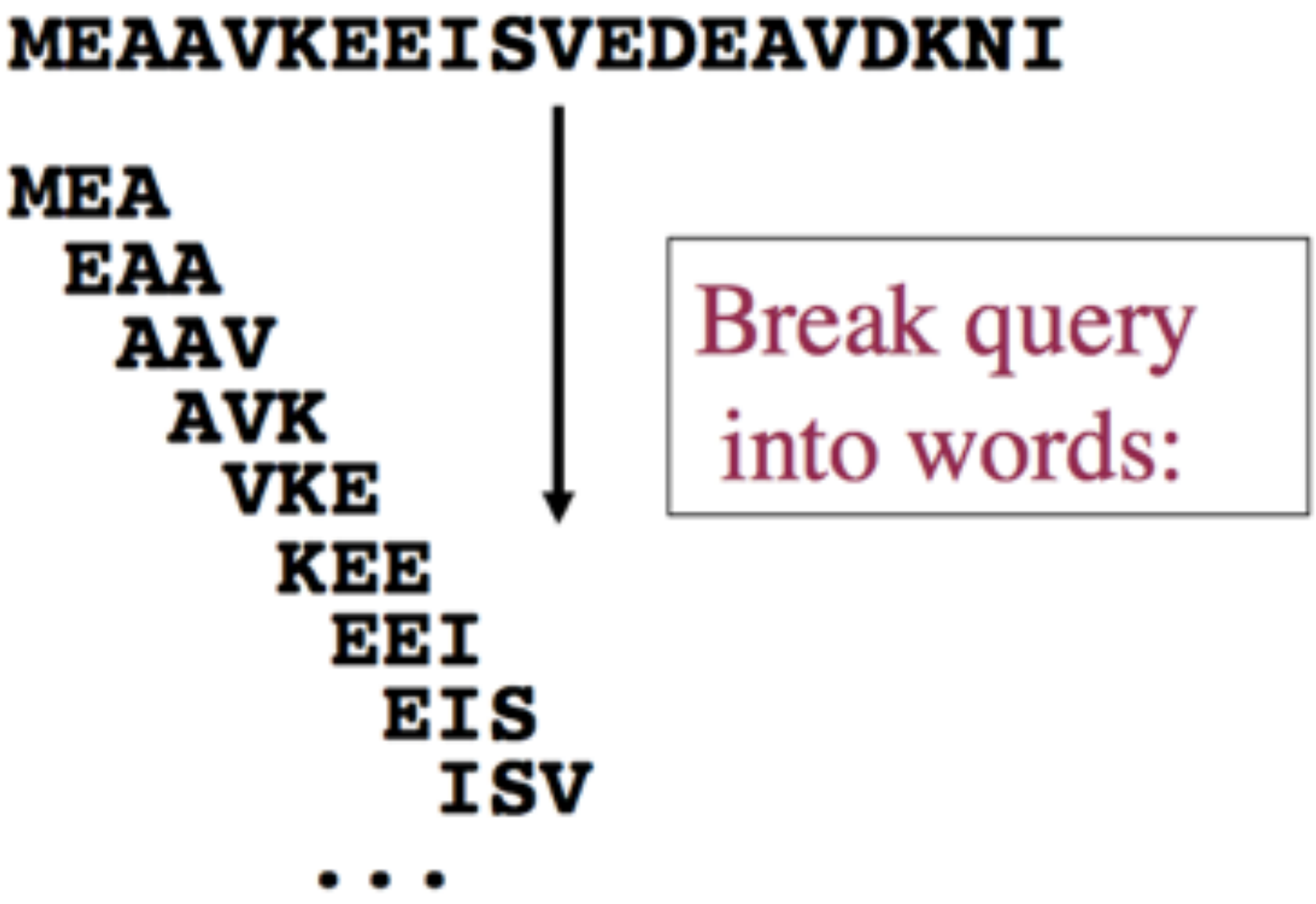
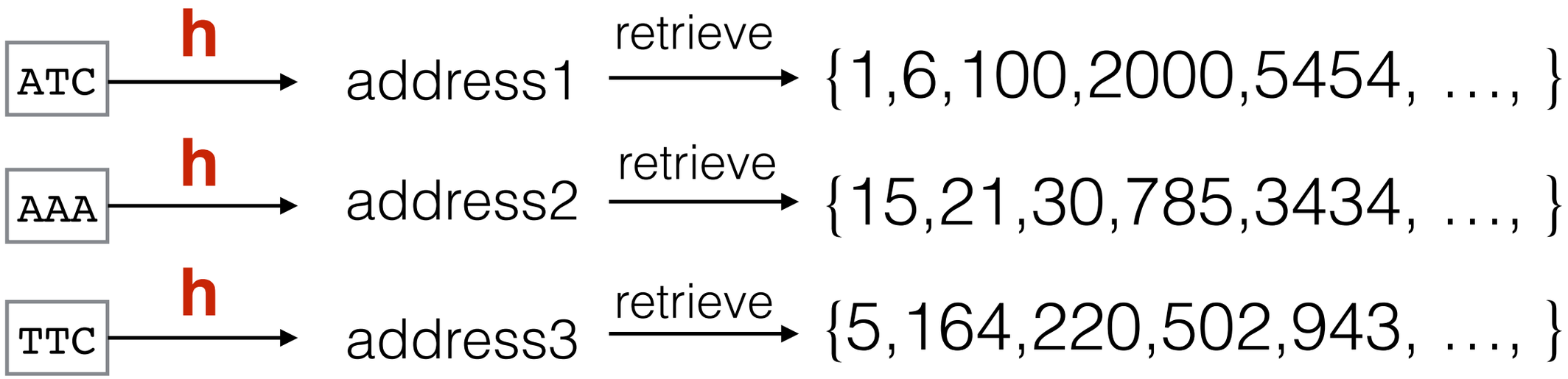
1. Break query sequence into words

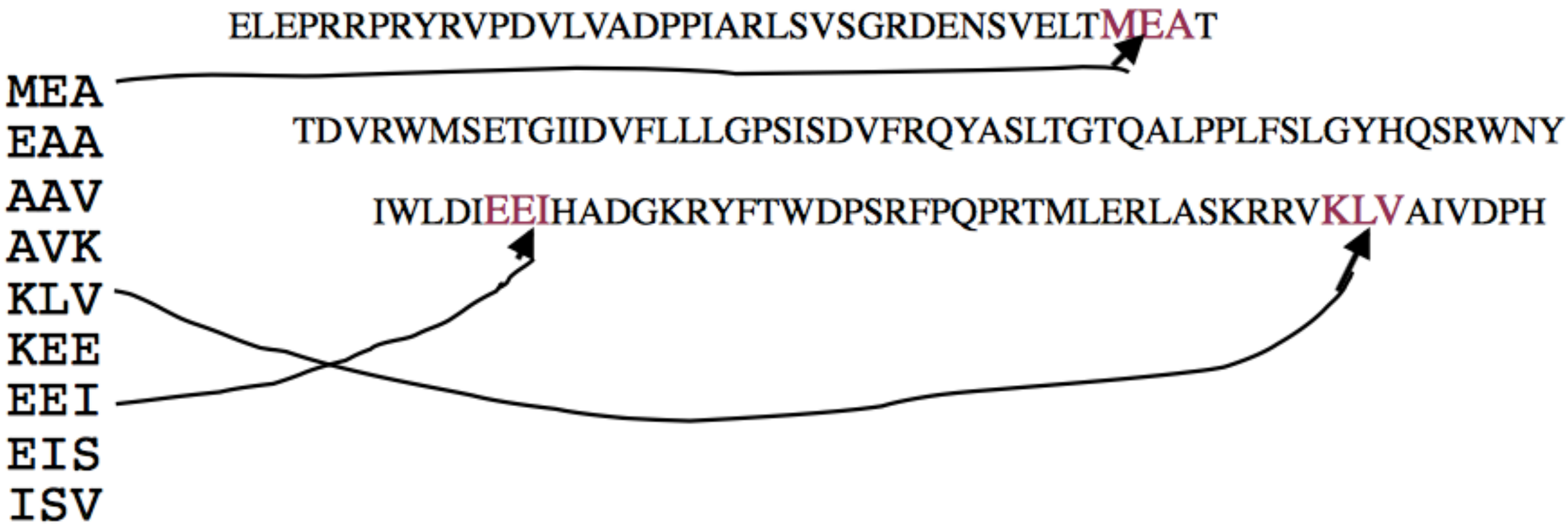


2. Find database hits

- Find exact matches to query words
- Can be done in efficiently
  - Hashing
  - Alternatively AC finite state machine



2. Find database hits



3. Extend hits

1. Find “seeds” (initial matches) of a fixed length (e.g. 11)
2. Try extending an alignment from each seed

How to handle possible mismatches in words?

MVRERKCILCHIVYGSKKEMDEHMRSMLHHRELENLKGRDIS

Query word, W=3 for proteins ↓

(W=11 for nucleotides)

