Read Mapping

de Novo Assembly

Peter N. Robinson

Institut für Medizinische Genetik und Humangenetik
Charité Universitätsmedizin Berlin

Genomics: Lecture #2 WS 2014/2015

Today

- Genome assembly: the basics
- Hamiltonian and Eulerian Graphs: Review
- De Bruijn graphs: Basics
- De Bruijn graphs for genome assembly: Simplified
- De Briun graphs for genome assembly: Simple pair end
- De Bruijn graphs: real life

Outline

Genome assembly: the basics

The process of puzzling together a complete genome sequence of an organism for which shotgun sequencing has been performed is referred to as **genome assembly**.

- As the costs for sequencing have declined, the major challenge becomes computational
- Can we sequence and de novo assemble a large ($>$ 100 Mb) genome with the short (50-250bp) reads typical of current NGS protocols?

Genome assembly: the basics

- There are two major classes of assembly algorithms
  1. Overlap-layout consensus (OLC)
  2. De bruijn graph (DBG)
- OLC was widely used back in the day when sequencing was performed by the low-throughput, longer-read Sanger method.
- DBG based methods have dominated the scene since the introduction of NGS

Sequencing data: Models and intuition

To get intuition about genome assembly, let us consider a **idealized genome** that represents a long random sequence of four bases and that does not contain repeats or other complex structures.

- Consider the simplest sequencing strategy: single-end, **whole-genome shotgun** (WGS).
- That is, we sample equal-length fragments with starting points randomly distributed across the genome
- For now, ignore sequencing errors and biases

Sequencing data: Models and intuition

- Thus, our shotgun sequencing can be likened to a process that samples bases from all genome positions at random
- The chance that any particular base is sampled is very low in a single sampling process
- However, we perform the sampling process a very large number of times

Any suggestions as to how we might model this?

Sequencing data: Models and intuition

The Poisson distribution expresses the probability of a given number of events occurring in a fixed interval of time and/or space if these events are iid.

$$f(k;\lambda) = P(X = k) = \frac{e^{-\lambda}\lambda^k}{k!} \quad (1)$$

- k refers to number of reads that overlap a certain genomic position ("coverage")
- λ mean sequencing depth

Sequencing data: Models and intuition

Let's look at the model more closely

- G: genome size (e.g., 3.2×10^9 nucleotides for humans)
- L: read length (e.g., 100 nucleotides for a typical Illumina run)
- N: read number
- n_b: total number of sequenced bases

It is now easy to calculate that

$$n_b = N \times L \tag{2}$$

- Similarly, the average coverage depth per base is $d_b = \frac{n_b}{G}$

Sequencing data: Models and intuition

k-mers: subsequences with k nucleotides (here: k=3)

- Consider the following k-mers in a small genome of 17 nucleotides

$\text{Composition}_k(\text{TCATTCTTCAGGTCAAA})$

TCA
CAT
ATT
TTC
TCT
CTT
TTC
TCA
CAG
AGG
GGT
GTC
TCA
CAA
AAA

- How many 3-mers are there in this genome?

Sequencing data: Models and intuition

- In general there are $L - k + 1$ k-mer subsequences in a sequence of length L with $k \leq L$.
- Let's say we want to know the total number of k-mers in our WGS data. Since we have N reads, each of which has $L - k + 1$ k-mer subsequences, the total number of k-mers (n_k) is

$$n_k = N \times (L - k + 1) \tag{3}$$

- The coverage depth for k-mers is then $d_k = \frac{n_k}{G}$.
- The ratio between the coverage depth for bases and that for k-mers is then

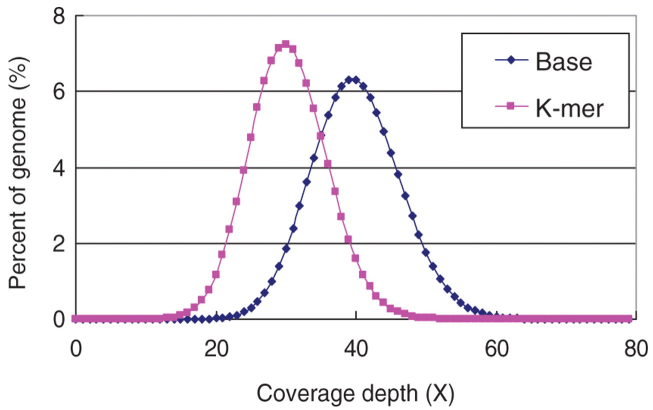
$$\frac{d_b}{d_k} = \frac{n_b/G}{n_k/G} = \frac{L}{L - k + 1} \tag{4}$$

Sequencing data: Models and intuition

Say we are performing de novo sequencing for an organism that has not been sequenced before. How can we estimate its overall genome size?

- The number of k-mers in the WGS reads (n_k) can be directly counted
- The mean coverage depth of k-mers can be estimated from the peak value of the empirical k-mer coverage depth distribution curve

peak depth value $d_k = 30.4$



Graphic: Zhenyu Li et al., Briefings in functional genomics (2012) 11 (1): 25-37.

Sequencing data: Models and intuition

With this data in hand, we can now estimate the genome size as

$$G \approx \frac{n_k}{d_k} \tag{5}$$

and we can estimate the actual base coverage by

$$d_b \approx \frac{L}{L - k + 1} \times d_k \tag{6}$$

- Here, we would use a value of k such that we do not expect to see a given k-mer more than once in a random genome
- In practice, these estimates are not exact even in a random genome because of sequencing errors (why?).

Sequencing data: Models and intuition

Getting back to our initial question, we can now use the mean base coverage estimate $\lambda = d_b$ to estimate the probability that a given base will not be covered

$$P(X = 0) = \frac{e^{-\lambda}\lambda^0}{0!} = e^{-\lambda} \tag{7}$$

Therefore, the probability of seeing at least one read at a given position is

$$P(X > 0) = 1 - e^{-\lambda} \tag{8}$$

Sequencing data: Models and intuition

- So if we want to estimate the mean read depth required such that at least 99% of the genome is covered once (and thus the probability of any base is at least 99%)[1], we have

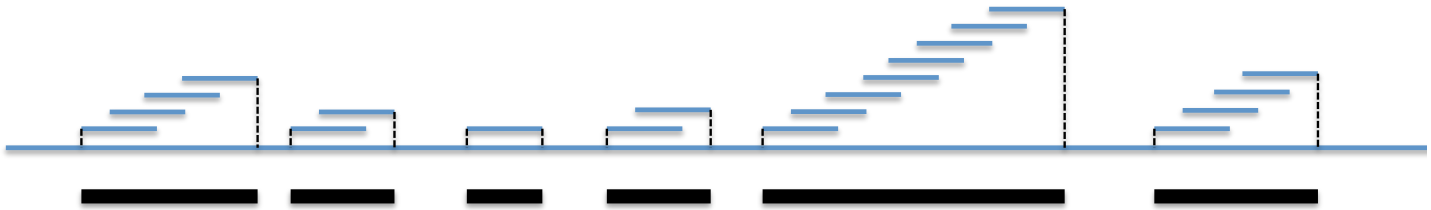
$$\begin{aligned} P(X > 0) = 0.99 &= 1 - e^{-\lambda} \\ e^{-\lambda} &= 0.01 \\ -\lambda &= -4.605 \end{aligned}$$

- Thus we need to sequence to an average depth of at least 4.6 to get at least 99% of the genome covered at least once.
- This roughly explains the goal of 6x coverage in initial Sanger sequencing projects of the human genome

[1] linearity of expectation

Sequencing data: Models and intuition

Now let us consider **contigs**: combinations of overlapping reads that represent contiguous sequence



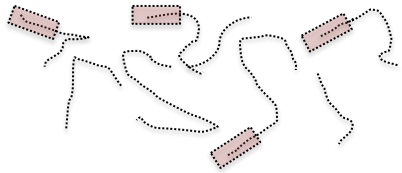
- Collection of $N = 21$ **reads** assembled into 6 **contigs**
- The contigs are assumed to be the best possible representation of the original DNA sequence
- Note that the actual locations of the contigs and their orientation to one another are unknown to us.

Sequencing data: Models and intuition

The initial steps of genome assembly are basically an attempt to find contigs

Genome: 3.2 Gb

Many copies of genome



Reads: 500bp

Only one end sequenced

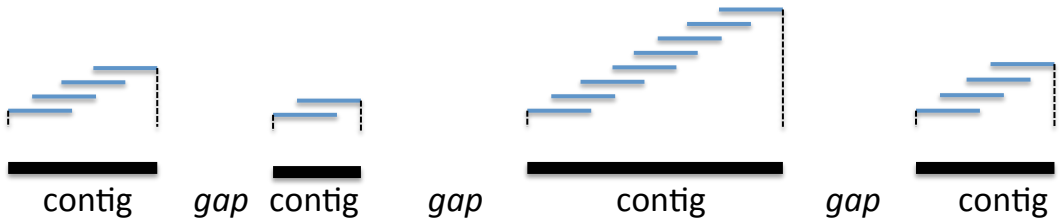
Not all fragments sequenced

```
tgctgctcctacaacatcggccgtgcctg
            atcggccgtgcctggaataagccct
```

Find overlapping reads

...tgctgctcctacaacatcggccgtgcctggaataagccct...

