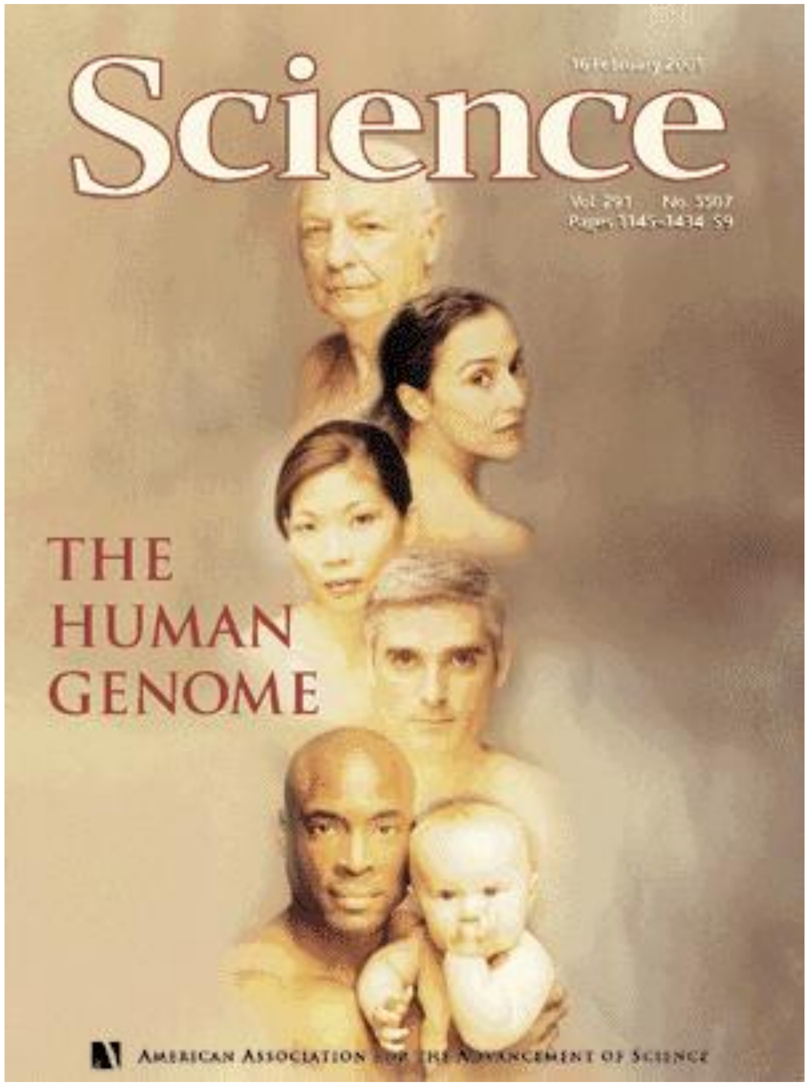
394C

March 5, 2012

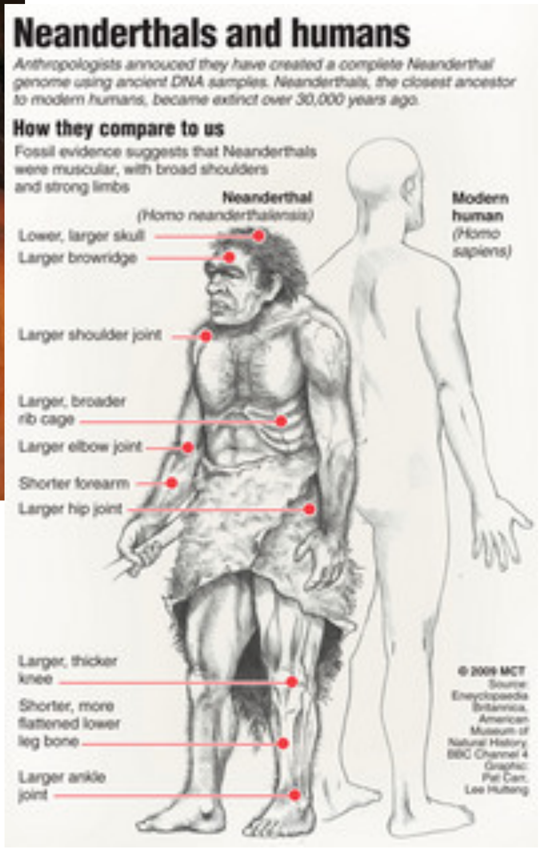
Introduction to Genome Assembly

Genome Sequencing Projects:

Started with the Human Genome Project



Other Genome Projects! (Neandertals, Wooly Mammoths, and more ordinary creatures…)

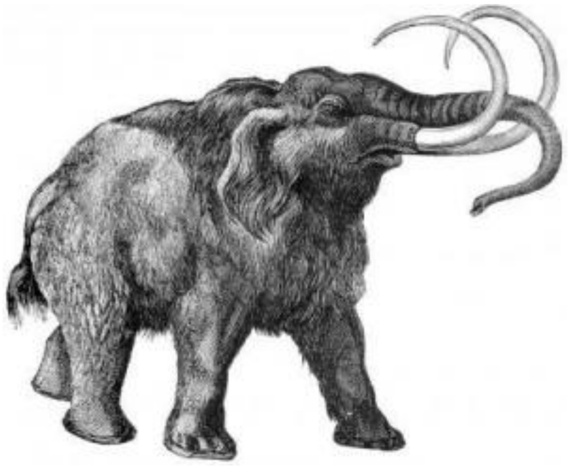

Neanderthals and humans

Anthropologists annouced they have created a complete Neanderthal genome using ancient DNA samples. Neanderthals, the closest ancestor to modern humans, became extinct over 30,000 years ago.

How they compare to us

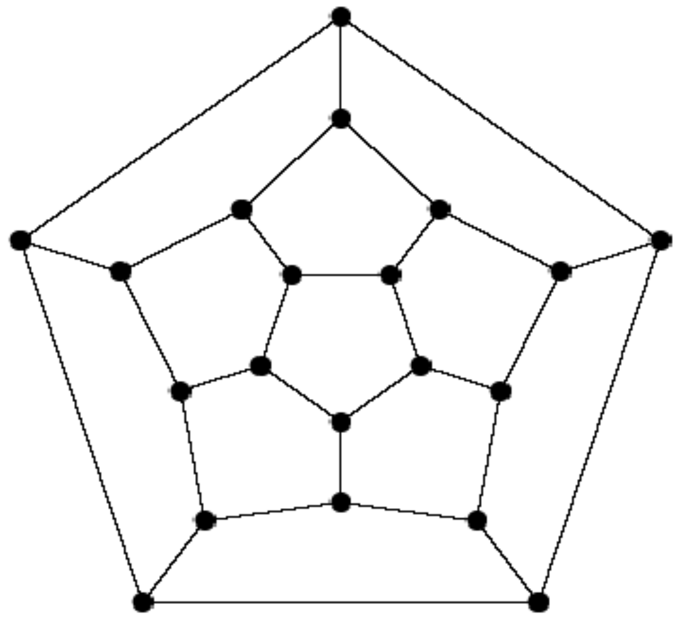
Fossil evidence suggests that Neanderthals were muscular, with broad shoulders and strong limbs

© 2009 MCT
Source: Encyclopaedia Britannica, American Museum of Natural History, BBC Channel 4
Graphic: Pat Carr, Lee Hulteng