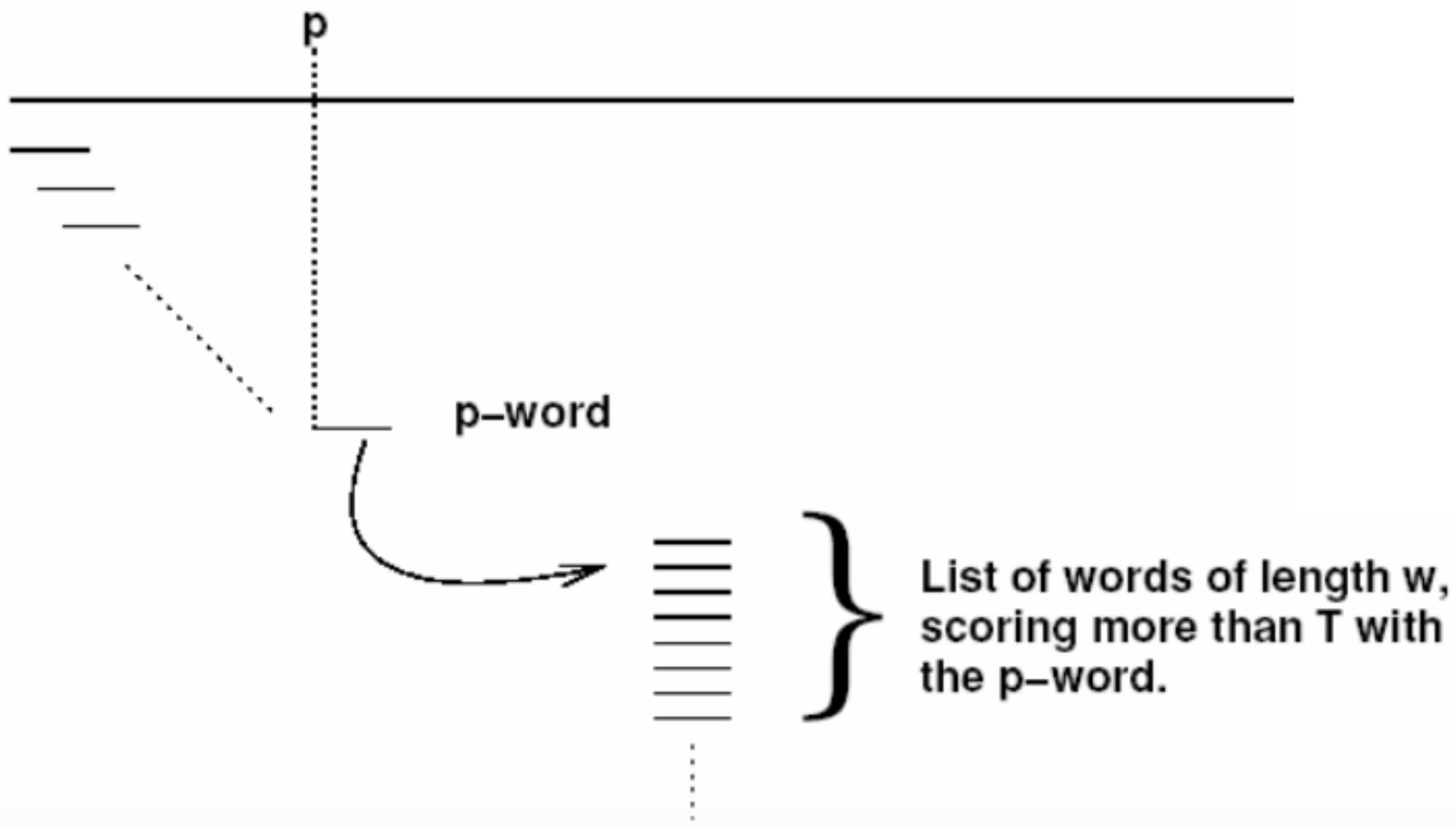
Neighbor words

Word	Score (BL-62)
GSK	15
GAK	12
GNK	12
GTK	12
GSR	12
GDK	11
GQK	11
GEK	11
GGK	11
GKK	11
GSQ	11
GSE	11

How to handle possible mismatches in words?

First step:

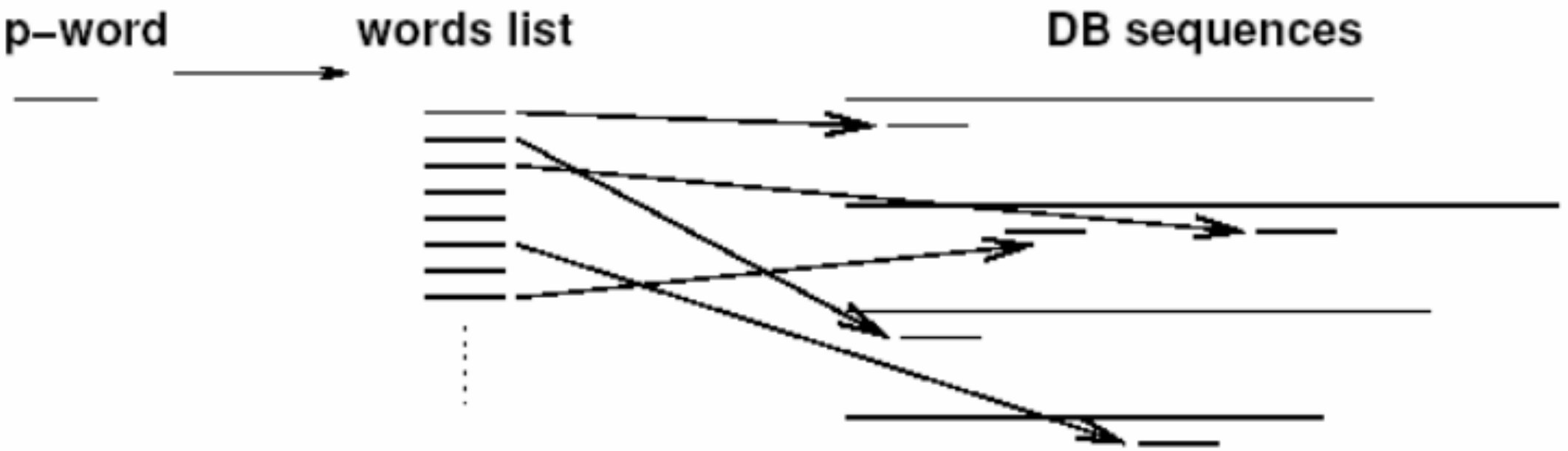
For each position p of the query, find the list or words of length w scoring more than T when paired with the word starting at p:



How to handle possible mismatches in words?

Second step:

For each words list, identify all exact matches with DB sequences:



How to handle possible mismatches in words?

Third step:

For each word match («hit»), extend ungapped alignment in both directions. Stop when S decreases by more than X from the highest value reached by S.

HSP = High Scoring Segment Pair

Statistics: Question

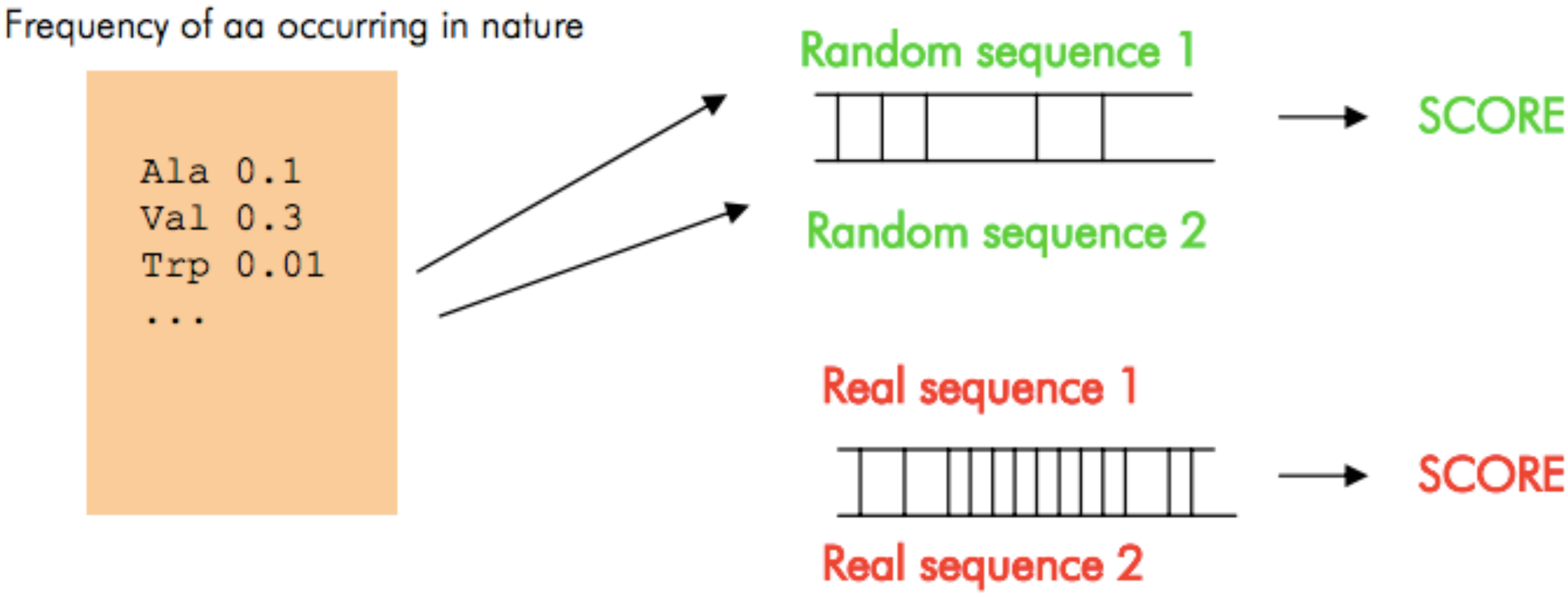
- Given two random sequences of lengths m and n
- What is the probability that they will produce an MSP score of >= S ?

Statistics: more intuition

The probability will depend on:

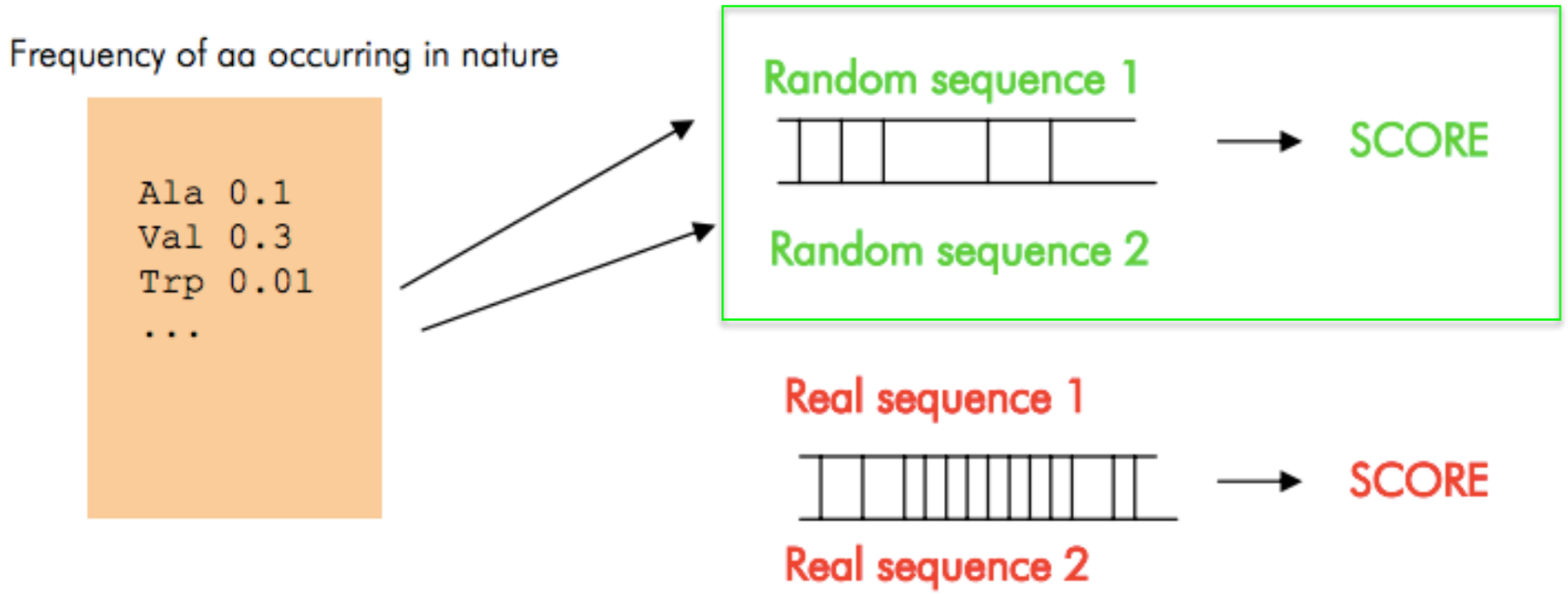
- How long is are the sequences (the longer the easier to get a local score above threshold by chance)
- Scoring matrix
- Distribution of amino acids in each sequence