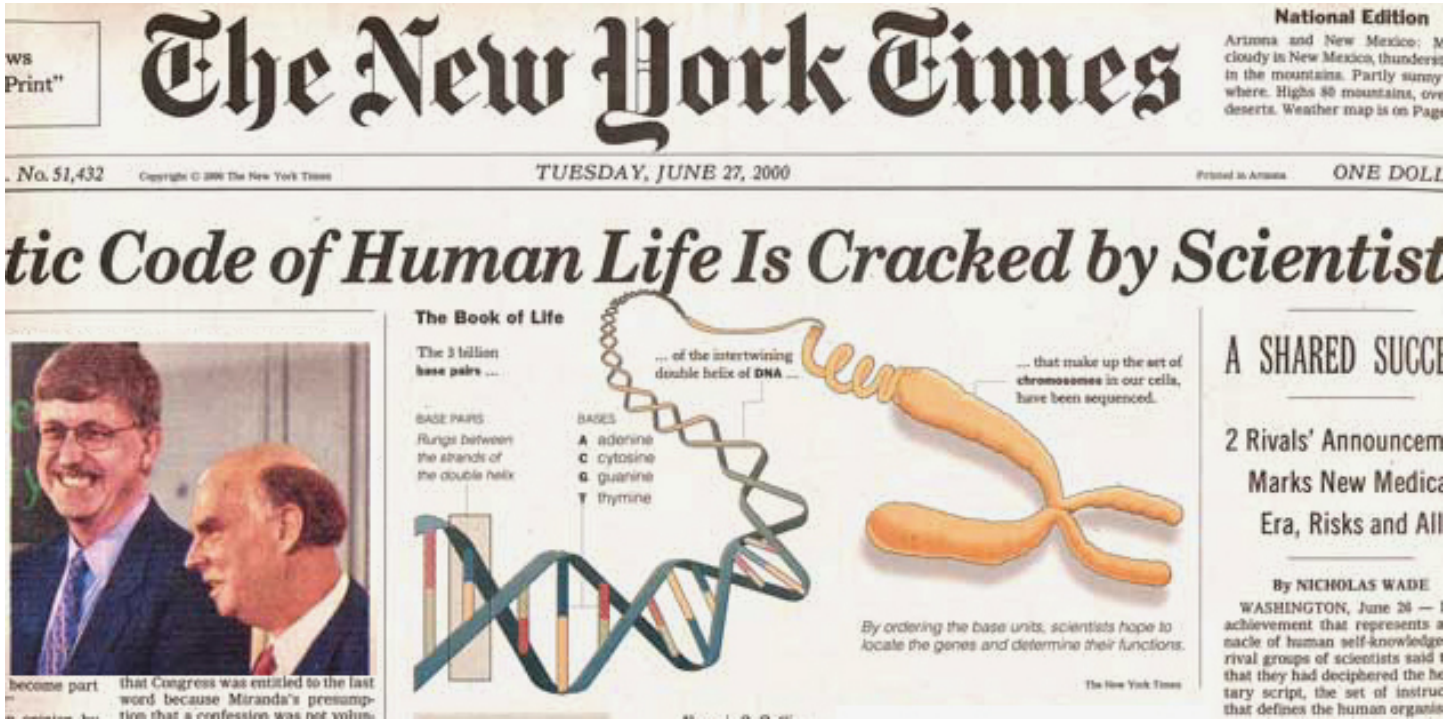
Merge overlapping reads into contigs



Result of assembly is set of contigs with gaps

Sequencing data: Models and intuition

Putting it all together...



The New York Times

National Edition

TUESDAY, JUNE 27, 2000

tic Code of Human Life Is Cracked by Scientist

The Book of Life

A SHARED SUCCE

2 Rivals' Announcem

Marks New Medica

Era, Risks and All

By NICHOLAS WADE

- Competition to assemble the human genome: whole-genome shotgun vs. BAC by BAC

The human genome project(s)

Public effort: Lander et al., Nature, Feb. 15, 2001.

- US, UK, France, Germany, Japan, China
- government labs and universities.
- BAC-by-BAC sequencing.

Commercial: Venter et al., Science, Feb. 16, 2001.

- whole genome random shotgun sequencing.
- Celera (www.celera.com)

BAC by BAC

- BAC: bacterial artificial chromosome
- BACs have inserts of 100,000–300,000 nucleotides
- Do shotgun sequencing on each separate BAC;
- BACs are much smaller than the human genome and correspondingly easier to assemble.
- First assemble individual BACs, then fit overlapping BACs together

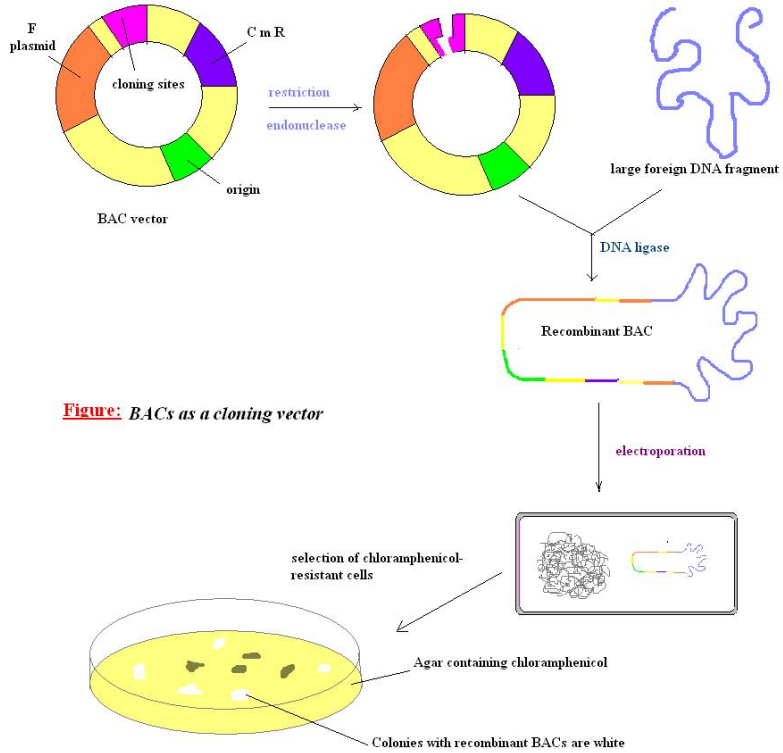
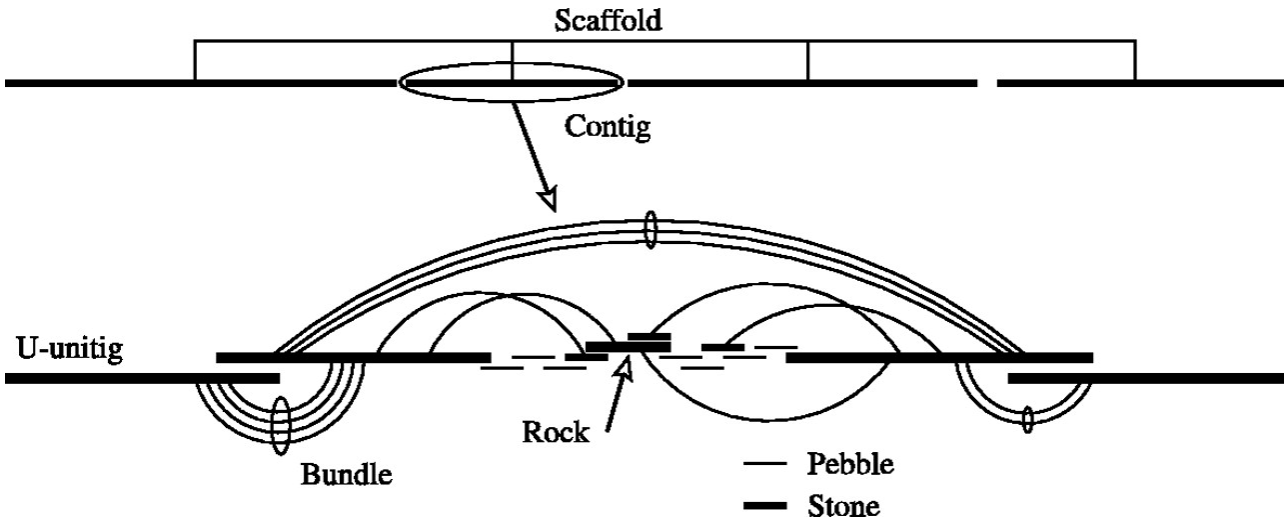


Figure: *BACs as a cloning vector*

Image: wikipedia

Advantage: highly accurate. Disadvantage: Slow

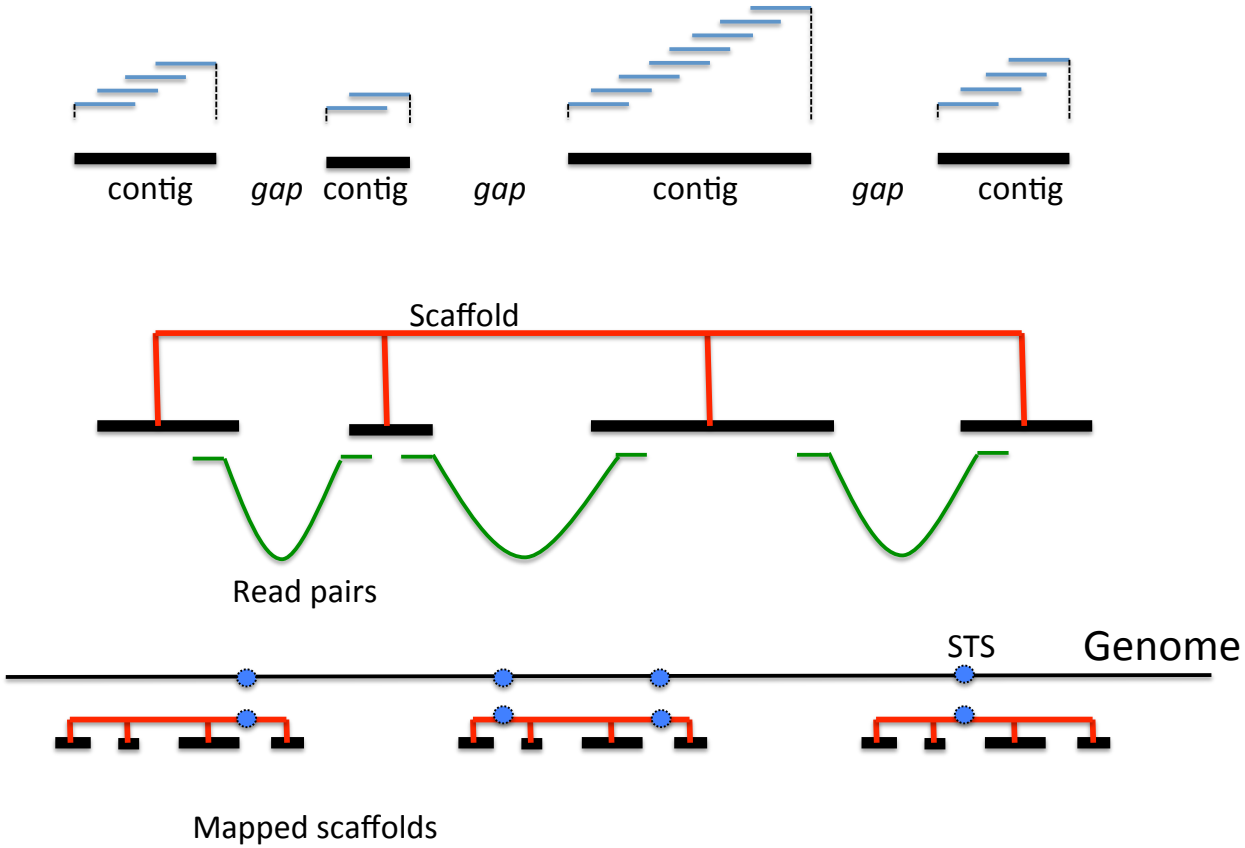
Whole genome shotgun



Myers EW (2000) A Whole-Genome Assembly of Drosophila *Science* **287**:2196–2204

- All against all pairwise alignment
- Merge to contigs if overlap big enough
- Nicknames for contigs: small = rock, smaller = stone, smaller = pebble.

