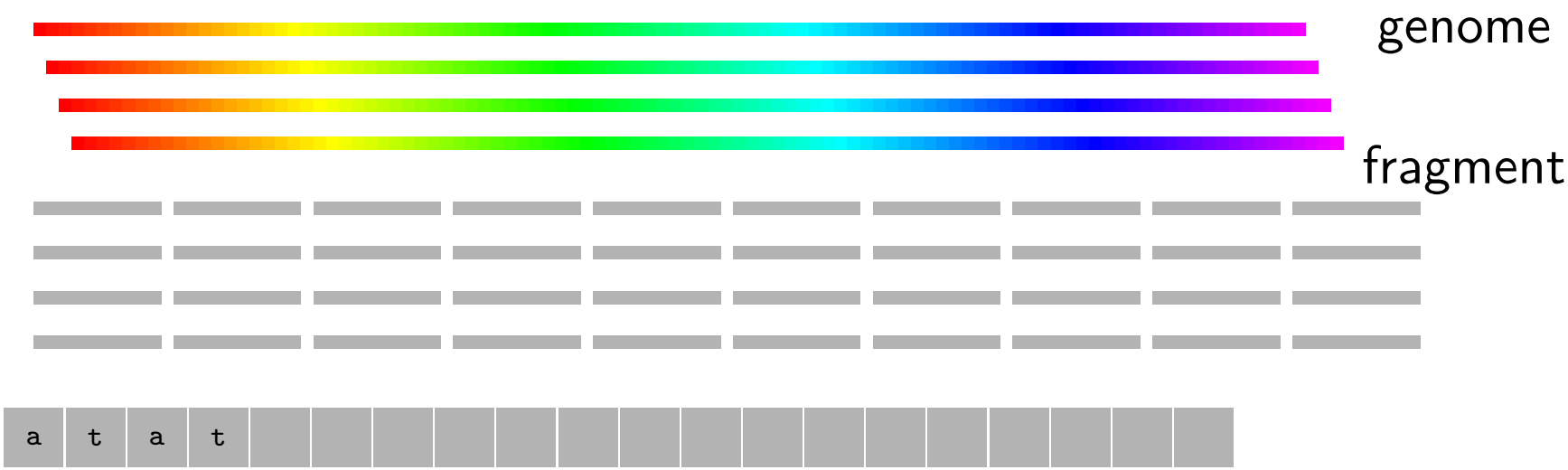
The overlap problem in read assembly



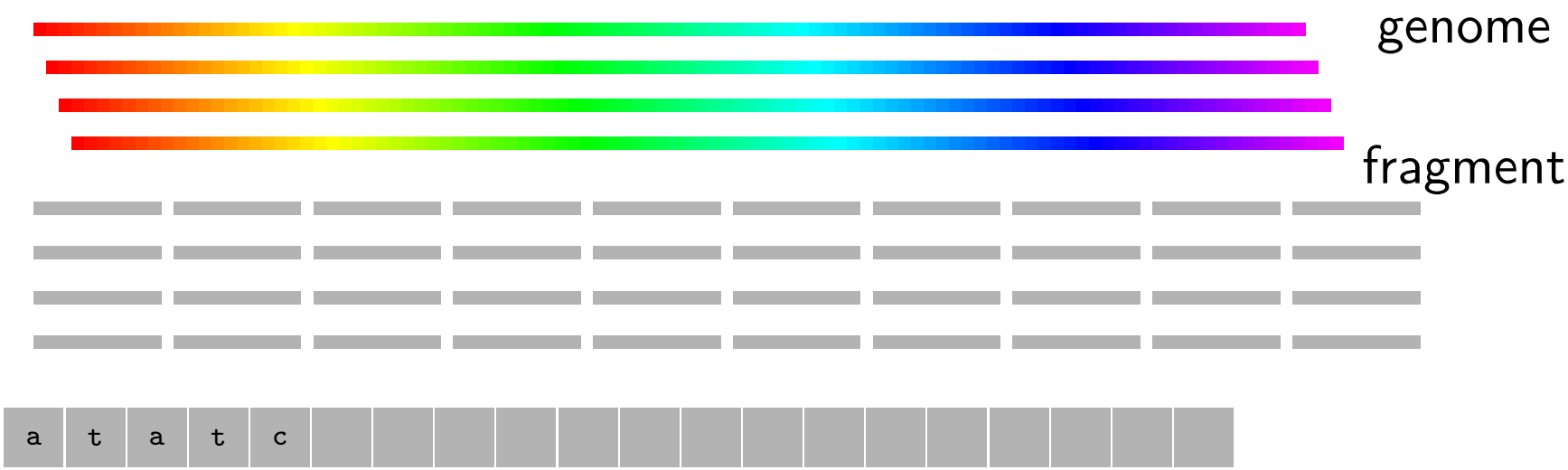
The overlap problem in read assembly



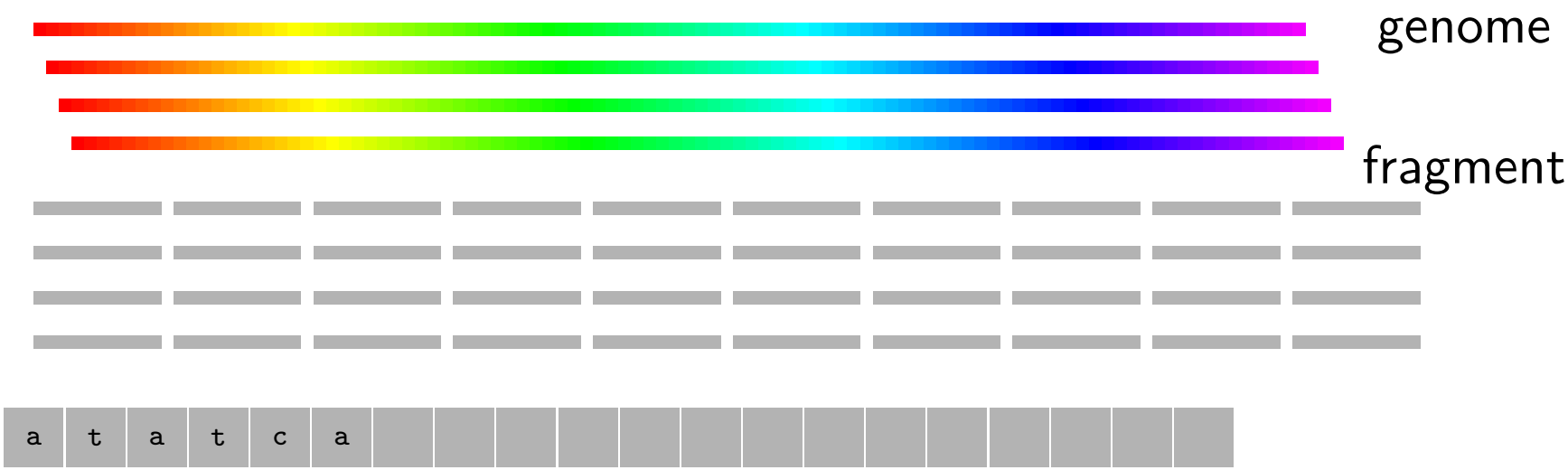
The overlap problem in read assembly



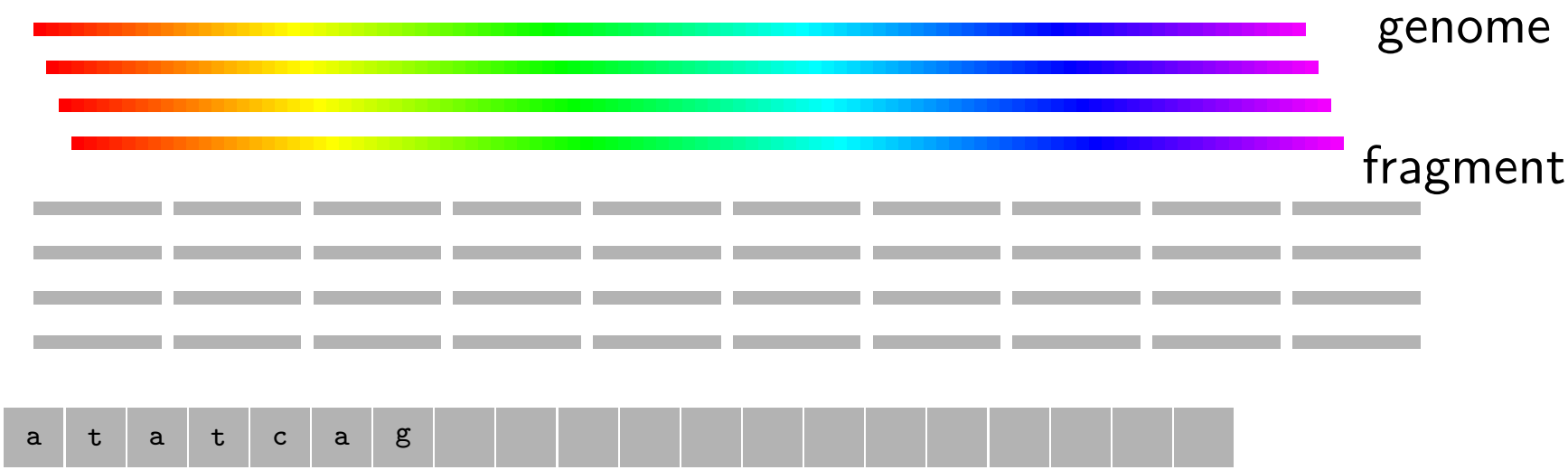
The overlap problem in read assembly



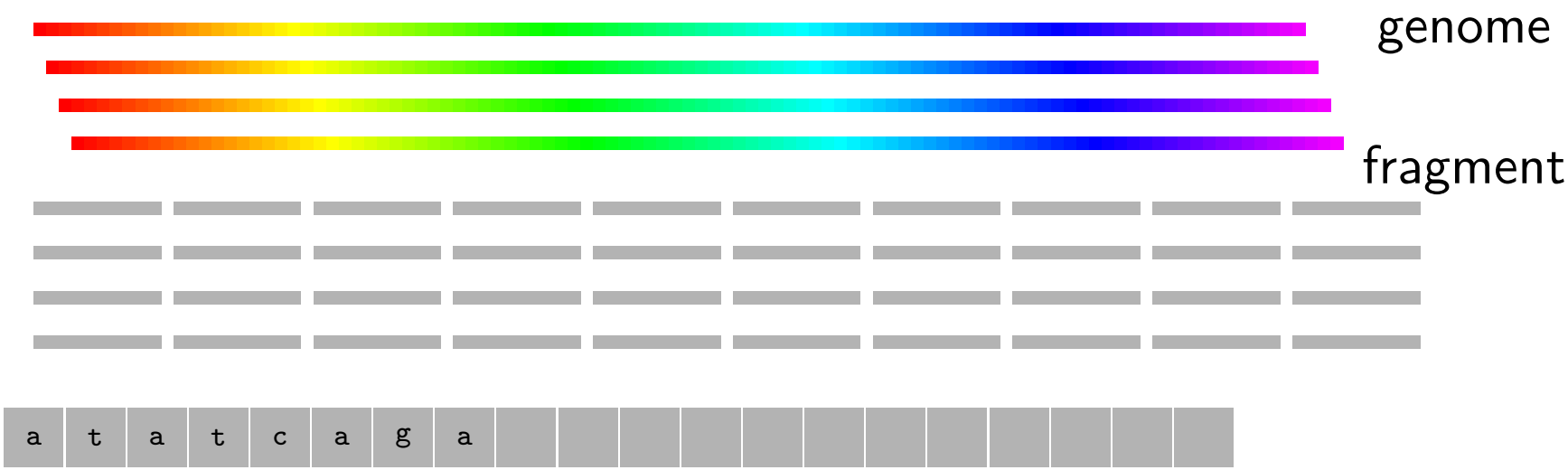