Statistics: Intuition

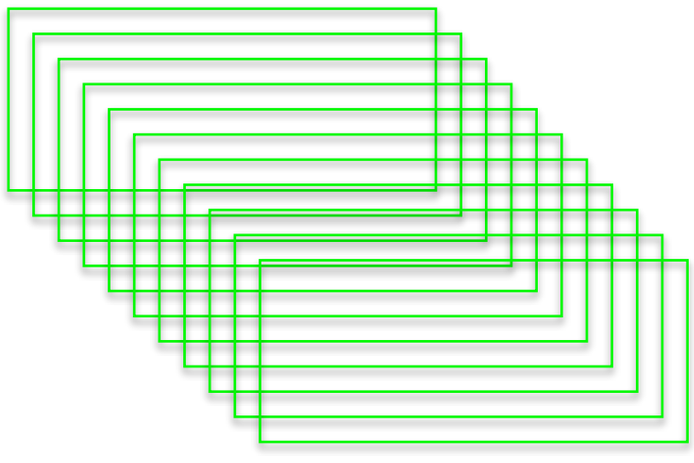


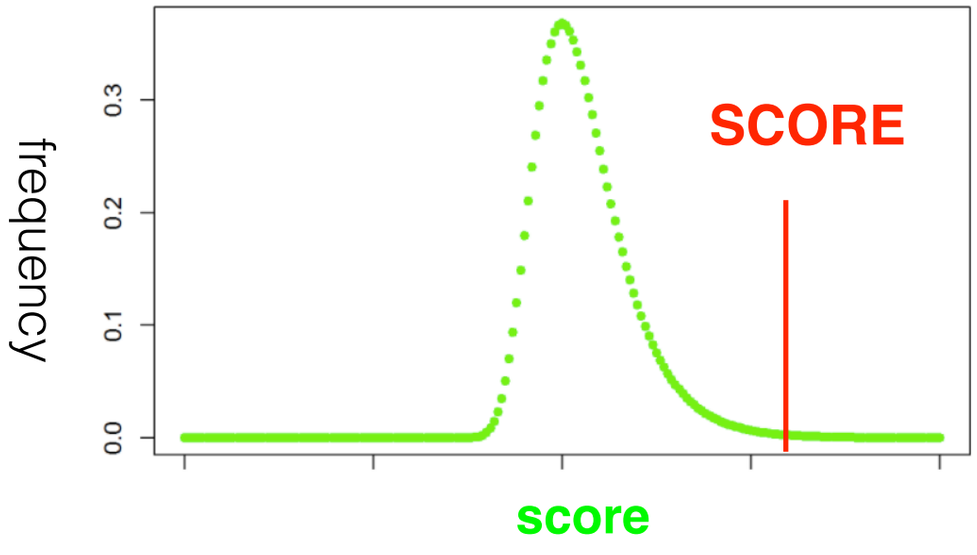
Simulation

1. Generate many random sequence pairs



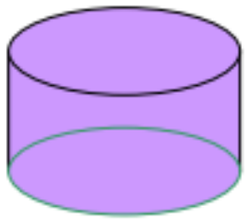
2. Compute the distribution of the SCOREs

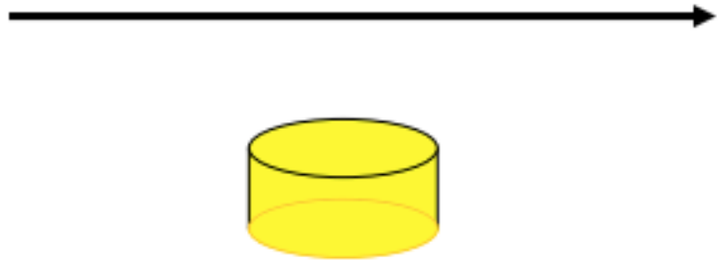




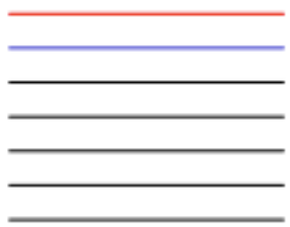
Statistical test

sequence DB





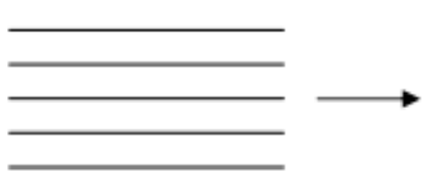
Hits list

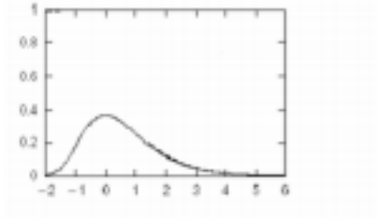


Score A

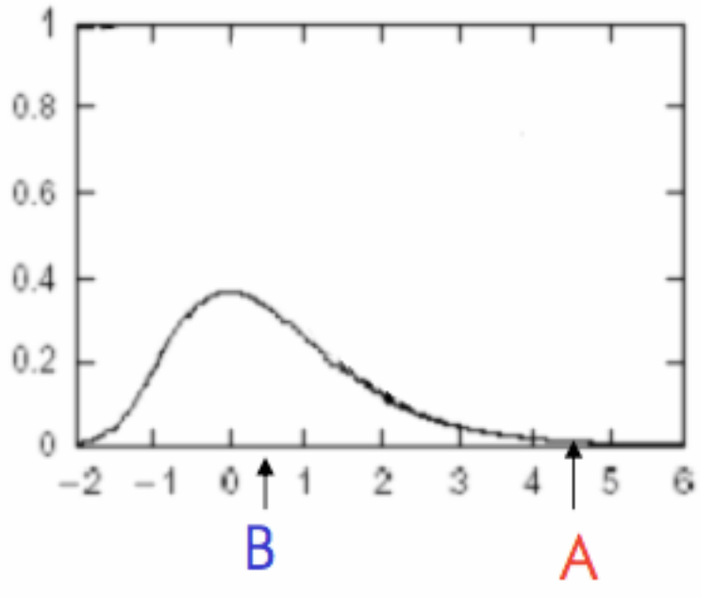
Score B

Simulation





Random DB (smaller)



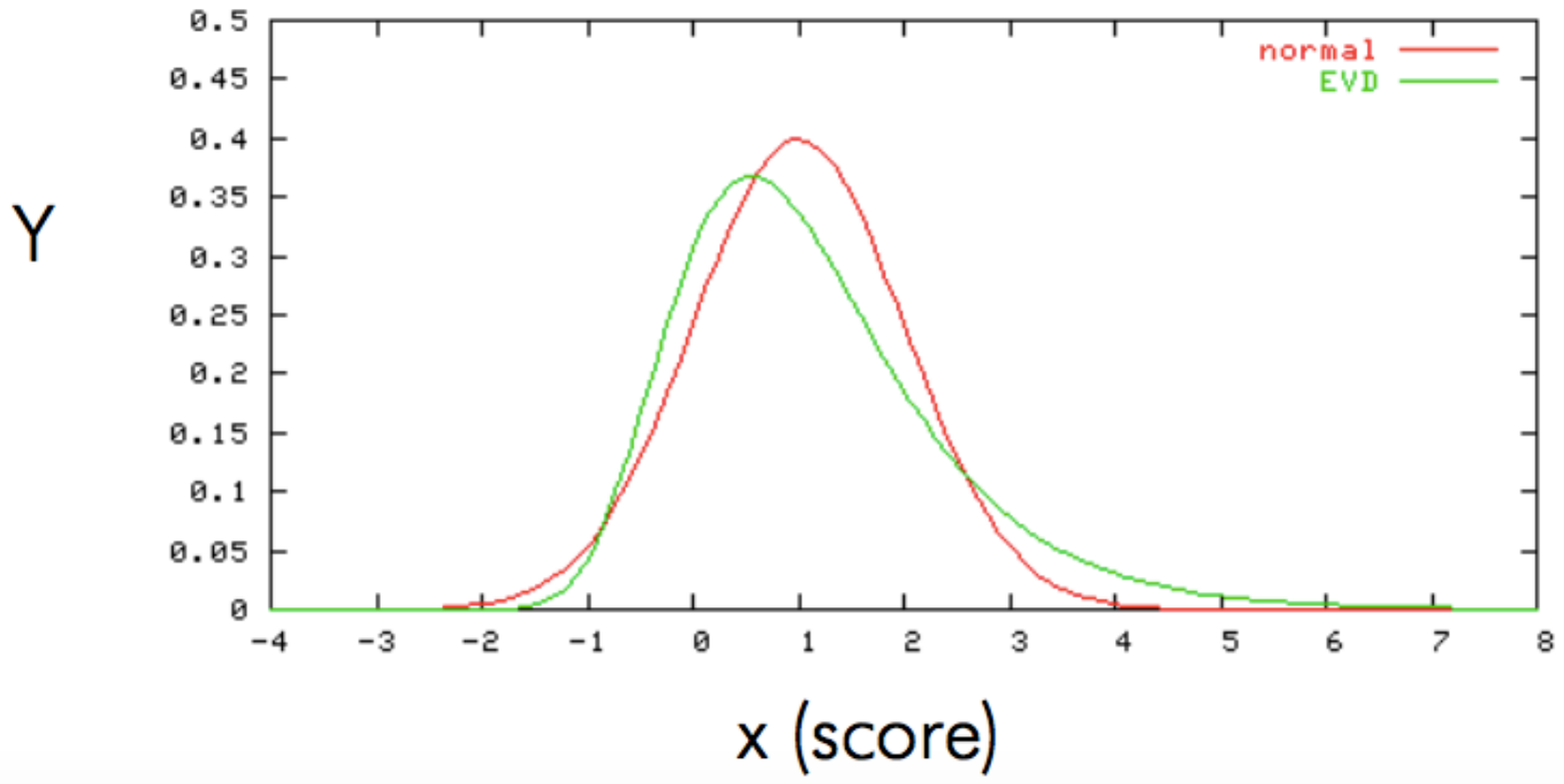
Score A: is significant

Score B: is NOT significant

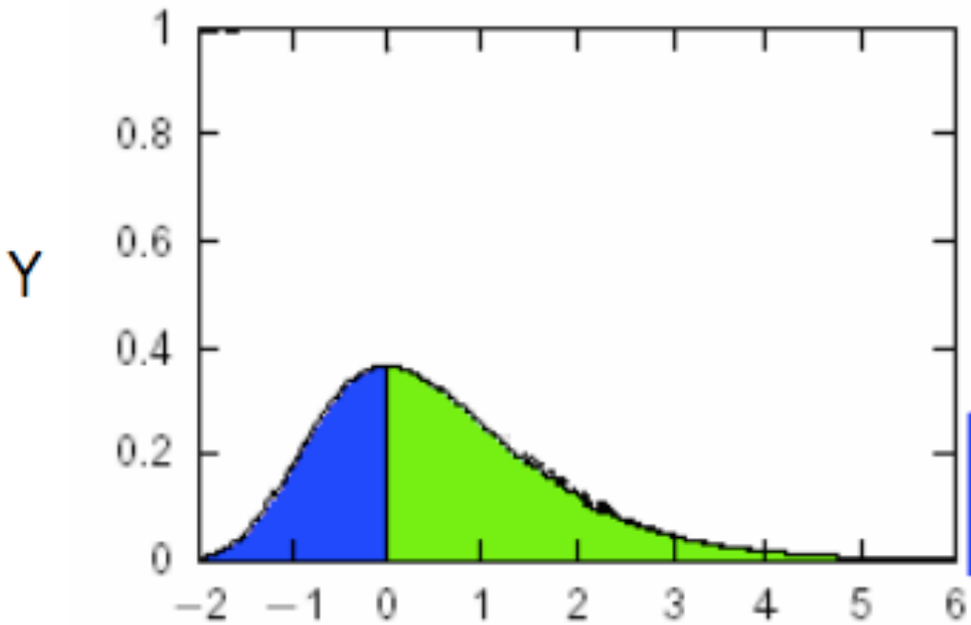
p-value = 0.45 p-value = 0.001

Extreme value distribution

Karlin and Altschul observed that in the framework of local alignments without gaps: the distribution of random sequence alignment scores follow an EVD.



