أكاديمية كاوست
KAUST ACADEMY

Advanced Bioinformatics

Instructor :Sakhaa Alsaedi
Sakhaa.Alsaedi@kaust.edu.sa

TA: Ebtihal Hani

Day 3: How Do We Assemble Genomes?

27th Feb. 2024

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

Outlines

- Overview of Genome Assembly and Sequencing
- Computational problem of genome assembly
- Hamiltonian paths and universal strings
- String Reconstruction as an Eulerian Path Problem
- De Bruijn Graphs
- Euler’s Theorem
- Assembling Read-Pair
- De Bruijn Graphs Face Harsh Realities of Assembly

References

Text books

- **Chapter 3, Bioinformatics Algorithms, Phillip Compeau and Pavel Pevzner**

Slide information:

- [Slides:Chapter 3: How Do We Assemble Genomes?]
  - [The String Reconstruction Problem]

Outlines

- Overview of Genome Assembly and Sequencing
- Computational Problem of Genome Assembly
- Hamiltonian paths and universal strings
- String Reconstruction as an Eulerian Path Problem
- De Bruijn Graphs
- Euler’s Theorem
- Assembling Read-Pair
- De Bruijn Graphs Face Harsh Realities of Assembly

Brief History of Genome Sequencing

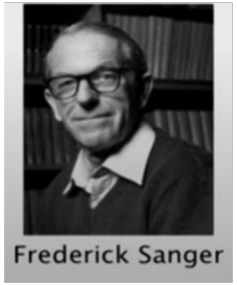
أكاديمية كاوست
KAUST ACADEMY

- 1977: Walter Gilbert and Frederick Sanger develop independent DNA sequencing methods.


Walter Gilbert

- 1980: They share the Nobel Prize.

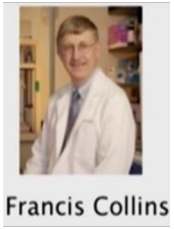
- Still, their sequencing methods were too expensive ($3 billion to sequence the human genome).


Frederick Sanger

The Race to Sequence the Human Genome

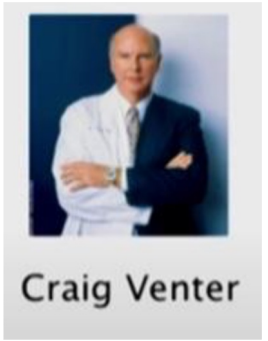
أكاديمية كاوست
KAUST ACADEMY

- 1990: The public Human Genome Project, headed by Francis Collins, aims to sequence the human genome by 2005.

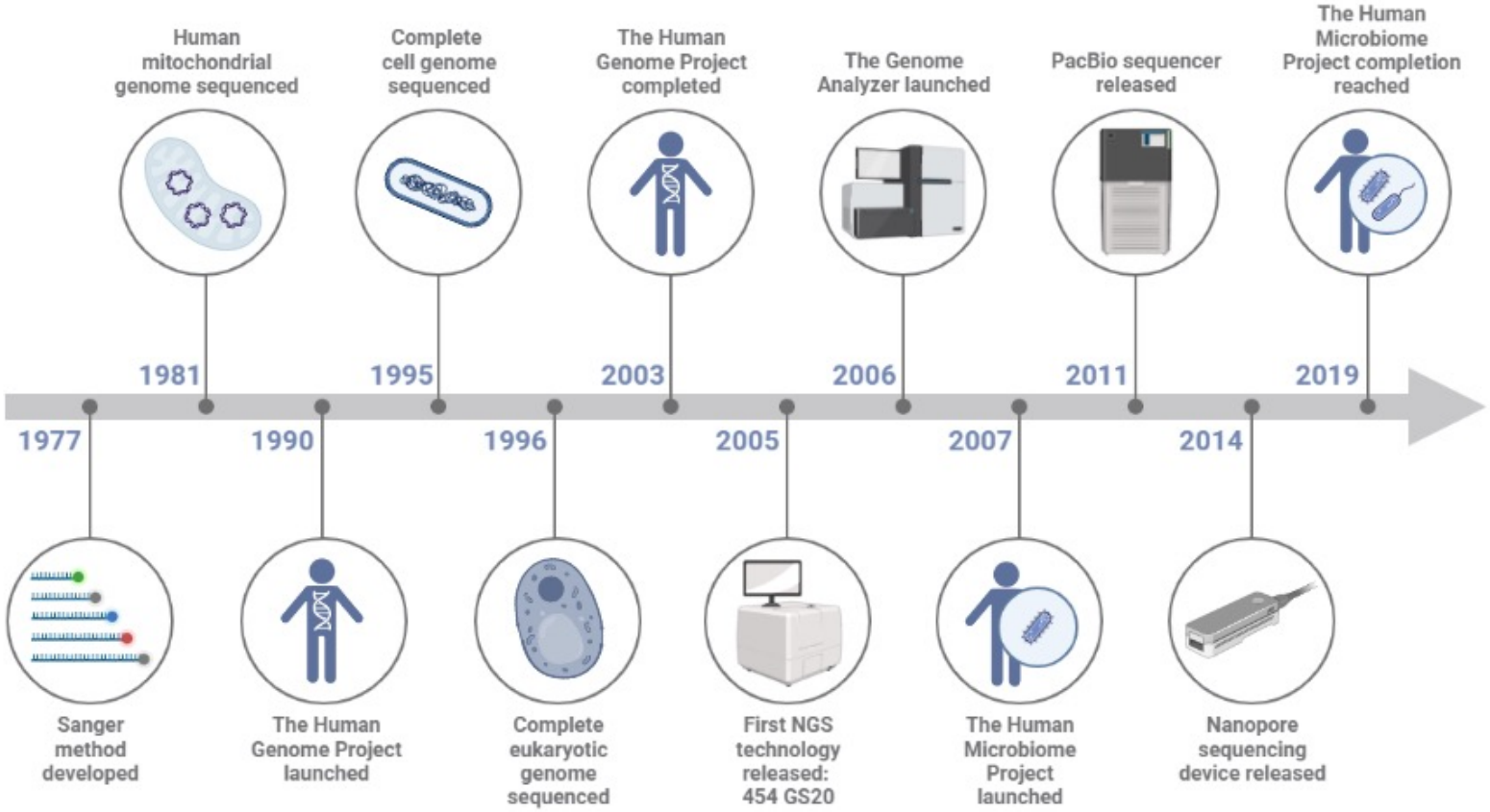

Francis Collins

- 1997: Craig Venter founds Celera Genomics, a private firm, with the same goal.

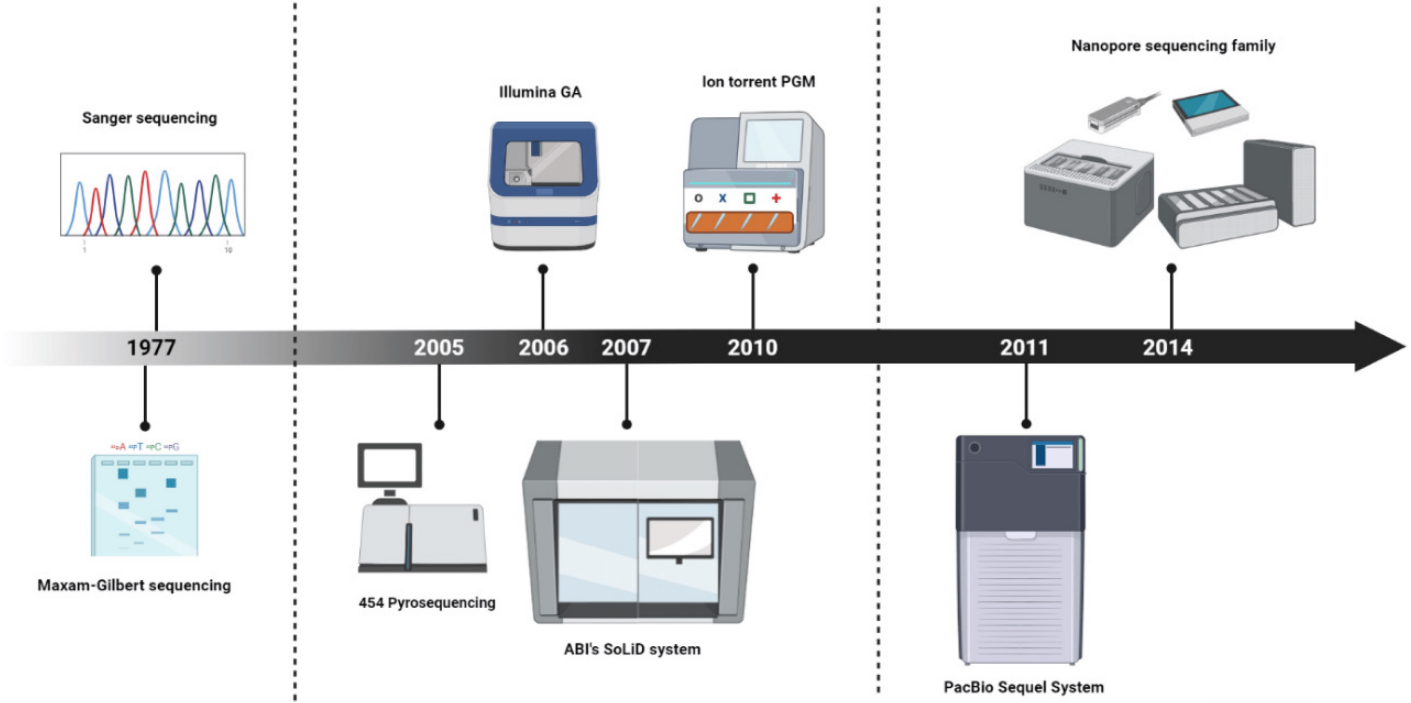
- 2000:


Craig Venter

History of Sequencing Technology



History of Sequencing Technology



First-generation sequencing

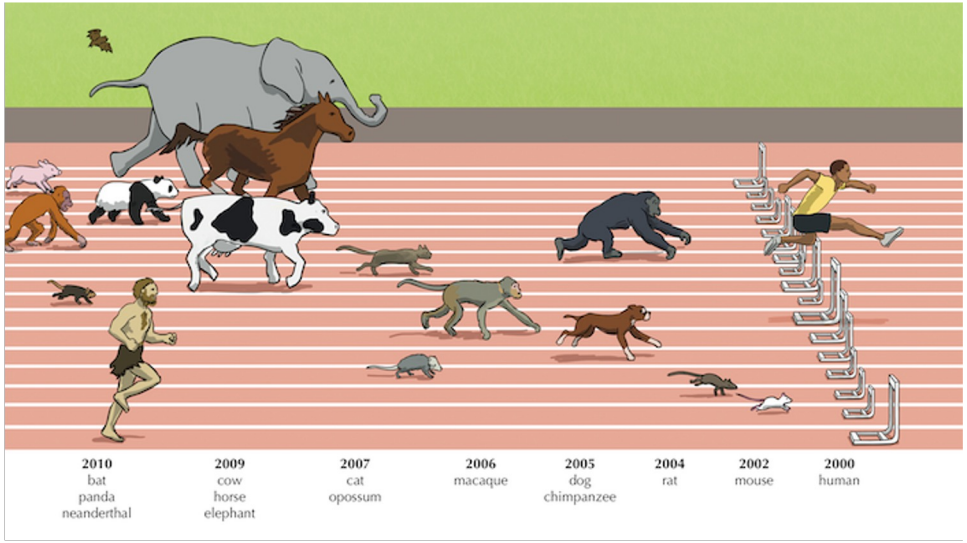
Next-generation sequencing (NGS)

Third-generation sequencing

From Human to Mouse to Rat to ...

أكاديمية كاوست
KAUST ACADEMY

- Early 2000s: Many more mammalian genomes are sequenced using the same Sanger sequencing method, but it is clear that new technology is needed for further progress



Next Generation Sequencing Technologies

أكاديمية كاوست
KAUST ACADEMY

- Late 2000s: The market for new sequencing machines takes off.

  - Illumina reduces the cost of sequencing a human genome from $3 billion to $10,000.

  - Complete Genomics builds a genomic factory in Silicon Valley that sequences hundreds of genomes per month.

  - Beijing Genome Institute orders hundreds of sequencing machines, becoming the world's largest sequencing center."

Personal Genome Sequencing

أكاديمية كاوست
KAUST ACADEMY

Why Do We Sequence Genomes?

Think and share your thoughts :)

جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

Why Do We Sequence Personal Genomes?

أكاديمية كاوست
KAUST ACADEMY

- 2010: Nicholas Volker became the first human being to be saved by genome sequencing.
  - Doctors could not diagnose his condition; he went through dozens of surgeries.
  - Sequencing revealed a rare mutation in a XIAP gene linked to a defect in his immune system.
  - This led doctors to use immunotherapy, which saved the child.

Nicholas Volker

Saudi gene hunters comb country's DNA to prevent rare diseases

Research could help prevent disorders that result from marriages between relatives

8 DEC 2016 · BY JOCELYN KAISER

Few Mutations Can Make a Big Difference...

أكاديمية كاوست
KAUST ACADEMY

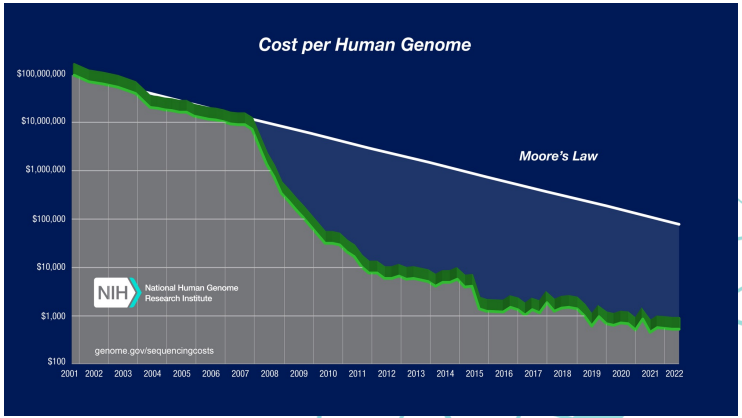
- Different people have slightly different genomes: on average, roughly 1 mutation in 1000 nucleotides.
- The 1 in 1000 nucleotides difference accounts for height, high cholesterol susceptibility, and 1000s of genetic diseases.

Few Mutations Can Make a Big Difference...

أكاديمية كاوست
KAUST ACADEMY

- 2010: Scientists launch a project to sequence 10,000 vertebrate genomes.