The overlap problem in read assembly



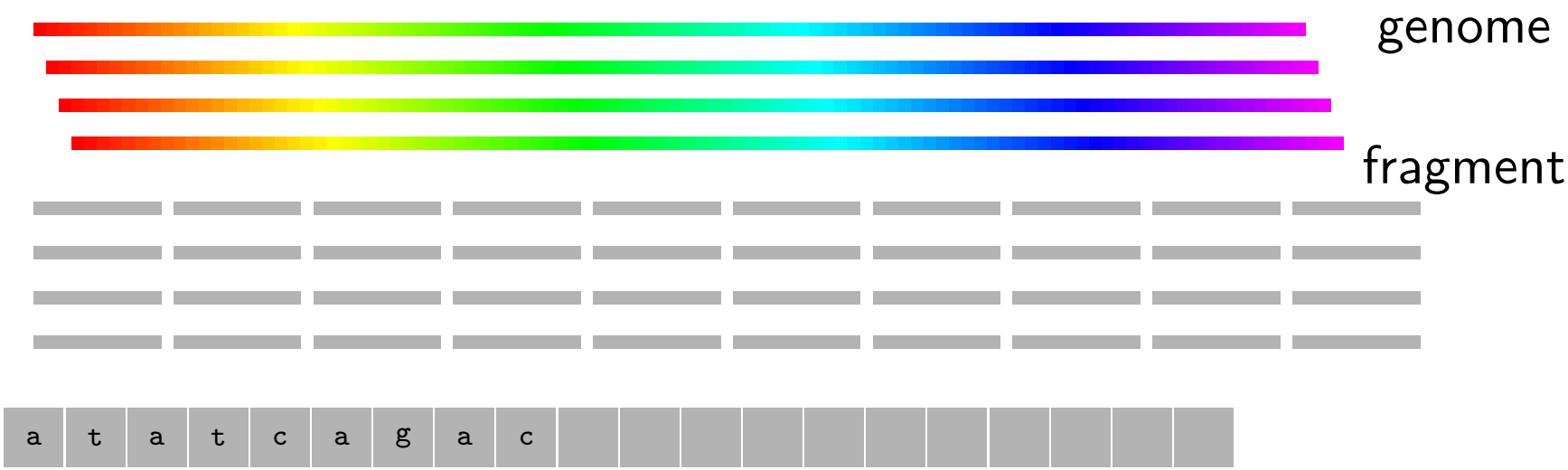
The overlap problem in read assembly



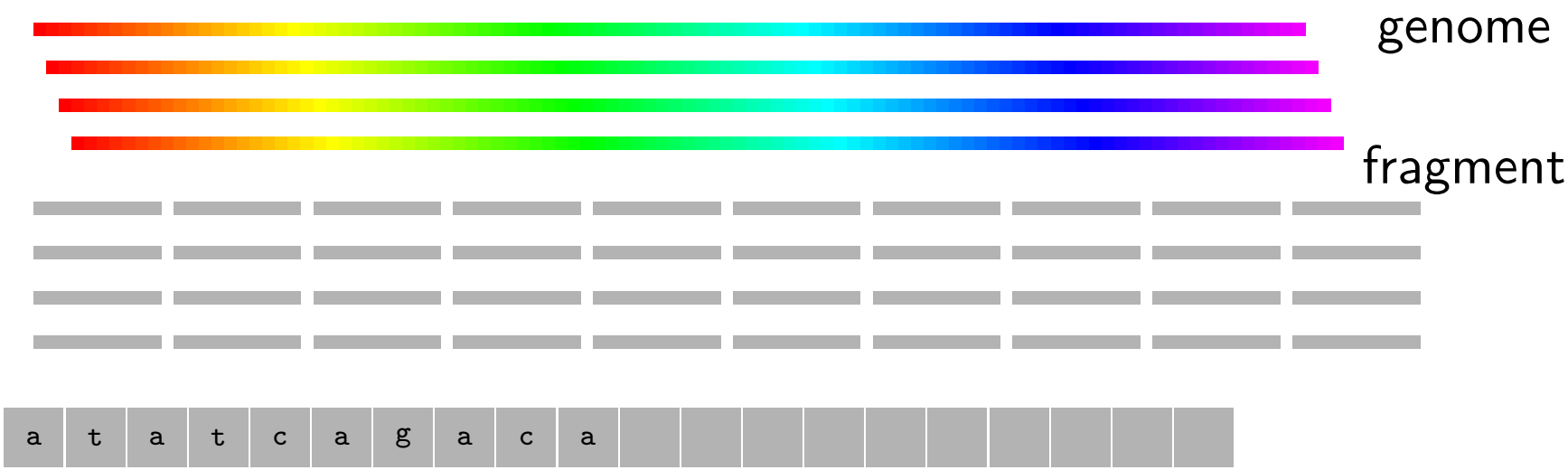
The overlap problem in read assembly



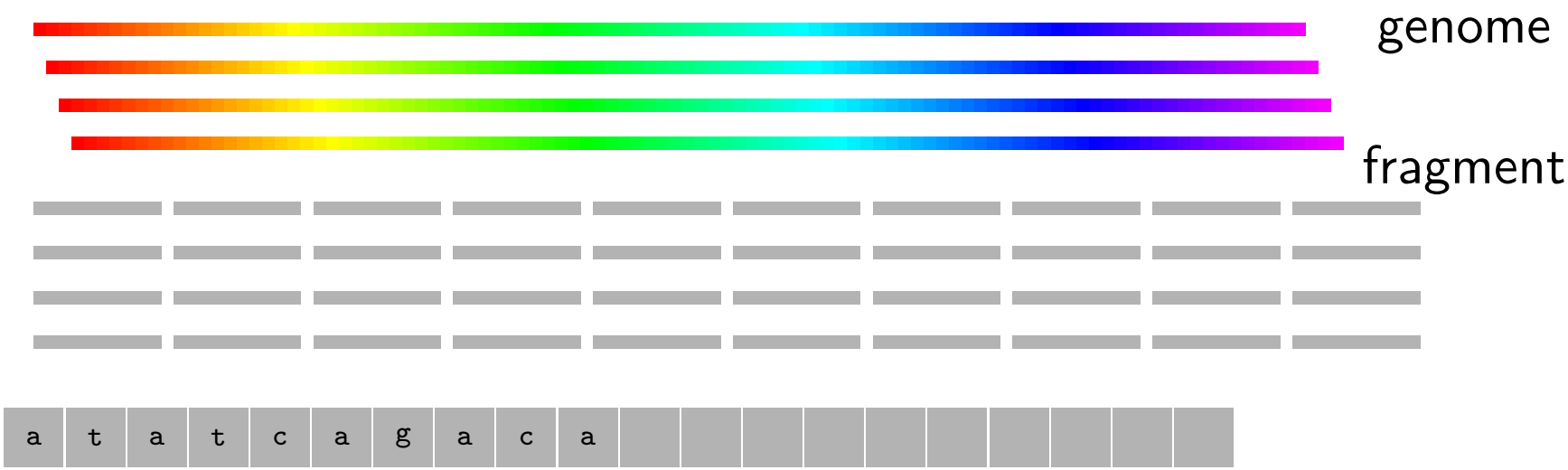
The overlap problem in read assembly



The overlap problem in read assembly



The overlap problem in read assembly