Extreme value distribution



$$Y = \lambda \exp[-\lambda(x-\mu) - e^{-\lambda(x-\mu)}]$$

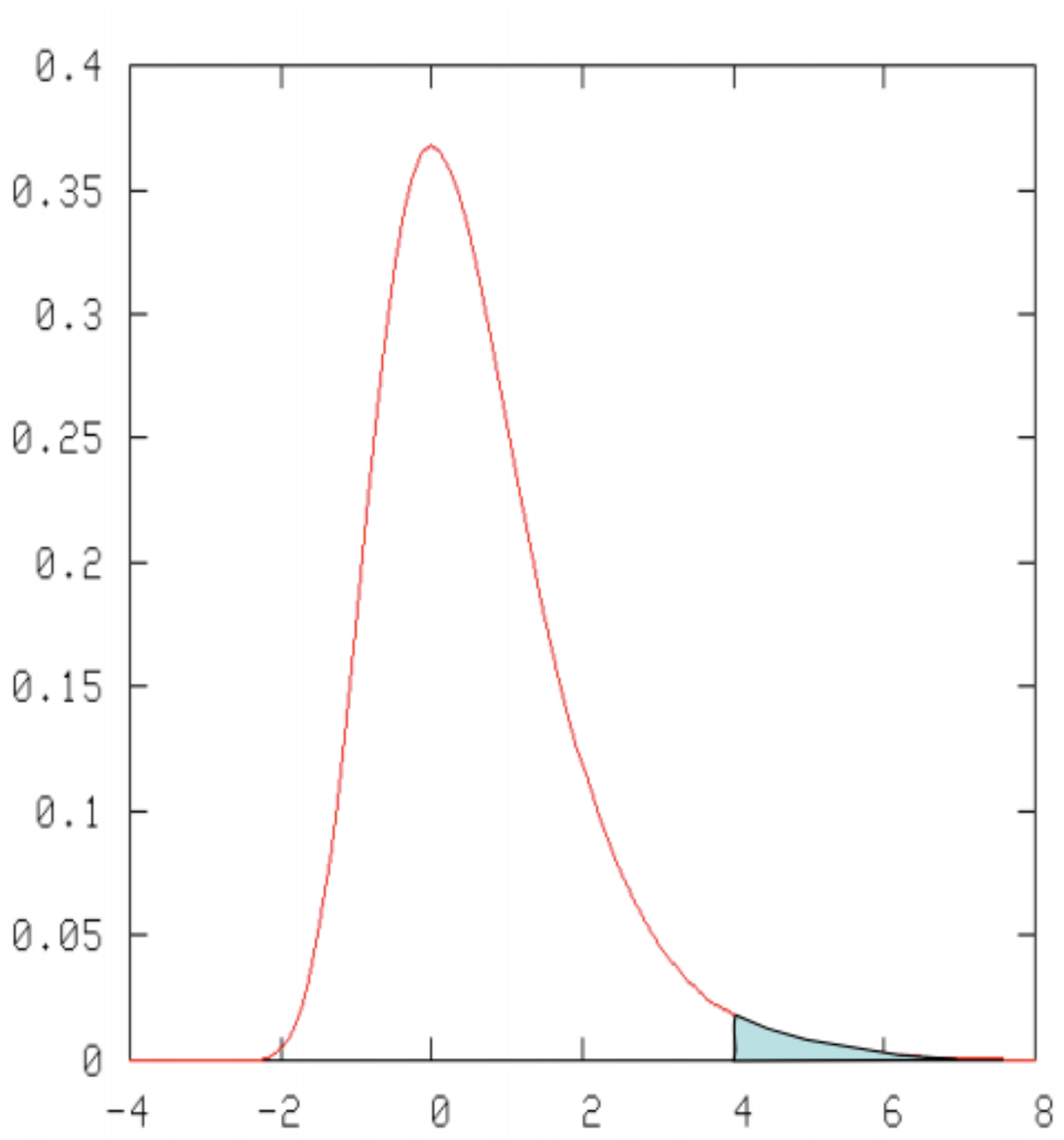
$$\int$$

$$P(Z < x) = \exp[-e^{-\lambda(x-\mu)}]$$

$$P(Z \geq x) = 1 - \exp[-e^{-\lambda(x-\mu)}]$$

P-value = the probability of obtaining a score equal or greater than *x* by chance

Compute a p-value



- The probability of observing a score >=4 is the area under the curve to the right of 4.
- <u>For an *Unscaled EVD:*</u>

$$P(S \geq x) = 1 - e^{(-e^{-x})}$$

$$P(S \geq 4) = 1 - e^{(-e^{-4})}$$

$$P(S \geq 4) = 0.018149$$

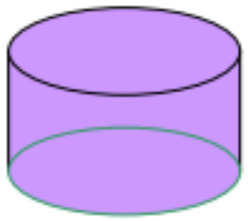
Parameters

$$P(Z \geq x) = 1 - exp[-e^{-\lambda(x-\mu)}] \qquad (1)$$

μ, λ : parameters depend on the length and composition of the sequences and on the scoring system: μ is the mode (highest point) of the distribution and λ is the decay parameter
-They can me estimated by making many alignments of random or shuffled sequences.

Statistical test

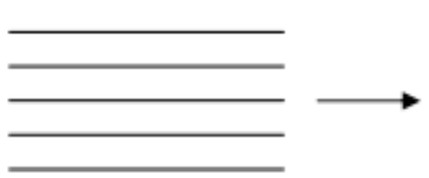
sequence DB



Hits list

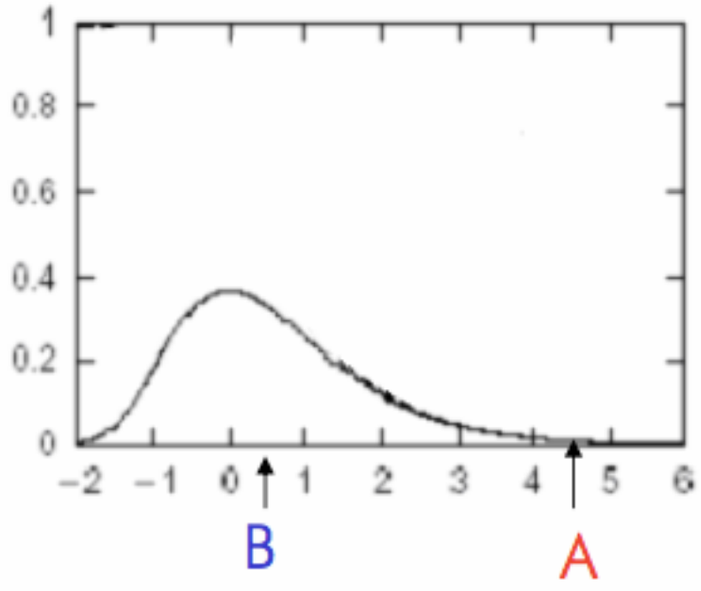
Score A

Score B



EVD

Random DB (smaller)



Score A: is significant

Score B: is NOT significant

p-value = 0.45 p-value = 0.001

Significance: P-value and E-value

In a database of size N: P x N = E

- P-value:

Probability that an alignment with this score occurs by chance in a database of size N.

The closer the P-value is towards 0, the better the alignment

- E-value:

Number of matches with this score one can expect to find by chance in a database of size N.

The closer the E-value is towards 0, the better the alignment

→ Smaller E-value, more significant in statistics

Bigger E-value , by chance

E[# occurrences of a string of length m in reference of length L] ~ $L/4^m$

Parameters

$$P(Z \geq x) = 1 - exp[-e^{-\lambda(x-\mu)}] \qquad (1)$$

μ, λ : parameters depend on the length and composition of the sequences and on the scoring system: μ is the mode (highest point) of the distribution and λ is the decay parameter
-They can me estimated by making many alignments of random or shuffled sequences.
- For alignments without gaps they can be calculated from the scoring matrix and then :

$$P(Z \geq x) = 1 - exp[-Kmne^{-\lambda x}] \qquad (2)$$

K: is a constant that depend on the scoring matrix values and the frequencies of the different residues in the sequences.
m,n : sequence lengths

E-value

Approximation:

if x is very small, then 1-exp(-x) can be approximated by x

Therefore,

$$P(Z>=x) \sim e^{-\lambda(x-\mu)} = Kmn\, e^{-\lambda x}$$

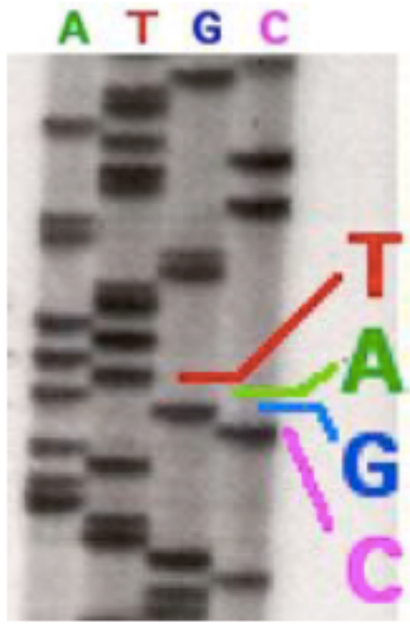
So E-value = DatabaseLength * p-value

$$\text{E-value} = KNme^{-\lambda x}$$

where N is the database size (not the aligned length n)