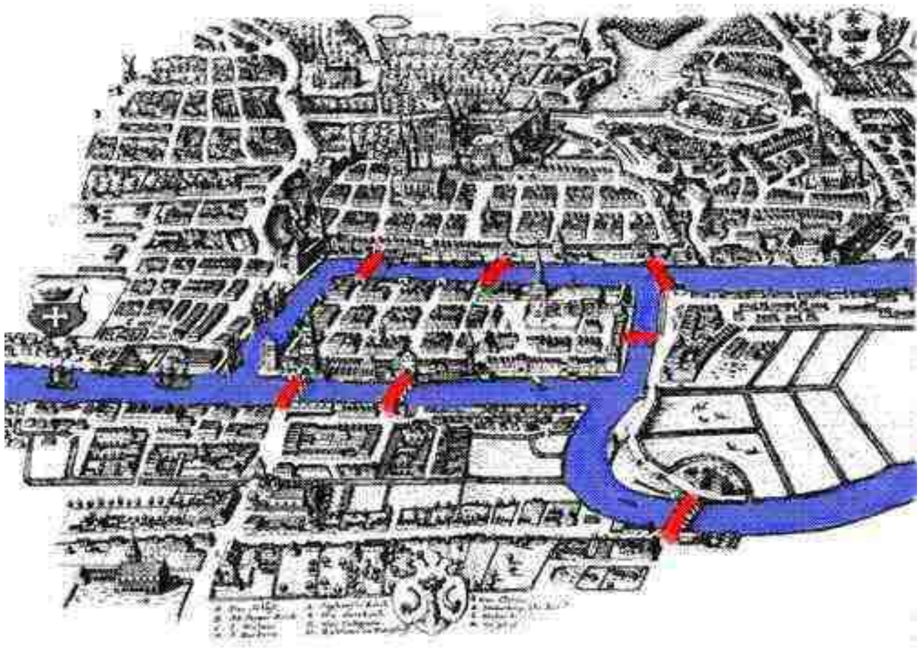
Hamiltonian Cycle Problem

- Find a cycle that visits every ***vertex*** exactly once
- NP – complete



Game invented by Sir William Hamilton in 1857

Bridges of Königsberg

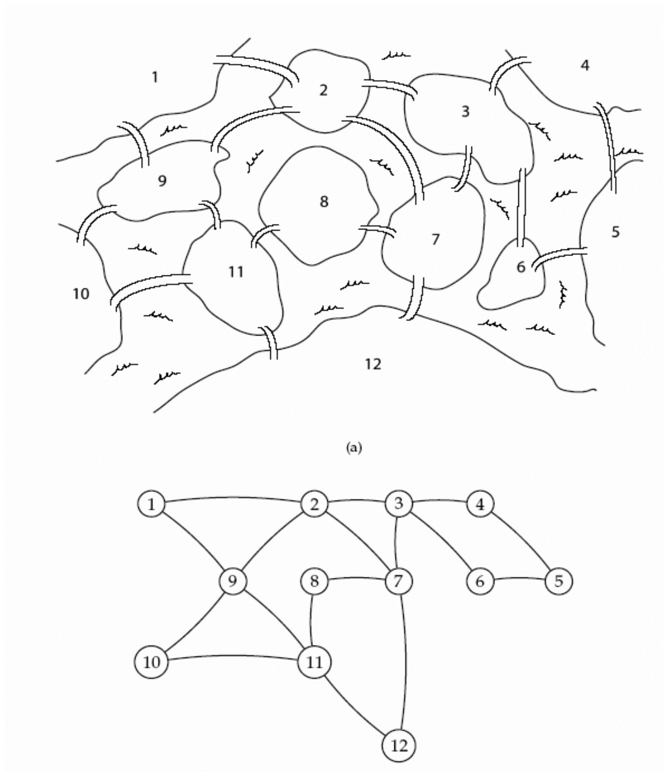


Find a tour crossing every bridge just once
Leonhard Euler, 1735

Eulerian Cycle Problem

- Find a cycle that visits every ***edge*** exactly once

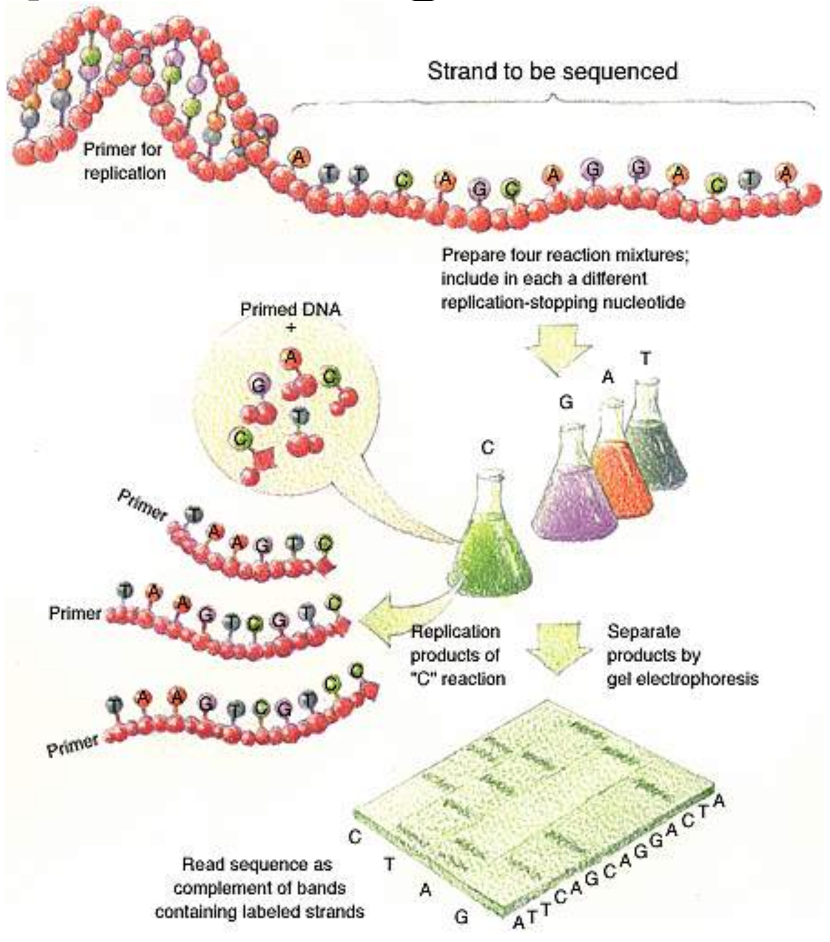
- Linear time



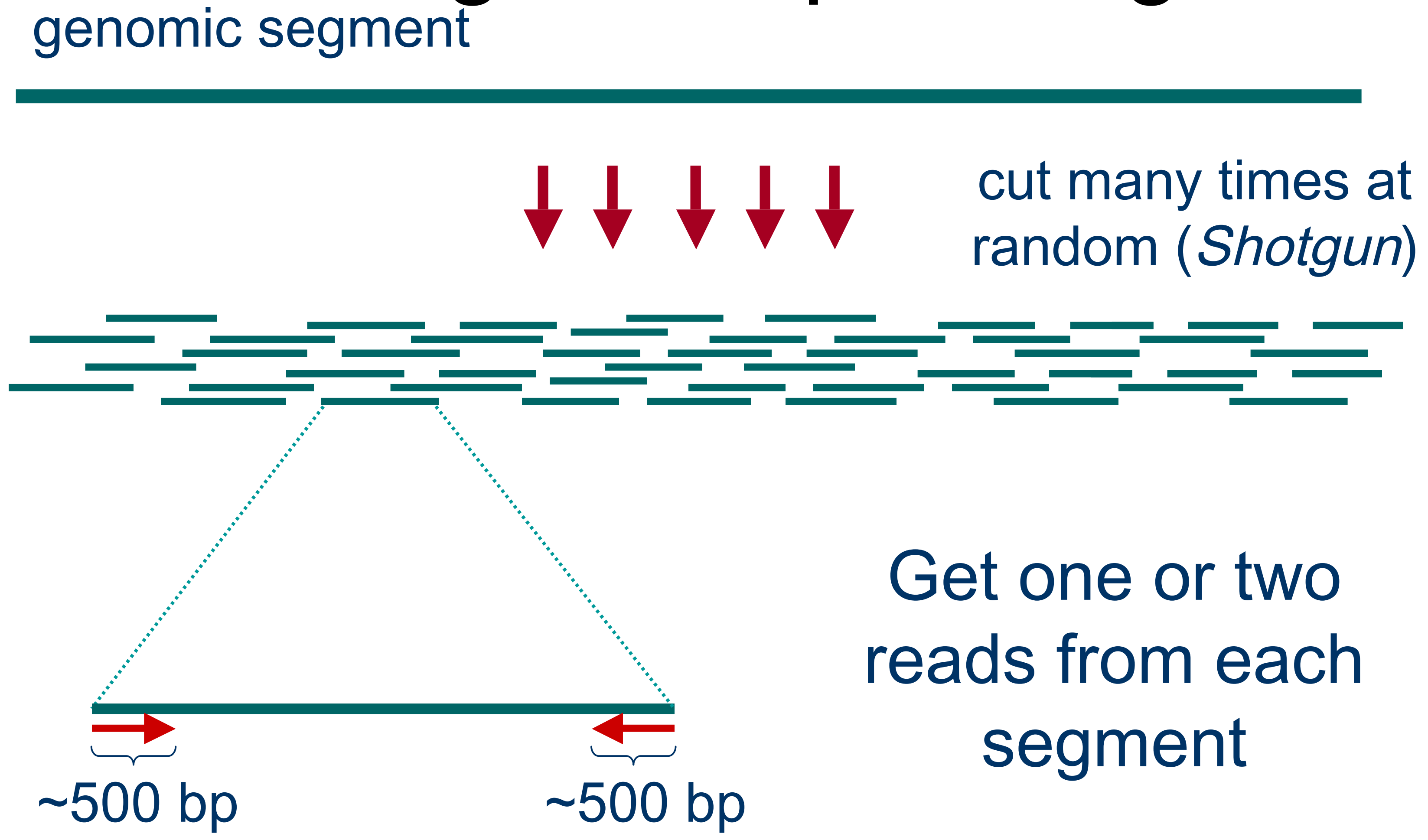
More complicated Königsberg

DNA Sequencing

- Shear DNA into millions of small fragments
- Read 500 – 700 nucleotides at a time from the small fragments (Sanger method)

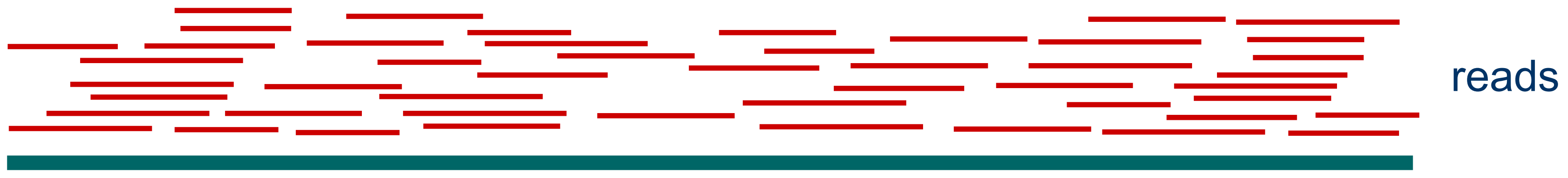

Strand to be sequenced
Primer for replication
Prepare four reaction mixtures; include in each a different replication-stopping nucleotide
Primed DNA +
G
A
T
C
Primer
Replication products of "C" reaction
Separate products by gel electrophoresis
Read sequence as complement of bands containing labeled strands
ATTCAGCAGGACTA

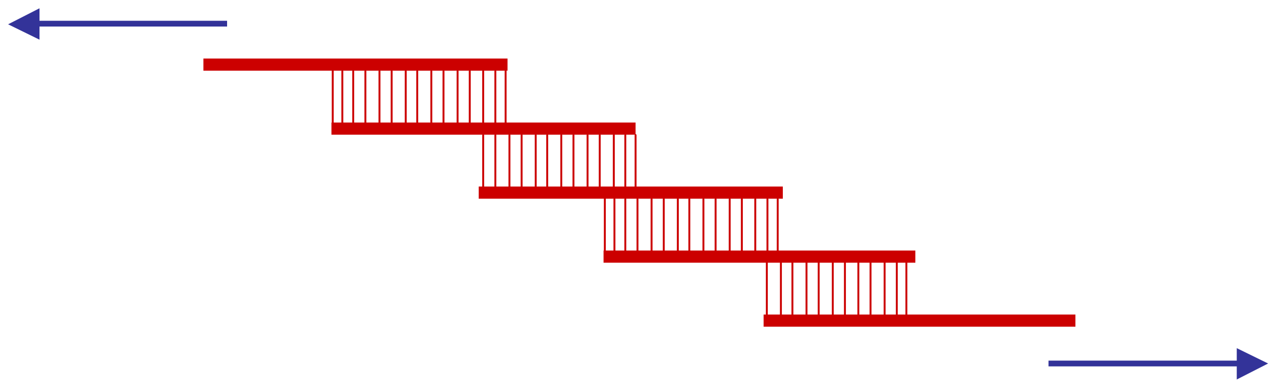
Shotgun Sequencing



Get one or two reads from each segment

Fragment Assembly





Cover region with ~7-fold redundancy

Overlap reads and extend to reconstruct the original genomic region

Fragment Assembly

- **Computational Challenge:** assemble individual short fragments (reads) into a single genomic sequence (“superstring”)