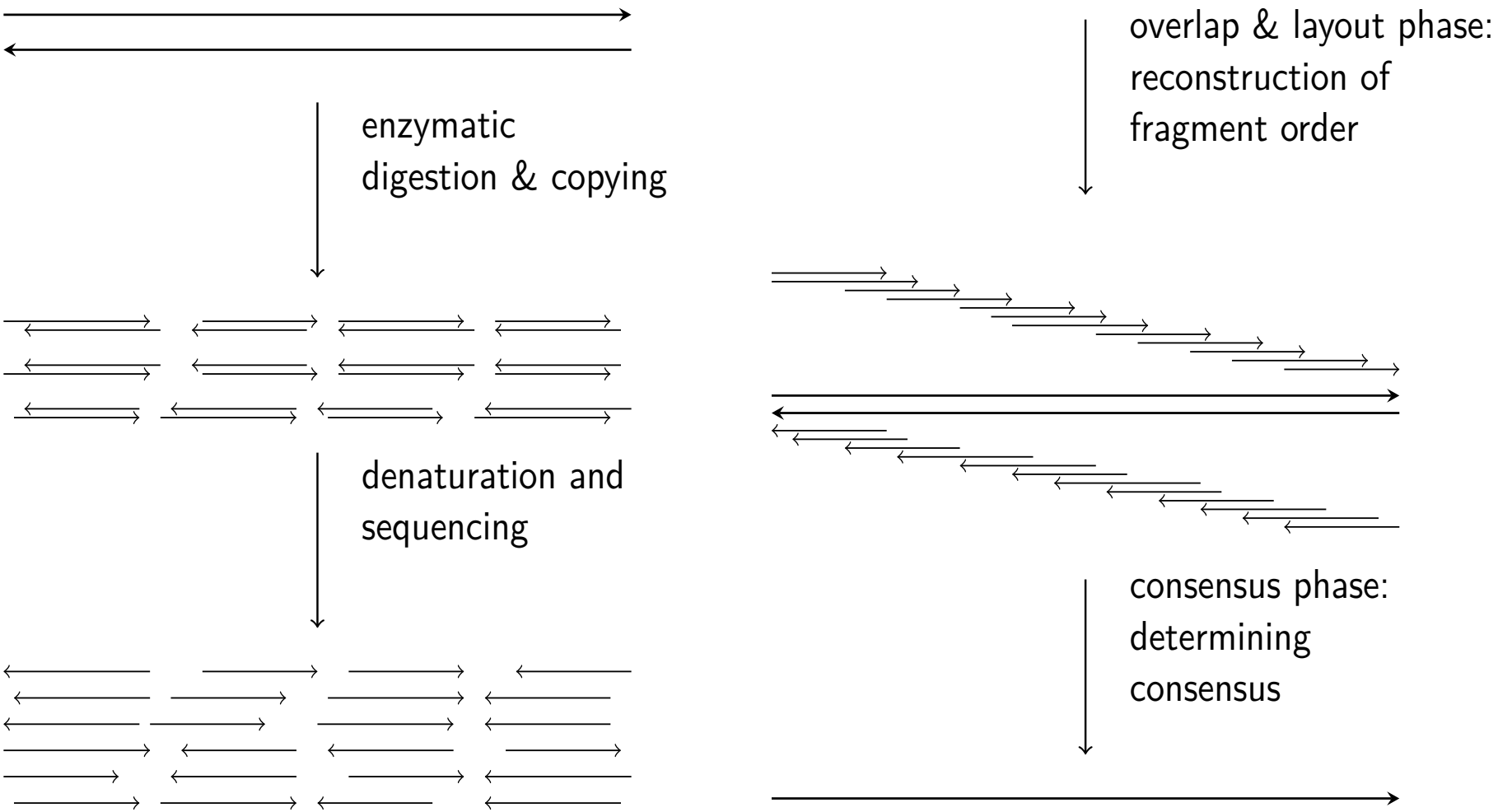
- genome lengths: $\approx 5 \cdot 10^6$ bp. (bacteria); $\approx 3 \cdot 10^9$ bp. (human genome)
- fragment length: $\approx 10^2 - 10^4$
- mean coverage (# sequenced fragment positions/#genome positions): $40 - 80$

$\Rightarrow$ $\frac{5 \cdot 10^6 \cdot 40}{10^2} = 2 \cdot 10^6$ fragments for sequencing a bacterial genome of average length (for 100 bp. fragments and $40\times$ coverage)

The overlap problem in read assembly

- The shotgun sequencing technique delivers many different reads.
- For each read we know the sequence of bases it consists of, but we do not know (*a*) the direction of the read (i.e. is it on the forward or reverse complementary strand?), and (*b*) the start position of the reads relative to the start of the corresponding strand.
- The *read assembly problem* now consists of determining the entire target sequence, by assembling the given reads in the correct orientation and order.
- The different phases of the shotgun assembly process are depicted in Figure 1.