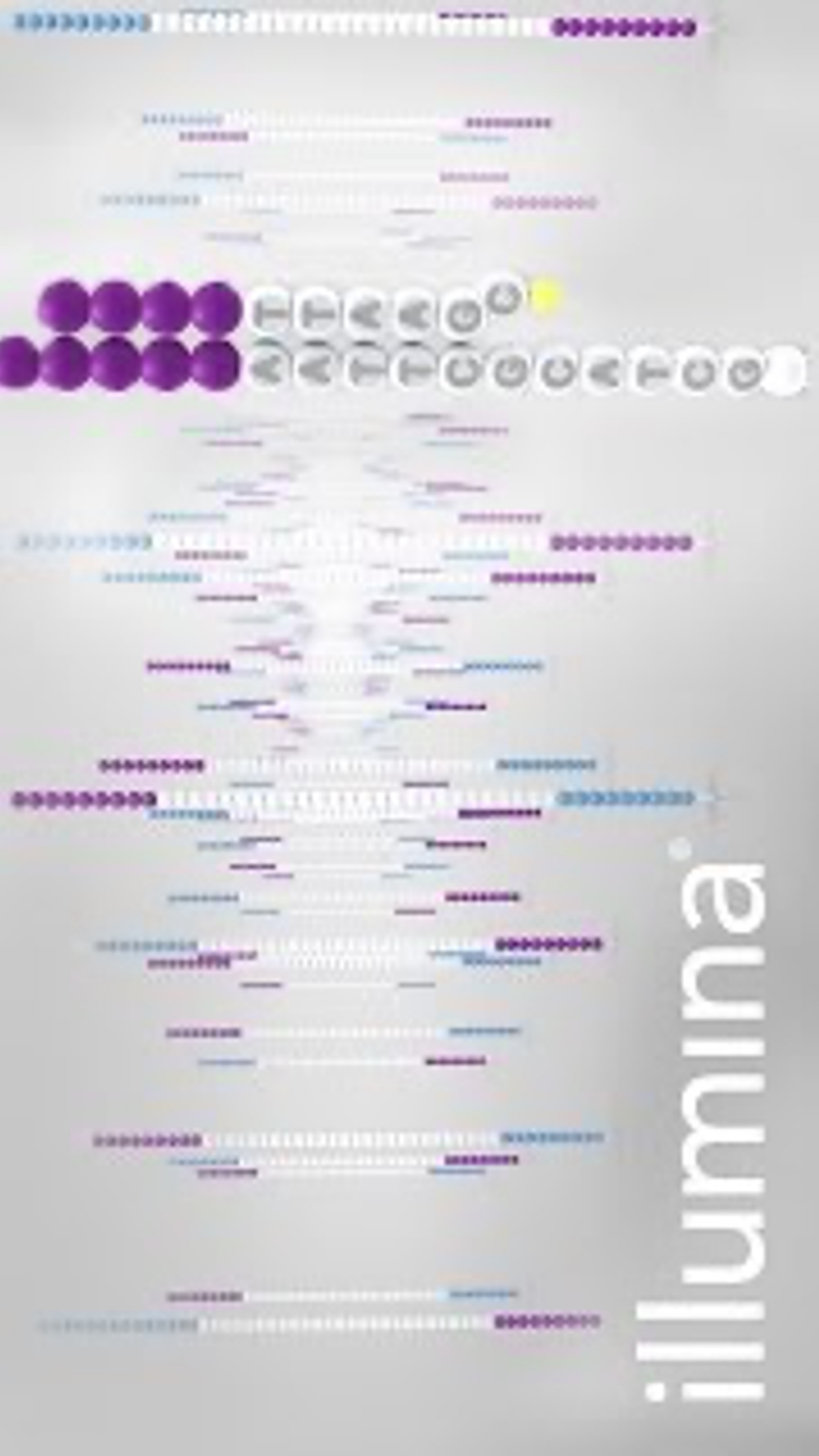
- Now: Human genome sequencing costs just a few thousand dollars and under $1,000 human genomes may arrive any day now.



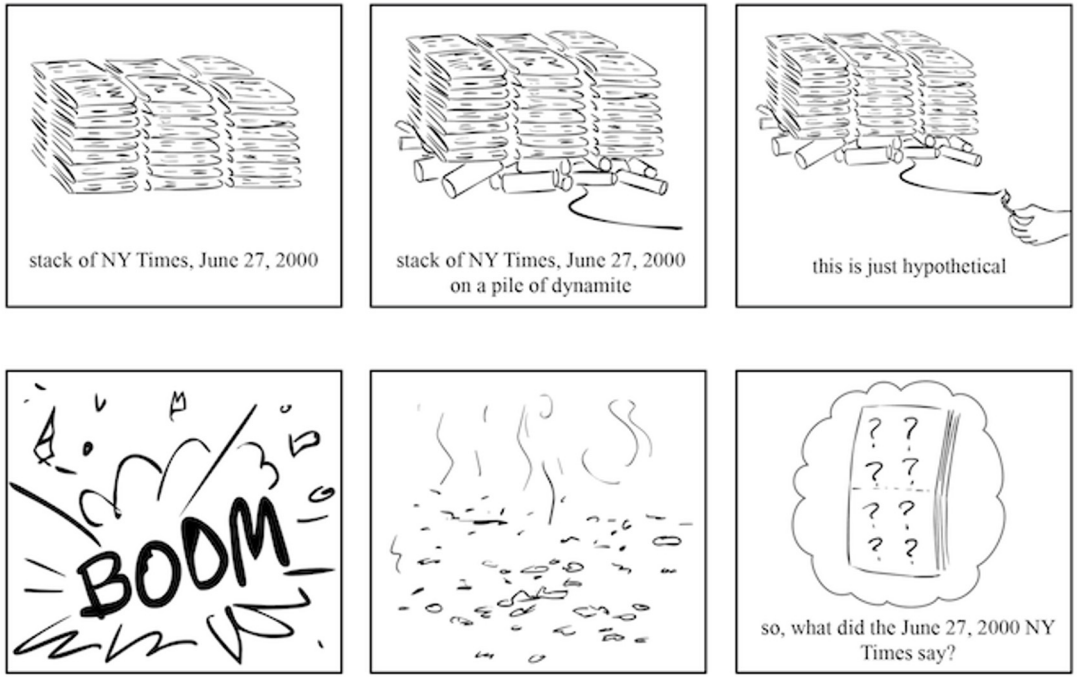
أكاديمية كاوست
KAUST ACADEMY

Bioinformatics in Saudi Arabia

Saudi Human reference genome project

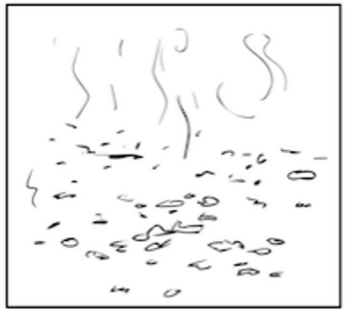

illumina

Genome Assembly Problem ← → The Newspaper Problem



The Newspaper Problem as an Overlapping Puzzle

أكاديمية كاوست
KAUST ACADEMY



atshirt, appr
e have not yet named a
nation is welc

shirt, approximately 6'2" 18
yet named any suspects
is welcomed. Please cal

Multiple Copies of a Genome (Millions of them)

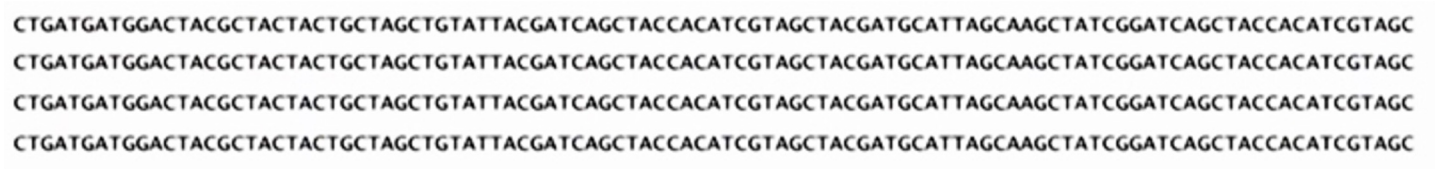
أكاديمية كاوست
KAUST ACADEMY

stack of NY Times, June 27, 2000

CTGATGATGGACTACGCTACTACTGCTAGCTGTATTACGATCAGCTACCACATCGTAGCTACGATGCATTAGCAAGCTATCGGATCAGCTACCACATCGTAGC
CTGATGATGGACTACGCTACTACTGCTAGCTGTATTACGATCAGCTACCACATCGTAGCTACGATGCATTAGCAAGCTATCGGATCAGCTACCACATCGTAGC
CTGATGATGGACTACGCTACTACTGCTAGCTGTATTACGATCAGCTACCACATCGTAGCTACGATGCATTAGCAAGCTATCGGATCAGCTACCACATCGTAGC
CTGATGATGGACTACGCTACTACTGCTAGCTGTATTACGATCAGCTACCACATCGTAGCTACGATGCATTAGCAAGCTATCGGATCAGCTACCACATCGTAGC

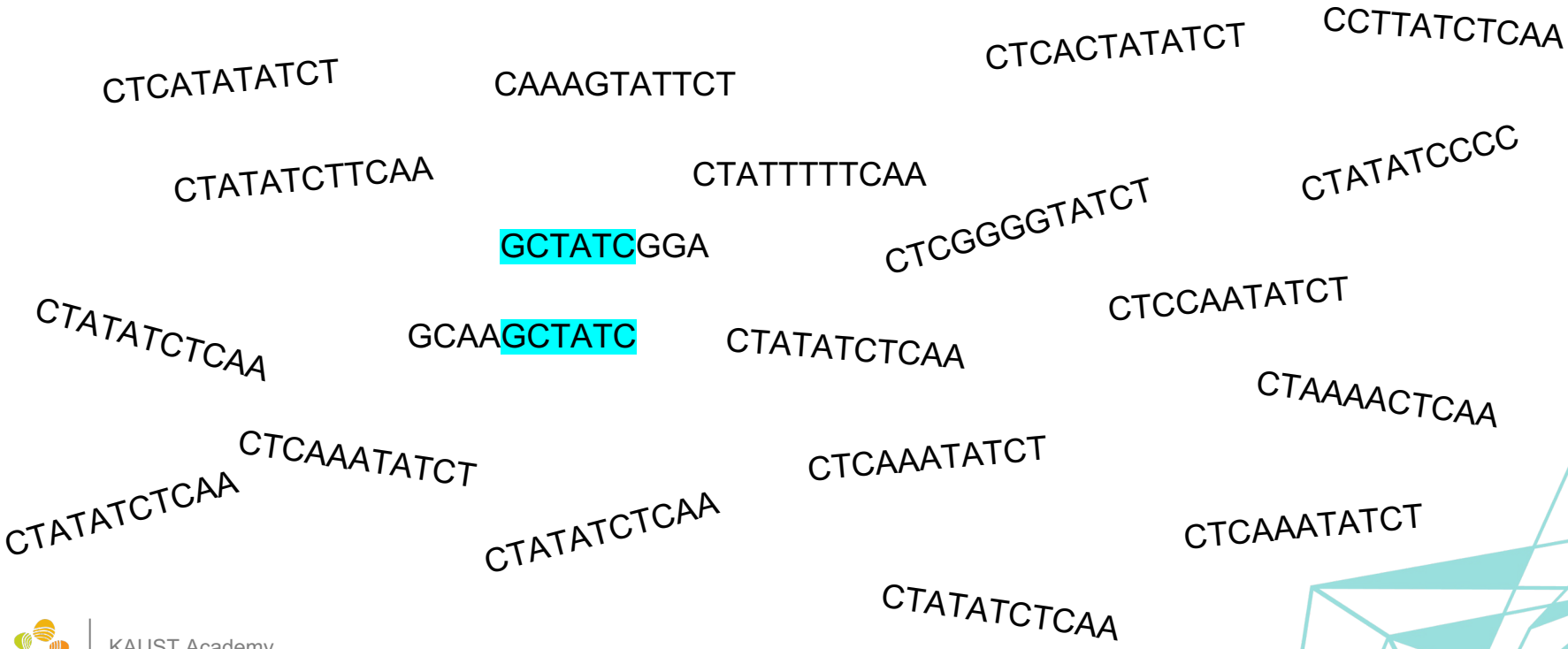
Determining the order of nucleotides in a genome, or **genome sequencing**, presents a fundamental task in bioinformatics.

Generating 'Reads'

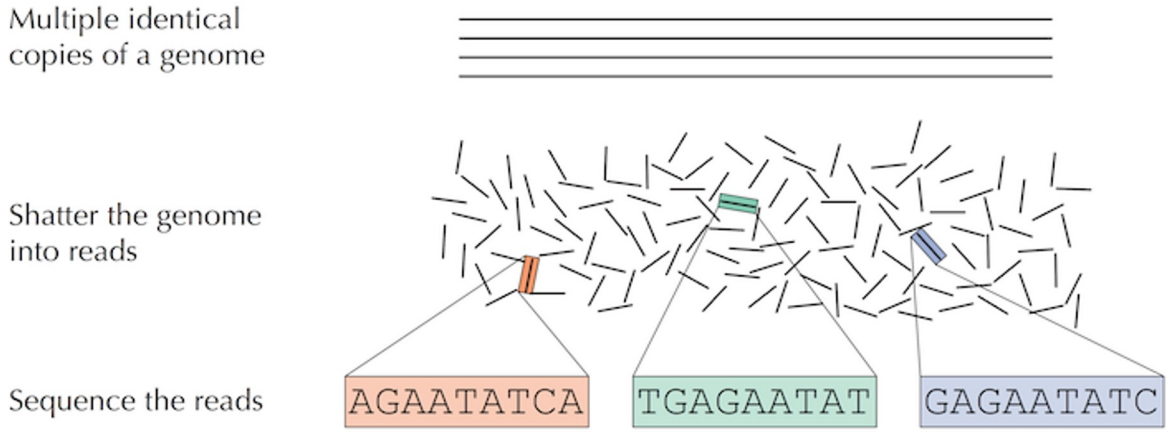


The largest obstacle is the fact that biologists still lack the technology to read the nucleotides of a genome from beginning to end in the same way that you would read a book. The best they can do is sequence much shorter DNA fragments called **reads**.

No Idea What Position Every Read Comes From



From Experimental to Computational Challenges



What Makes Genome Sequencing Difficult?

أكاديمية كاوست
KAUST ACADEMY

- Modern sequencing machines cannot read an entire genome one nucleotide at a time from beginning to end (like we read a book)

- They can only shred the genome and generate short reads.

- The genome assembly is not the same as a jigsaw puzzle: we must use overlapping reads to reconstruct the genome, a giant overlap puzzle!

Outlines

- Overview of Genome Assembly and Sequencing
- Computational Problem of Genome Assembly
- Hamiltonian paths and universal strings
- String Reconstruction as an Eulerian Path Problem
- De Bruijn Graphs
- Euler’s Theorem
- Assembling Read-Pair
- De Bruijn Graphs Face Harsh Realities of Assembly

Computational Problem of Genome Assembly

أكاديمية كاوست
KAUST ACADEMY

- The String Composition Problem
- The String Reconstruction Problem
- The String Reconstruction Problem as Hamiltonian Path Problem
- The String Reconstruction as an Eulerian Path Problem

Computational Problem of Genome Assembly

أكاديمية كاوست
KAUST ACADEMY

- The String Composition Problem
- The String Reconstruction Problem
- The String Reconstruction Problem as Hamiltonian Path Problem
- The String Reconstruction as an Eulerian Path Problem

What is 3-mer Composition?