Whole genome shotgun



- Additional processing to piece together contigs

Whole genome shotgun: Overlaps

- We will be covering mainly algorithms for creating contigs from reads in these lectures
- Let us begin with another topic to build intuition: How much **overlap** do we need?
- Key questions: How many contigs are there? How big are the gaps? How long are the contigs?

Overlap between communism and capitalism

Whole genome shotgun: Contigs and Overlaps

Recall our definitions

- G: genome size. Assume 3×10^9 nucleotides
- L: read length. Assume 500 nucleotides
- N: read number
- $n_b = N \cdot L$: total number of sequenced bases
- $\lambda = NL/G$ is the coverage

For instance, 10x coverage of the human genomes requires

$$N = \lambda G/L = 10 \cdot 3 \times 10^9/500 = 60 \text{ million reads}$$

Reads: Probability to start at a given base

- In a genome of length G, a read of length L can start anywhere except at the very ends of the chromosomes
- In humans there are $c = 23$ chromosomes, so $c \times (L - 1)$ positions cannot represent start positions.
- For $L = 500$, we have $23 \times 499 = 11477$ such positions, but these can be ignored in a genome of 3×10^9 nucleotides
- Thus, the probability that a read starts at base i is well approximated by $P(\text{read starts at } i) \approx N/G$ if there are a total of N reads

Reads: Probability to start in an interval

- Consider an interval I that is as long as a read (L nucleotides).
- The expected number of reads that start in I is then $\lambda = L \times N/G$.
- Assuming a Poisson distribution, the probability that no read starts in I is then

$$P(X = 0) = \frac{e^{-\lambda}\lambda^0}{0!} = e^{-\lambda} \tag{9}$$

- The probability that at least one read starts in I is then

$$P(X > 0) = 1 - P(X = 0) = 1 - e^{-\lambda} \tag{10}$$

Reads: Probability to start in an interval

- Consider a nucleotide at position i
- This nucleotide is in a gap between contigs if no read starts in the interval

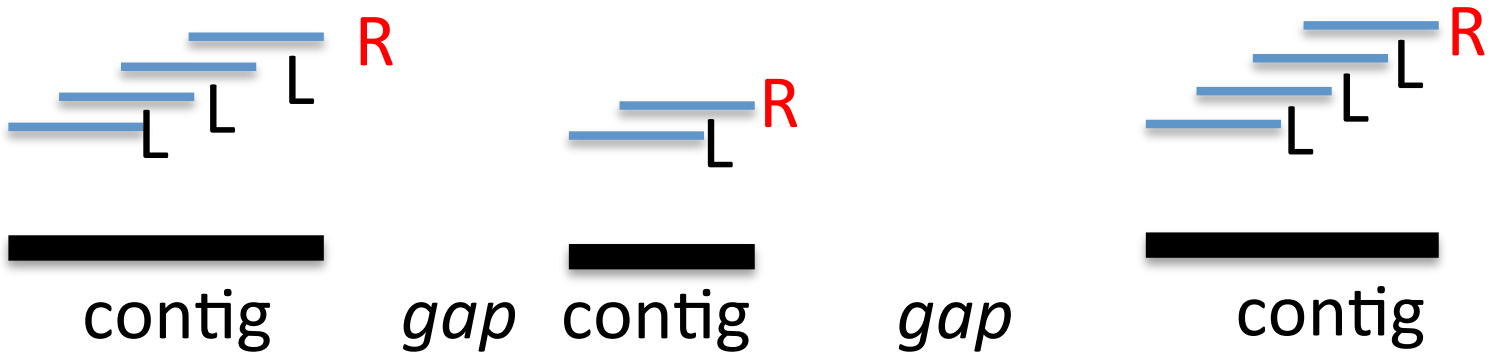
$$[i - L + 1, i]$$

- This interval has length L, and thus, the probability that no read starts in it is $e^{-\lambda}$
- By linearity of expectation, we can estimate the number of nucleotides in gaps across the entire assembly as

$$G \cdot e^{-\lambda}$$

- Correspondingly, the number of nucleotides included in contigs is $G \cdot \left(1 - e^{-\lambda}\right)$

Contigs: How many are there?



- Each contig has a unique rightmost read ("R")
- The probability that a given read is the rightmost read is the same as the probability that no other read starts within the read
- If the read starts at position i, this is the probability that no read starts within the interval $[i - L + 1, i]$, which we have already calculated as $e^{-\lambda}$

Contigs: How many are there?

- The number of contigs must be equal to the number of rightmost reads
- There are a total of N reads, each of which has a probability of $e^{-\lambda}$ of being an R reads. Thus, the expected number of contigs is

$$Ne^{-\lambda}$$

- The expected number of reads per contig is then $1/e^{-\lambda}$

Contigs: How big are they?

- We have seen that the expected size of the sequenced region of the genome is $(1 - e^{-\lambda}) \cdot G$
- The expected number of contigs is $Ne^{-\lambda}$ Therefore, the expected size of a contig is simply

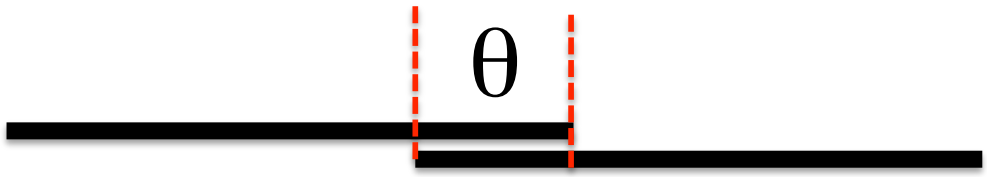
$$\frac{(1 - e^{-\lambda}) \cdot G}{Ne^{-\lambda}}$$

- Thus if we go for a coverage of $\lambda = 6$ of the human genome with 500 nt reads, we would expect roughly
  1. $N = \lambda G/L$ =36 million reads
  2. 100% $\times(1 - e^{-\lambda})$ =99.8% of the genome being sequenced
  3. A total of $Ne^{-\lambda}$=89,235 contigs
  4. An average contig length of $\frac{(1 - e^{-\lambda}) \cdot G}{Ne^{-\lambda}}$=33,536 nucleotides

Contigs and overlaps

But we have completely neglected the topic of **how much of an overlap** is required to connect two reads?

- Let say we require an overlap of one nucleotide only
- Then any two random reads will overlap with a probability of $1/4$ – not exactly what we want...
- Let θ refer to the proportion of L required to detect an overlap



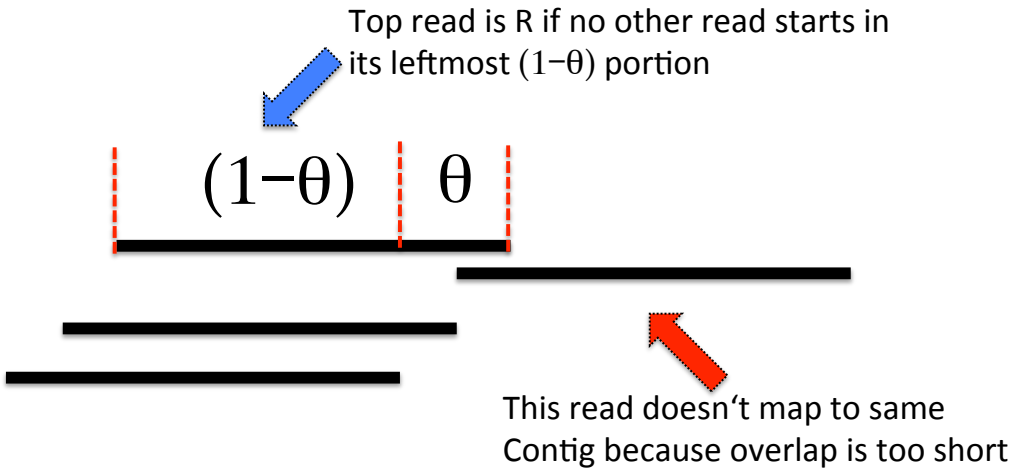
- We will now combine a group of reads to a contig if they are connected by overlaps of length $\geq \theta L$

Expected number of contigs

Let us now calculate the expected number of contigs, given that we demand an overlap of at least θL between combined reads

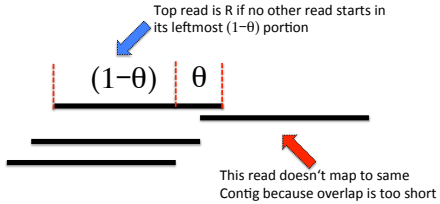
- , As before the probability that a read starts at a given position is N/G
- The probability that k reads start in an interval that is L long is again approximated by the Poisson
- The calculation that a given read at posistion i is the rightmost read now requires not that there is no read in the interval $[i - L + 1, i]$, but instead that there is no read in the leftmost $(1 - \theta)$ proportion of this interval

Expected number of contigs



- We need to calculate the probability that zero reads start in $(1-\theta)L$.
- Above, the expected number of reads that start in I of length L is then $\lambda = L \times N/G$.
- Here, we adjust this to reflect the expected number of reads that start in $(1-\theta)L$ to be $(1-\theta)\lambda$.