Figure 1: The phases of the shotgun read assembly process



Example 1

Consider the read assembly problem for the following 4 reads:

```
ACCGT
CGTGC
TTAC
TACCGT
```

Assume we know that the target sequence has length 10. Then a possible assembly (shown as a multiple alignment) is as follows:

```
--ACCGT--
----CGTGC
TTAC-----
-TACCGT--
TTACCGTGC
```

The overlap problem in read assembly

The read assembly process can be divided into different phases:

- **Overlap phase:**
  - Determine suffix/prefix overlaps for all pairs of reads.
  - If suffix of one read is similar to a prefix of another read, the two reads are likely from the same regions of the DNA.
  - In Example 1, there is an overlap of length 3 between sequences `ACCGT` and `CGTGC`.
- **Layout phase:**
  - determine a multiple alignment, derived from the positioning of the different reads above each other, and from the introduction of gap symbols.
- **Consensus phase:**
  - determine the target sequence by considering the different columns of the multiple alignment.
  - Choose the base which is dominating a column (majority choice).

The overlap problem in read assembly

In this section, we will focus on the overlap phase. The other phases will be considered in the Genome Informatics lecture.

- Note that the reads delivered by the sequencers are not always exact.
- The amount of errors (i.e. the number inserted, deleted and replaced bases) in the reads varies with the sequencing technology, see the following table:

instrument	primary error	errors per bp
Illumina (all models)	substitution	≈ 0.001
Ion Torrent (all chips)	indel	≈ 0.01
Oxford Nanopore	deletions	≥ 0.04
Pacific Biosciences RS	CG deletions	≈ 0.15

source: `http://www.molecularecologist.com/next-gen-table-3c/`

The overlap problem in read assembly

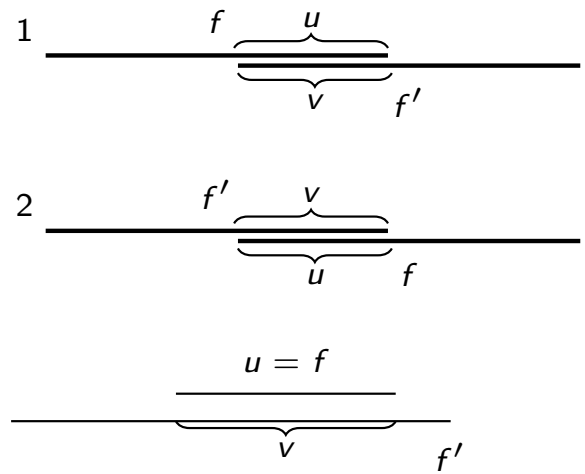
- To handle these errors in the reads we therefore have to compute approximate overlaps.
- Consider two reads f and f', which stem from overlapping parts of the sequenced DNA.
- Suppose a fixed error rate ϵ, $0 \leq \epsilon \leq 1$, i.e. ϵ is the number of errors per base of the read.
- By assumption, f can be derived from from the original by at most $\epsilon \cdot |f|$ errors.
- The corresponding holds for f'.
- As in the worst case, all errors may be in the overlapping parts of the reads, the overlapping part of f can be transformed with at most $k = \epsilon \cdot |f| + \epsilon \cdot |f'| = \epsilon \cdot (|f| + |f'|)$ edit operations into the overlapping part of f'.
- So for reads of the same length, say m, we allow $2m\epsilon$ errors, i.e. 2ϵ errors per base.
- Hence the following definitions make sense:

The overlap problem in read assembly

Definition 1

An alignment A is an *overlap of f and f'*, if there are substrings u of f, v of f', A is an alignment of u and v and at least one of the following properties holds:

1. u is a suffix of f and v is a prefix of f'
2. u is a prefix of f and v is a suffix of f'
3. $u = f$ and v is a substring of f'
4. u is a substring of f and $v = f'$



$len(A) = (|u| + |v|)/2$ is the length of the overlap A. Two reads f and f' *overlap with an error rate* ϵ if and only if there is an overlap A of f and f' such that $\delta(A) \leq k$, where $k = \epsilon \cdot (|f| + |f'|)$.

The overlap problem in read assembly

- The dynamic programming techniques developed earlier can be modified to solve the overlap problem.
- For a read f of length m and a read f' of length n, we compute an $(m+1) \times (n+1)$-table Ov_δ, such that for all (i,j), $0 \leq i \leq m$, $0 \leq j \leq n$ we have

$$Ov_\delta(i,j) = \min\{edist_\delta(u,v) \mid u \text{ is a suffix of } f[1 \ldots i] \text{ and } v = f'[1 \ldots j] \text{ or } u = f[1 \ldots i] \text{ and } v \text{ is a suffix of } f'[1 \ldots j]\}$$

The overlap problem in read assembly

- One can show that there is an overlap of f and f', if and only if either

1. $Ov_\delta(i, n) \leq k$ for some i, $0 \leq i \leq n$ or
2. $Ov_\delta(m, j) \leq k$ for some j, $0 \leq j \leq m$.