Composition(TAATGCCATGGGATGTT)=

TAA
AAT
ATG
TGC
GCC
CCA
CAT
ATG
TGG
GGG
GGA
GAT
ATG
TGT
GTT
TAATGCCATGGGATGTT

What is k-mer Composition?

$Composition_3$(TAATGCCATGGGATGTT) =

TAA AAT ATG TGC GCC CCA CAT ATG TGG GGG GGA GAT ATG TGT GTT

=

AAT ATG ATG ATG CAT CCA GAT GCC GGA GGG GTT TAA TGC TGG TGT

e.g., lexicographic order (like in a dictionary)

Reconstructing strings from *k*-mers

Given a string *Text*, its ***k*-mer composition** $Composition_k(Text)$ is the collection of all *k*-mer substrings of *Text* (including repeated *k*-mers). For example,

$Composition_3$(TATGGGGTGC) =

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

Note that we have listed *k*-mers in **lexicographic order**

String Reconstruction Problem:

In turn, AAT can only be extended by ATG, which can only be extended by TGT, and so on, leading us to reconstruct **TAATGTT**:

TAA
AAT
ATG
TGT
GTT
TAATGTT

String Reconstruction Problem

It looks like we are finished with the String Reconstruction Problem and can let you move on to the next chapter. To be sure, let's consider an additional 3-mer composition:

AAT ATG ATG ATG CAT CCA GAT GCC GGA GGG GTT TAA TGC TGG TGT

STOP and Think: Reconstruct a string with this composition.

String Reconstruction Problem

If we start again with TAA, then the next 3-mer in the string should start with AA, and there is only one such 3-mer, AAT. In turn, AAT can only be extended by ATG.

TAA
AAT
ATG
TAATG

String Reconstruction Problem

ATG can be extended either by TGC, or TGG, or TGT. Now we must decide which of these 3-mers to choose. Let's select TGT:

```
TAA
 AAT
  ATG
   TGT
TAATGT
```

String Reconstruction Problem

After TGT, our only choice is GTT:

```
TAA
 AAT
  ATG
   TGT
    GTT
TAATGTT
```

Unfortunately, now we are stuck at GTT because no 3-mers in the composition start with TT! We could try to extend TAA to the left, but no 3-mers in the composition end with TA.

String Reconstruction Problem:

أكاديمية كاوست
CADEMY

You may have found this trap on your own and already discovered how to escape it. Like a good chess player, if you think a few steps ahead, then you would never extend ATG by TGT until reaching the end of the genome. With this thought in mind, let's take a step back, extending ATG by TGC instead:

```
TAA
 AAT
  ATG
   TGC
TAATGC
```

String Reconstruction Problem:

Continuing the process, we obtain the following assembly:

```
TAA
 AAT
  ATG
   TGC
    GCC
     CCA
      CAT
       ATG
        TGG
         GGA
          GAT
           ATG
            TGT
             GTT
TAATGCCATGGATGTT
```

Yet this assembly is incorrect because we have only used fourteen of the fifteen 3-mers in the composition (we omitted GGG), making our reconstructed genome one nucleotide too short.

Is this the only solution to the String Reconstruction Problem for this collection of 3-mers?

Code Challenge: Solve the String Composition Problem.

أكاديمية كاوست
KAUST ACADEMY

String Composition Problem: *Generate the k-mer composition of a string.*
Input: An integer *k* and a string *Text*.
Output: $Composition_k(Text)$.

Bioinformatics Challenges

GitHub

- **3A: Generate the k-mer Composition of a String**
- 3B: Reconstruct a String from its Genome Path
- 3C: Construct the Overlap Graph of a Collection of k-mers
- 3D: Construct the De Bruijn Graph of a String
- 3E: Construct the De Bruijn Graph of a Collection of k-mers
- 3G: Find an Eulerian Path in a Graph
- 3H: Reconstruct a String from its k-mer Composition
- 3I: Find a k-Universal Circular String

What is k-mer Composition?

In the figure below, consecutive 3-mers in **TA**ATGCCATGGGATGTT are linked together to form this string's **genome path**.

Figure: The fifteen color-coded 3-mers making up **TA**ATGCCATGGGATGTT are joined into the genome path according to their order in the genome.

String Spelled by a Genome Path Problem.

→ *Reconstruct a string from its genome path.*

أكاديمية كاوست
KAUST ACADEMY

Input: A sequence *path* of *k*-mers $Pattern_1, \ldots, Pattern_n$ such that the last $k - 1$ symbols of $Pattern_i$ are equal to the first k-1 symbols of $Pattern_{i+1}$ for $1 \le i \le n$-1.
Output: A string *Text* of length $k+n-1$ such that the i-th k-mer in *Text* is equal to $Pattern_i$ (for $1 \le i \le n$).

Reconstructing the genome from its genome path is easy: as we proceed from left to right, the 3-mers "spell' out **TA**ATGCCATGGGATGTT, adding one new symbol to the genome at each new 3-mer. This yields a function **PathToGenome**(*path*)**.**

Bioinformatics Challenges

GitHub

- 3A: Generate the k-mer Composition of a String
- **3B: Reconstruct a String from its Genome Path**
- 3C: Construct the Overlap Graph of a Collection of k-mers
- 3D: Construct the De Bruijn Graph of a String
- 3E: Construct the De Bruijn Graph of a Collection of k-mers
- 3G: Find an Eulerian Path in a Graph
- 3H: Reconstruct a String from its k-mer Composition
- 3I: Find a k-Universal Circular String

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

Outlines

- Overview of Genome Assembly and Sequencing
- Computational problem of genome assembly
- Hamiltonian paths and universal strings
- String Reconstruction as an Eulerian Path Problem
- De Bruijn Graphs
- Euler’s Theorem
- Assembling Read-Pair
- De Bruijn Graphs Face Harsh Realities of Assembly

Representing a Genome as a Path

Composition(TAATGCCATGGGATGTT)=

Can we construct this genome path without knowing the genome **TAATGCCATGGGATGTT**, only from its composition?

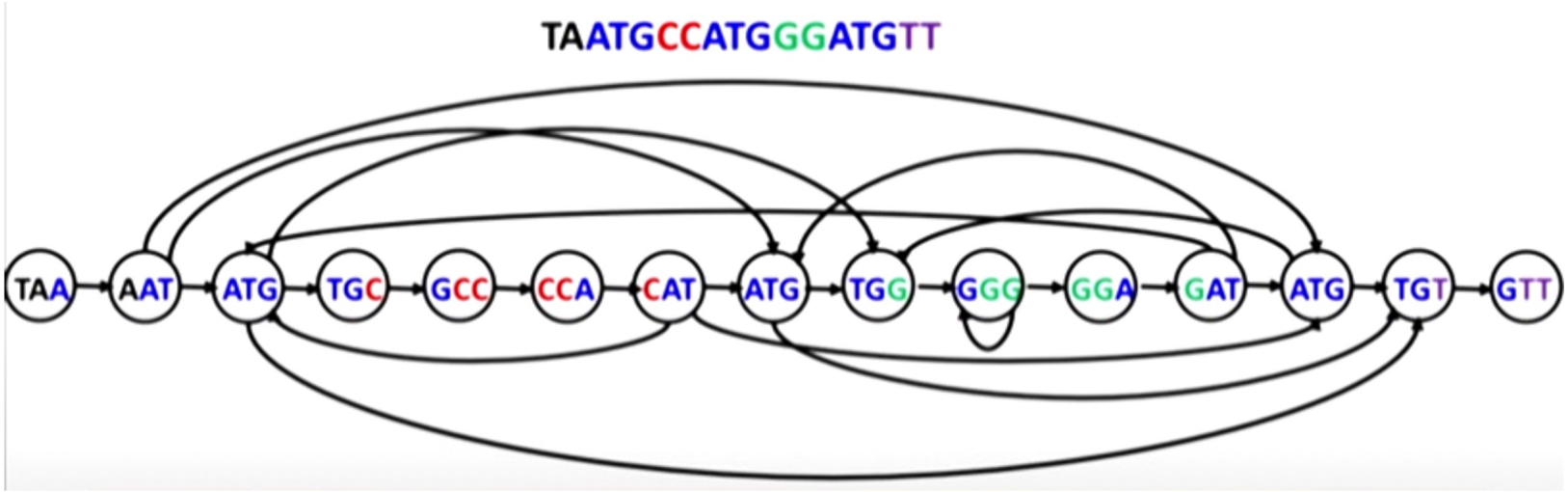
Yes. We simply need to connect k-mer$_1$ with k-mer$_2$ if ***suffix***(k-mer$_1$)=***prefix***(k-mer$_2$).
E.g. **TAA** → **AAT**

A Path Turns into a Graph

TAATGCCATGGGATGTT

Yes. We simply need to connect k-mer$_1$ with k-mer$_2$ if ***suffix***(k-mer$_1$)=***prefix***(k-mer$_2$).
E.g. **TAA** → **AAT**

A Path Turns into a Graph



Yes. We simply need to connect k-mer$_1$ with k-mer$_2$ if ***suffix***(k-mer$_1$)=***prefix***(k-mer$_2$).

E.g. **TAA** → **AAT**

Summary: String Reconstruction as a Walk in the Overlap Graph

We will use the terms **prefix** and **suffix** to refer to the first $k - 1$ nucleotides and last $k - 1$ nucleotides of a k-mer, respectively. For example, *Prefix*(**TAA**) = **TA** and *Suffix*(**TAA**) = **AA**. We note that the suffix of a 3-mer in the genome path is equal to the prefix of the following 3-mer in the path. For example, *Suffix*(**TAA**) = *Prefix*(**AAT**) = **AA** in the genome path for **TAATGCCATGGGATGTT**, shown again below.

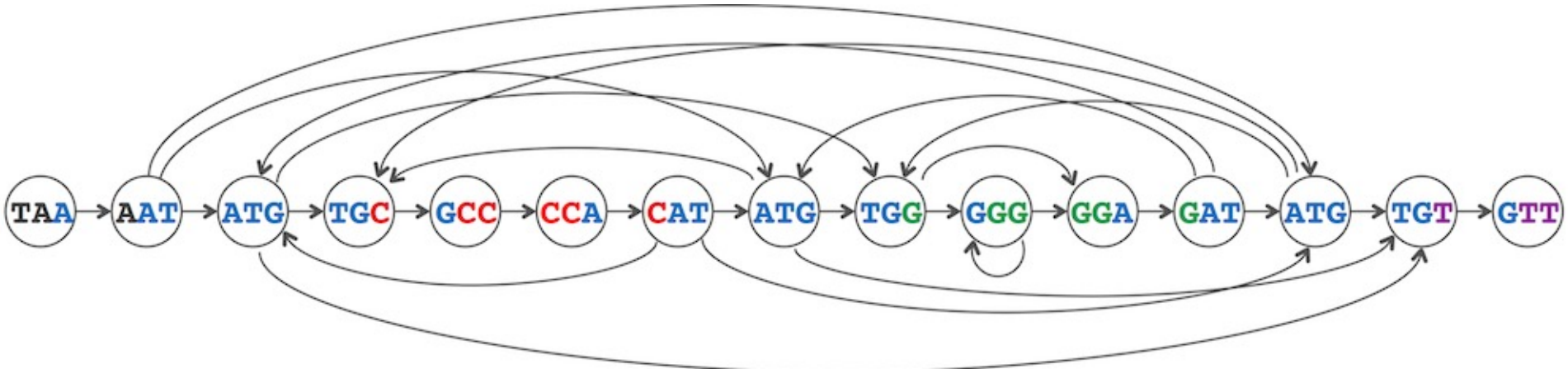
String Reconstruction as a Walk in the Overlap Graph

This observation suggests a method of constructing a string's genome path from its *k*-mer composition: we will use an arrow to connect any *k*-mer *Pattern* to a *k*-mer *Pattern'* if the suffix of *Pattern* is equal to the prefix of *Pattern'*.

STOP and Think: Apply this rule to the 3-mer composition of **TA**ATGCCATGGGATGTT. Are you able to reconstruct the genome path for **TA**ATGCCATGGGATGTT?

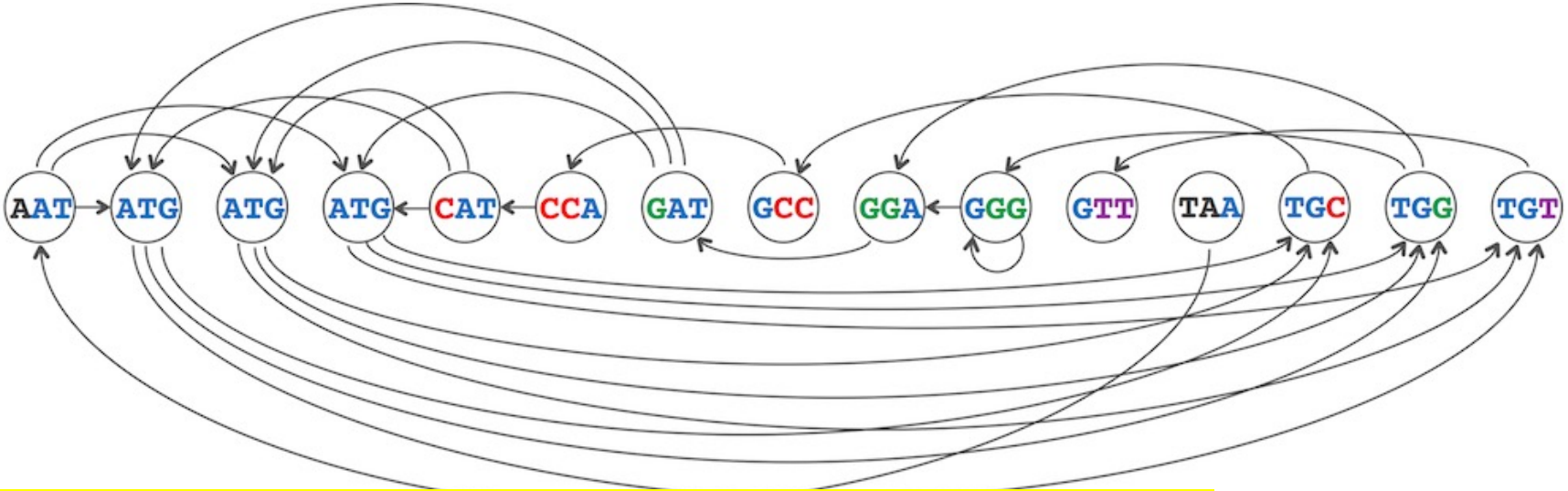
String Reconstruction as a Walk in the Overlap Graph

If we strictly follow the rule of connecting two 3-mers with an arrow every time the suffix of one is equal to the prefix of the other, then we will connect all consecutive 3-mers in TAATGCCATGGGATGTT. However, because we don't know this genome in advance, we wind up having to connect many other pairs of 3-mers as well. For example, each of the three occurrences of ATG should be connected to TGC, TGG, and TGT, as shown in the figure below.