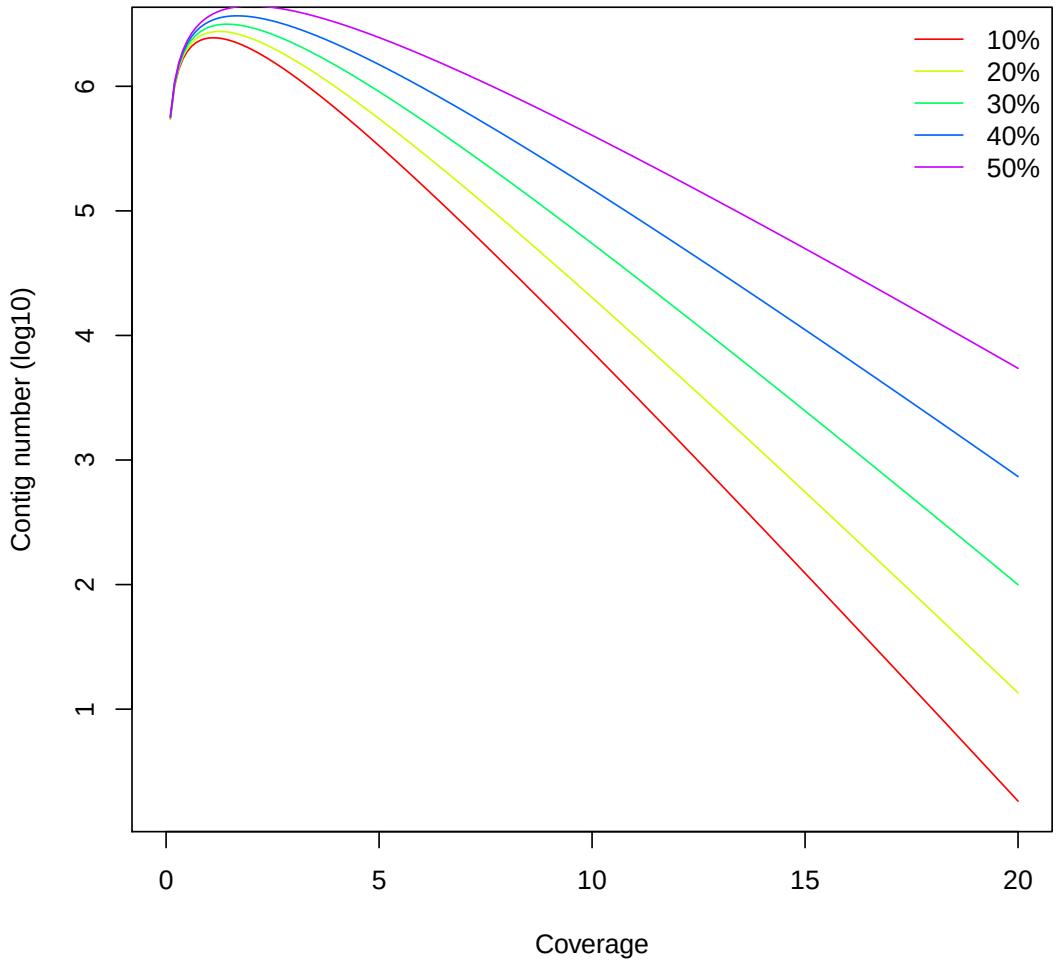
Expected number of contigs



- The expected number of contigs is then N (number of reads) time the probability that a read is the rightmost read of an island, which is equivalent to their being no reads starting in $(1-\theta)L$

$$\begin{aligned}\mathbb{E}[\#\text{contigs}] &= N \times P(\text{no read starts in } (1-\theta)L) \\ &= Ne^{-(1-\theta)\lambda} \\ &= Ne^{-(1-\theta)LN/G} \quad \blacksquare \quad \text{by definition of } \lambda\end{aligned}$$

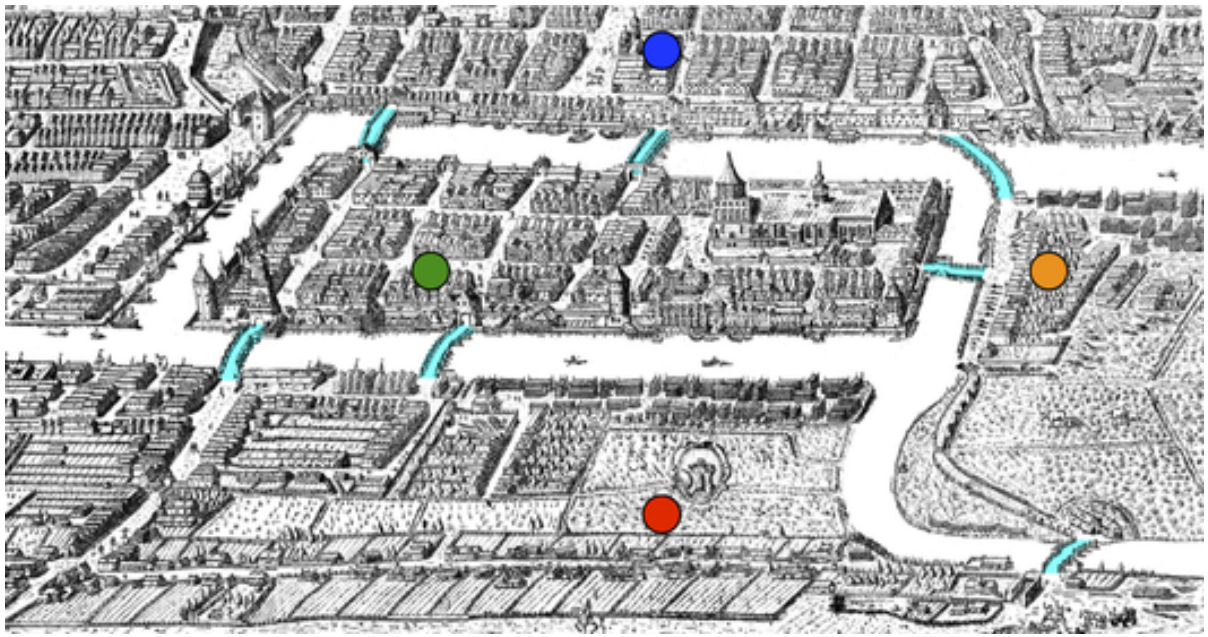
Expected number of contigs



Outline

Königsberg

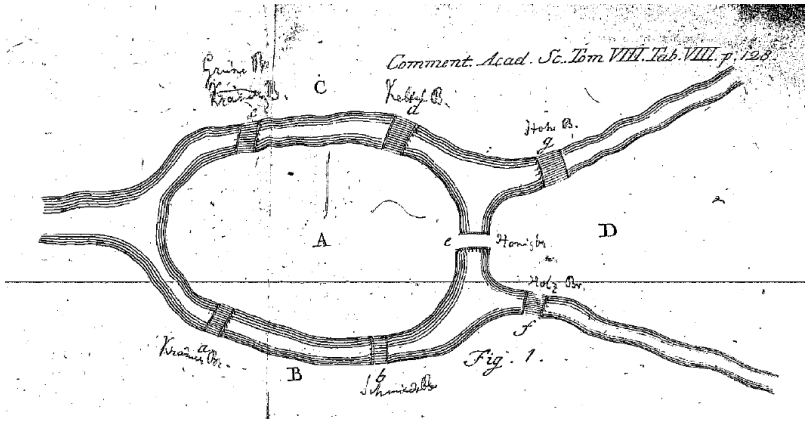
The "Hello World" of Eulerian graphs is of course Königsberg with its seven bridges. Königsberg is located on both sides of the Pregel River, and comprises two large islands which were connected to each other and the mainland by seven bridges.

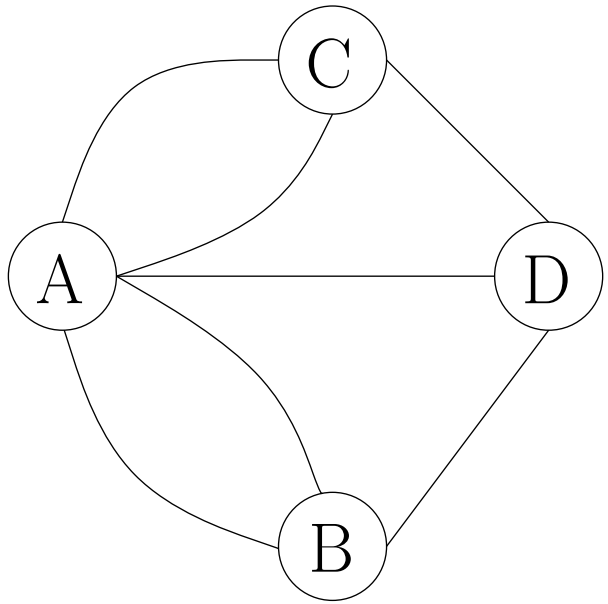


Königsberg

The problem was to find a walk through the city that would cross each bridge once and only once.

- Euler formulated the problem as a graph problem





Genome Sequencing And Graphs

Our goal today is to find an algorithm that will allow us to take a collection of **short NGS sequence reads** – say, strings of 100–250 nucleotides in length with the lettersd ACGT – and to output a longer string representing the **Genome** that was sequenced.

- We will present several simplified scenarios with the goal of motivating and explaining the de Bruijn graph and its use in genome assembly algorithms.

Naive Genome Assembly

We will begin by discussing a ridiculously naive string reconstruction problem. Here, and in the following, the "string" will represent a genome that we have sequenced, and the k-mer subsequences (with k=3) will represent our short reads.

- We will begin by examining a small genome of 17 nucleotides

TCATTCTTCAGGTCAAA

Naive Genome Assembly

Imagine we have a function called composition_k that takes a DNA sequence and returns a set of all k-mers contained in it

- In the following examples we will choose $k = 3$

Composition_k(TCATTCTTCAGGTCAAA)

TCA
CAT
ATT
TTC
TCT
CTT
TTC
TCA
CAG
AGG
GGT
GTC
TCA
CAA
AAA

Naive Genome Assembly

Although we can display the k-mers in "genome order", which would make it ridiculously easy to reconstruct the original genome, in actuality we do not know the original order of the kmers. Therefore, we might as well show them lexicographically.