- To compute table Ov_δ, we have to consider that prefixes or suffixes of the reads can be deleted or inserted at no costs.
- Hence, one needs to set the first row and first column of Ov_δ to 0 and derives the following recurrences:

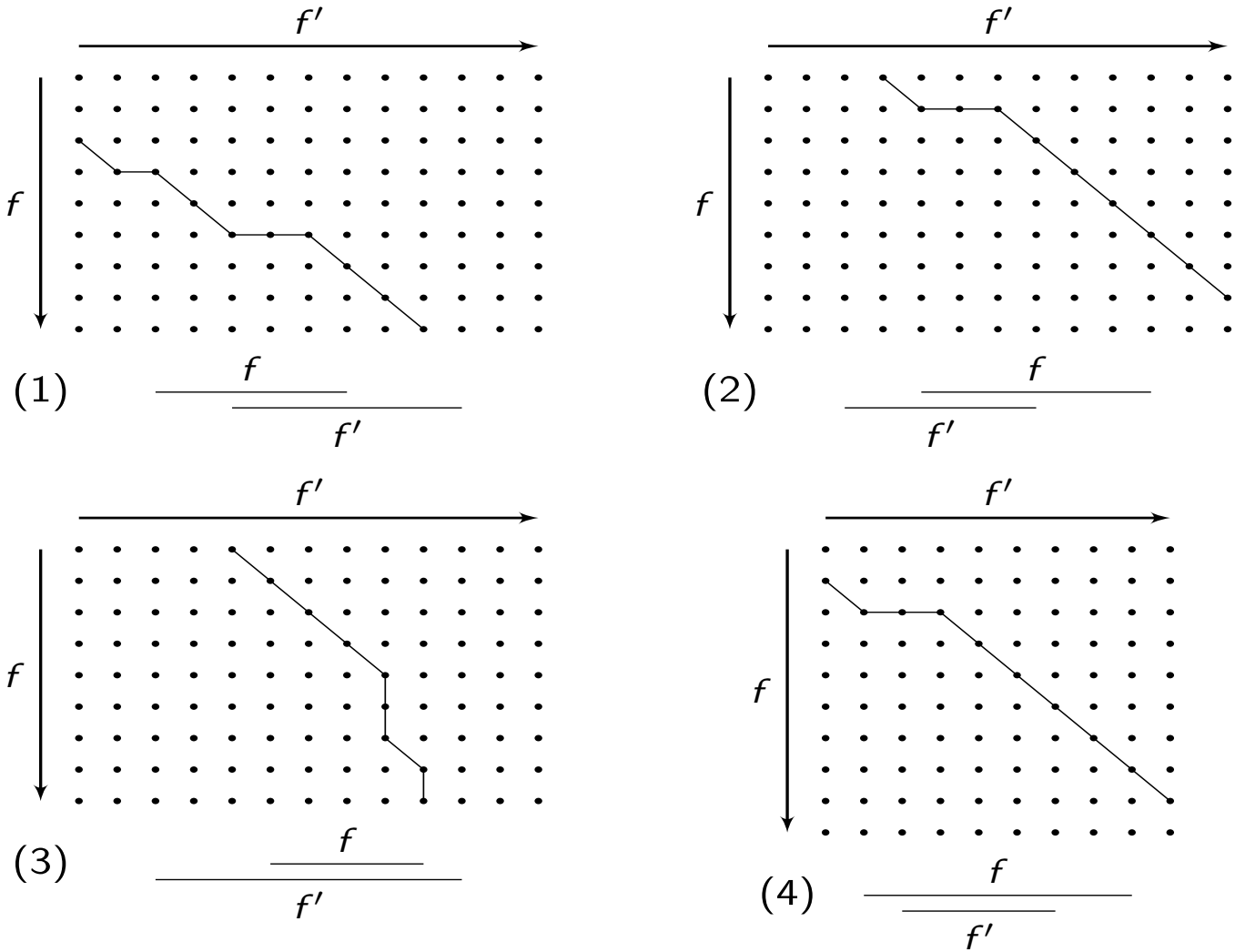
$$Ov_\delta(i,j) = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ \min \left\{ \begin{array}{ll} Ov_\delta(i-1,j) & + \delta(f[i] \to \varepsilon) \\ Ov_\delta(i,j-1) & + \delta(\varepsilon \to f'[j]) \\ Ov_\delta(i-1,j-1) & + \delta(f[i] \to f'[j]) \end{array} \right\} & \text{otherwise} \end{cases}$$

The overlap problem in read assembly

- The four different forms of overlaps between f and f' correspond to four different kinds of paths in the edit graph $G(f, f')$.
- Each path begins in the first row or first column and it ends in the last row or last column.
- See the following table and Figure 2, to which the column with header *example* refers.
- Note that $m = |f|$ and $n = |f'|$.

type	example	start position	end position
suffix of f with prefix of f'	(1)	$(i', 0)$	(m, j)
prefix of f with suffix of f'	(2)	$(0, j')$	(i, n)
f with substring of f'	(3)	$(0, j')$	(m, j)
substring of f with f'	(4)	$(i', 0)$	(i, n)

Figure 2: The paths from a node in the first row or first column of $G(f, f')$ to a node in the last row or last column represents the overlaps between f and f'. Different forms of paths represent different forms of overlaps.