Genomics

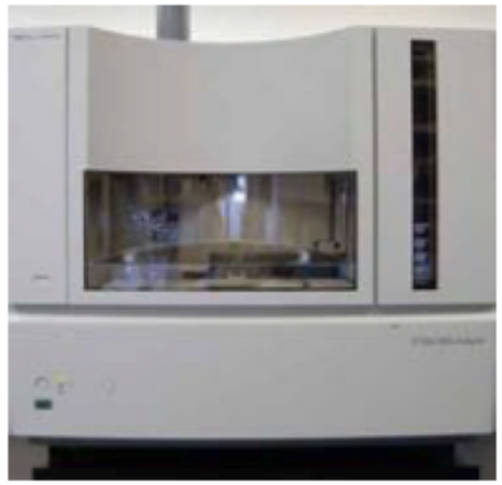
Sequencing tech



1970s: 0th Gen

Radioactive Chain Termination

5000bp / week



1980s-1990s: 1st Gen

Automated Capillary Sequencing

384kbp / day

2000s: 2nd Gen

Pyrosequencing, SOLiD Sequencing-by-Synthesis

1Gbp+ / day

Sequencing tech: next generation

Illumina HiSeq 2000
Sequencing by Synthesis

>60Gbp / day
100bp reads

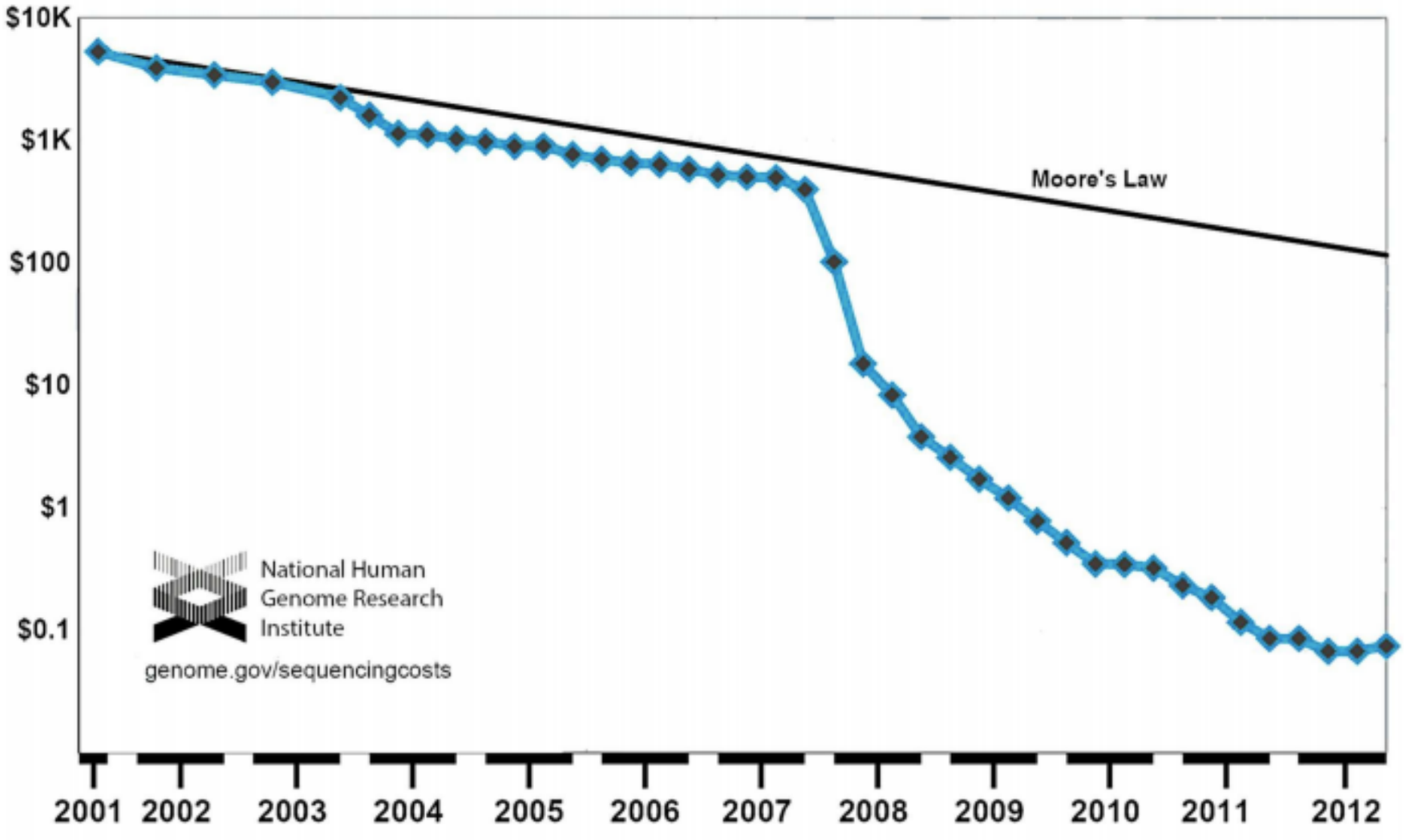
PacBio
SMRT-sequencing

~1Gbp / day
Long Reads

Oxford Nanopore
Nanopore sensing

Many GB / day?
Very Long Reads?

Cost per raw megabase of DNA sequence



Until 2007: Sanger sequencing

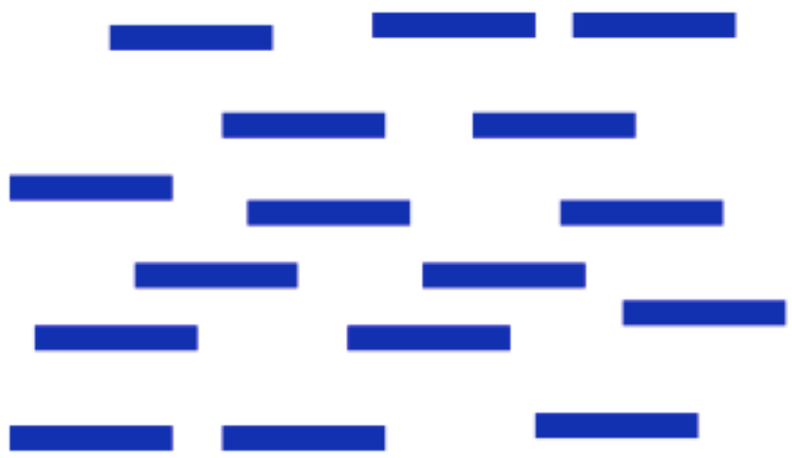
Starting in 2008: next-generation (454, Illumina, SOLiD)

What do we get from sequencing?

Sequencing technologies produce short **reads** from random locations in the DNA sample

How to analyze these reads?

Position of individual reads on the target DNA is not known



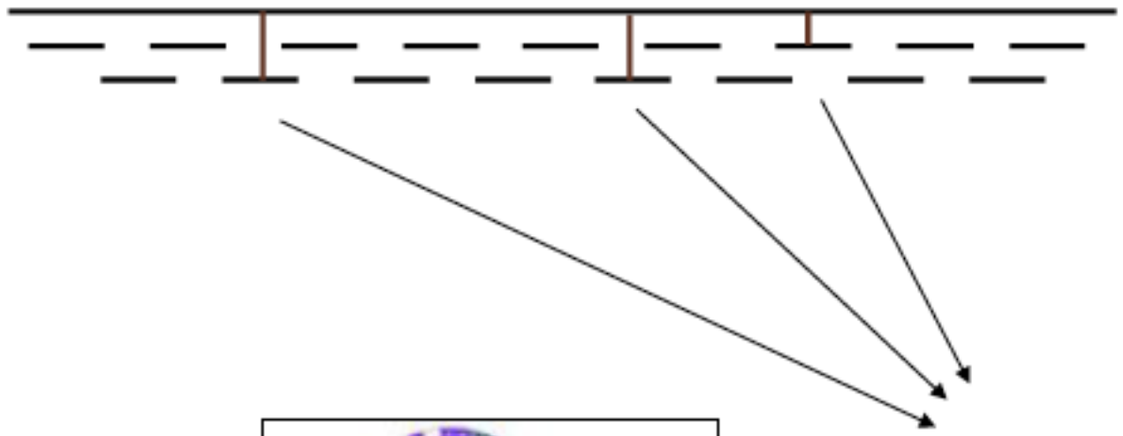
Solved by computational methods:

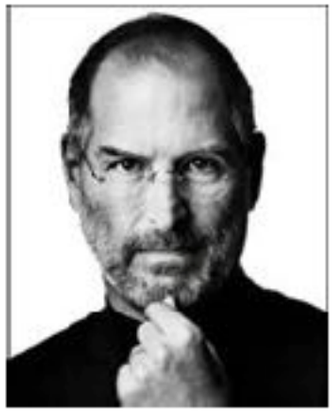
– **mapping** if target DNA is known

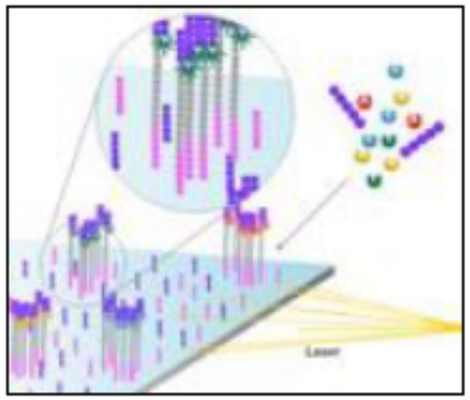
– **assembly** if it is not known

Mutation identification: Mapping

How does your genome compare to the reference?







Cancer
Heart Disease
Brain Disease

Genome projects: Assembly

1976	MS2 (RNA virus) 40 kB
1988	**Human genome sequencing project (15 years)**
1995	bacterium H. influenzae 2 MB, shotgun (TIGR)
1996	S. cerevisiae 10 MB, BAC-by-BAC (Belgium, UK)
1998	C. elegans 100 MB, BAC-by-BAC (Wellcome Trust)
1998	**Celera: human genome in three years!**
2000	D. melanogaster 180 MB, shotgun (Celera, Berkeley)
2001	**2x human genome 3 GB (NIH, Celera)**
after 2001	mouse, rat, chicken, chimpanzee, dog,. . .
2007	**Genomes of Watson and Venter (454)**

Use sequencing for other types of data

Molecular Biology

Mathematics & Computer Science

Computational Biology

Desired measurement → **reduce to sequencing** → Sequence → Solve inverse problem → Analyze