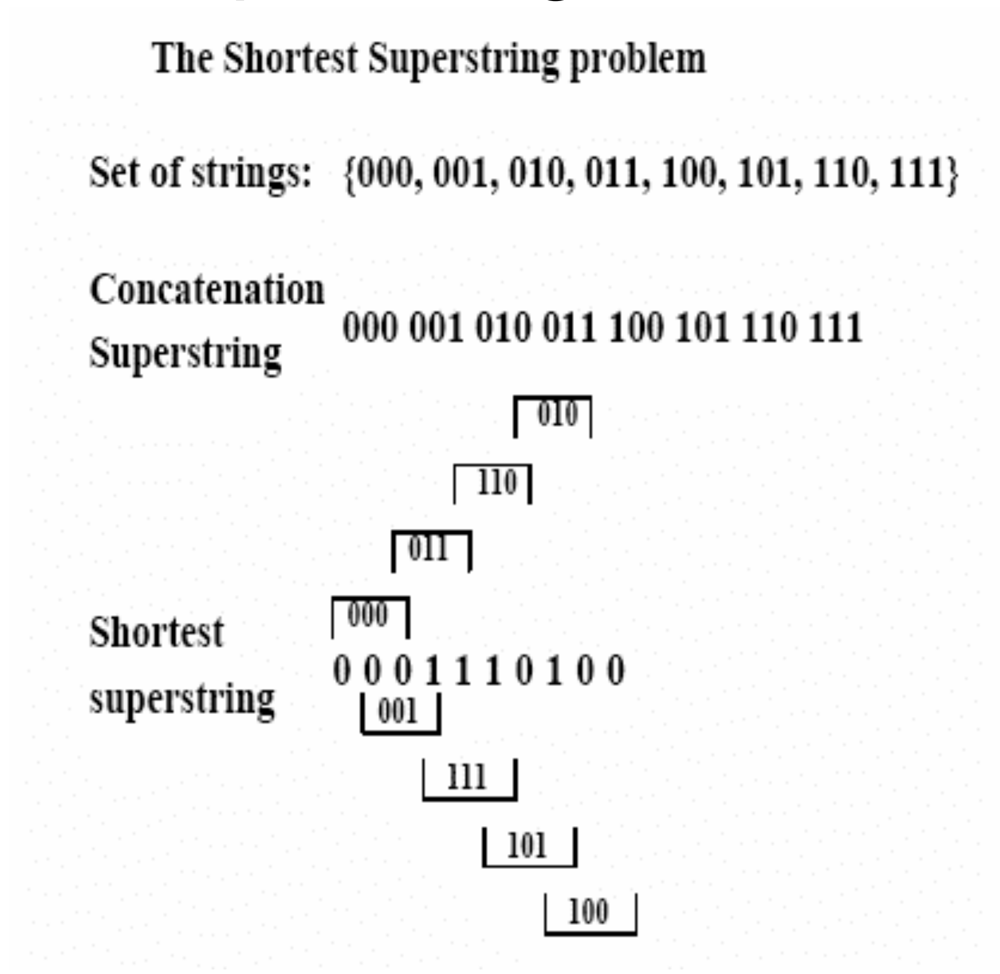
- Until late 1990s the shotgun fragment assembly of human genome was viewed as intractable problem

Shortest Superstring Problem

- 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

- **Complexity:** NP – complete
- **Note:** this formulation does not take into account sequencing errors

Shortest Superstring Problem: Example



The Shortest Superstring problem

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

Concatenation Superstring 000 001 010 011 100 101 110 111

Shortest superstring 0 0 0 1 1 1 0 1 0 0

Reducing SSP to TSP

- Define *overlap (s_i, s_j)* as the length of the longest prefix of s_j that matches a suffix of s_i.

  aaaggcatcaaatctaaaggcatcaaa

  aaaggcatcaaatctaaaggcatcaaa

What is** overlap (s_i, s_j) **for these strings?

Reducing SSP to TSP

- Define *overlap* *(s_i, s_j)* as the length of the longest prefix of s_j that matches a suffix of s_i.

  aaaggcatcaaatctaaaggcatcaaa

  aaaggcatcaaatctaaaggcatcaaa
                        aaaggcatcaaatctaaaggcatcaaa

overlap=12

Reducing SSP to TSP

- Define *overlap (s_i, s_j)* as the length of the longest prefix of s_j that matches a suffix of s_i.

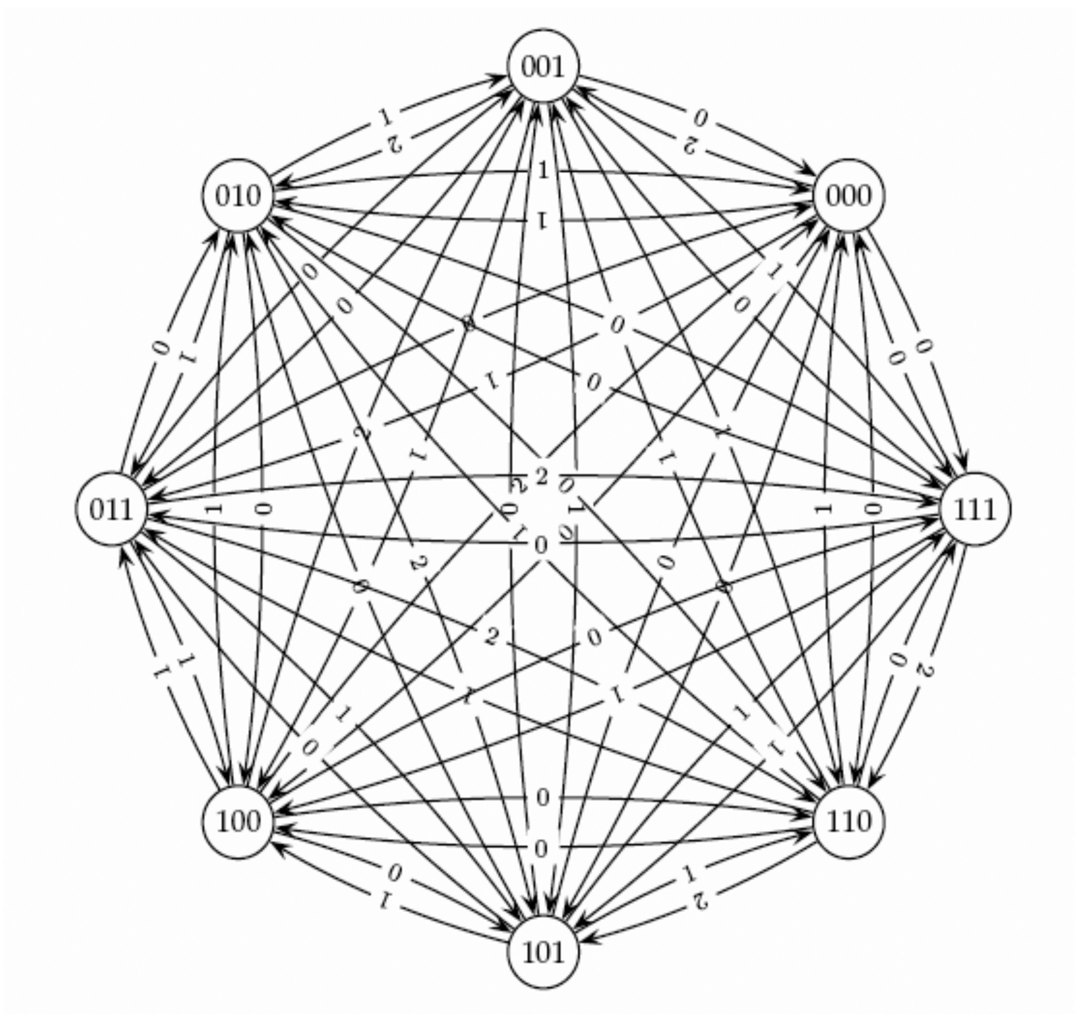
aaaggcatcaaatctaaaggcatcaaa

aaaggcatcaaatctaaaggcatcaaa

aaaggcatcaaatctaaaggcatcaaa

- Construct a graph with *n* vertices representing the *n* strings s_1, s_2, …., s_n.
- Insert edges of length *overlap (s_i, s_j)* between vertices s_i and s_j.
- Find the shortest path which visits every vertex exactly once. This is the **Traveling Salesman Problem** (TSP), which is also NP – complete.

Reducing SSP to TSP (cont'd)



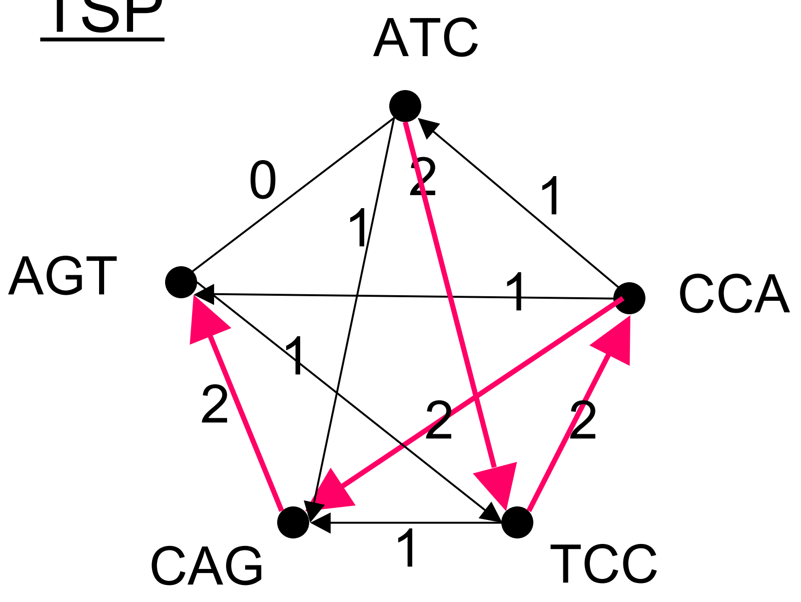
SSP to TSP: An Example

S = { ATC, CCA, CAG, TCC, AGT }

<u>SSP</u>

```
        AGT
      CCA
    ATC
    ATCCAGT
     TCC
       CAG
```

<u>TSP</u>



ATCCAGT

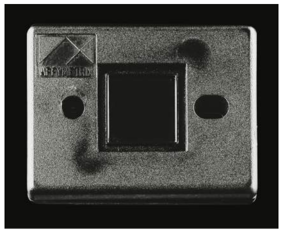
Sequencing by Hybridization (SBH): History

- **1988:** SBH suggested as an an alternative sequencing method. Nobody believed it would ever work

*First microarray prototype **(1989)***

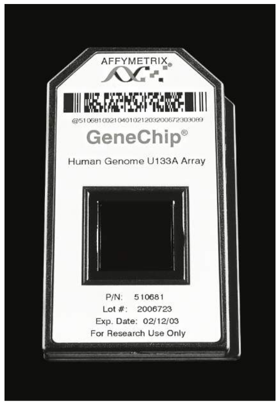
- **1991:** Light directed polymer synthesis developed by Steve Fodor and colleagues.