String Reconstruction as a Walk in the Overlap Graph

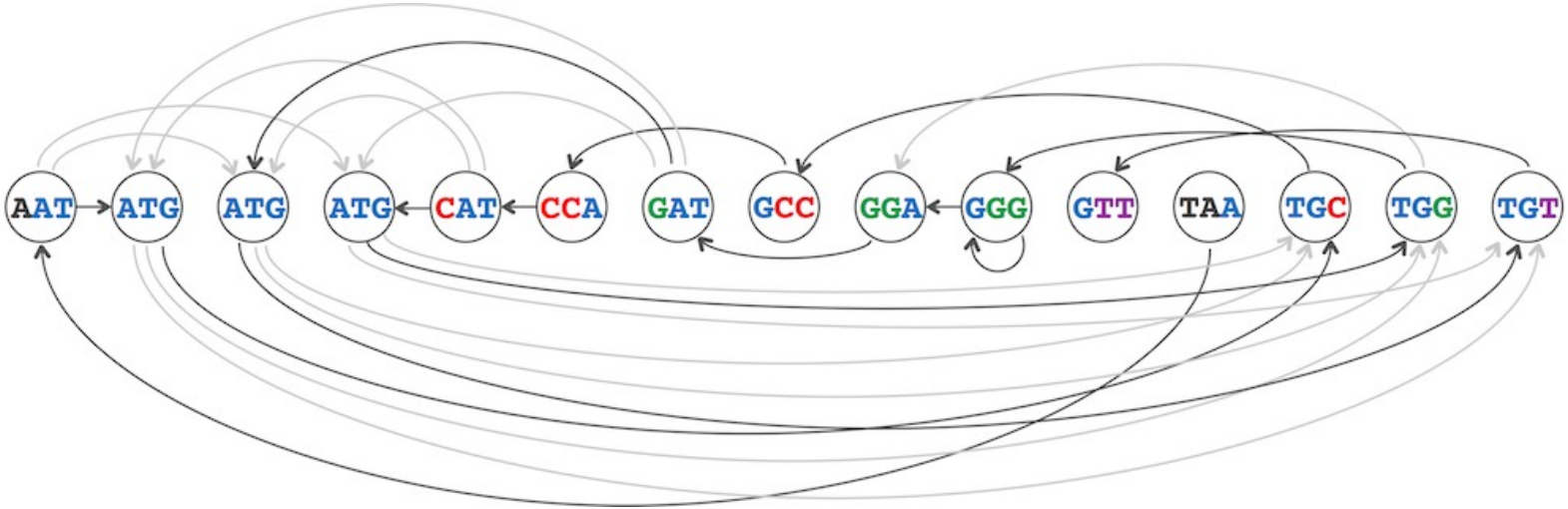
In genome sequencing, we do not know in advance how to correctly order reads. Therefore, we will arrange the 3-mers lexicographically, which produces the overlap graph shown in the figure below.



The path through the graph representing the correct assembly is now harder to see.

String Reconstruction as a Walk in the Overlap Graph

Although finding such a path is currently just as difficult as trying to assemble the genome by hand, the graph nevertheless gives us a nice way of visualizing the overlap relationships between reads.



The genome path spelling out **TA**ATGCCATGGGATGTT, highlighted in the overlap graph.

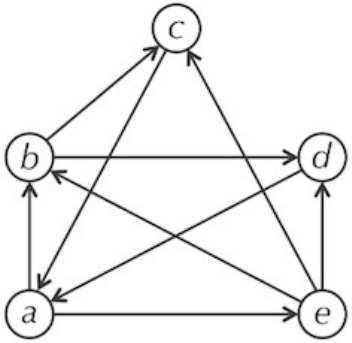
String Reconstruction Problem: as a Walk in the Overlap Graph

Overlap Graph Problem: *Construct the overlap graph of a collection of k-mers.*
Input: A collection Patterns of *k*-mers.
Output: The overlap graph *Overlap*(*Patterns*).

Two graph representations

There are two standard ways of representing a graph. For a directed graph with n nodes, the $n \times n$ **adjacency matrix** ($A_{i,j}$) is defined by the following rule: $A_{i,j}$ = 1 if a directed edge connects node i to node j, and $A_{i,j}$ = 0 otherwise. Another (more memory-efficient) way of representing a graph is to use an **adjacency list**, for which we simply list all nodes connected to each node. See figure below.

Graph



Adjacency Matrix

	a	*b*	*c*	*d*	*e*
a	0	1	0	0	1
b	0	0	1	1	0
c	1	0	0	0	0
d	1	0	0	0	0
e	0	1	1	1	0

Adjacency List

a is adjacent to *b* and *e*
b is adjacent to *c* and *d*
c is adjacent to *a*
d is adjacent to *a*
e is adjacent to *b*, *c*, and *d*

Code Challenge: Solve the Overlap Graph Problem

Input: A collection *Patterns* of *k*-mers.
Output: The overlap graph *Overlap*(*Patterns*), in the form of an adjacency list. (You may return the nodes and their edges in any order.)

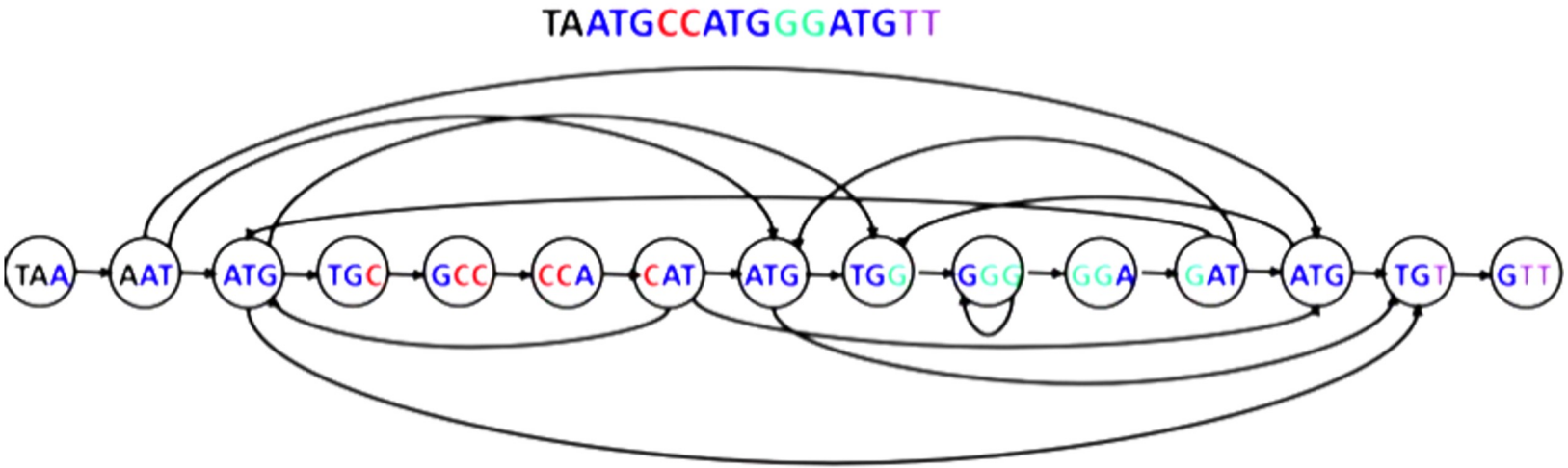
Bioinformatics Challenges

GitHub

- 3A: Generate the k-mer Composition of a String
- 3B: Reconstruct a String from its Genome Path
- **3C: Construct the Overlap Graph of a Collection of k-mers**
- 3D: Construct the De Bruijn Graph of a String
- 3E: Construct the De Bruijn Graph of a Collection of k-mers
- 3G: Find an Eulerian Path in a Graph
- 3H: Reconstruct a String from its k-mer Composition
- 3I: Find a k-Universal Circular String

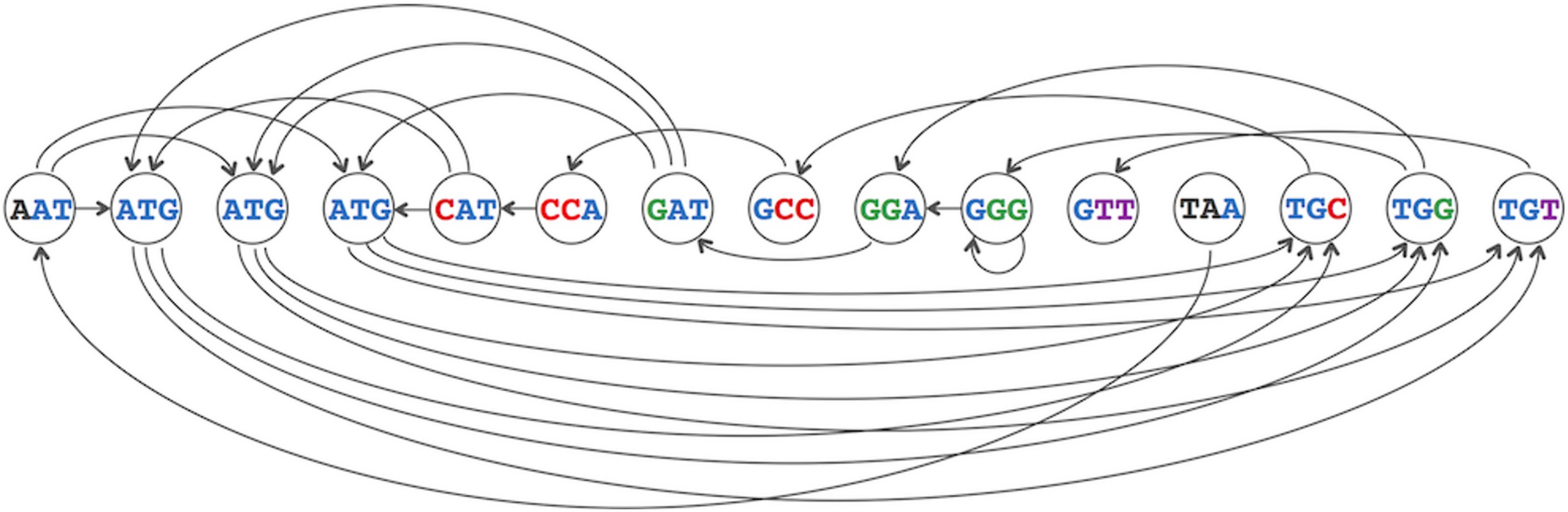
جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

A Path Turns into a Graph



Can We still find the genome path in this graph?

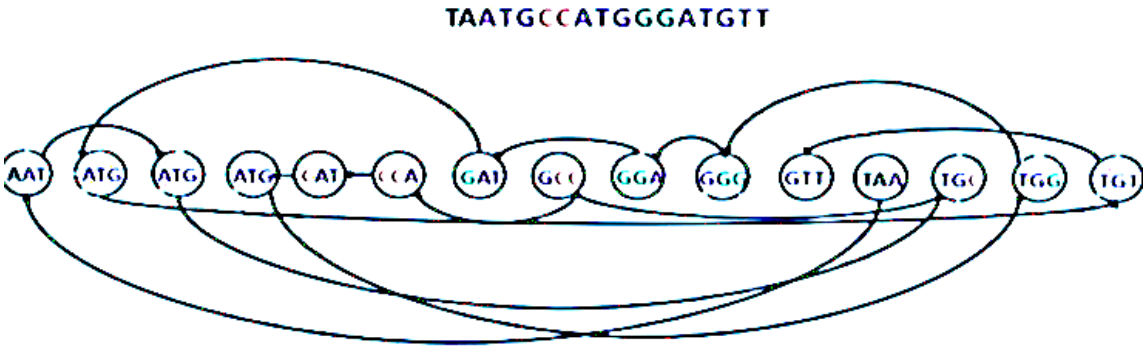
Where is the Genomic Path?



Nodes are arranged from left to right in lexicographic order

Where is the Genomic Path?

Hamiltonian Path: a path that visits each node in a graph exactly one

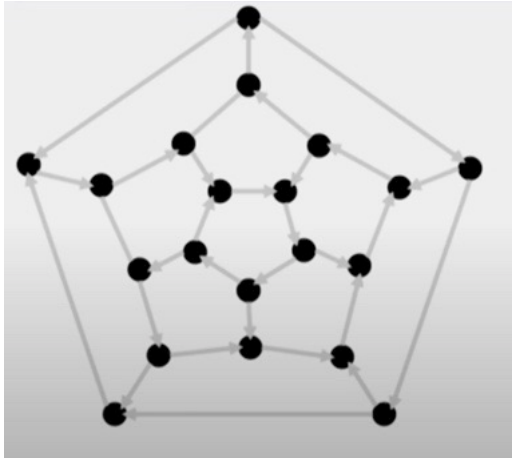


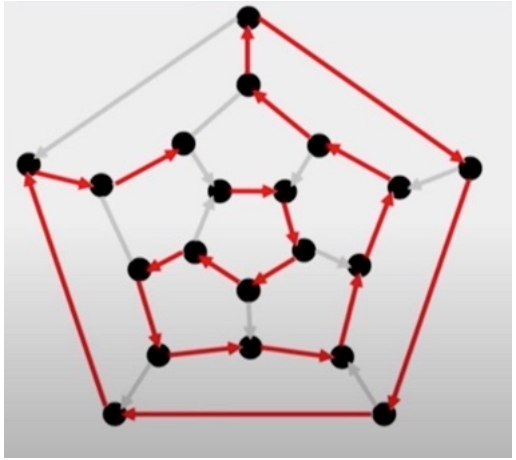
What are we trying to find in this graph?

Hamiltonian paths and universal strings

Hamiltonian Path Problem: *Construct a Hamiltonian path in a graph.*

- **Input:** A directed graph.
- **Output:** A path visiting every **node** in the graph exactly once





Hamiltonian paths and universal strings

A **binary string** is a string composed only of 0's and 1's; a binary string is ***k*-universal** if it contains every binary *k*-mer exactly once.

For example:

0001110100 is a 3-universal string, as it contains each of the eight binary 3-mers (000, 001, 011, 111, 110, 101, 010, and 100) exactly once.