Creativity

Chemistry & Physics

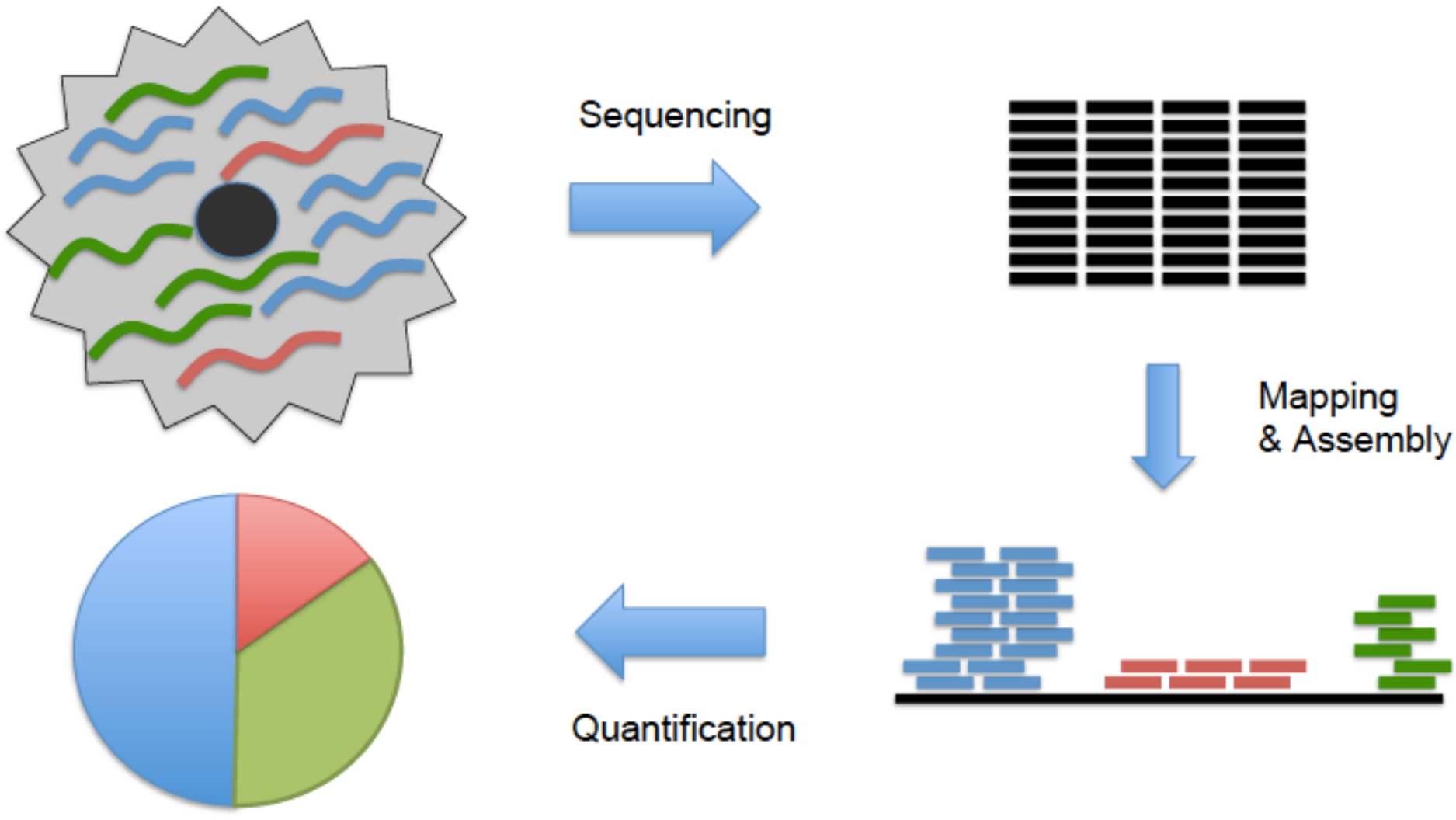
Statistics

Biology

$$\mathbb{P}(f=(p,t,l)) \approx \frac{\lambda_l \cdot \frac{\tau_t}{l(t)} \cdot w_{p|t,l} \cdot \phi_{f|p,t,l}}{\sum_{(q,r,m)\in\mathcal{A}(f)} \lambda_m \cdot \frac{\tau_r}{l(r)} \cdot w_{q|r,m} \cdot \phi_{f|q,r,m}}$$

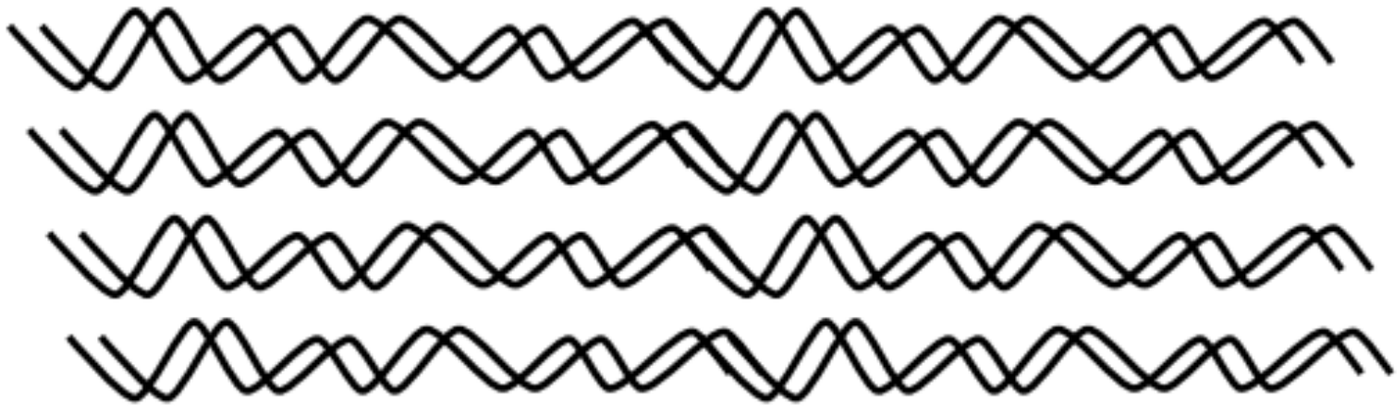
X-seq technology

RNA-seq

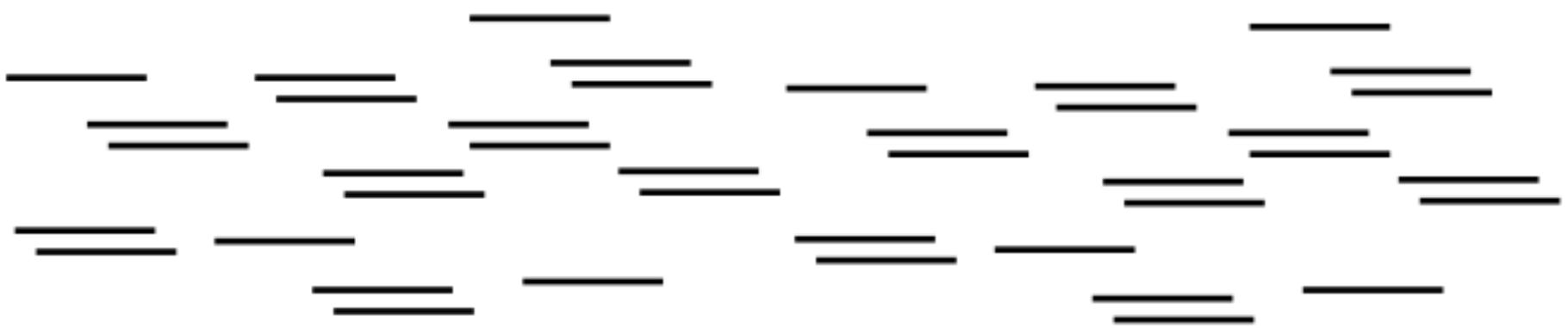


Assembly

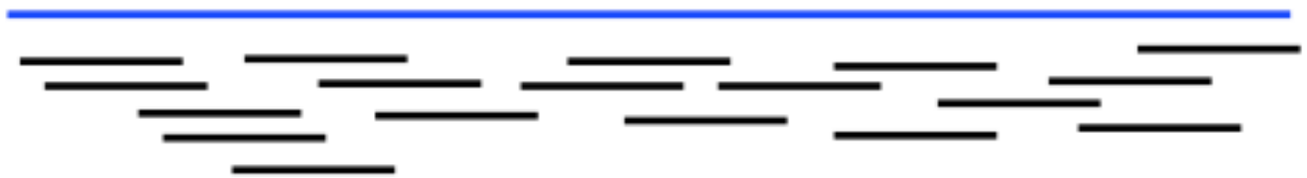
Many copies of the DNA



Shear it, randomly breaking them into many small pieces, read ends of each:



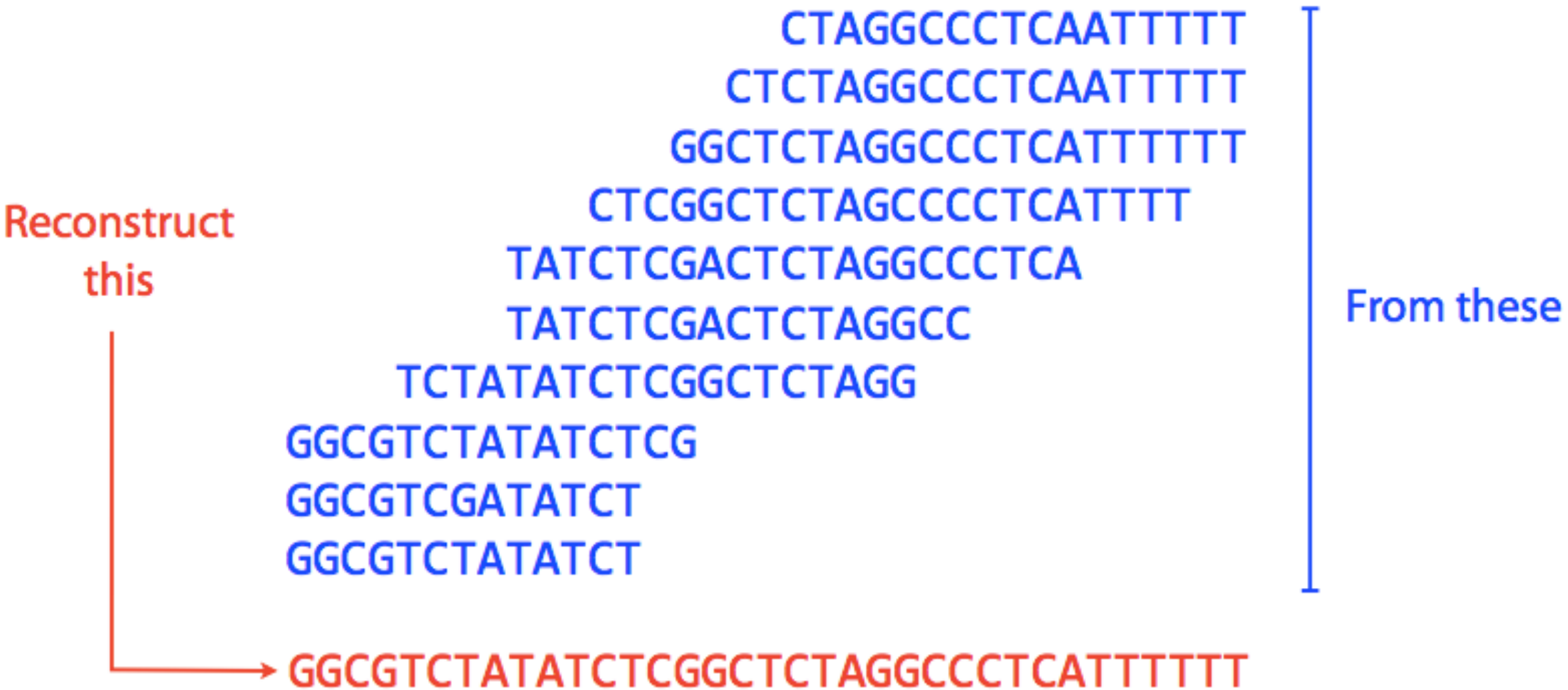
Assemble into original genome:



* Slide from Carl Kingsford

Assembly

<u>Computational Challenge</u>: assemble individual short fragments (reads) into a single genomic sequence (“superstring”)



Shortest common superstring

Problem: Given a set of strings, find a shortest string that contains all of them