*First commercial DNA microarray prototype w/16,000 features **(1994)***



- **1994:** Affymetrix develops first 64-kb DNA microarray

*500,000 features per chip **(2002)***



How SBH Works

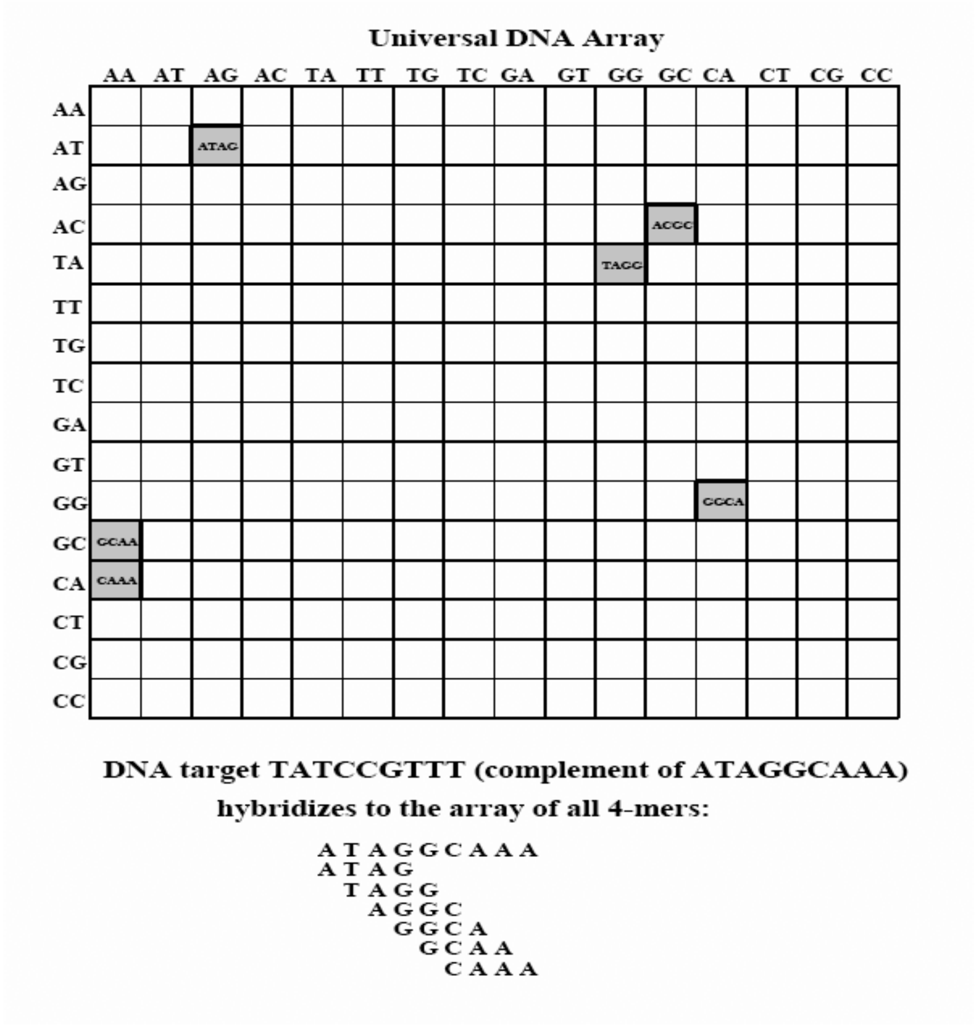
- Attach all possible DNA probes of length *l* to a flat surface, each probe at a distinct and known location. This set of probes is called the DNA array.
- Apply a solution containing fluorescently labeled DNA fragment to the array.
- The DNA fragment hybridizes with those probes that are complementary to substrings of length *l* of the fragment.

How SBH Works (cont’d)

- Using a spectroscopic detector, determine which probes hybridize to the DNA fragment to obtain the l–mer composition of the target DNA fragment.

- Apply the combinatorial algorithm (below) to reconstruct the sequence of the target DNA fragment from the l – mer composition.

Hybridization on DNA Array



Universal DNA Array

	AA	AT	AG	AC	TA	TT	TG	TC	GA	GT	GG	GC	CA	CT	CG	CC
AA																
AT			ATAG													
AG																
AC												AGGC				
TA											TAGG					
TT																
TG																
TC																
GA																
GT																
GG													GGCA			
GC	GCAA															
CA	CAAA															
CT																
CG																
CC																

DNA target TATCCGTTT (complement of ATAGGCAAA)
hybridizes to the array of all 4-mers:

```
A T A G G C A A A
A T A G
  T A G G
    A G G C
      G G C A
        G C A A
          C A A A
```

l-mer composition

- ***Spectrum (s, l)*** - *unordered* multiset of all possible *(n – l + 1)* *l*-mers in a string *s* of length *n*
- The order of individual elements in *Spectrum (s,l)* does not matter
- For *s* = TATGGTGC all of the following are equivalent representations of *Spectrum (s,3):*

  {TAT, ATG, TGG, GGT, GTG, TGC}

  {ATG, GGT, GTG, TAT, TGC, TGG}

  {TGG, TGC, TAT, GTG, GGT, ATG}

l-mer composition

- ***Spectrum (s, l)*** - *unordered* multiset of all possible *(n – l + 1)* *l*-mers in a string *s* of length *n*
- The order of individual elements in *Spectrum (s,l)* does not matter
- For *s* = TATGGTGC all of the following are equivalent representations of *Spectrum (s,3):*

  {TAT, ATG, TGG, GGT, GTG, TGC}

  {ATG, GGT, GTG, TAT, TGC, TGG}

  {TGG, TGC, TAT, GTG, GGT, ATG}
- We usually choose the lexicographically maximal representation as the canonical one.

Different sequences – the same spectrum

- Different sequences may have the same spectrum:

  Spectrum(GTATCT,2)=

  Spectrum(GTCTAT,2)=

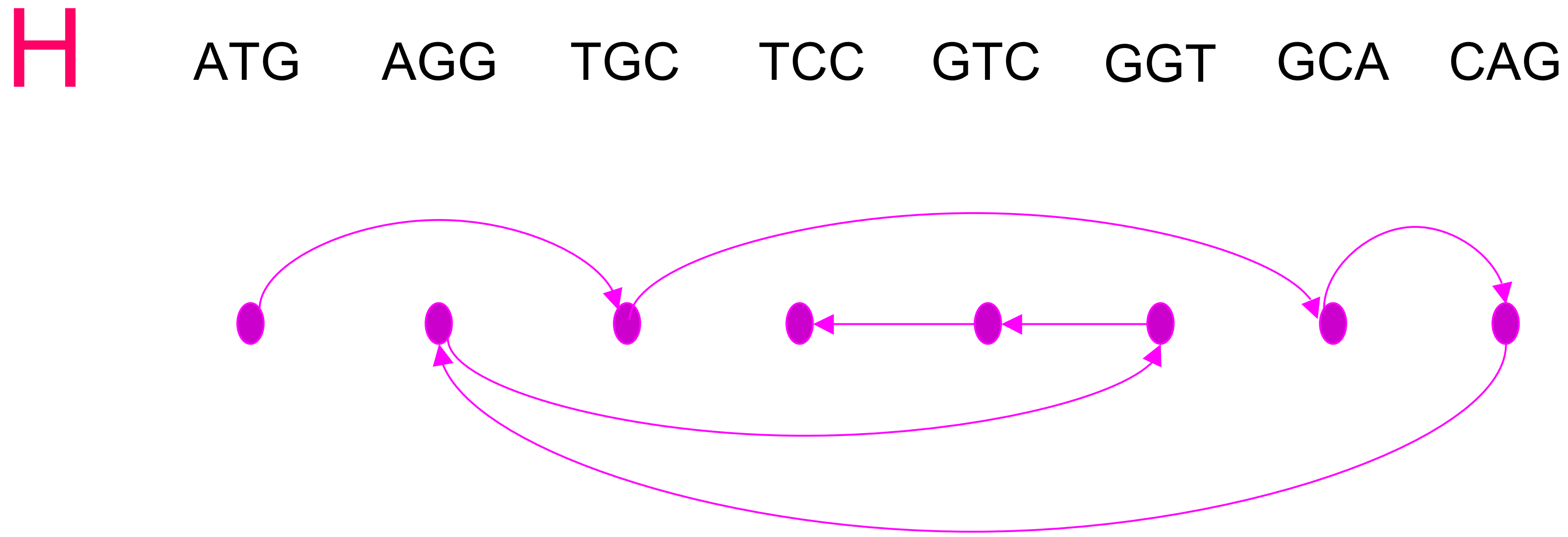
  {AT, CT, GT, TA, TC}

The SBH Problem

- Goal: Reconstruct a string from its *l*-mer composition
- Input: A set *S*, representing all *l*-mers from an (unknown) string *s*
- Output: String *s* such that *Spectrum (s,l) = S*