$\text{Composition}_k(\text{TCATTCTTCAGGTCAAA})$

„Genome order“:

TCA CAT ATT TTC TCT CTT TTC TCA CAG AGG GGT GTC TCA CAA AAA

„Lexicographic order“

AAA AGG ATT CAA CAG CAT CTT GGT GTC TCA TCA TCA TCT TTC TTC

Naive Genome Assembly

Let us now put each of the k-mers into the node of a graph and connect the graph by edges

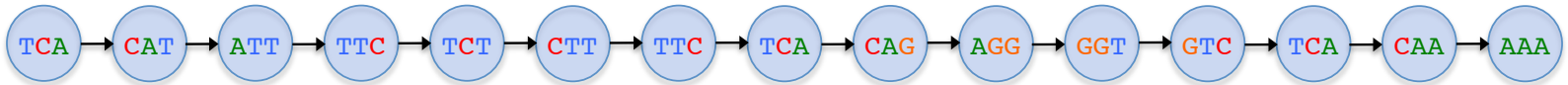
$\text{Composition}_k(\text{TCATTCTTCAGGTCAAA})$

„Genome order“:

TCA CAT ATT TTC TCT CTT TTC TCA CAG AGG GGT GTC TCA CAA AAA

Put k-mers into nodes

Connect nodes with edges



Naive Genome Assembly

But what if the nodes are not connected? Can we order them and put them back together again?

TCATTCTTCAGGTCAAA

Naive Genome Assembly

The question is whether we can reconstruct the original string if we only have the nodes and do not know what order they are on?

- Challenge: Find the following sequence based only on a collection of 3-mer subsequences:

TCATTCTTCAGGTCAAA

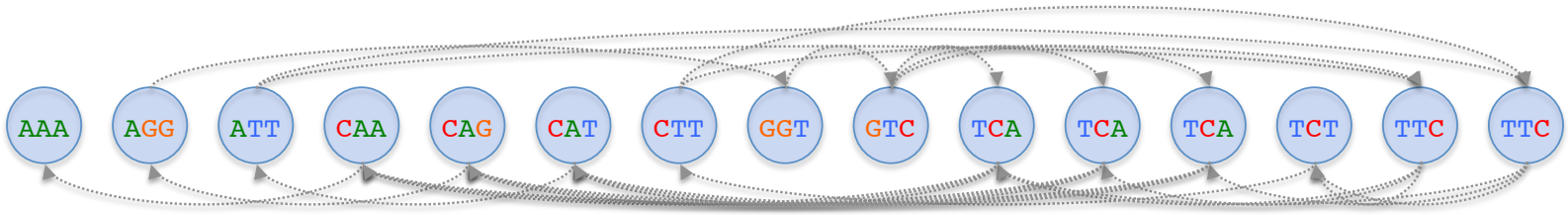
- The basic strategy to do this involves searching for overlaps between k-mers.
- E.g., connect k-mer$_i$ with k-mer$_j$ if

$$\text{suffix}(\text{k-mer}_i) = \text{prefix}(\text{k-mer}_j)$$

Naive Genome Assembly

If we do not know the order of the nodes the task seems rather difficult...

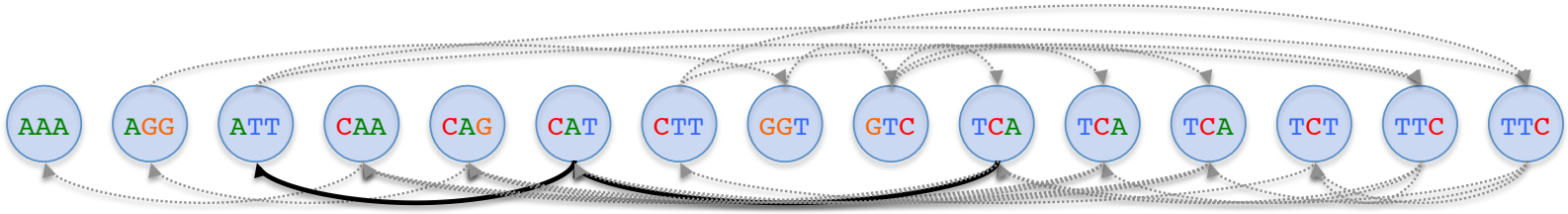
TCATTCTTCAGGTCAAA



Naive Genome Assembly

However, just to demonstrate how to generate a path that represents the sequence, let us pretend we are omnisicent start with the k-mer TCA and connect it to CAT and ATT.

TCATTCTTCAGGTCAAA

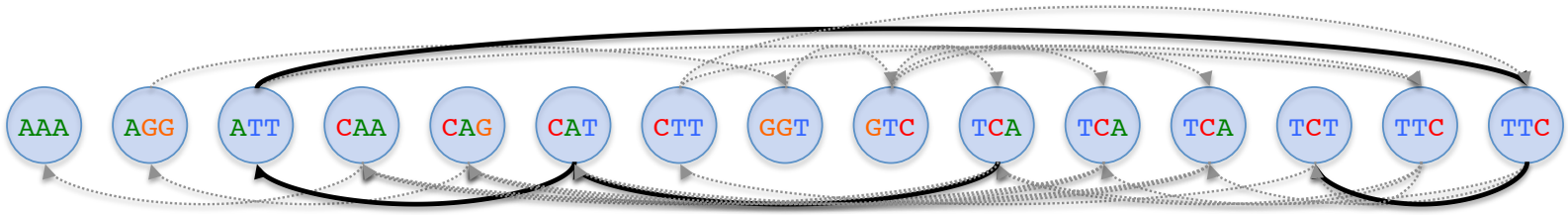


TCATT

Naive Genome Assembly

Conitnuing in this way ...

TCATTCTTCAGGTCAAA

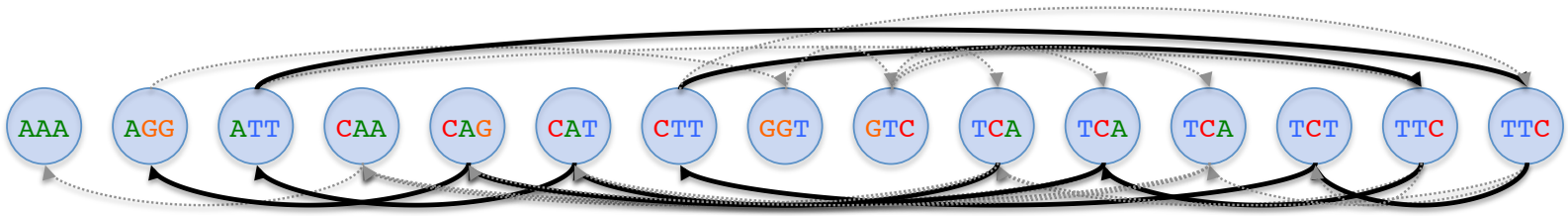


TCATTCT

Naive Genome Assembly

Further ...

TCATTCTTCAGGTCAAA

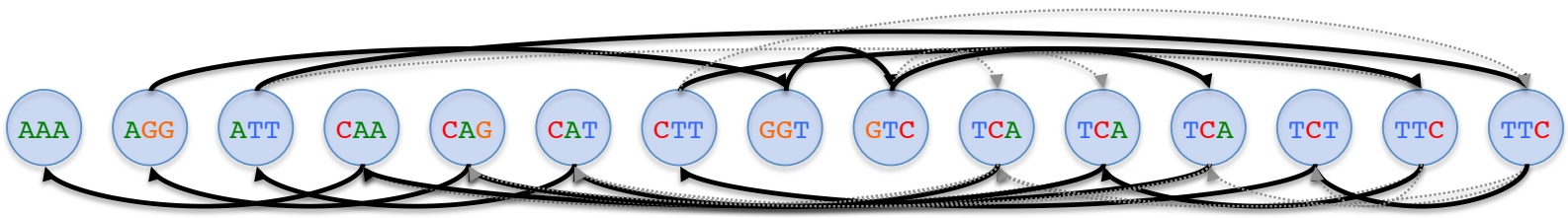


TCATTCTTCAG

Naive Genome Assembly

And finally ...

TCATTCTTCAG GTCAAA

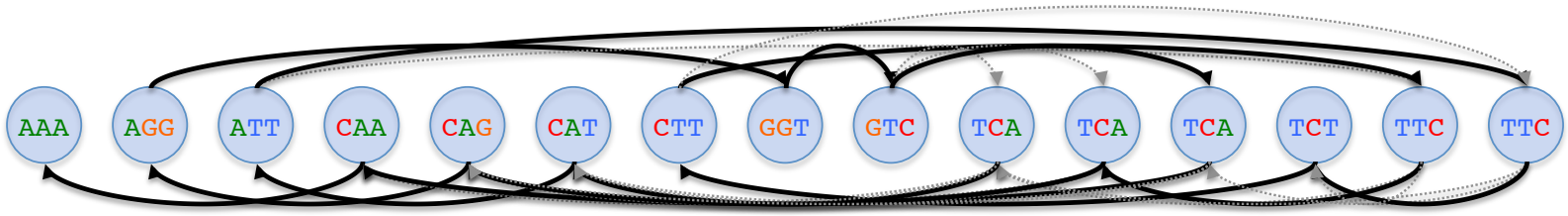


TCATTCTTCAGGTCAAA

Naive Genome Assembly

Notice that the solution to our problem was a path that visited every node exactly once.

TCATTCTTCAG GTCAAA

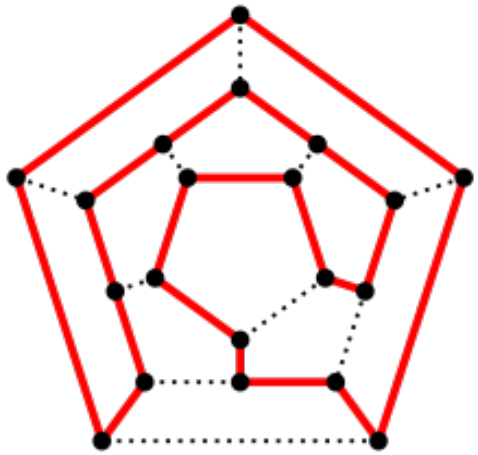


TCATTCTTCAGGTCAAA

Hamiltonian path

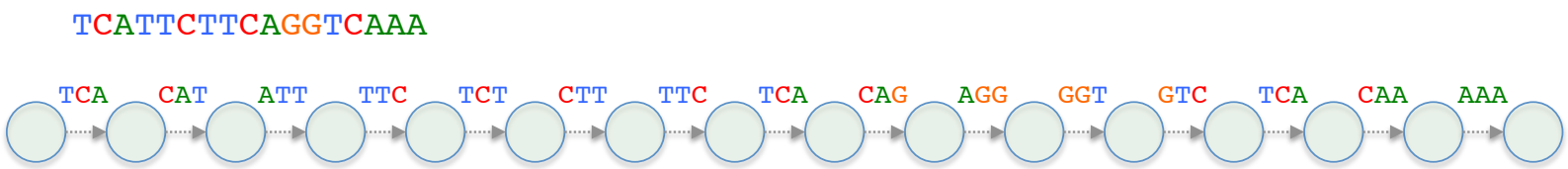
A **Hamiltonian path** is a path in an undirected or directed graph that visits each vertex exactly once. A Hamiltonian cycle is a Hamiltonian path that is a cycle.