The overlap problem in read assembly

Besides the number of errors one also requires that the length $len(A)$ of the overlap A of f and f' is at least τ, where τ is some parameter. To determine $len(A)$ one maintains the following information:

- for each overlap beginning in the first column, one keeps track of the row from which it starts, minimizing the row number if in the edit graph there is more than one minimizing edge into the current node.
- for each overlap beginning in the first row, one keeps track of the column from which it starts, minimizing the column number if in the edit graph there is more than one minimizing edge into the current node.

The *minimizing choice* in these cases maximizes the overlap length. As we only consider values in the last row and last column of matrix Ov_δ we can determine the length of the start and end positions of the string involved in the overlap A. This allows to compute $len(A)$ as to verify that $len(A) \geq \tau$ holds.

Figure 3: A prefix of f matches a substring of f'. So the minimizing path begins in the first row at column j'.

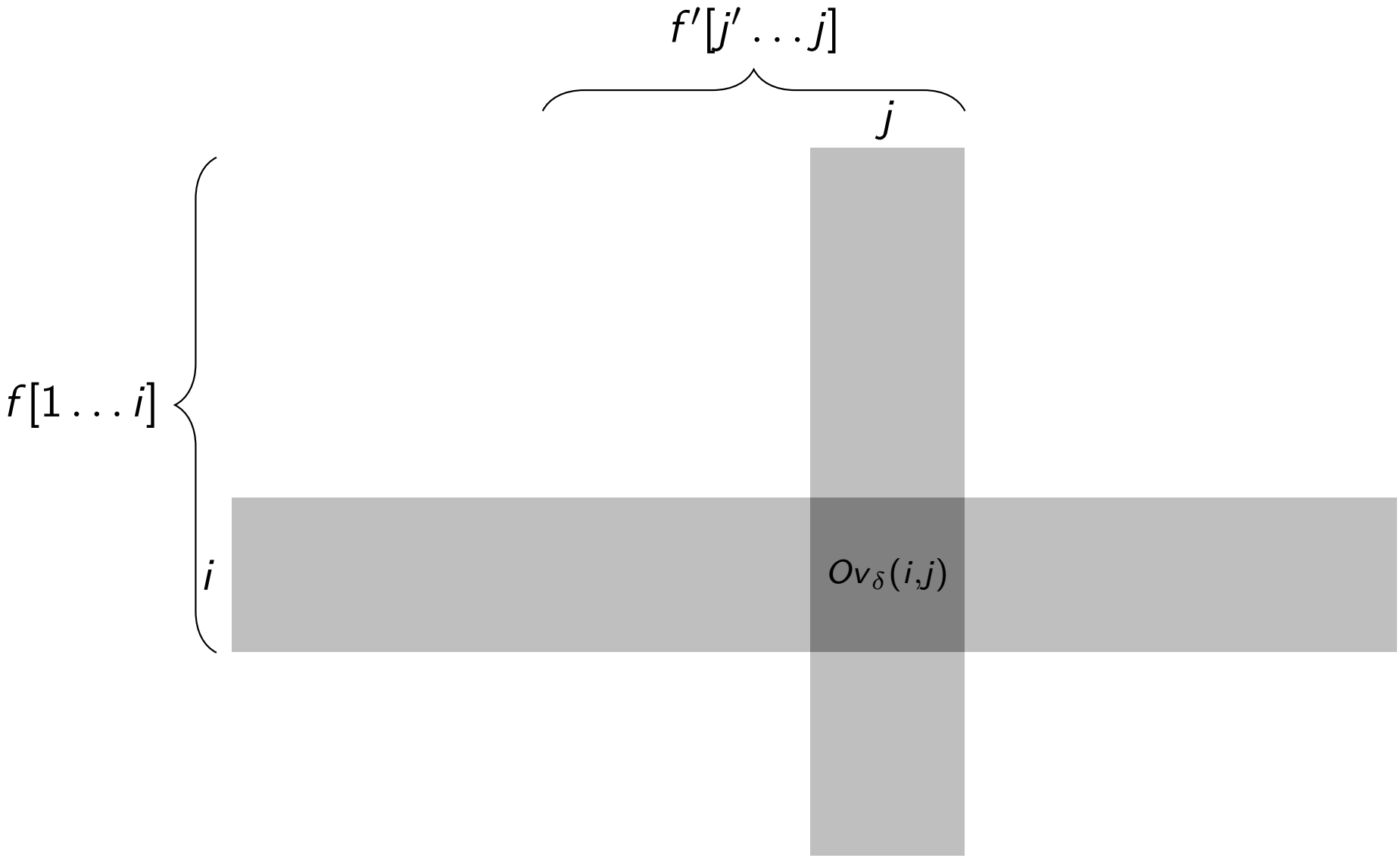


Figure 4: A prefix of f' matches a substring of f. So the minimizing path begins in first column and row i'.