SBH: Hamiltonian Path Approach

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



ATGCAGGTCC

Path visited every VERTEX once

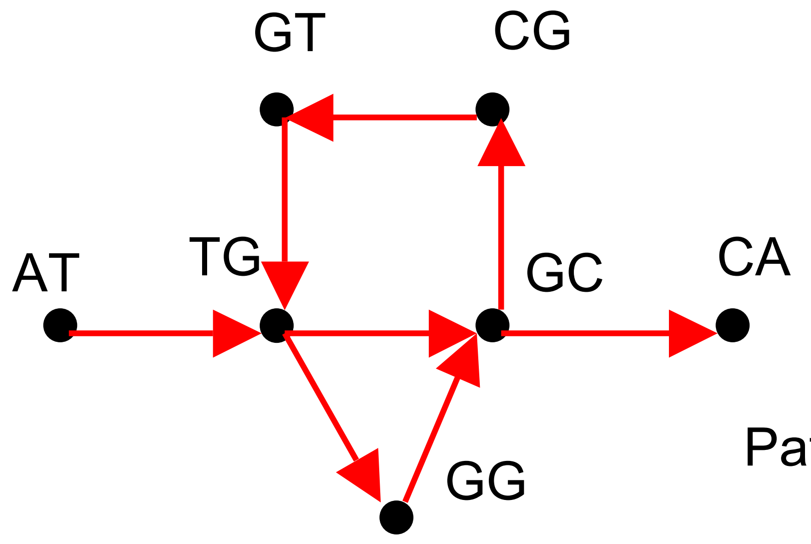
SBH: Eulerian Path Approach

S = { ATG, TGC, GTG, GGC, GCA, GCG, CGT }

Vertices correspond to (*l* –1)–mers :

{ AT, TG, GC, GG, GT, CA, CG }

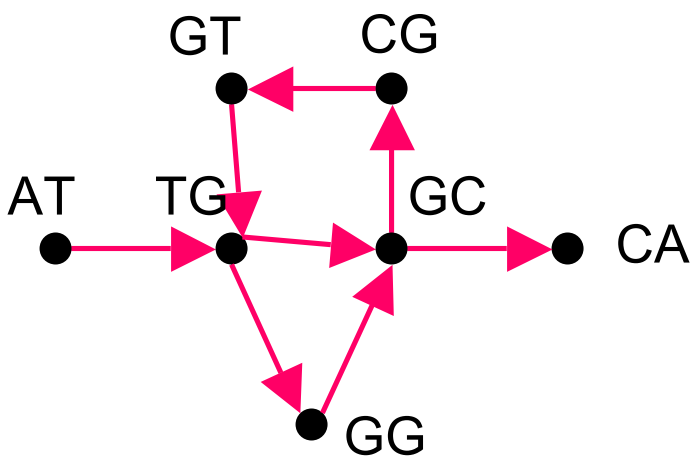
Edges correspond to *l* – mers from *S*



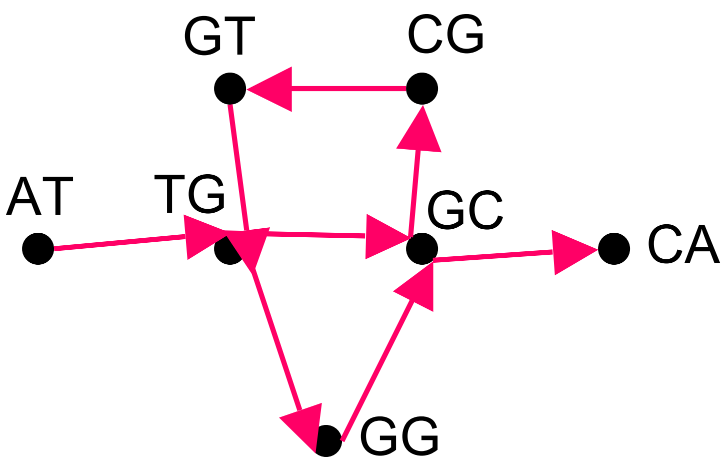
Path visited every EDGE once

SBH: Eulerian Path Approach

S = { AT, TG, GC, GG, GT, CA, CG } corresponds to two different paths:



ATGGCGTGCA



ATGCGTGGCA

Euler Theorem

- A graph is balanced if for every vertex the number of incoming edges equals to the number of outgoing edges:

  $$in(v)=out(v)$$

- **Theorem**: *A connected graph is Eulerian if and only if each of its vertices is balanced.*

Euler Theorem: Proof

- Eulerian → balanced

  for every edge entering *v* (incoming edge) there exists an edge leaving *v* (outgoing edge). Therefore

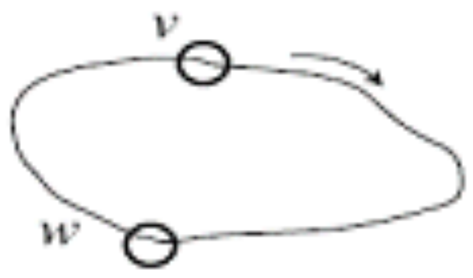
  $$in(v)=out(v)$$

- Balanced → Eulerian

  ???

Algorithm for Constructing an Eulerian Cycle

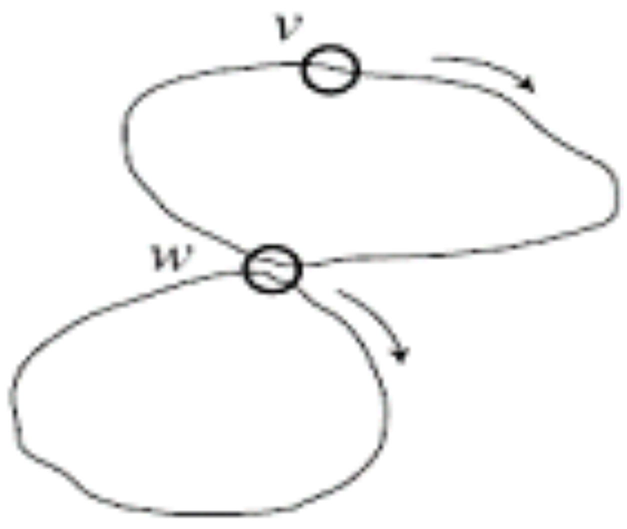
- Start with an arbitrary vertex *v* and form an arbitrary cycle with unused edges until a dead end is reached. Since the graph is Eulerian this dead end is necessarily the starting point, i.e., vertex *v*.



(a)

Algorithm for Constructing an Eulerian Cycle (cont'd)

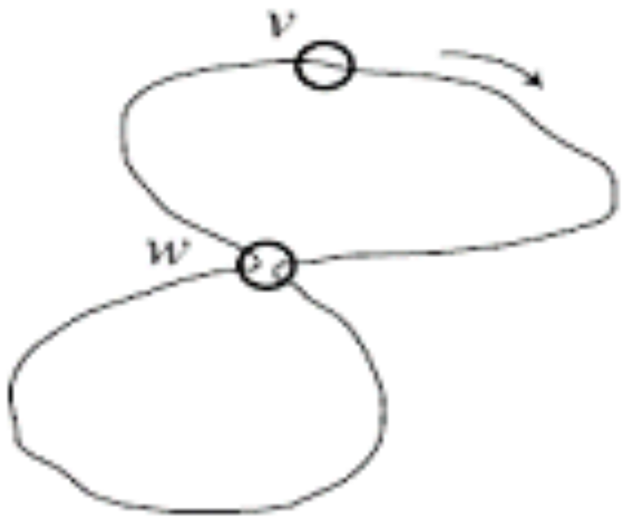
b. If cycle from (a) above is not an Eulerian cycle, it must contain a vertex *w*, which has untraversed edges. Perform step (a) again, using vertex *w* as the starting point. Once again, we will end up in the starting vertex *w.*



(b)

Algorithm for Constructing an Eulerian Cycle (cont'd)

c. Combine the cycles from (a) and (b) into a single cycle and iterate step (b).



(c)