- Finding a *k*-universal string is equivalent to solving the String Reconstruction Problem when the *k*-mer composition is the collection of all binary *k*-mers. Thus, finding a *k*-universal string can be reduced to finding a Hamiltonian path in the overlap graph formed on all binary *k*-mers

Hamiltonian paths and universal strings

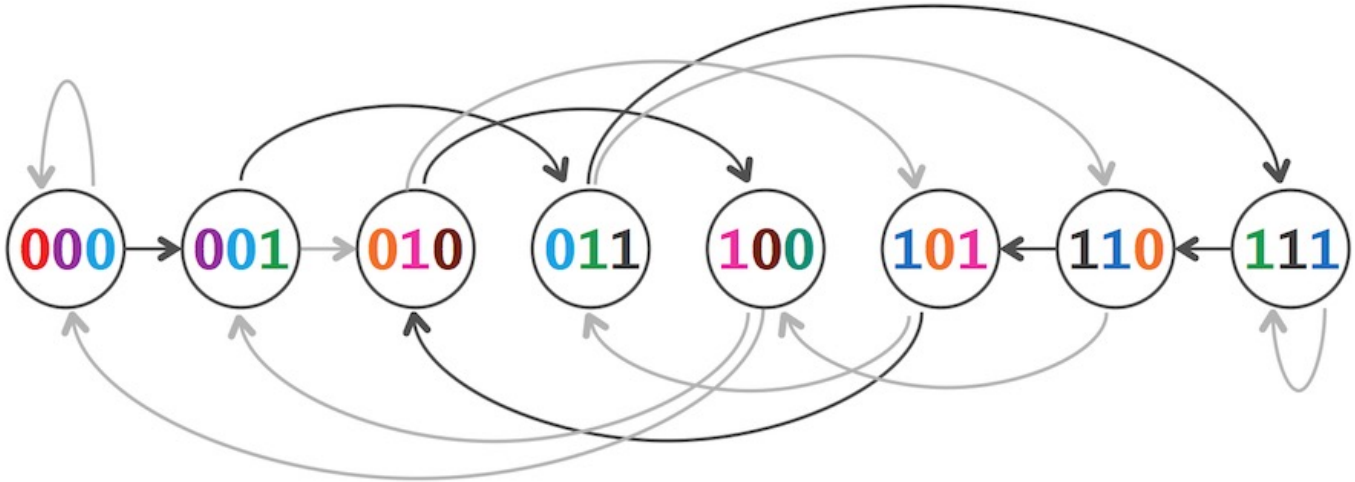
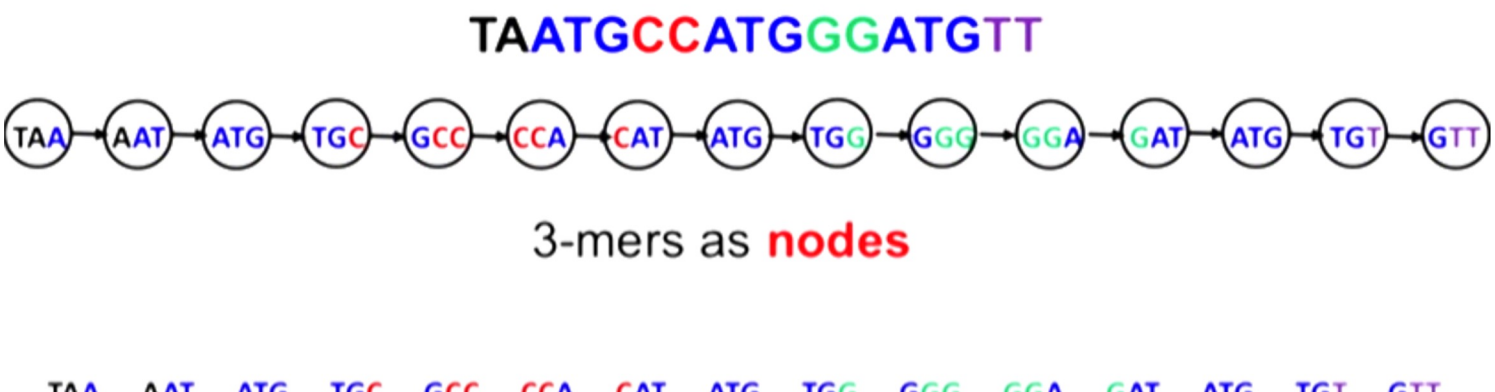


Figure: A Hamiltonian path (connecting node 000 to 100) in the overlap graph of all binary 3-mers.

Outlines

- Overview of Genome Assembly and Sequencing
- Computational Problem of Genome Assembly
- Hamiltonian paths and universal strings
- String Reconstruction as an Eulerian Path Problem
- De Bruijn Graphs
- Euler's Theorem
- Assembling Read-Pair
- De Bruijn Graphs Face Harsh Realities of Assembly

A Slightly Different Path



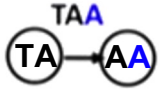
A Slightly Different Path

TAATGCCATGGGATGTT

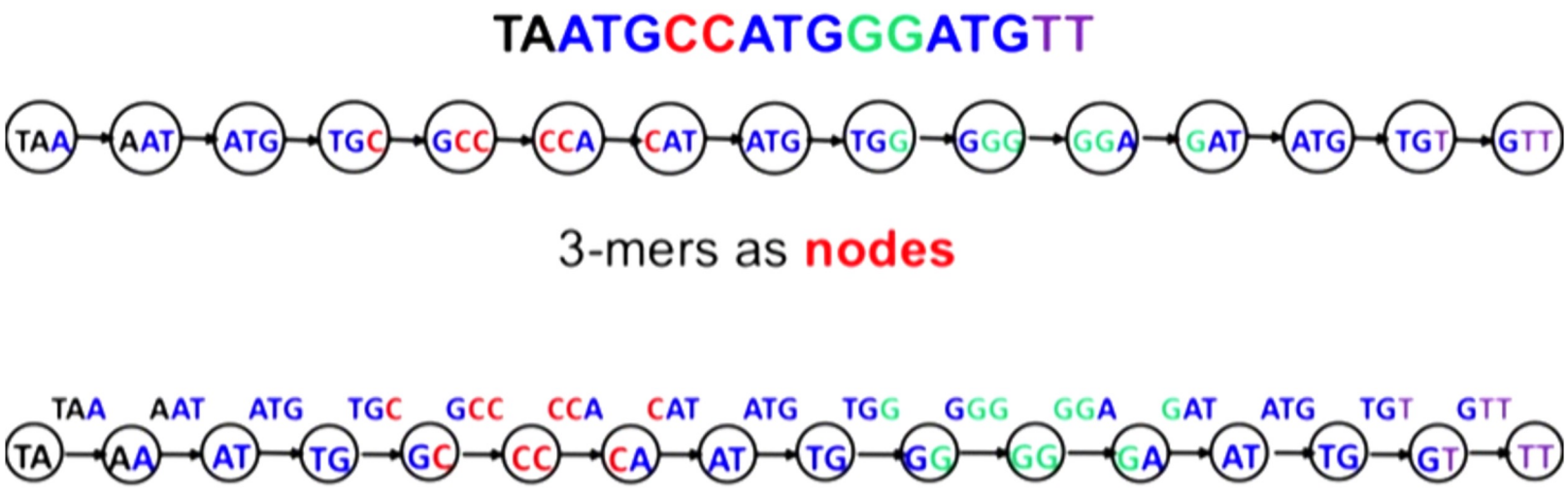
3-mers as **nodes**

3-mers as **edges**

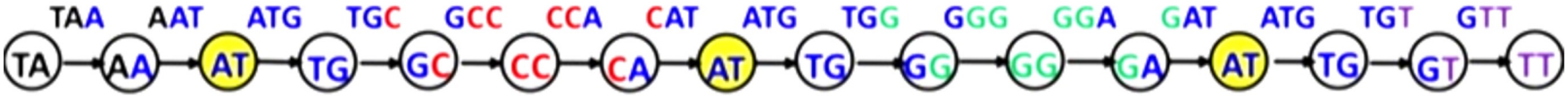
How do we label the starting and ending nodes of an edge?