- Determining whether Hamiltonian paths and cycles exist in graphs is the Hamiltonian path problem, which is NP-complete.



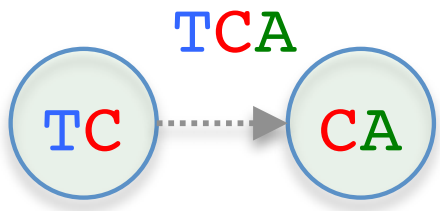
Another approach

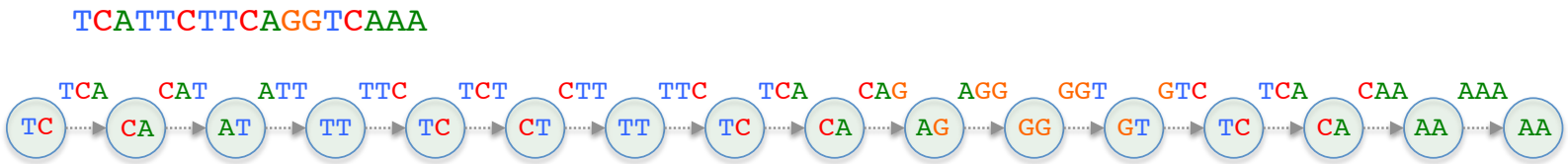
Instead of labeling the nodes with the k-mer subsequences, let us label the edges with these k-mers



Another approach

We will then label the nodes with the (k-1)-mer, i.e., 2-mer suffixes and prefixes

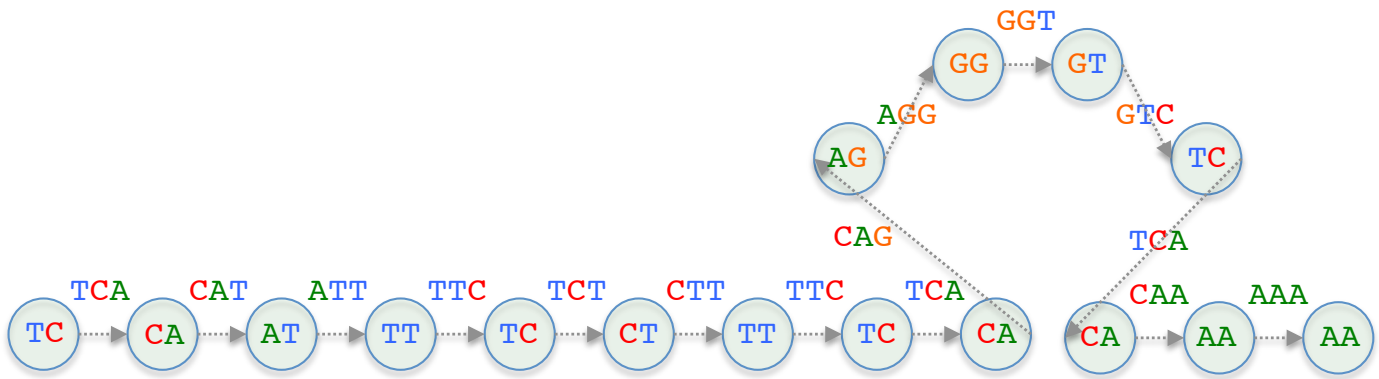




Constructing a de Bruijn graph

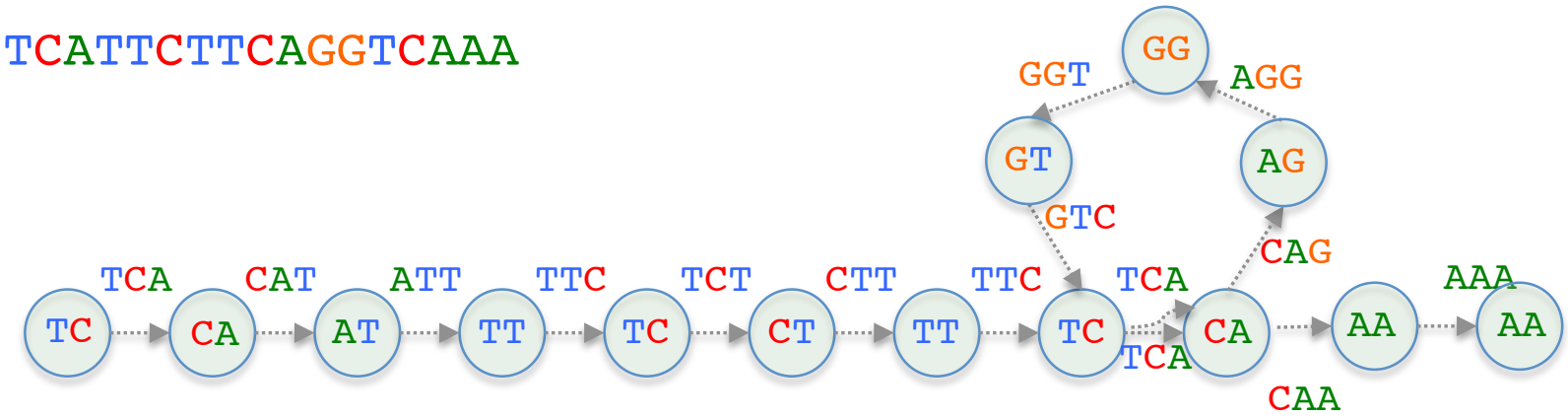
Let us now merge identically labels nodes in this graph. We will show the steps along the way for our example graph. A key idea is that we will **merge identical nodes whilst retaining the edges**.