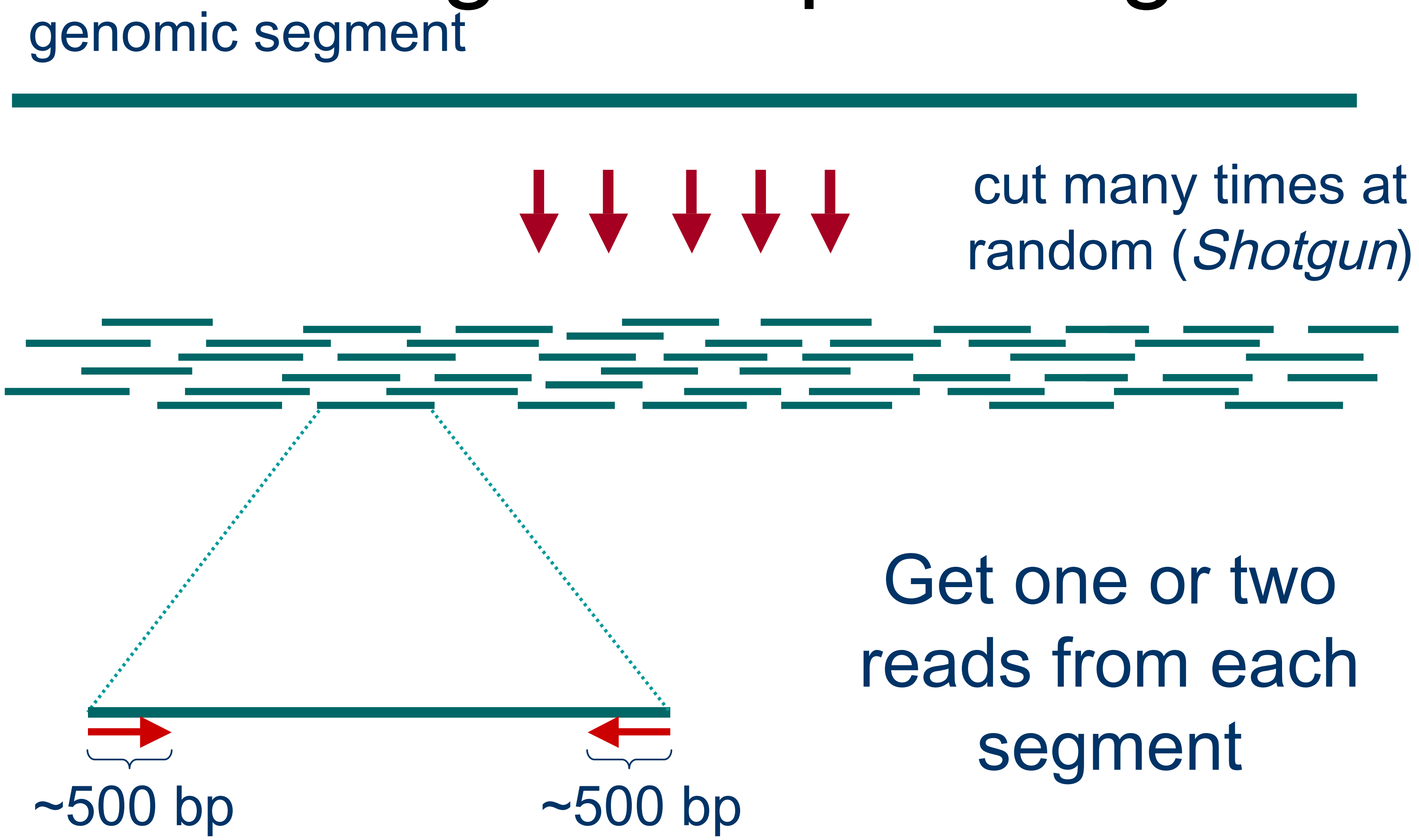
Euler Theorem: Extension

- **Theorem**: *A connected graph has an Eulerian path if and only if it contains at most two semi-balanced vertices and all other vertices are balanced.*

Some Difficulties with SBH

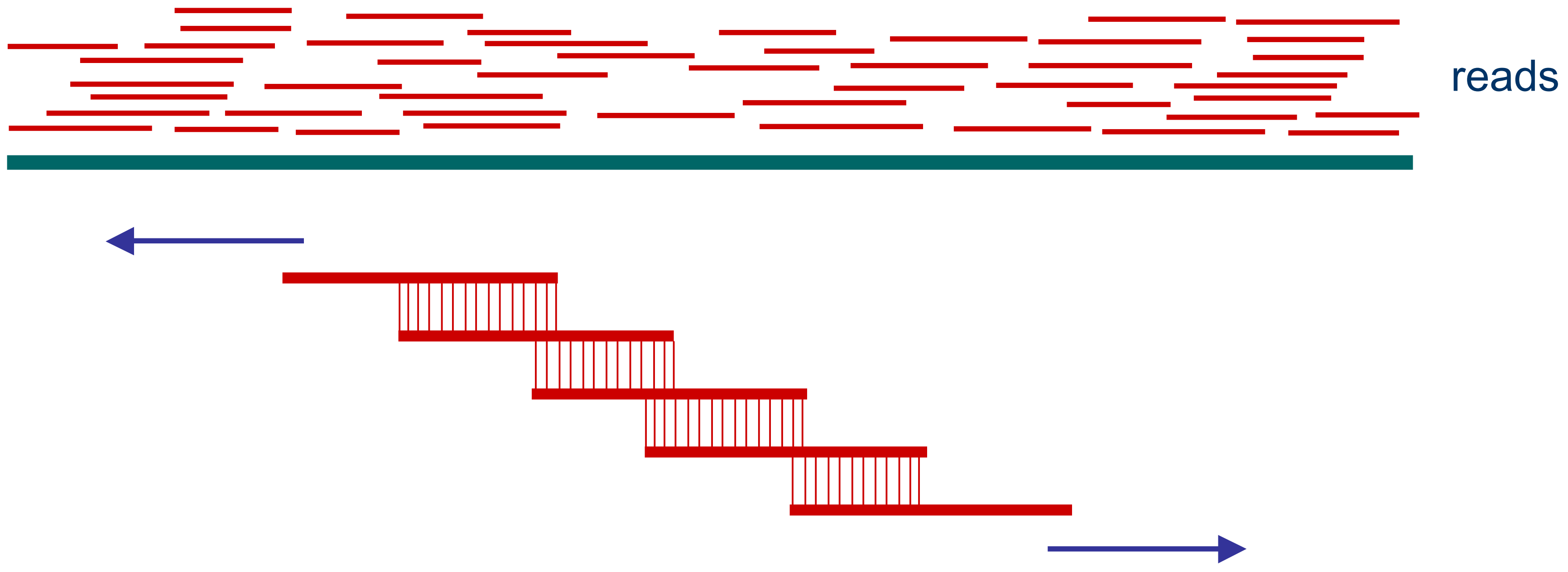
- **Fidelity of Hybridization:** difficult to detect differences between probes hybridized with perfect matches and 1 or 2 mismatches
- **Array Size:** Effect of low fidelity can be decreased with longer *l*-mers, but array size increases exponentially in *l.* Array size is limited with current technology.
- **Practicality:** SBH is still impractical. As DNA microarray technology improves, SBH may become practical in the future
- **Practicality again**: Although SBH is still impractical, it spearheaded expression analysis and SNP analysis techniques

Shotgun Sequencing



Get one or two reads from each segment

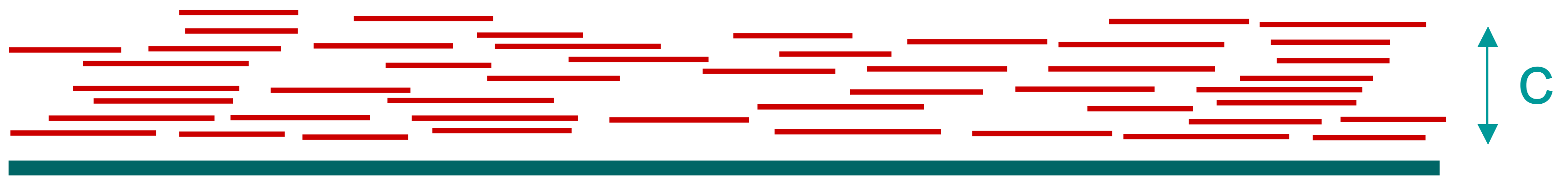
Fragment Assembly



Cover region with ~7-fold redundancy

Overlap reads and extend to reconstruct the original genomic region

Read Coverage



Length of genomic segment: ***L***

Number of reads: ***n***

Length of each read: ***l***

Coverage $\boldsymbol{C = n\,l\,/\,L}$

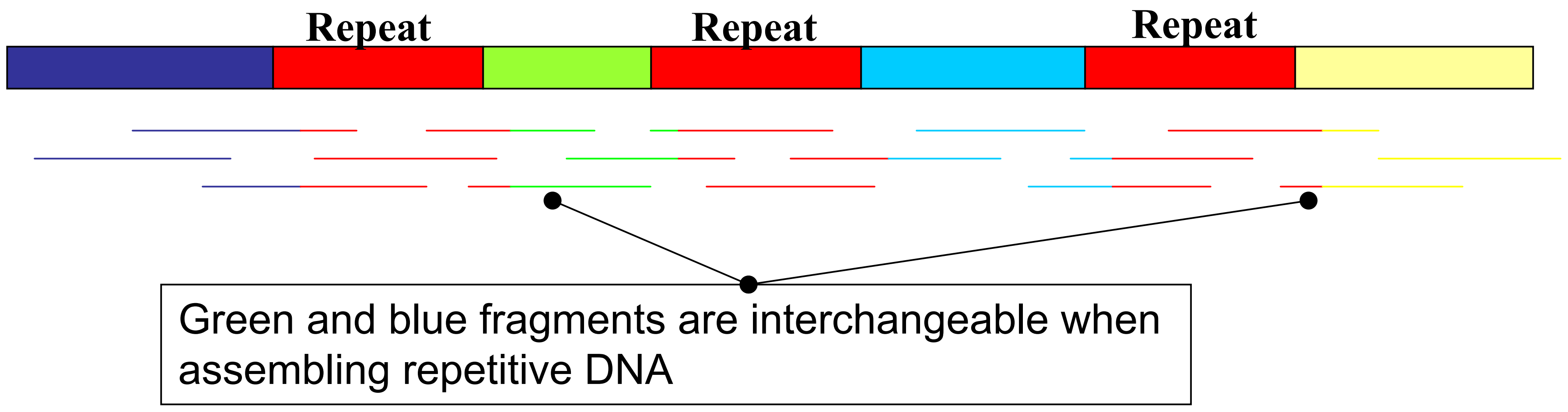
How much coverage is enough?

Lander-Waterman model:

Assuming uniform distribution of reads, *C*=10 results in 1 gapped region per 1,000,000 nucleotides