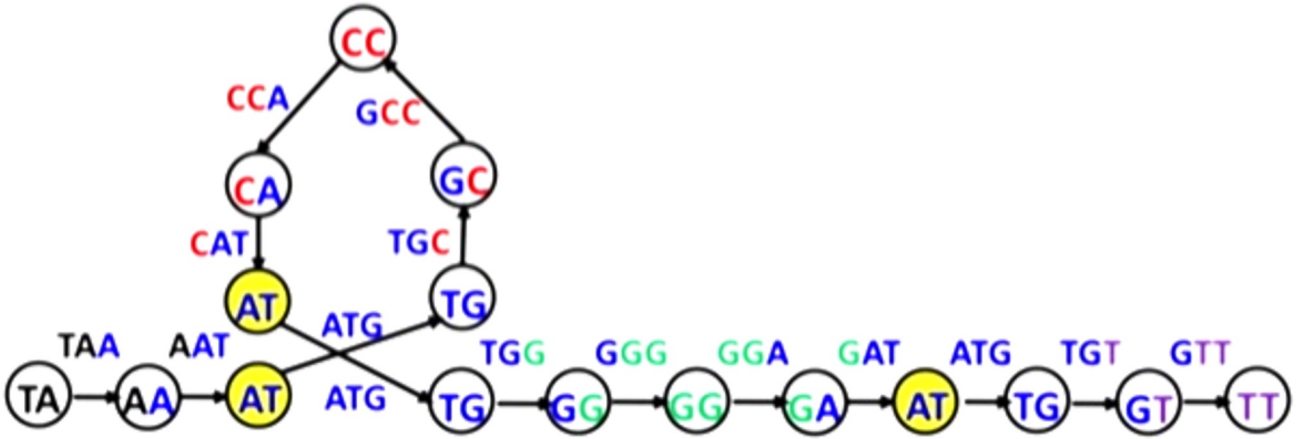
Prefix of TAA  **Suffix of TAA**

Labeling Node in the New Path

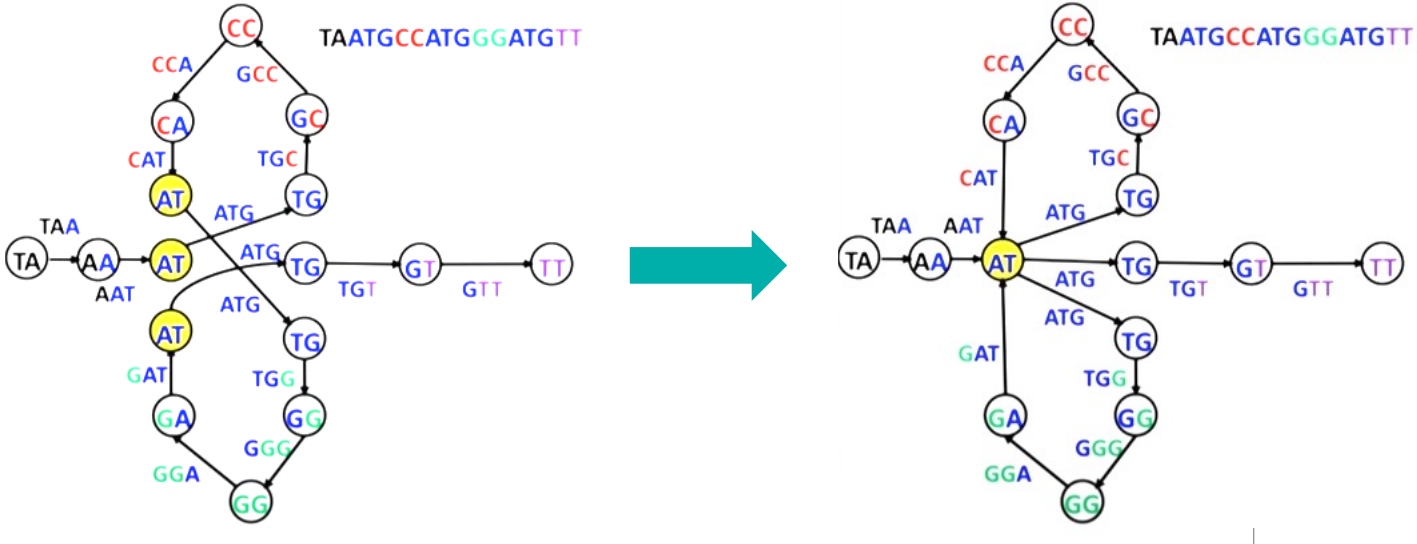


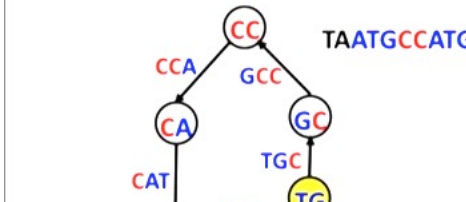
Gluing Identically Labeled Nodes



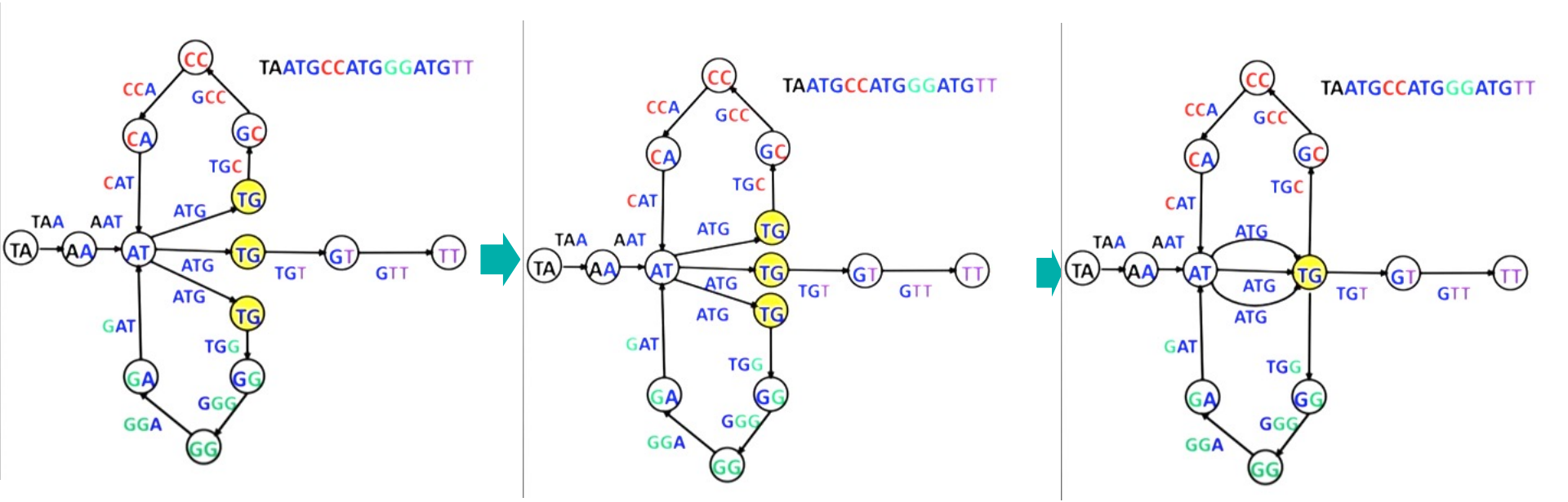


Gluing Identically Labeled Nodes

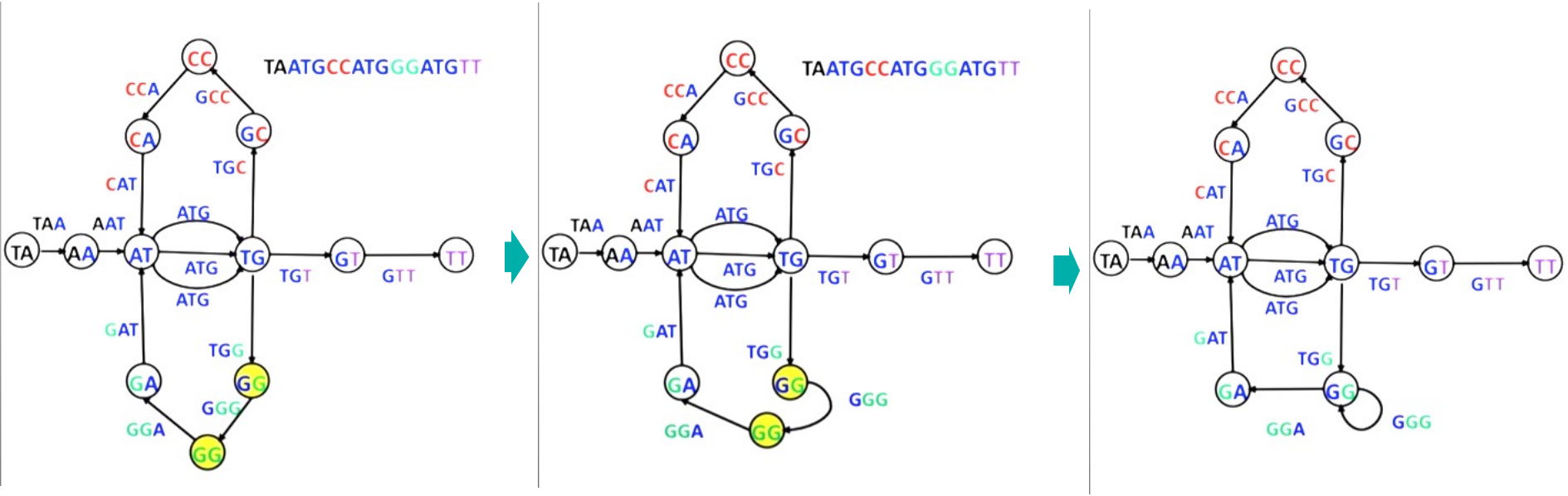




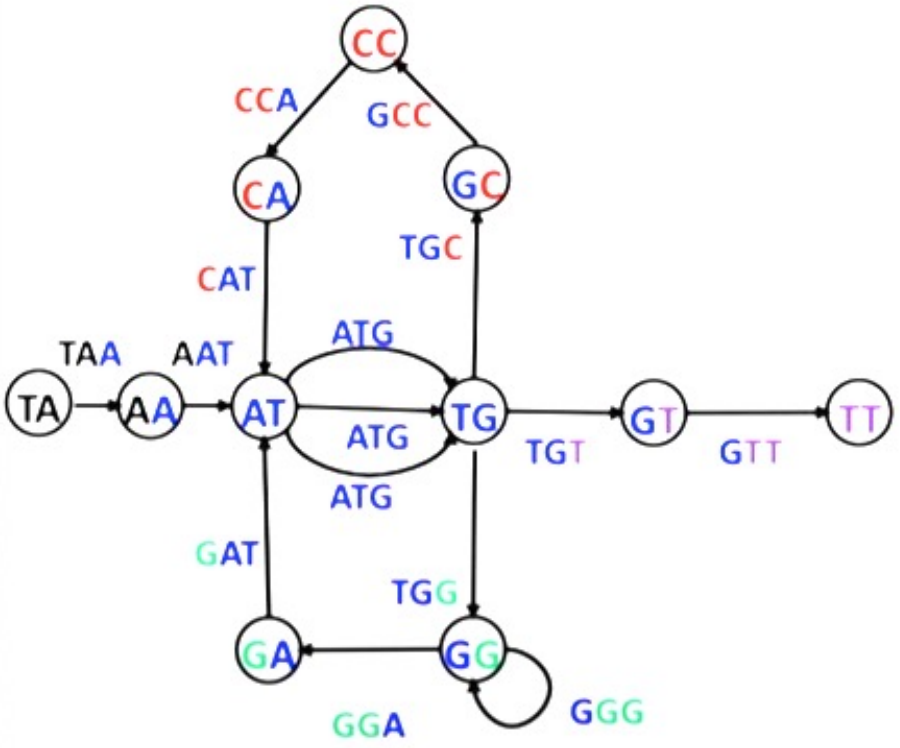
Gluing Identically Labeled Nodes



Gluing Identically Labeled Nodes

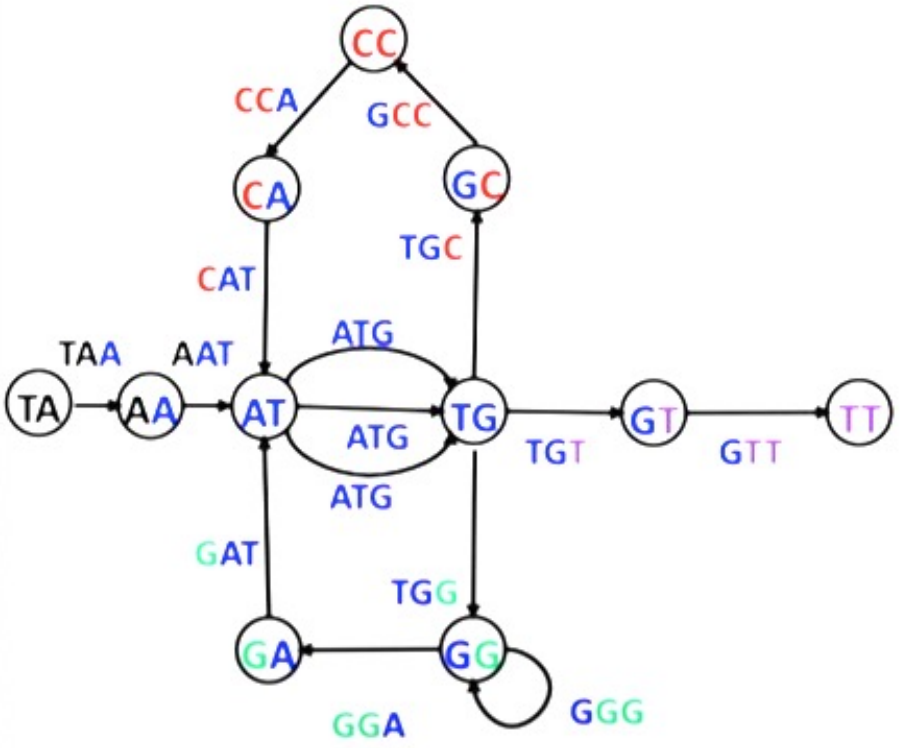


De Bruijn Graph of **TA**ATGCCATGGGATGTT



Where is the *Genome* hiding in this graph?

De Bruijn Graph of **TA**ATGCCATGGGATGTT



An **Eulerian path** in a graph is a path that visits each **edge** exactly once.

What Problem Would You Prefer to Solve?

Eulerian Path Problem. Find an **Eulerian** path in a graph.

- **Input.** A graph.
- **Output.** A path visiting every **edge** in the graph exactly once.

Hamiltonian Path Problem. Find a **Hamiltonian** path in a graph.

- **Input.** A graph.
- **Output.** A path visiting every **node** in the graph exactly once.

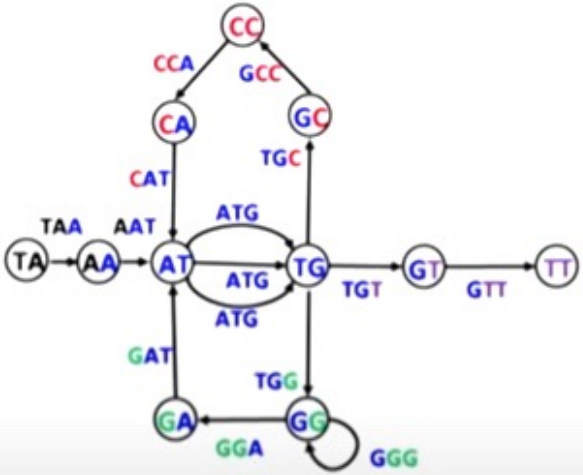
Outlines

- Overview of Genome Assembly and Sequencing
- Computational Problem of Genome Assembly
- Hamiltonian paths and universal strings
- De Bruijn Graphs
- Euler’s Theorem
- Assembling Read-Pair
- De Bruijn Graphs Face Harsh Realities of Assembly

Eulerian Path Problem

Eulerian Path Problem. Find an **Eulerian** path in a graph.

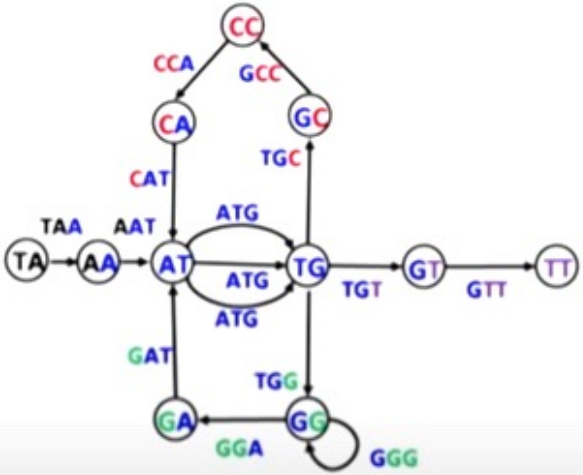
- **Input.** A graph.
- **Output.** A path visiting every **edge** in the graph exactly once.



Eulerian Path Problem

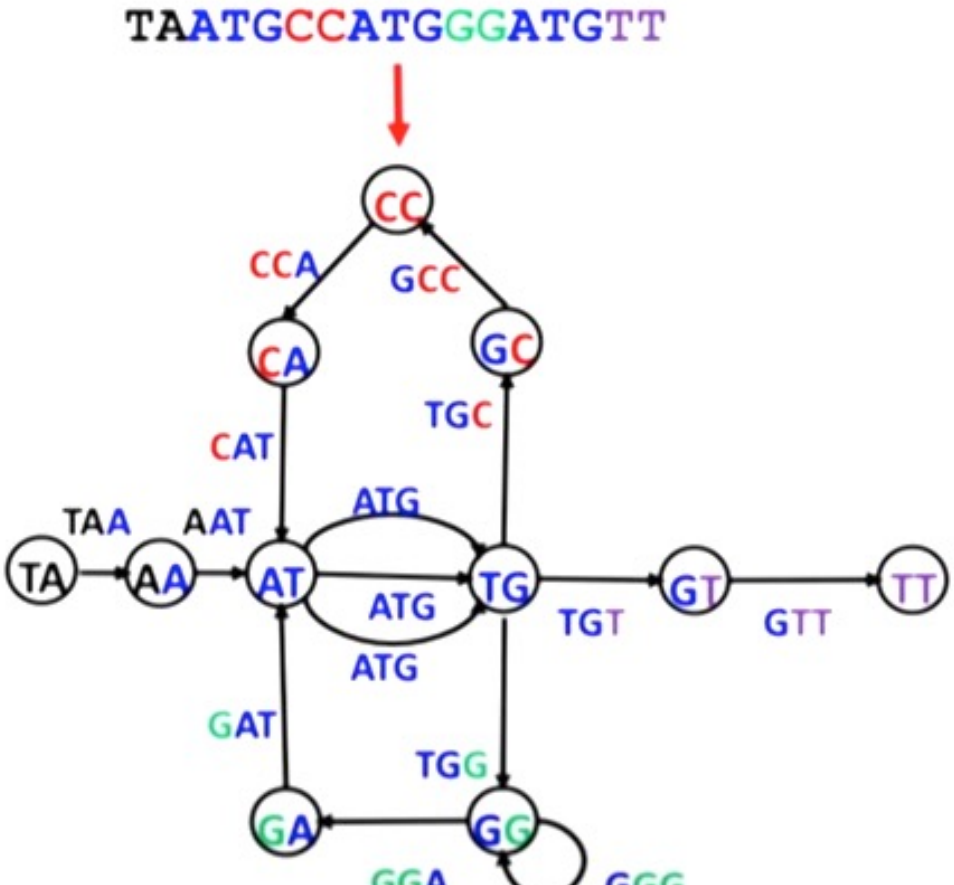
Eulerian Path Problem. Find an **Eulerian** path in a graph.

- **Input.** A graph.
- **Output.** A path visiting every **edge** in the graph exactly once.

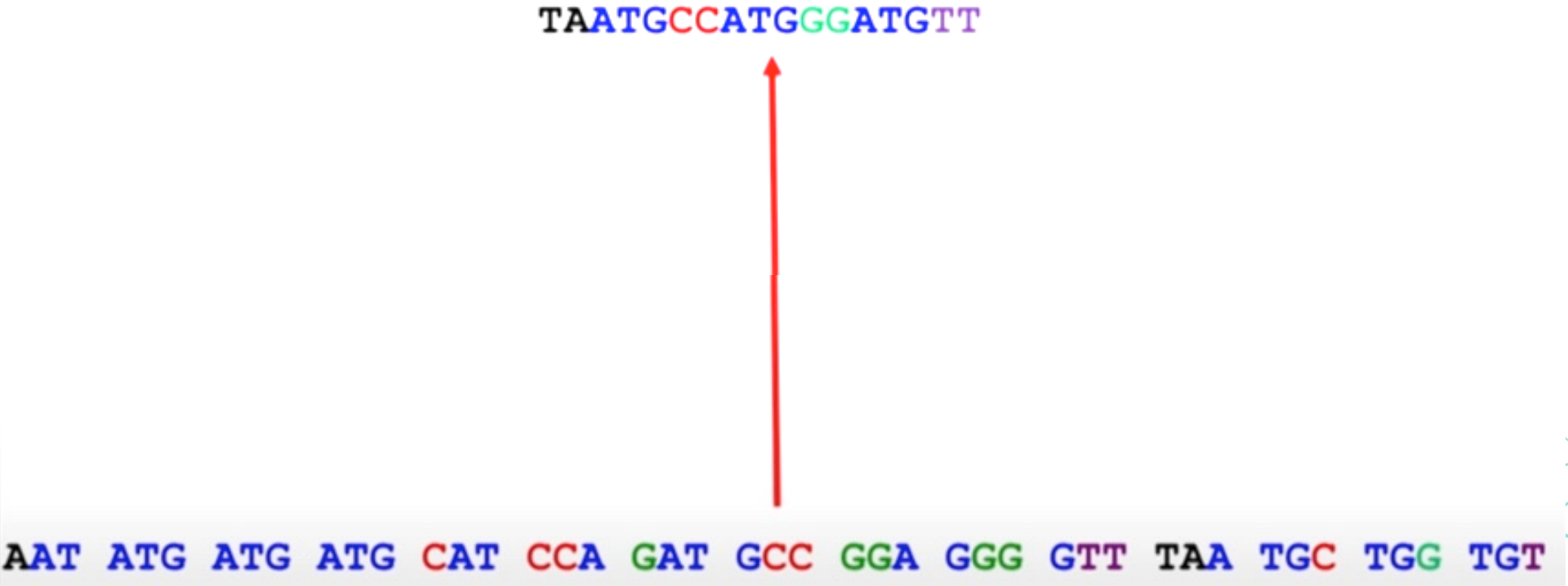


We constructed the de Bruijn graph from *Genome*, but in reality, *Genome* is unknown!