TCATTCTTCAGGTCAAA



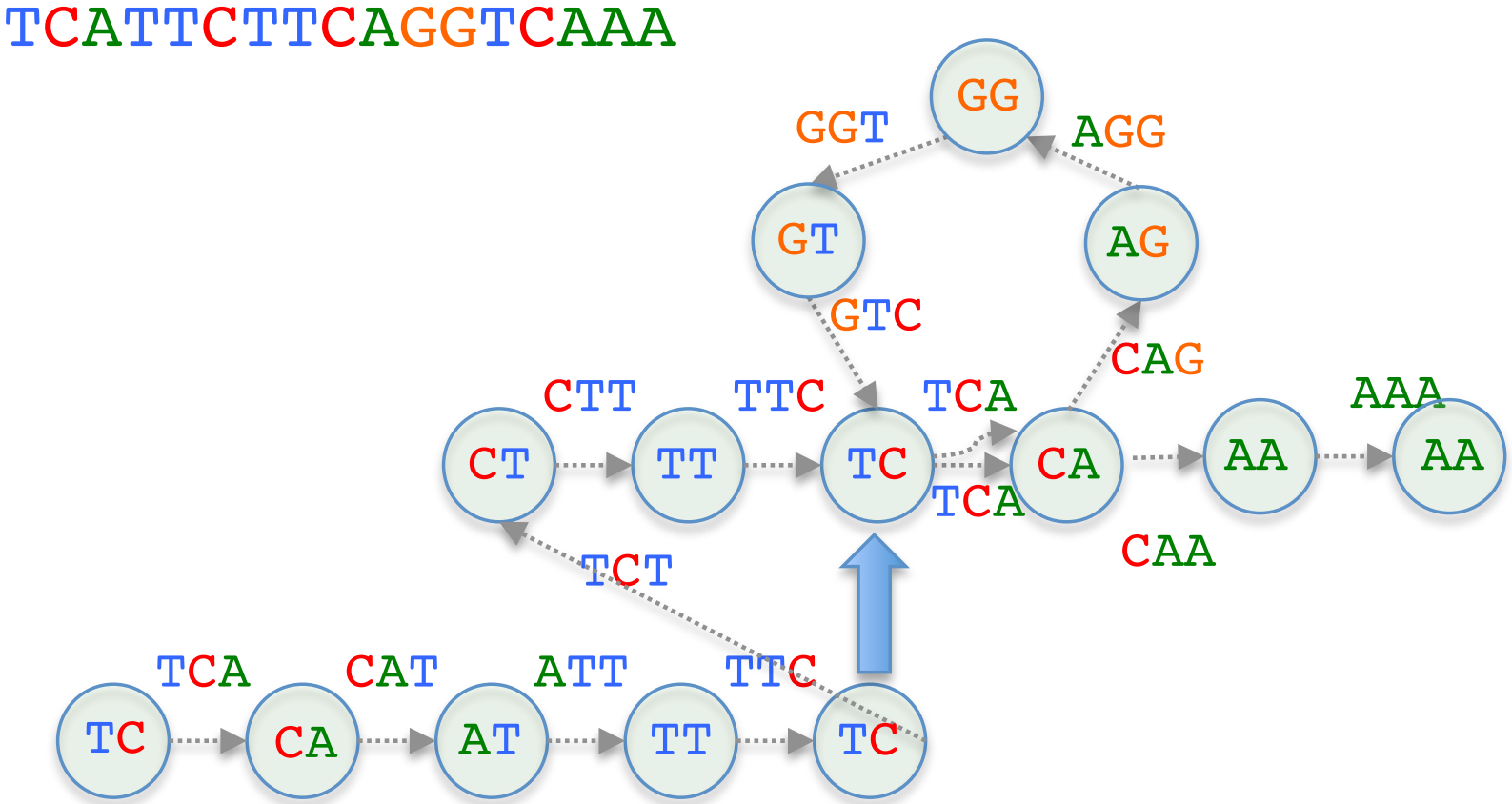
Constructing a de Bruijn graph

Merge two CA nodes whilst retaining their edges



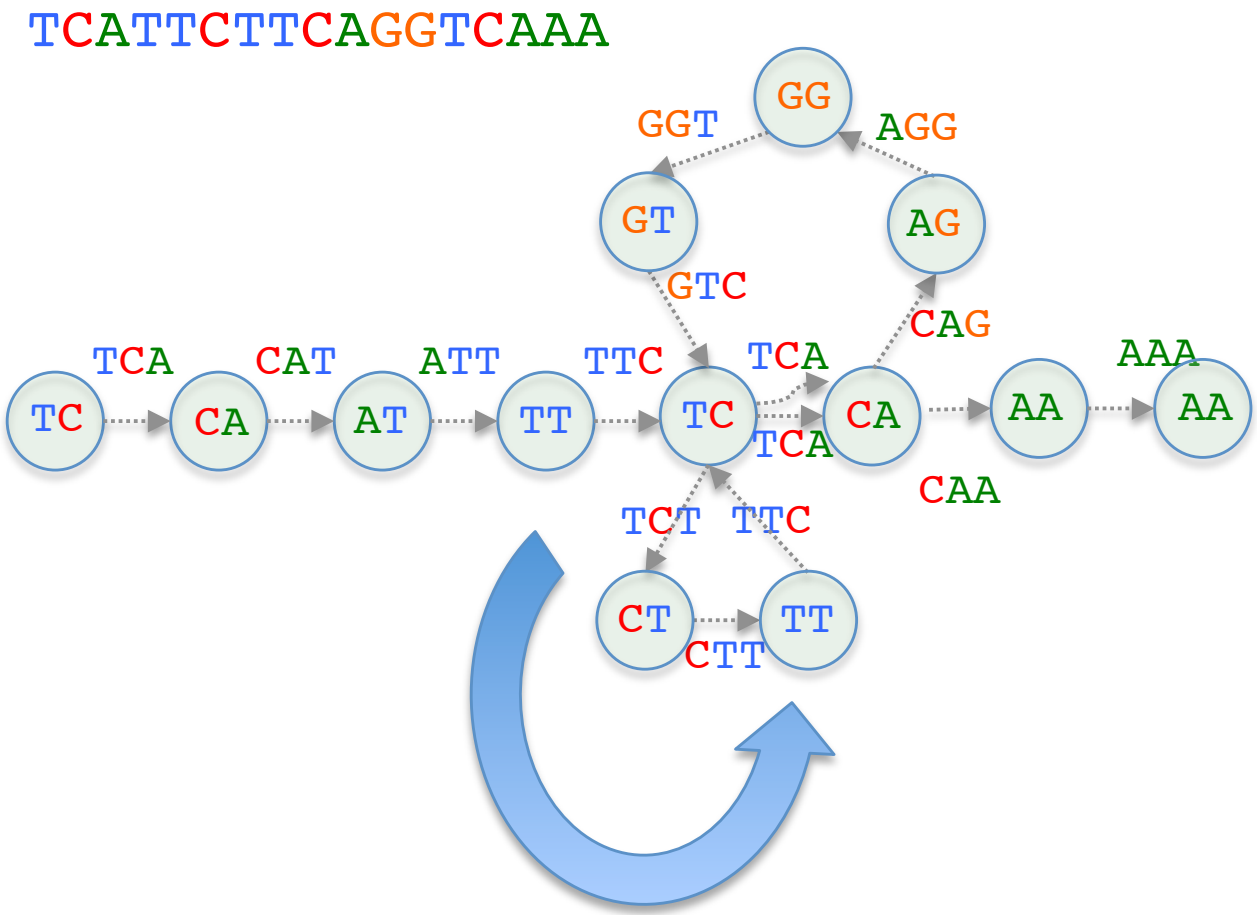
Constructing a de Bruijn graph

Continuing...



Constructing a de Bruijn graph

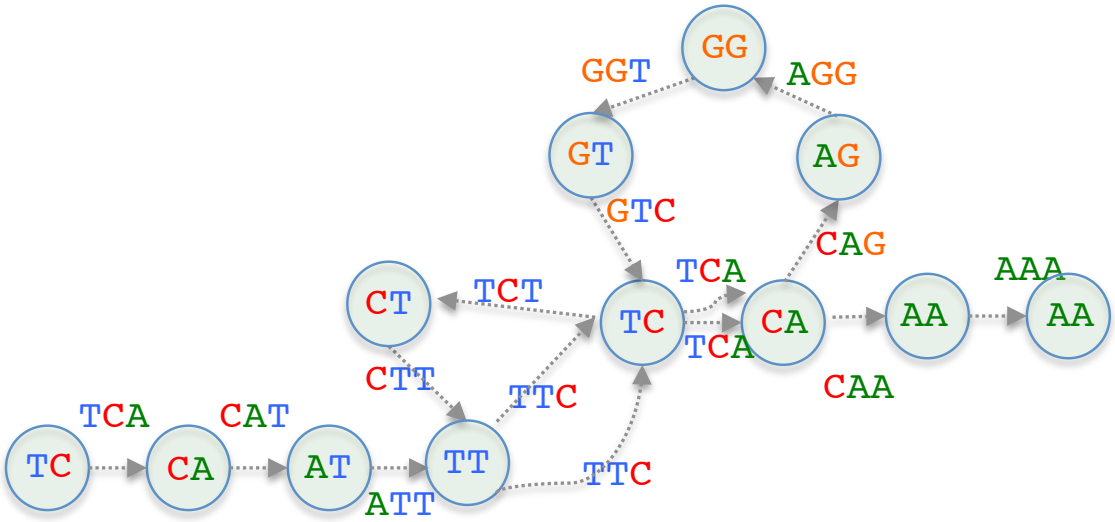
Merge two TC nodes whilst retaining their edges. Continuing...



Constructing a de Bruijn graph

Merge two TT nodes whilst retaining their edges. Continuing...

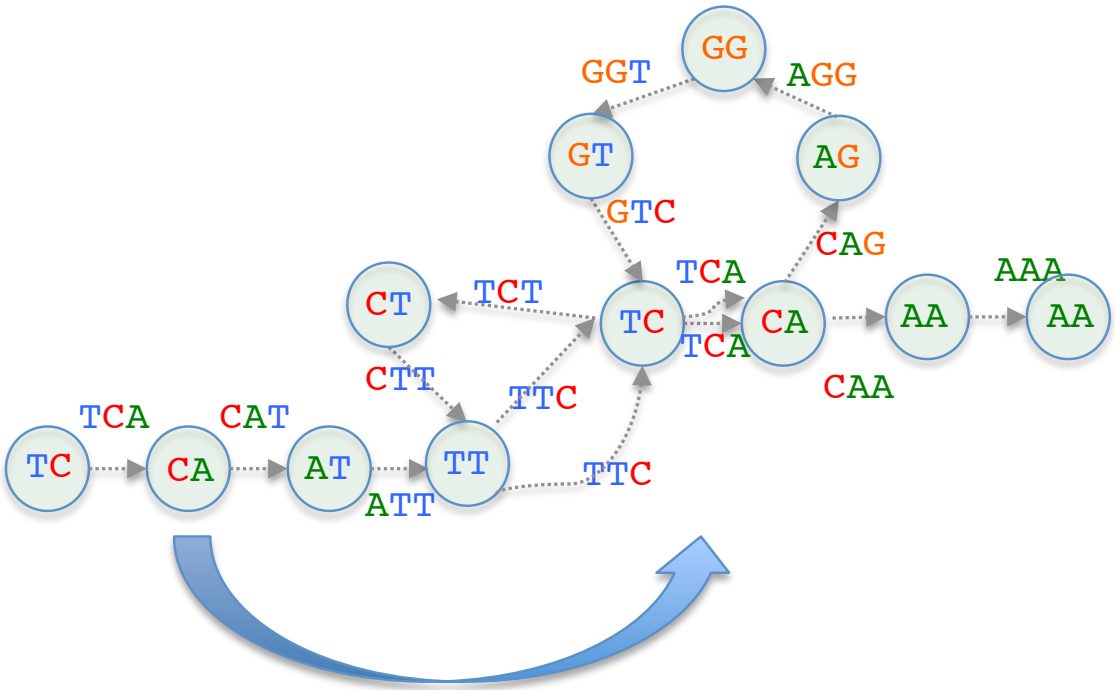
TCATTCTTCAGGTCAAA



Constructing a de Bruijn graph

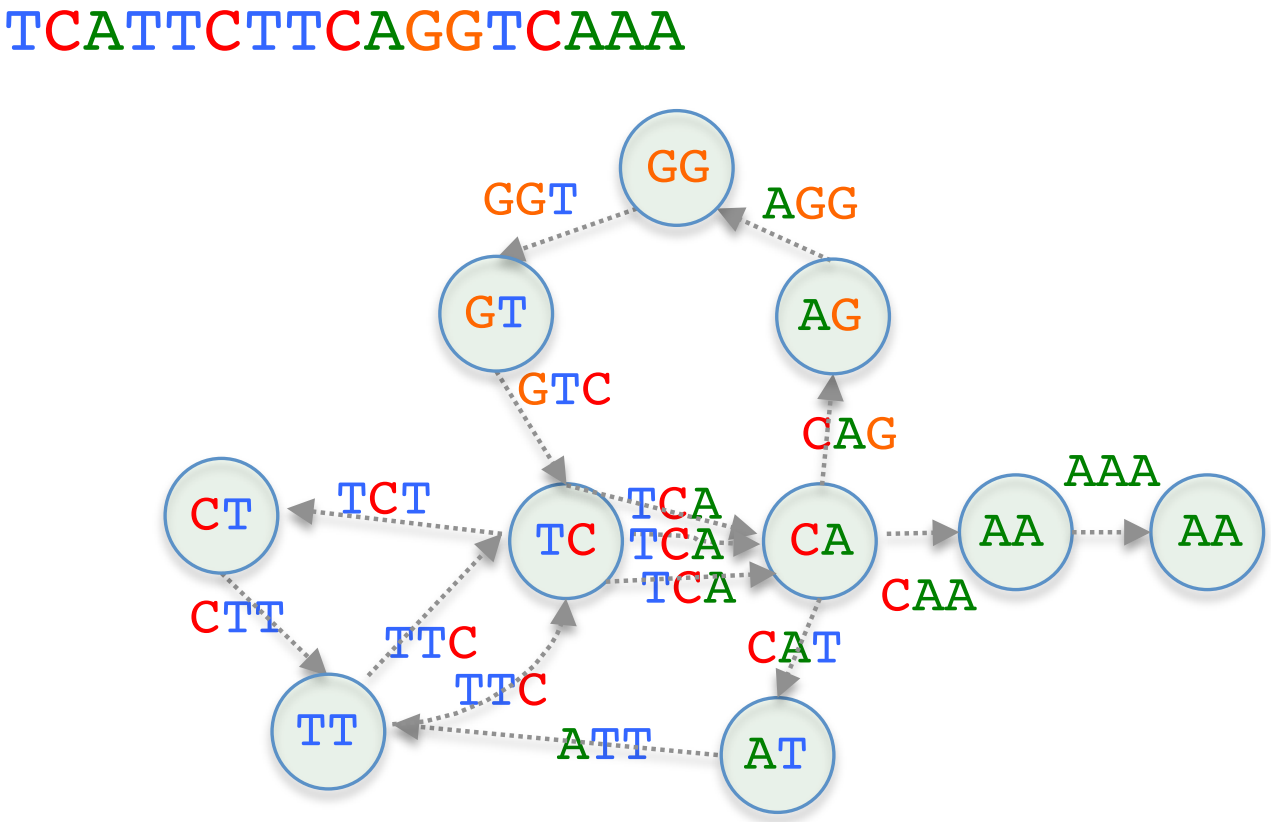
Continuing...