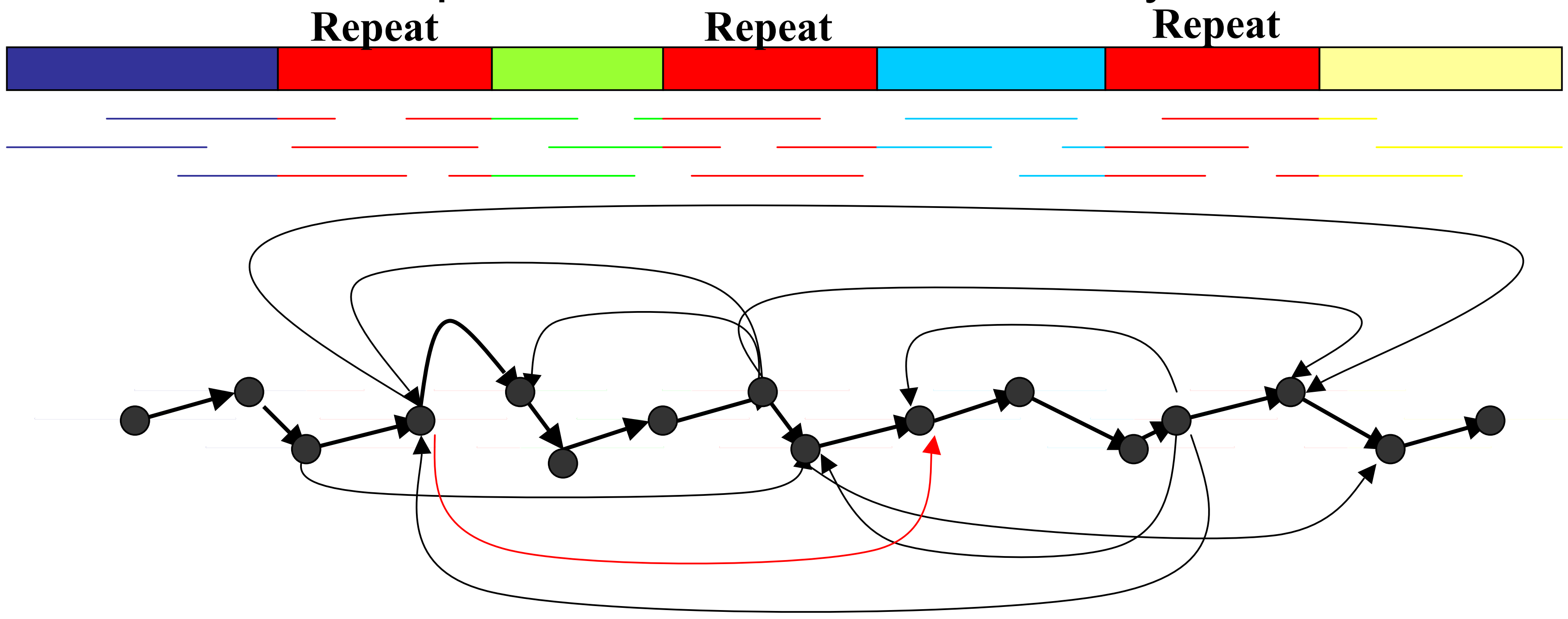
Challenges in Fragment Assembly

- Repeats: A **major** problem for fragment assembly
- > 50% of human genome are repeats:
  - over 1 million *Alu* repeats (about 300 bp)
  - about 200,000 LINE repeats (1000+ bp)


Repeat
Repeat
Repeat
Green and blue fragments are interchangeable when assembling repetitive DNA

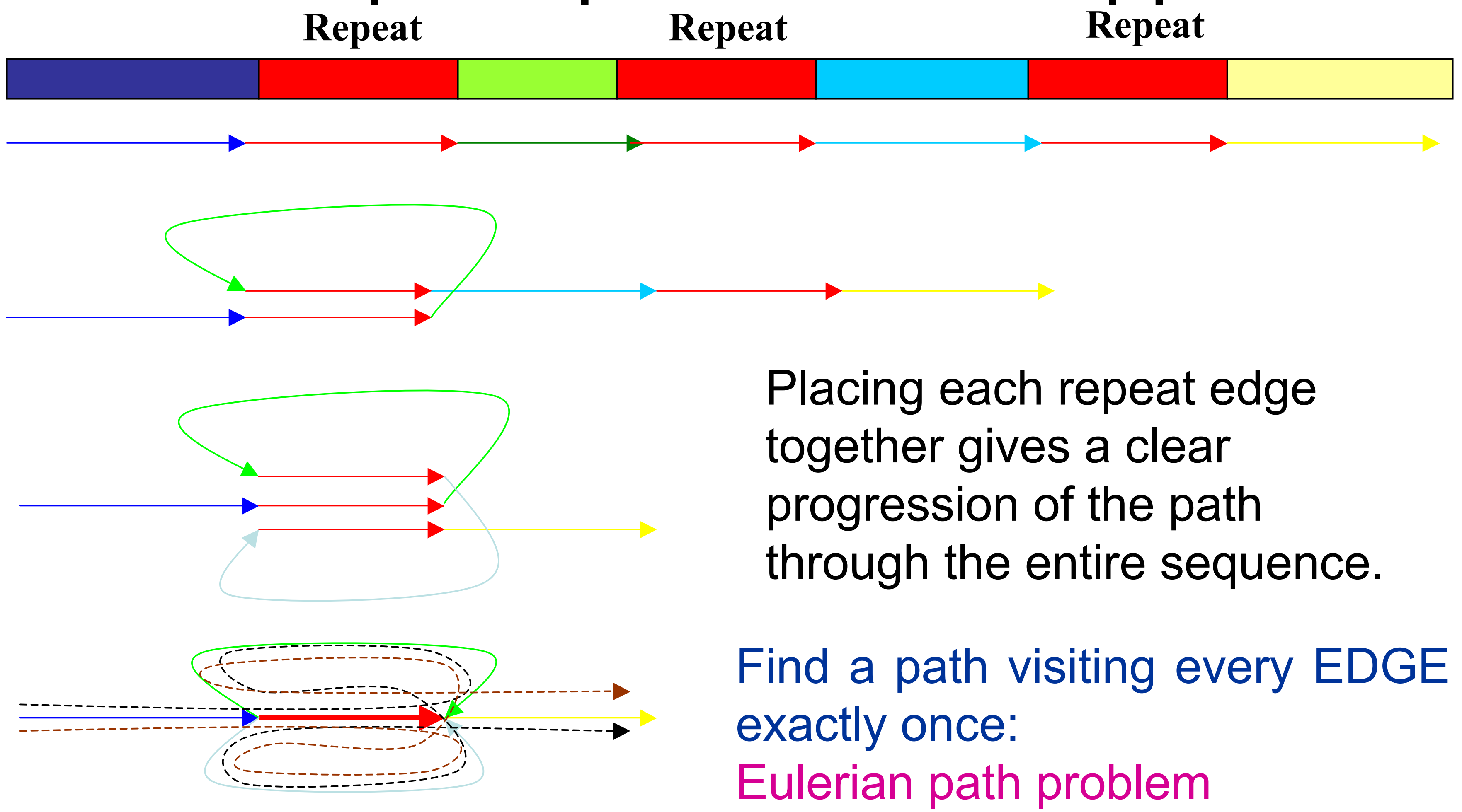
Overlap Graph: Hamiltonian Approach

Each vertex represents a read from the original sequence.
Vertices from repeats are connected to many others.



Find a path visiting every VERTEX exactly once: Hamiltonian path problem

Overlap Graph: Eulerian Approach



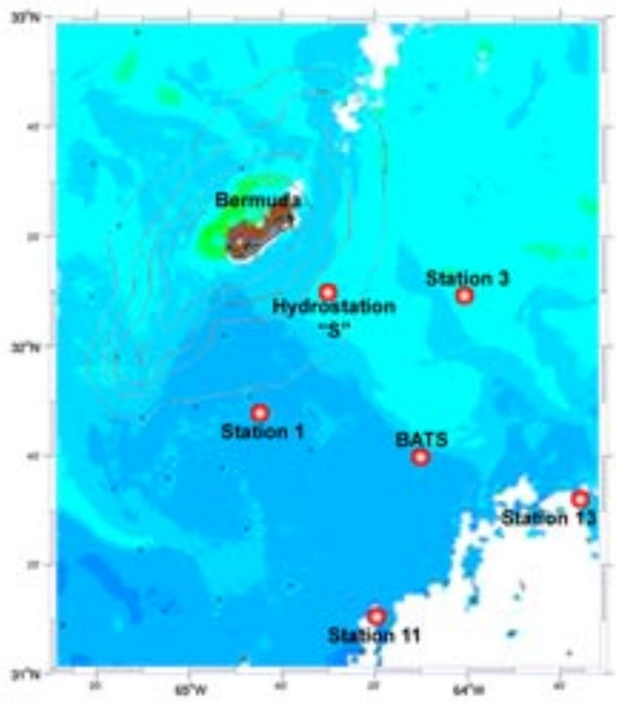
Placing each repeat edge together gives a clear progression of the path through the entire sequence.

Find a path visiting every EDGE exactly once:
Eulerian path problem

Metagenomics:

C. Venter et al., Exploring the Sargasso Sea:

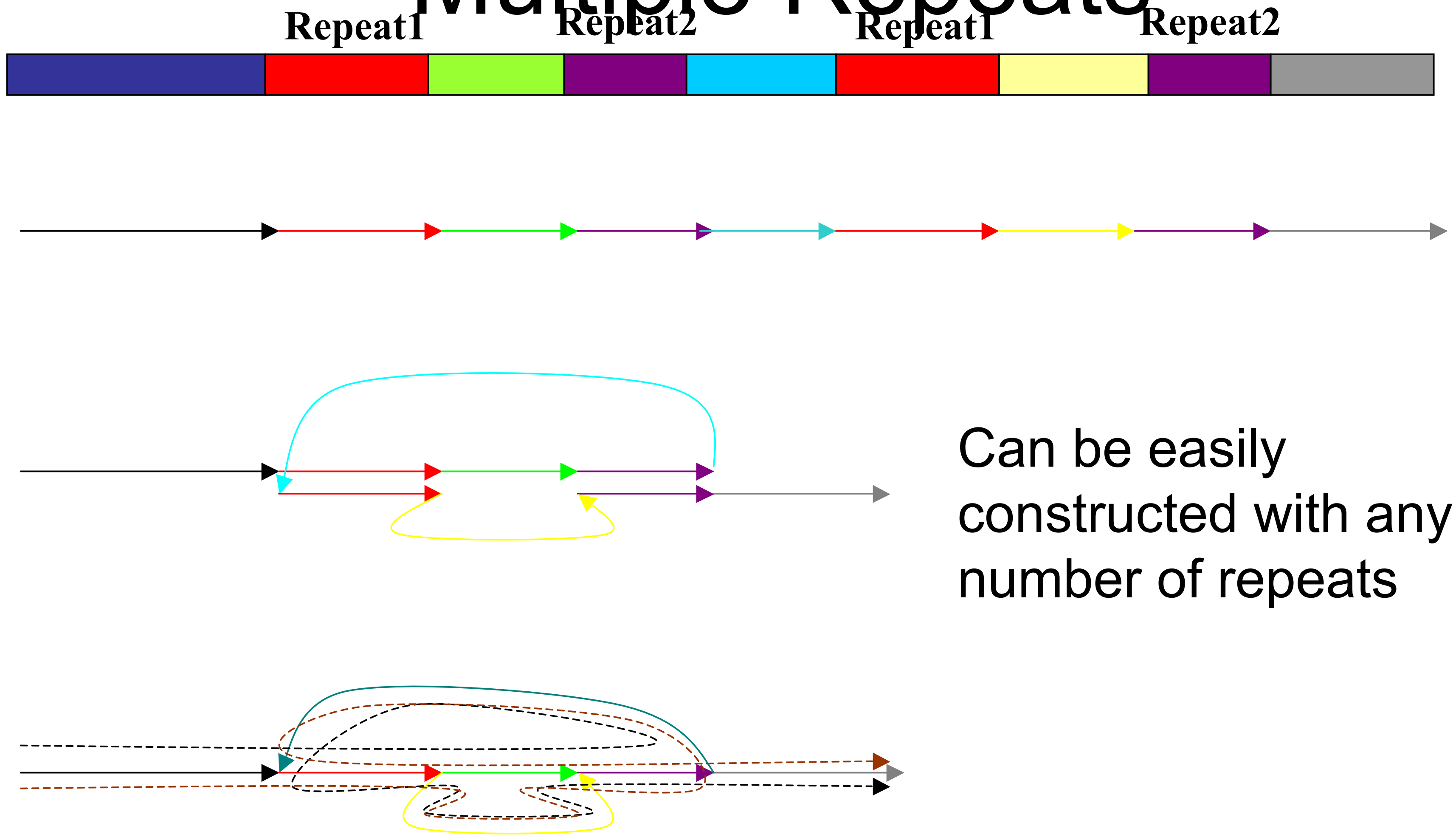
Scientists Discover One Million New Genes in Ocean Microbes