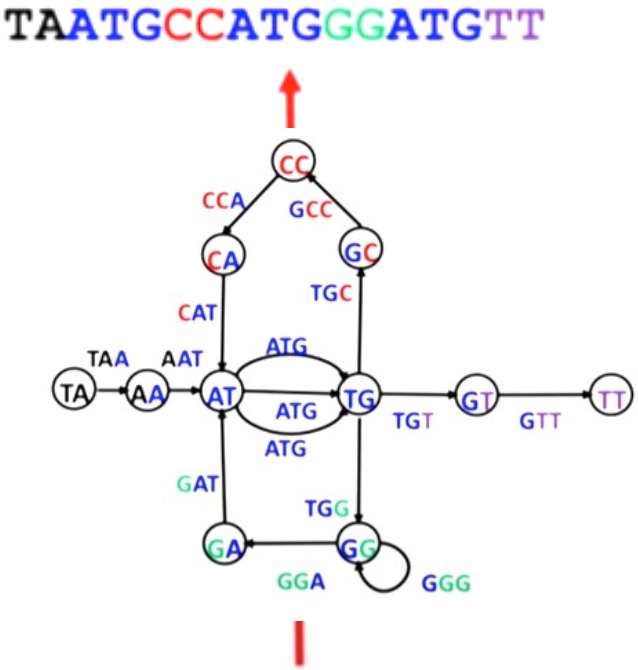
What We Have Done: From *Genome* to de Bruijn Graph



From Reada (K-mers) to Genome

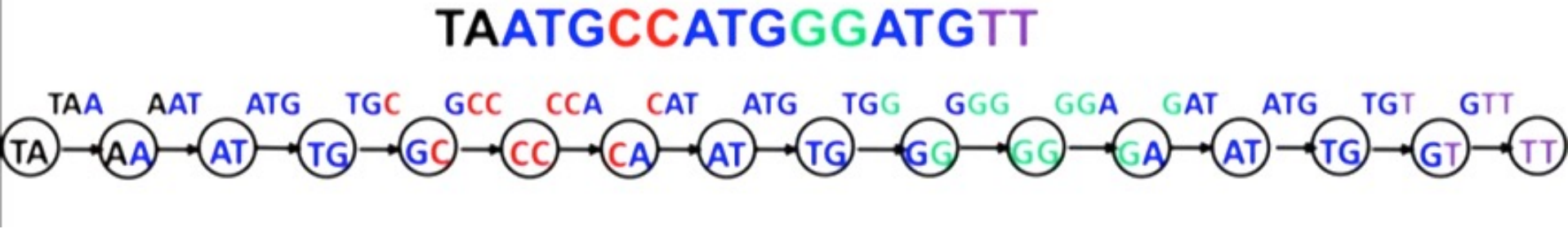


From Reads to de Bruijn Graph to Genome

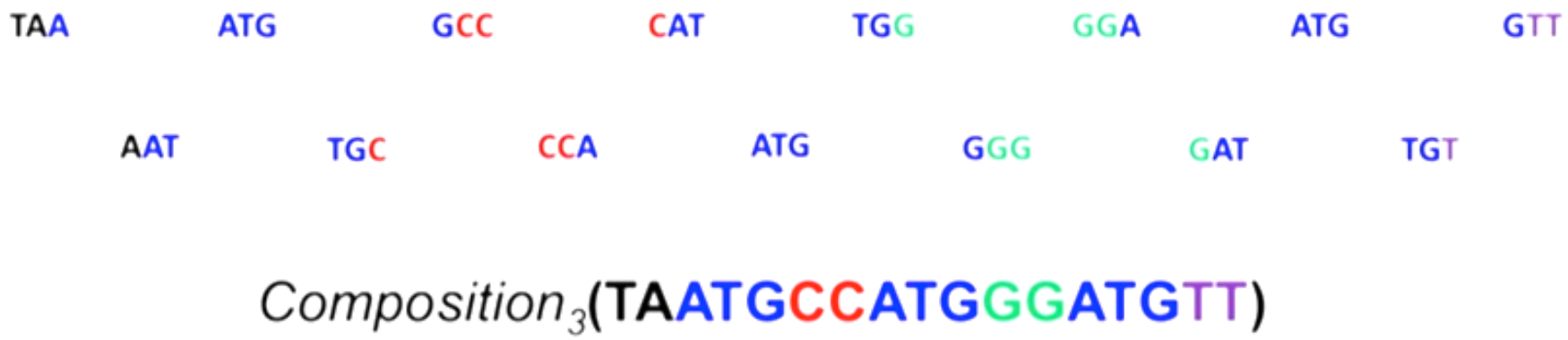


AAT ATG ATG ATG CAT CCA GAT GCC GGA GGG GTT TAA TGC TGG TGT

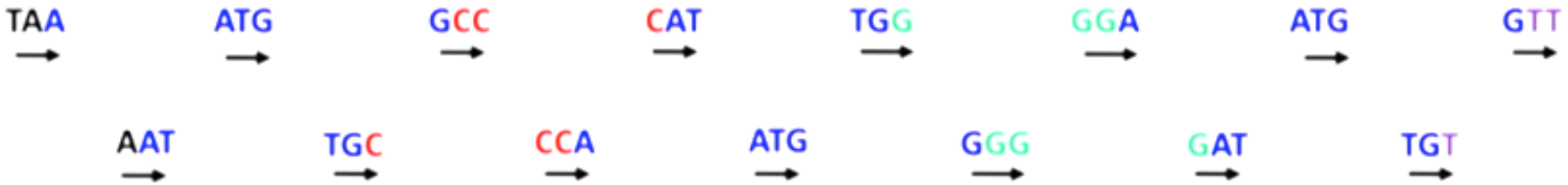
Constructing de Bruijn Graph when Genome is Known



Constructing de Bruijn Graph when Genome is Known

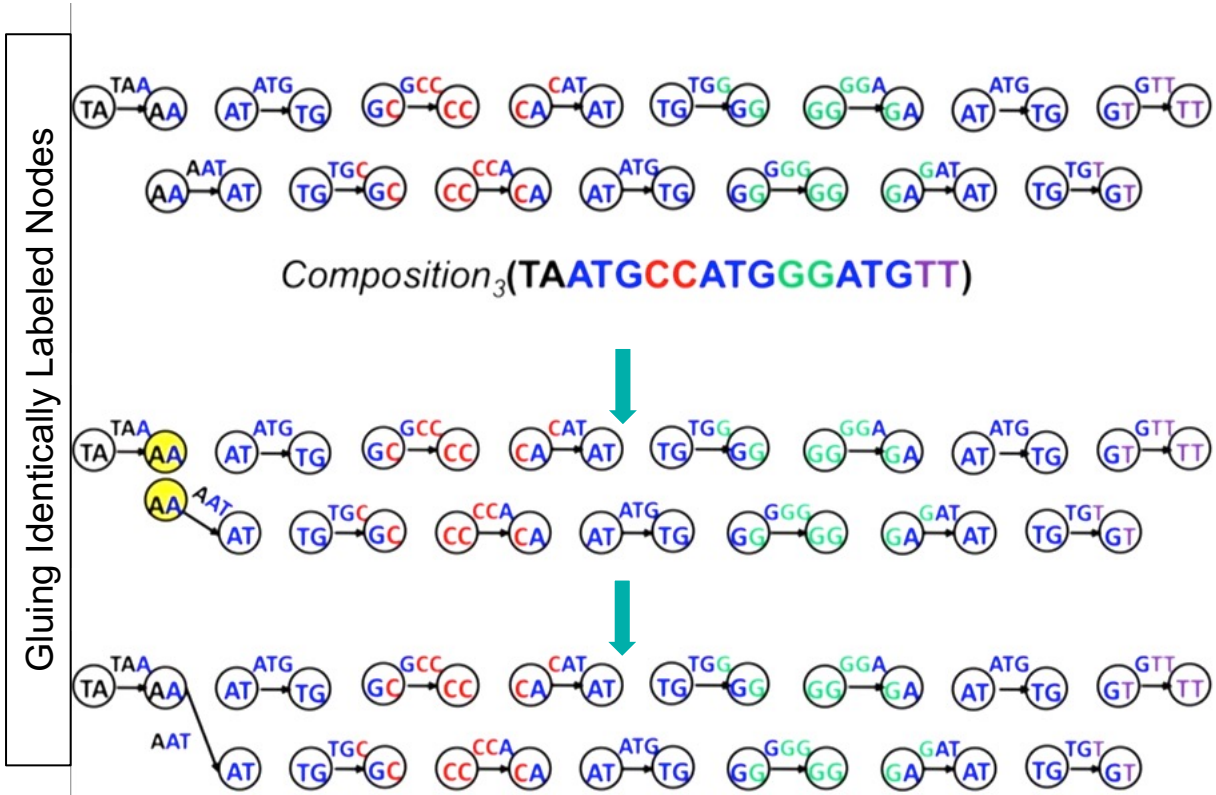


Represnting Composition as Graph Of Isolated Eges



$Composition_3$(TAATGCCATGGGATGTT)

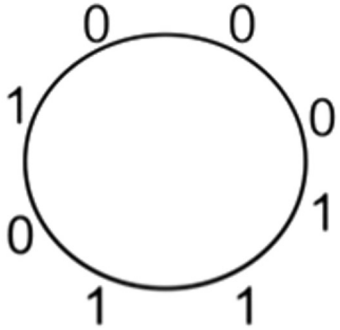
Constructing de Bruijn Graph from k–mer composition



From Hamilton  to Euler  to de Bruijn 

Universal String Problem (De Bruijn, 1946). Find a circular string containing each binary *k*-mer exactly once.

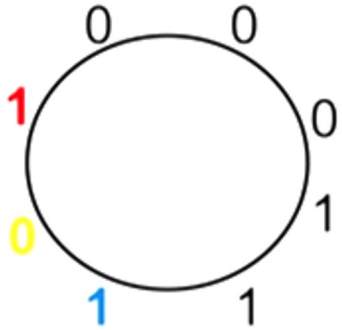
000 001 010 011 100 101 110 111



From Hamilton  to Euler  to de Bruijn 

Universal String Problem (Nicolaas de Bruijn, 1946). Find a circular string containing each binary *k*-mer exactly once.

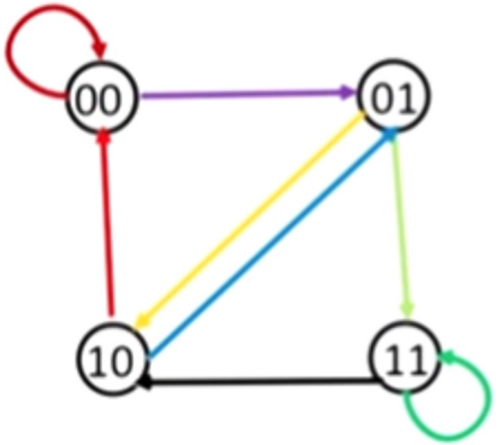
000 001 010 011 100 101 110 111

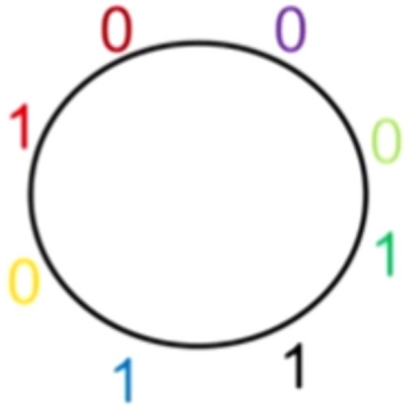


Universal String Problem (Nicolaas de Bruijn, 1946). Find a circular string containing each binary *k*-mer exactly once.

000 001 010 011 100 101 110 111

From Hamilton  to Euler  to de Bruijn 





Constructing de Bruijn Graph

أكاديمية كاوست
KAUST ACADEMY

De Bruijn graph of a collection of *k*-mers:

- Represent every *k*-mer as an edge between its prefix and suffix
- Glue **ALL** nodes with identical labels.

DeBruijn(k-mers)
form a node for each (*k-1*)-mer from ***k-mers***
for each *k*-mer in ***k-mers***
connect its prefix node with its suffix node by an edge

Bioinformatics Challenges

GitHub

- 3A: Generate the k-mer Composition of a String
- 3B: Reconstruct a String from its Genome Path
- 3C: Construct the Overlap Graph of a Collection of k-mers
- **3D: Construct the De Bruijn Graph of a String**
- **3E: Construct the De Bruijn Graph of a Collection of k-mers**
- 3G: Find an Eulerian Path in a Graph
- 3H: Reconstruct a String from its k-mer Composition
- 3I: Find a k-Universal Circular String

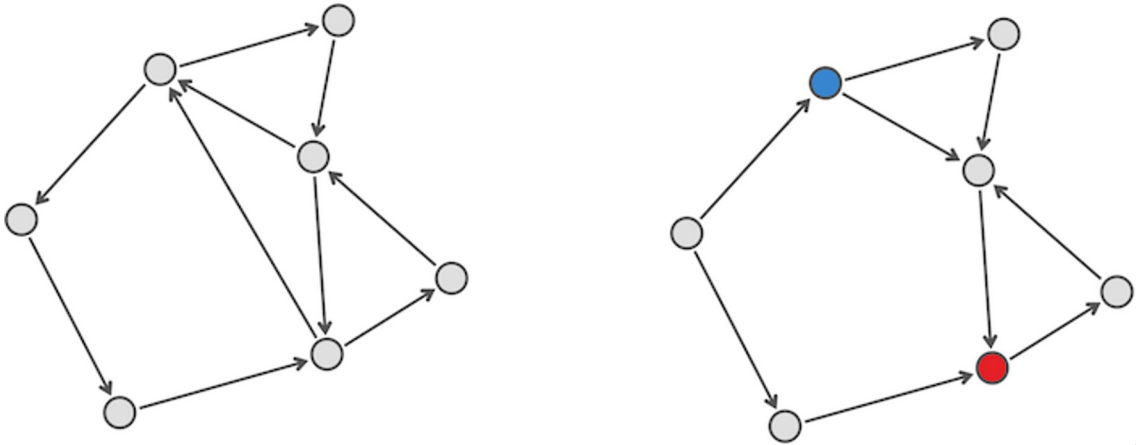
جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

Outlines

- Overview of Genome Assembly and Sequencing
- Computational Problem of Genome Assembly
- Hamiltonian paths and universal strings
- De Bruijn Graphs
- Euler’s Theorem

Euler's Theorem

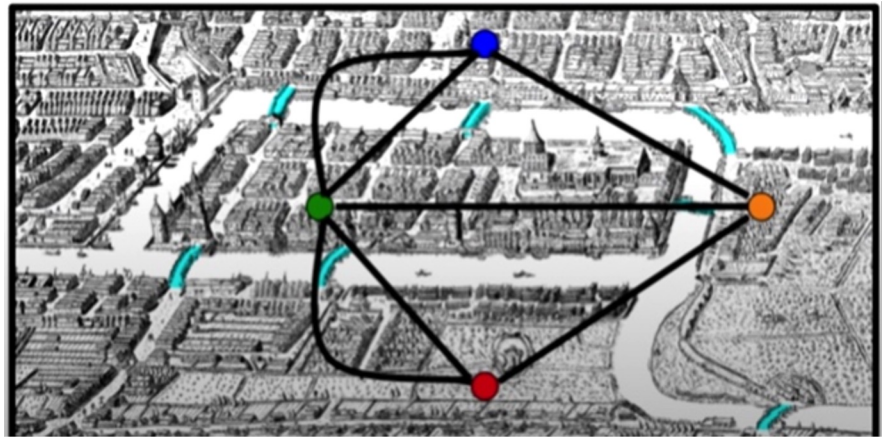
In order for a graph to be Eulerian, the number of incoming edges at any node must be equal to the number of outgoing edges at that node. We define the **indegree** and **outdegree** of a node v (denoted $in(v)$ and $out(v)$, respectively) as the number of edges leading into and out of v. A node v is **balanced** if $in(v) = out(v)$, and a graph is **balanced** if all its nodes are balanced. Because Leo must always be able to leave a node by an unused edge, any Eulerian graph must be balanced. The figure below shows a balanced graph and an unbalanced graph.