Input: Strings $s_1, s_2, \ldots, s_n$

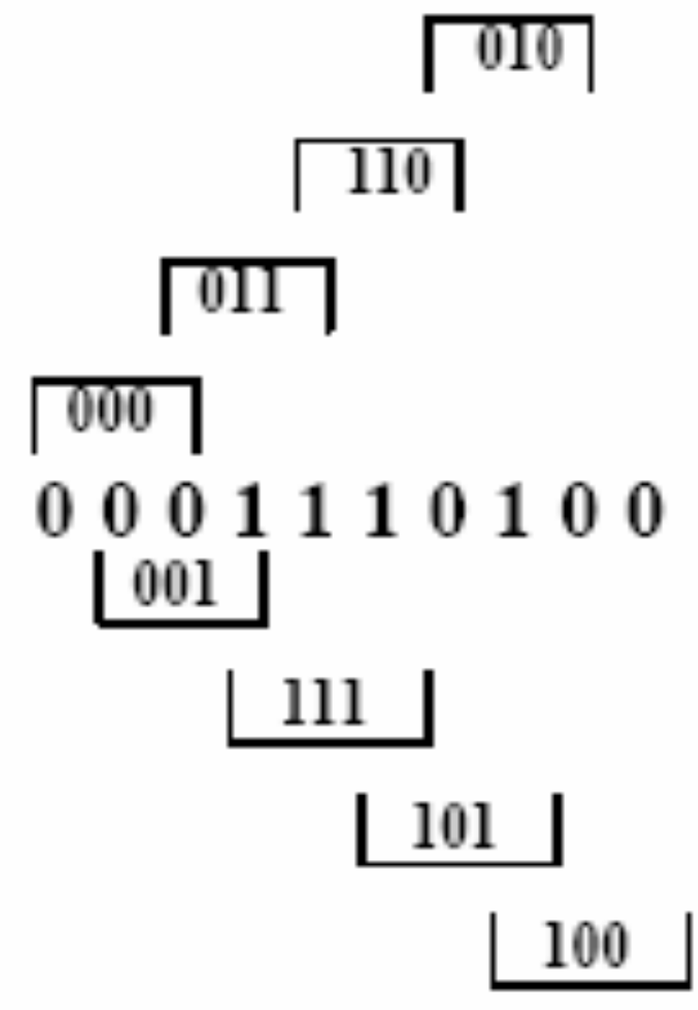
Output: A string s that contains all strings $s_1, s_2, \ldots, s_n$ as substrings, such that the length of s is minimized

Shortest common superstring

Set of strings: {000, 001, 010, 011, 100, 101, 110, 111}

Concatenation Superstring 000 001 010 011 100 101 110 111

Shortest superstring



Any ideas?

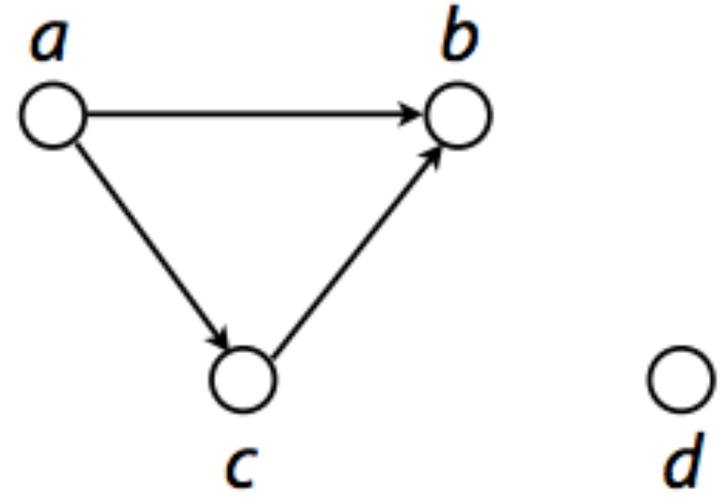
Directed Graph

Directed graph $G(V, E)$ consists of set of *vertices*, V and set of *directed edges*, E

Directed edge is an *ordered pair* of vertices.
First is the *source*, second is the *sink*.

Vertex is drawn as a circle

Edge is drawn as a line with an arrow connecting two circles



Vertex also called *node* or *point*

Edge also called *arc* or *line*

Directed graph also called *digraph*

$V = \{ a, b, c, d \}$

$E = \{ (a, b), (a, c), (c, b) \}$

Source Sink

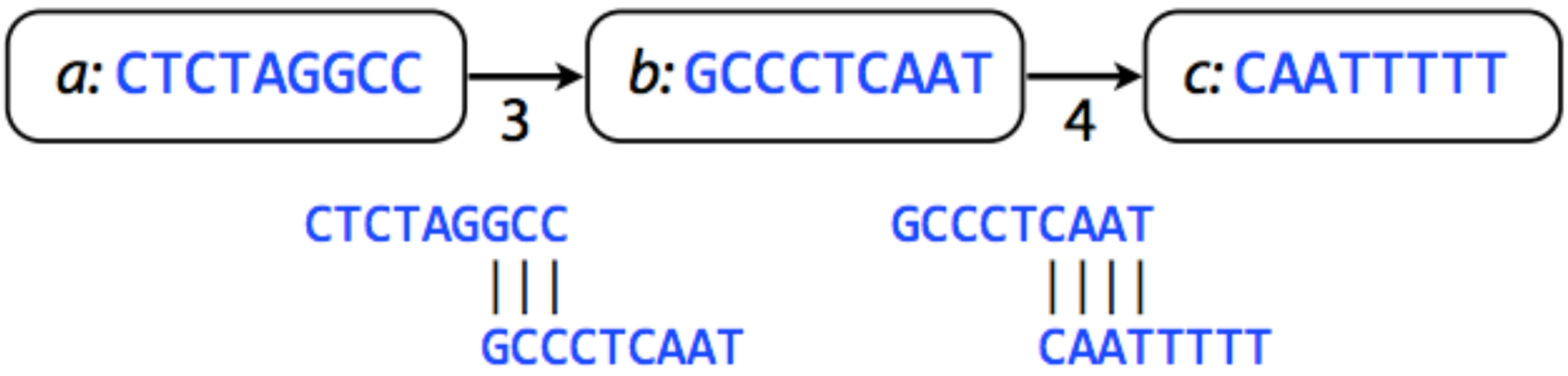
Overlap Graph

Below: overlap graph, where an overlap is a suffix/prefix match of at least 3 characters

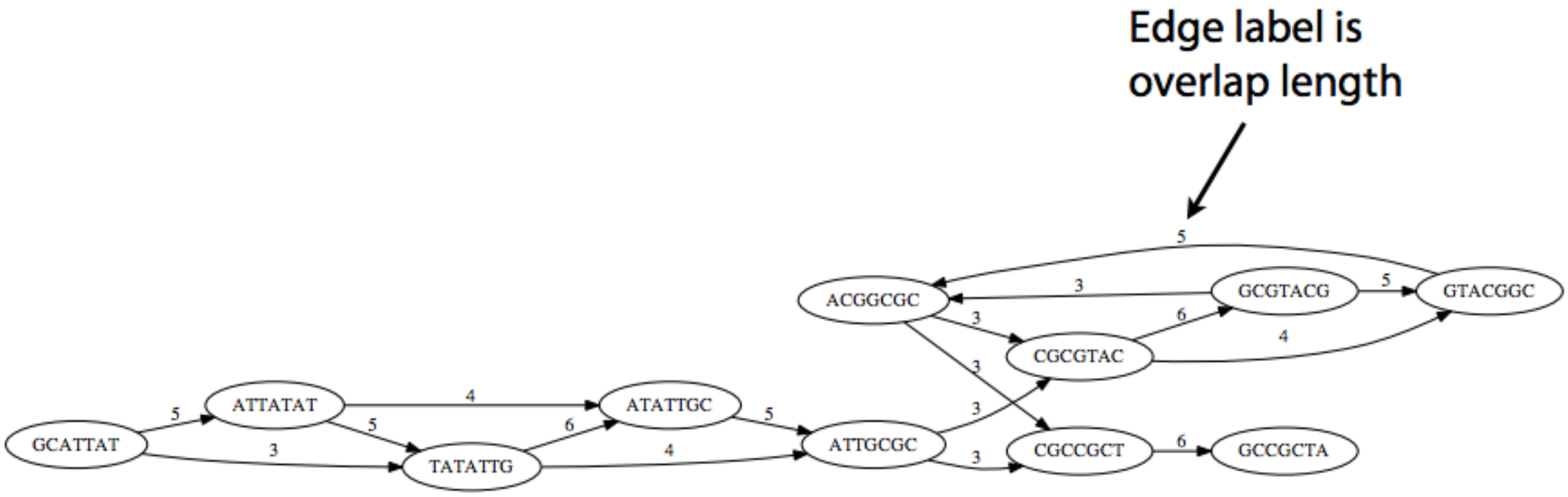
A vertex is a read, a directed edge is an overlap between suffix of source and prefix of sink

Vertices (reads): { *a:* CTCTAGGCC, *b:* GCCCTCAAT, *c:* CAATTTTT }

Edges (overlaps): { (*a*, *b*), (*b*, *c*) }



Example



Original string: GCATTATATATTGCGCGTACGGCGCCGCTACA

Shortest common superstring problem is hard

Can we solve it?

Imagine a modified overlap graph where each edge has cost = - (length of overlap)

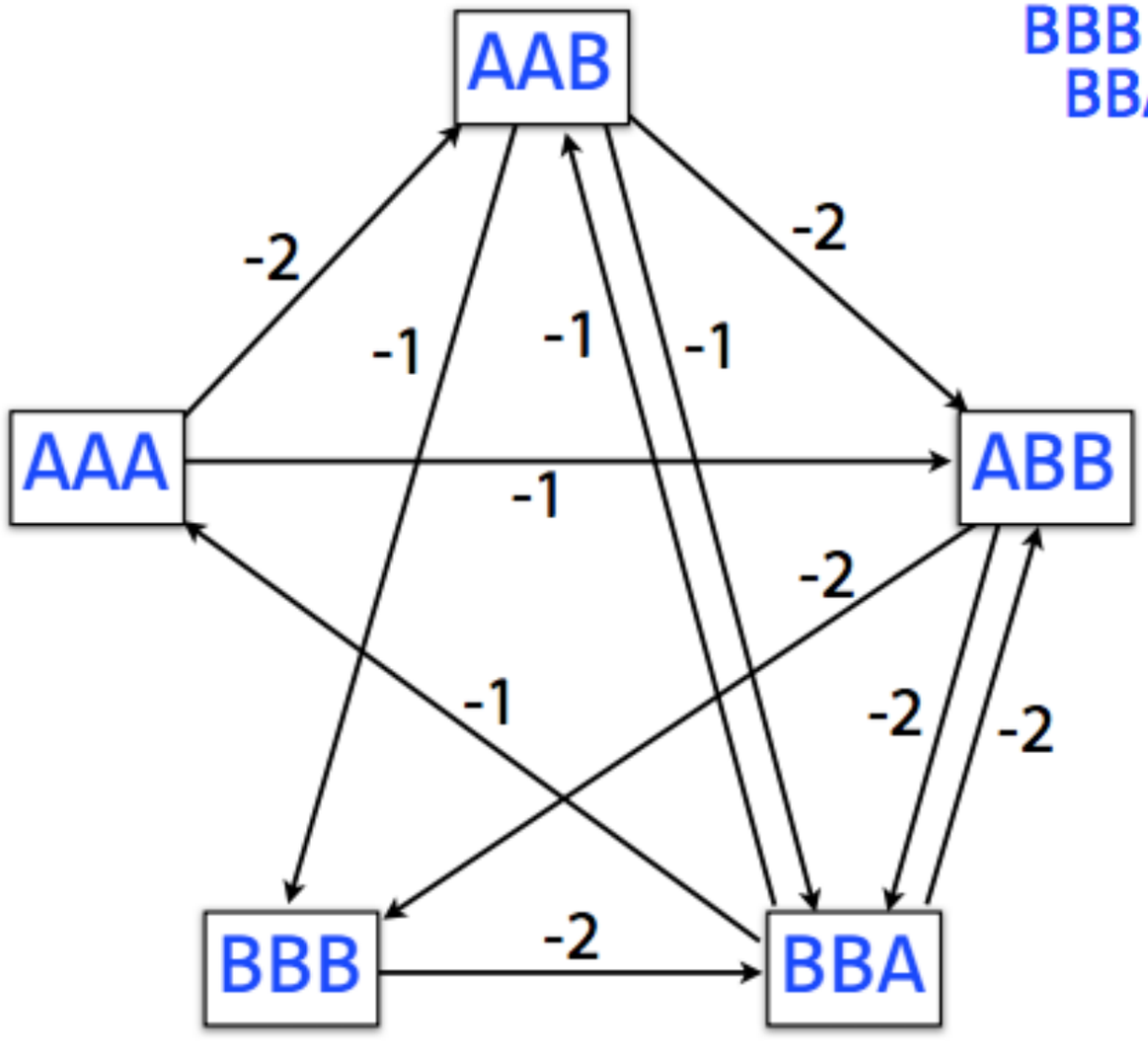
SCS corresponds to a path that visits every node once, minimizing total cost along path