Conclusions

- Graph theory is a vital tool for solving biological problems
- Wide range of applications, including sequencing, motif finding, protein networks, and many more

Multiple Repeats



Can be easily constructed with any number of repeats

Construction of Repeat Graph

- Construction of repeat graph from *k* – mers: emulates an SBH experiment with a huge (virtual) DNA chip.
- Breaking reads into *k* – mers: Transform sequencing data into virtual DNA chip data.

Construction of Repeat Graph
(cont’d)

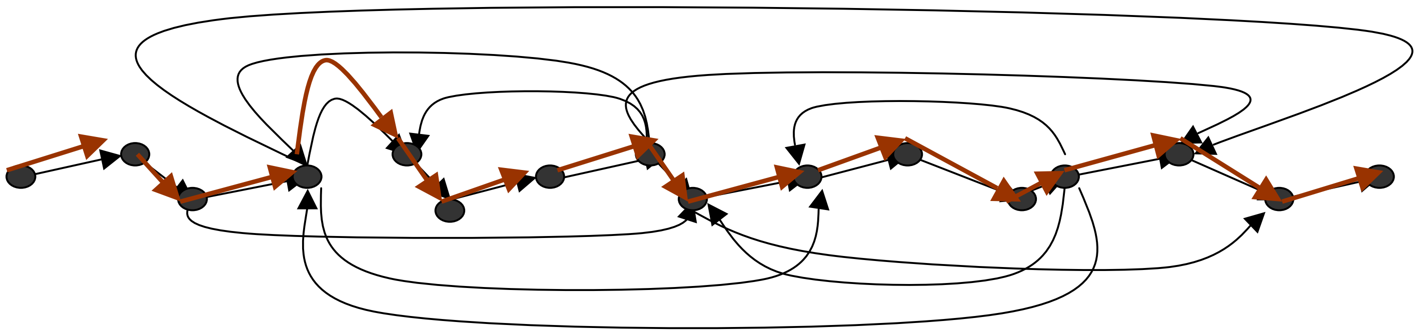
- Error correction in reads: “consensus first” approach to fragment assembly. Makes reads (almost) error-free BEFORE the assembly even starts.

- Using reads and mate-pairs to simplify the repeat graph (Eulerian Superpath Problem).

Approaches to Fragment Assembly

Find a path visiting every VERTEX exactly once in the OVERLAP graph:

Hamiltonian path problem

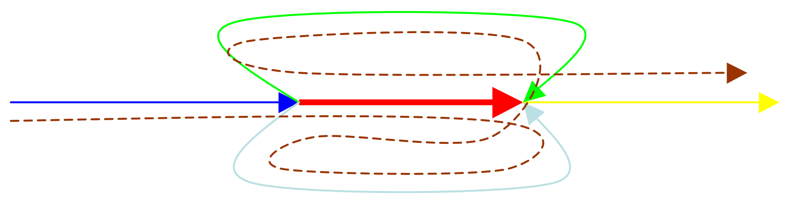


NP-complete: algorithms unknown

Approaches to Fragment Assembly (cont'd)

Find a path visiting every EDGE exactly once in the REPEAT graph:

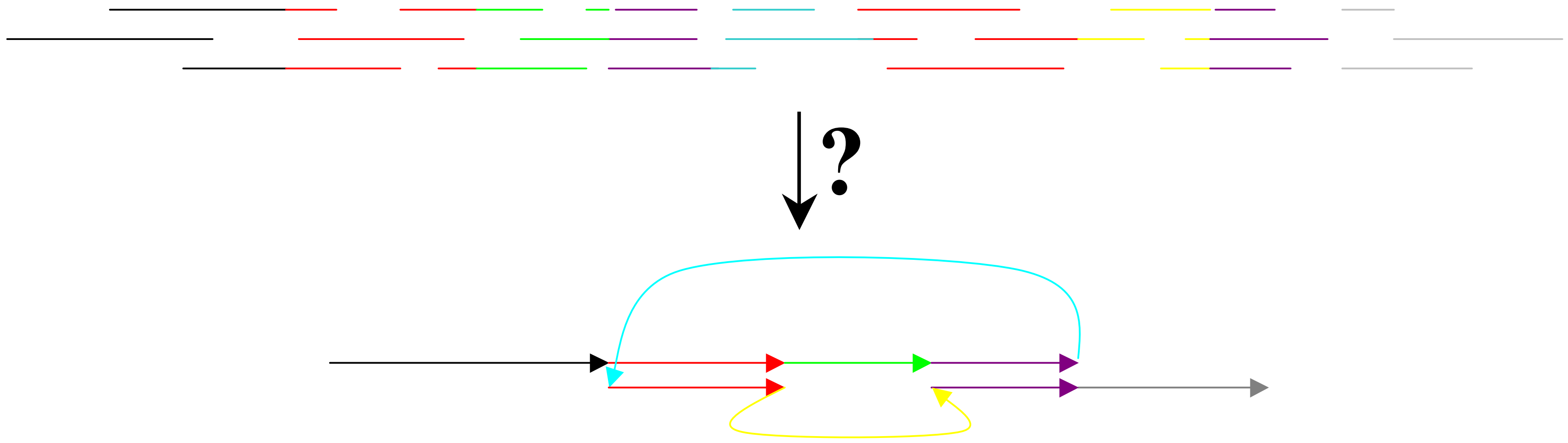
Eulerian path problem



Linear time algorithms are known

Making Repeat Graph Without DNA

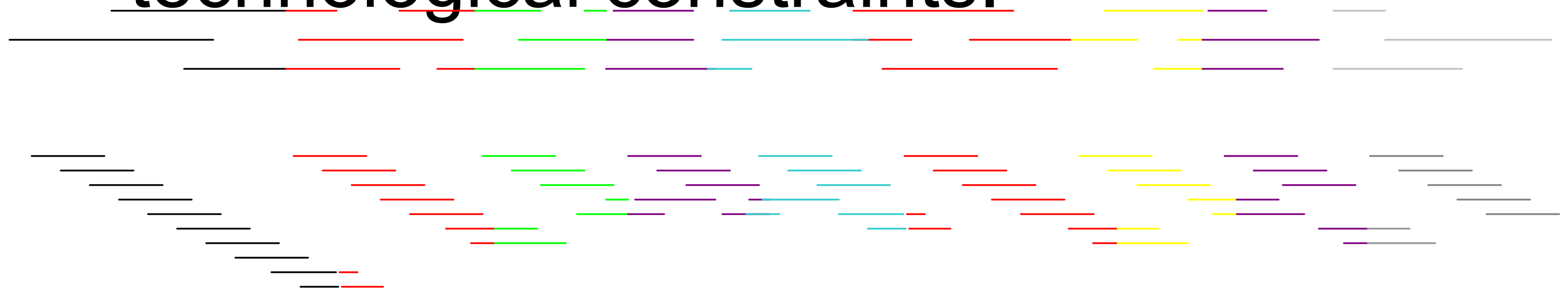
- Problem: Construct the repeat graph from a collection of reads.



- Solution: Break the reads into smaller pieces.

Repeat Sequences: Emulating a DNA Chip

- Virtual DNA chip allows the biological problem to be solved within the technological constraints.

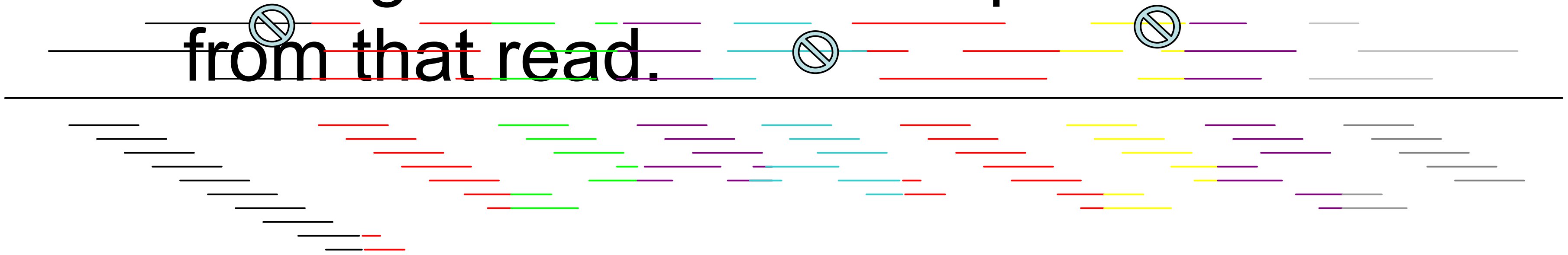


Repeat Sequences: Emulating a DNA Chip (cont’d)

- Reads are constructed from an original sequence in lengths that allow biologists a high level of certainty.
- They are then broken again to allow the technology to sequence each within a reasonable array.

Minimizing Errors

- If an error exists in one of the 20-mer reads, the error will be perpetuated among all of the smaller pieces broken from that read.



Minimizing Errors (cont’d)

- However, that error will not be present in the other instances of the 20-mer read.
- So it is possible to eliminate most point mutation errors before reconstructing the original sequence.