Eulerian CYCLE Problem

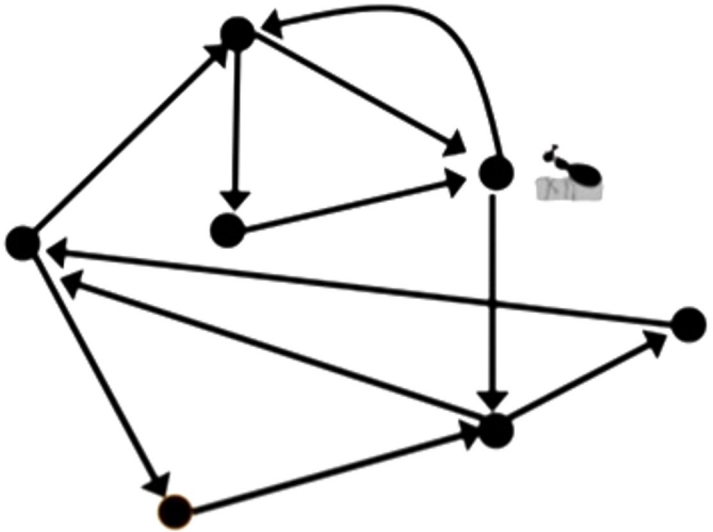
Eulerian CYCLE Problem. Find an Eulerian cycle in a graph.

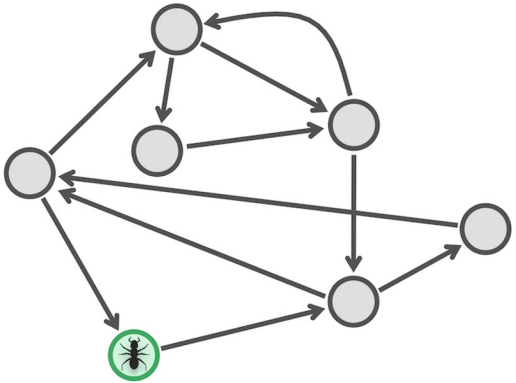
- **Input.** A graph.
- **Output.** A cycle visiting every edge in the graph exactly once.

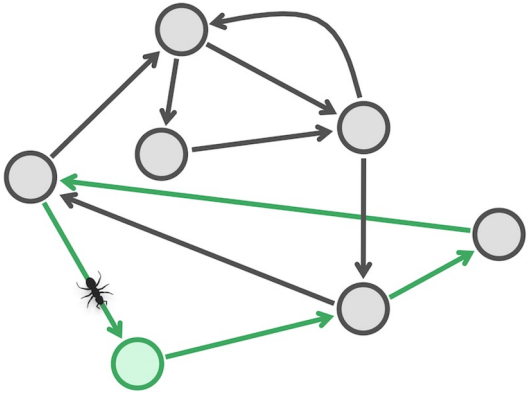


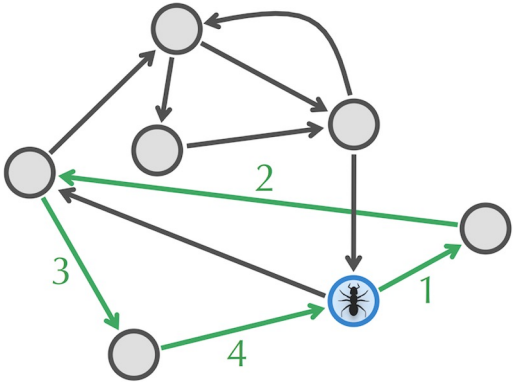
Let an ant randomly walk through the graph.

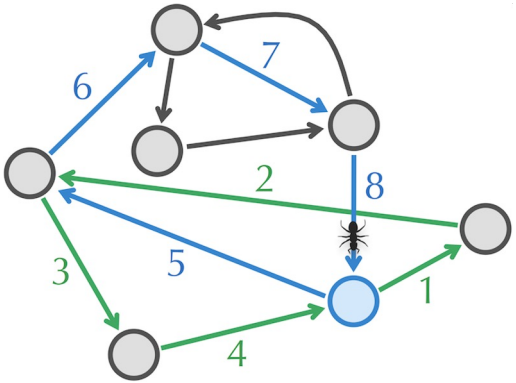
The ant cannot use the same edge twice!

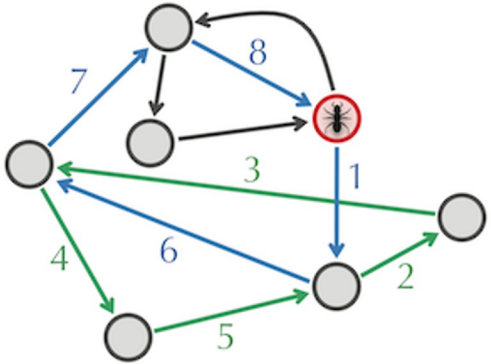


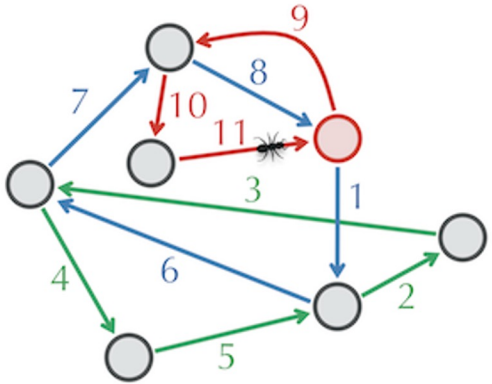





2
3
1
4

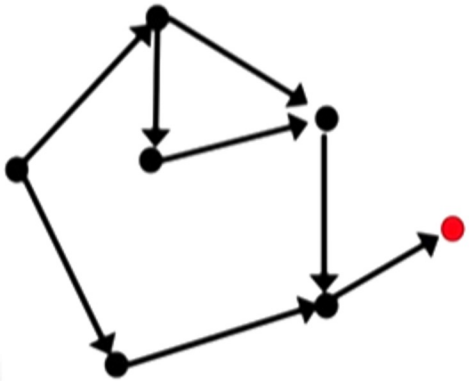

6
7
2
8
3
5
1
4

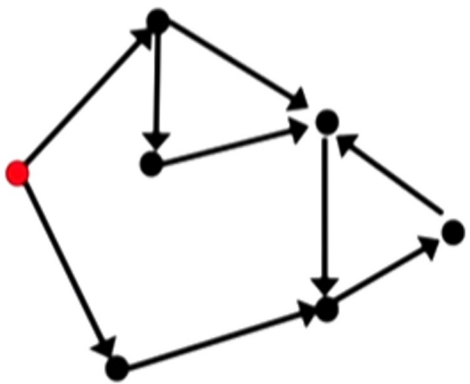

8
7
3
1
4
6
2
5

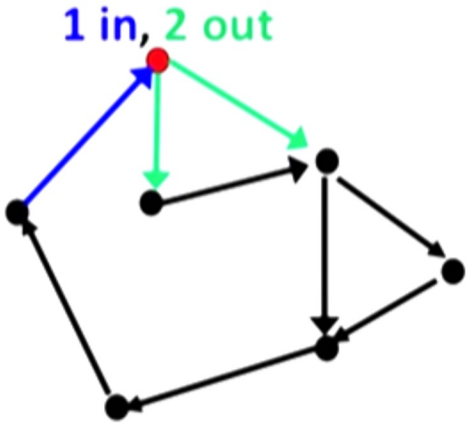

9
8
7
10
11
3
1
4
6
2
5

A Graph is **Eulerian** if It Contains an Eulerian Cycle.

Is this graph Eulerian?

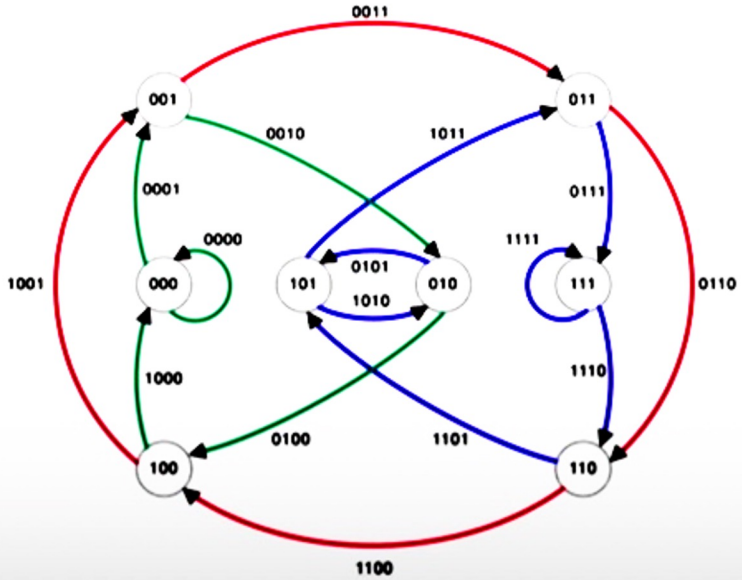


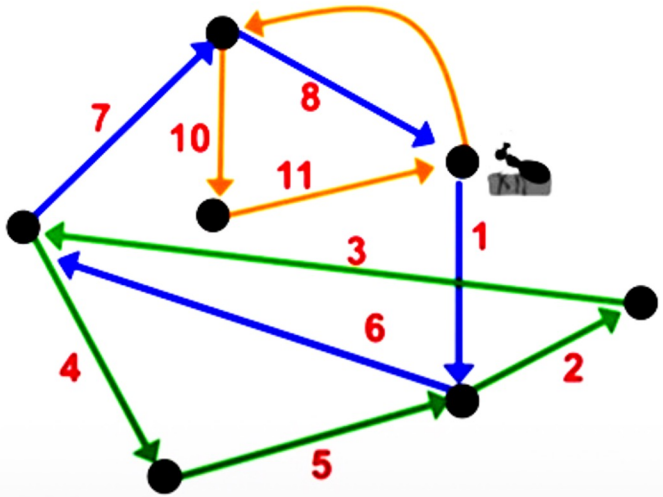




A graph is **balanced** if ***indegree*** = ***outdegree*** for each node

EulerianCycle(*BalancedGraph*)
  form a *Cycle* by randomly walking in *BalancedGraph* (avoiding already visited edges)
    while *Cycle* is not Eulerian
      select a node *newStart* in *Cycle* with still unexplored outgoing edges
      form a *Cycle'* by traversing *Cycle* from *newStart* and randomly walking afterwards
      Cycle ← *Cycle'*
  return *Cycle*

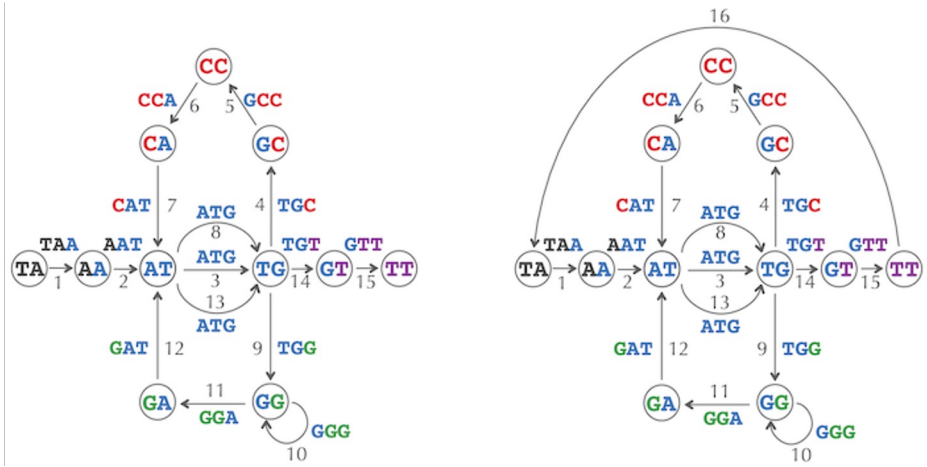




From Eulerian cycles to Eulerian paths

We can now check if a directed graph has an Eulerian *cycle*, but what about an Eulerian path? Consider the de Bruijn graph on the left in the figure below, which we already know has an Eulerian path, but which does not have an Eulerian cycle because nodes `TA` and `TT` are not balanced. However, we can transform this Eulerian path into an Eulerian cycle by adding a single edge connecting `TT` and `TA`, as shown in the figure below.

STOP and Think: How many unbalanced nodes does a graph with an Eulerian path have?



Code Challenge: Solve the Eulerian Path Problem.

- **Input:** The adjacency list of a directed graph that has an Eulerian path.
- **Output:** An Eulerian path in this graph.

Extra: Bioinformatics Challenges

GitHub

- 3A: Generate the k-mer Composition of a String
- 3B: Reconstruct a String from its Genome Path
- 3C: Construct the Overlap Graph of a Collection of k-mers
- 3D: Construct the De Bruijn Graph of a String
- 3E: Construct the De Bruijn Graph of a Collection of k-mers
- **3G: Find an Eulerian Path in a Graph**
- **3H: Reconstruct a String from its k-mer Composition**
- **3I: Find a k-Universal Circular String**

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology

Done with Day 3, Heyyyyy!

Thank You !

جامعة الملك عبدالله
للعلوم والتقنية
King Abdullah University of
Science and Technology