That's the *Traveling Salesman Problem* (*TSP*), which is NP-hard!

S: AAA AAB ABB BBB BBA

SCS(S): AAABBBA

AAA
AAB
ABB
BBB
BBA



Shortest common superstring problem is hard

Say we disregard edge weights and just look for a path that visits all the nodes exactly once

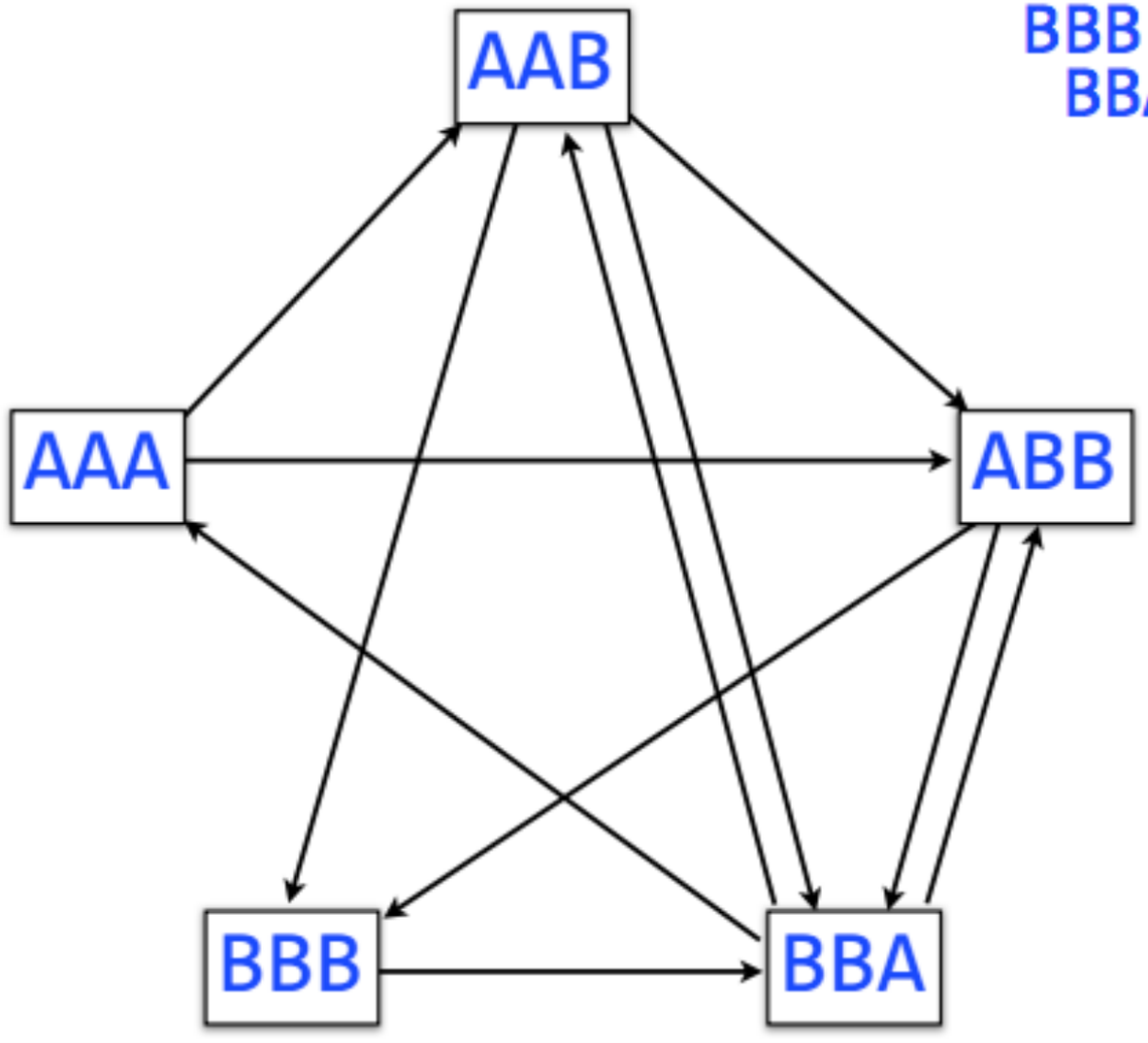
That's the *Hamiltonian Path* problem: NP-complete

Indeed, it's well established that SCS is NP-hard

S: AAA AAB ABB BBB BBA

SCS(S): AAABBBA

AAA
AAB
ABB
BBB
BBA



Matching a superstring to a set of short reads

Assume we have a set S of reads with length k (k-mers)

Goal: Find a string that can be exactly split in to set S.

$$S = \{\ \mathrm{ATG}\ \ \mathrm{AGG}\ \ \mathrm{TGC}\ \ \mathrm{TCC}\ \ \mathrm{GTC}\ \ \mathrm{GGT}\ \ \mathrm{GCA}\ \ \mathrm{CAG}\ \}$$

Overlap graph approach

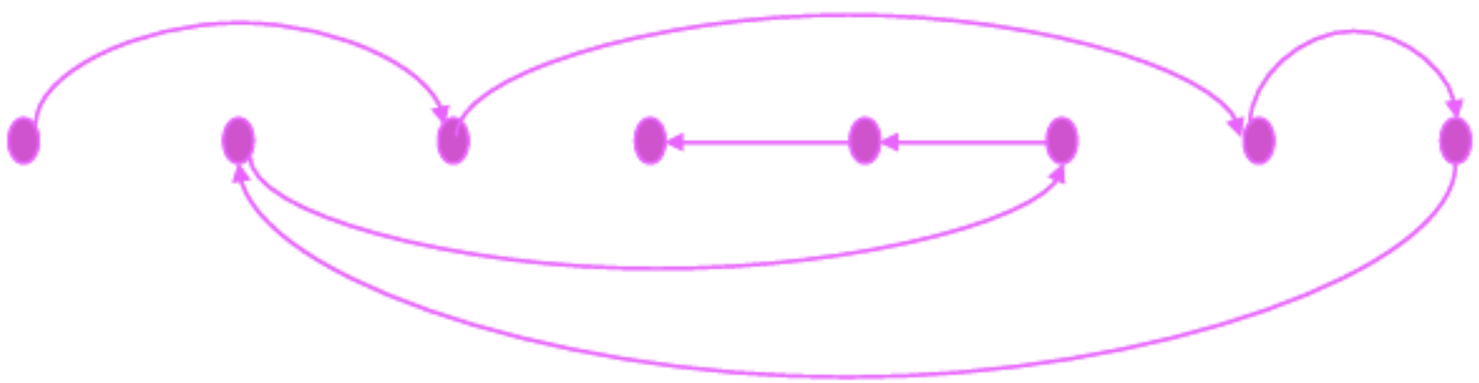
Assume we have a set S of reads with length k (k-mers)

Goal: Find a string that can be exactly split in to set S.

$$S = \{\ \text{ATG}\ \ \text{AGG}\ \ \text{TGC}\ \ \text{TCC}\ \ \text{GTC}\ \ \text{GGT}\ \ \text{GCA}\ \ \text{CAG}\ \}$$

H

ATG AGG TGC TCC GTC GGT GCA CAG



ATG C A G G T C C

Path visited every VERTEX once

Overlap graph approach is hard

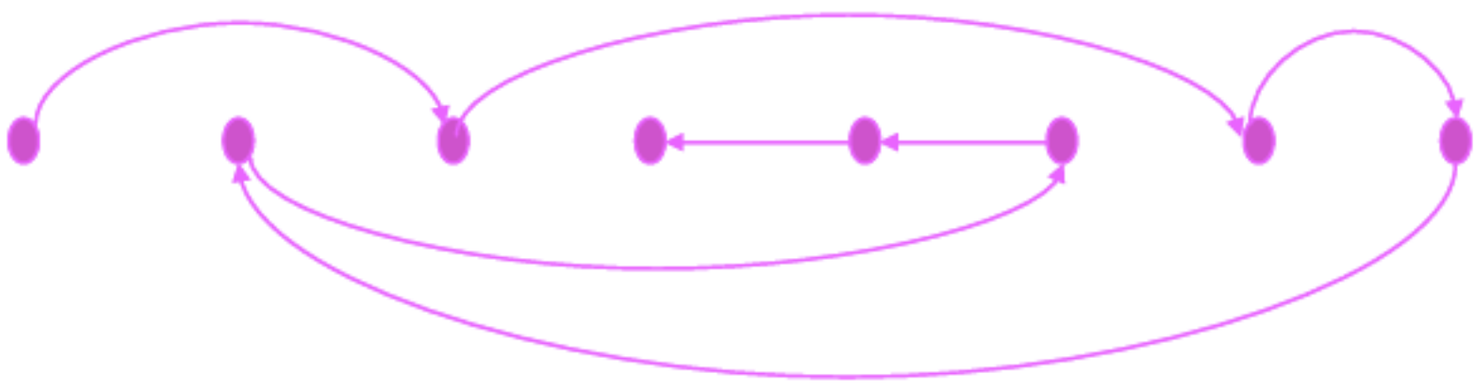
Assume we have a set S of reads with length k (k-mers)

Goal: Find a string that can be exactly split in to set S.

$$S = \{\ \text{ATG}\ \ \text{AGG}\ \ \text{TGC}\ \ \text{TCC}\ \ \text{GTC}\ \ \text{GGT}\ \ \text{GCA}\ \ \text{CAG}\ \}$$

H

ATG AGG TGC TCC GTC GGT GCA CAG



ATG C A G G T C C

Path visited every VERTEX once