TCATTCTTCAGGTCAAA



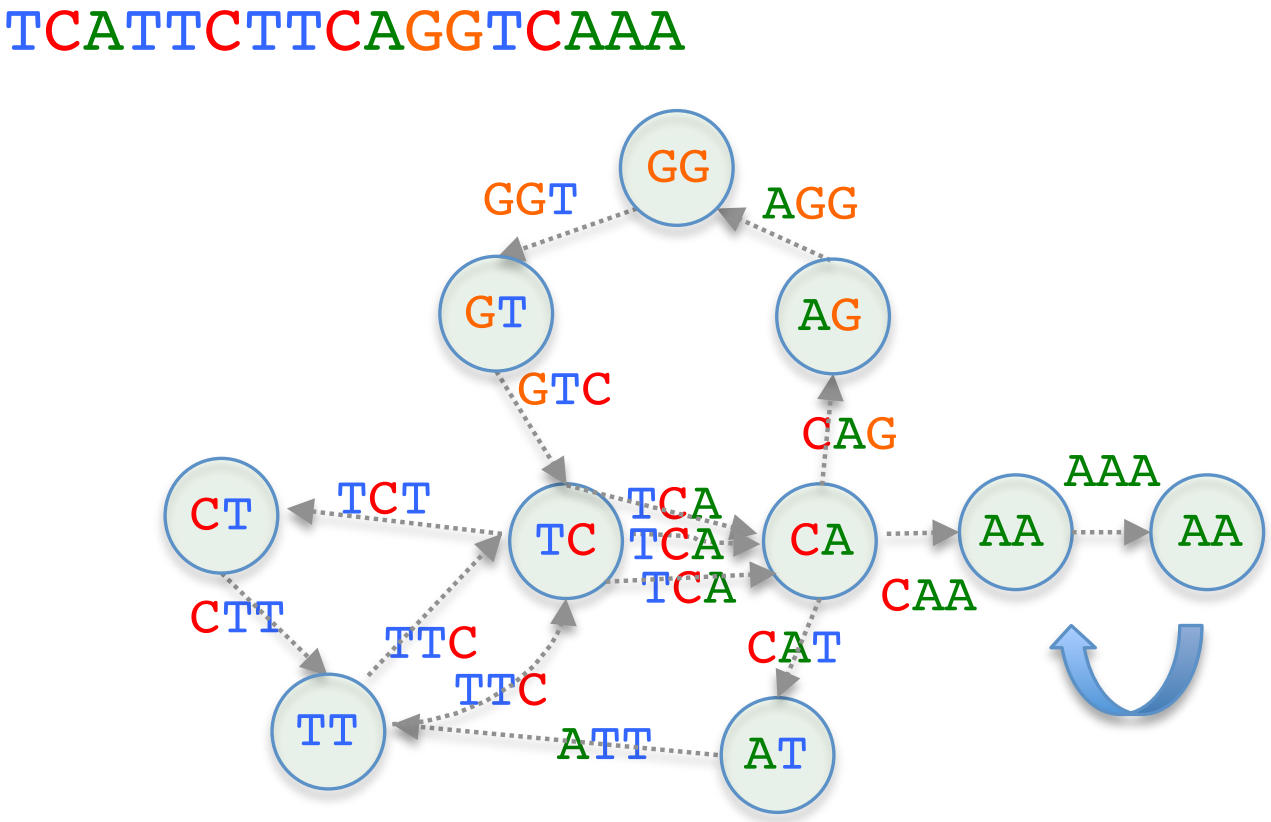
Constructing a de Bruijn graph

Merged two CA nodes whilst retaining their edges.



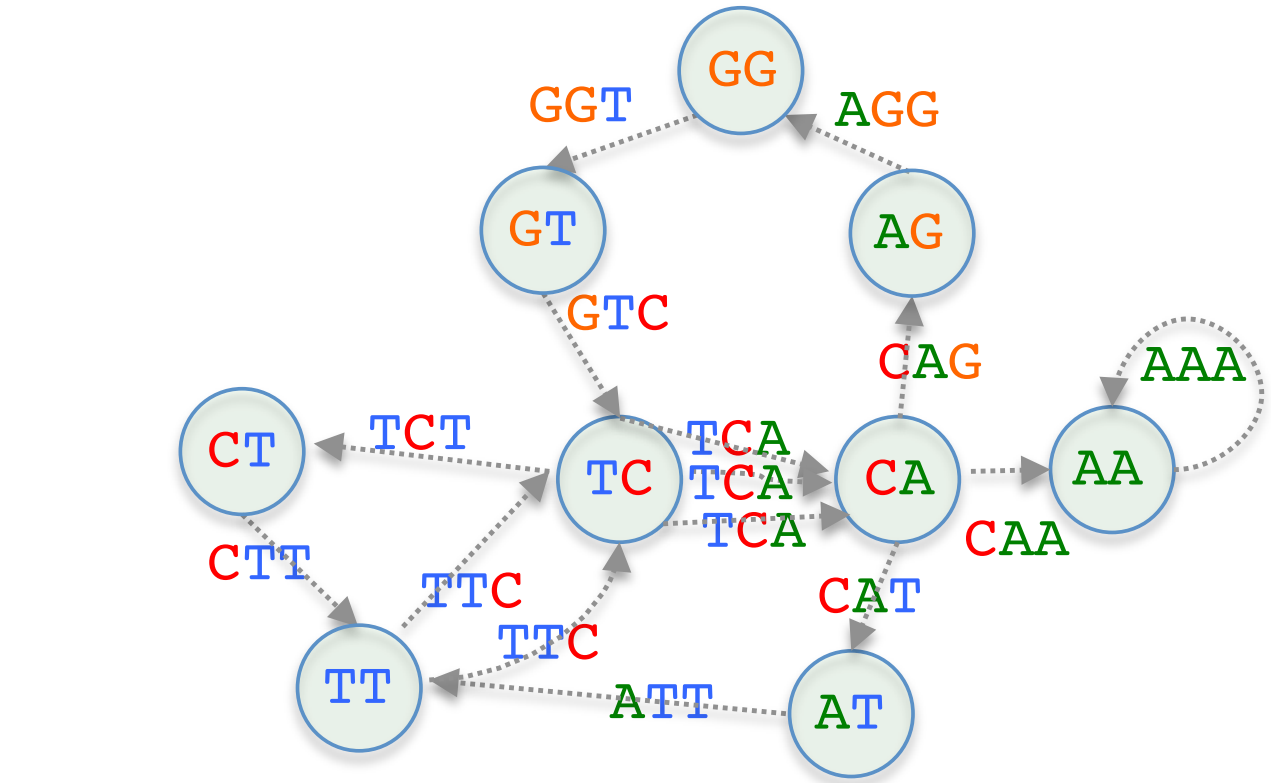
Constructing a de Bruijn graph

Continuing...



Constructing a de Bruijn graph

Merged two AA nodes whilst retaining their edges.

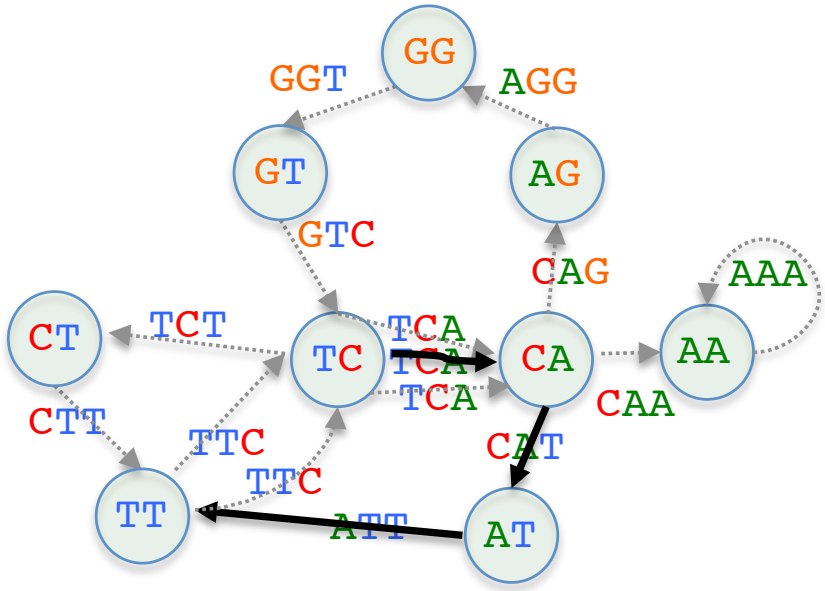


This is the de Bruijn graph of the string TCATTCTTCAGGTCAAA.

Traversing a de Bruijn graph

Let us now examine how we might reconstruct the original sequence from this de Bruijn graph

TCATTCTTCAGGTCAAA



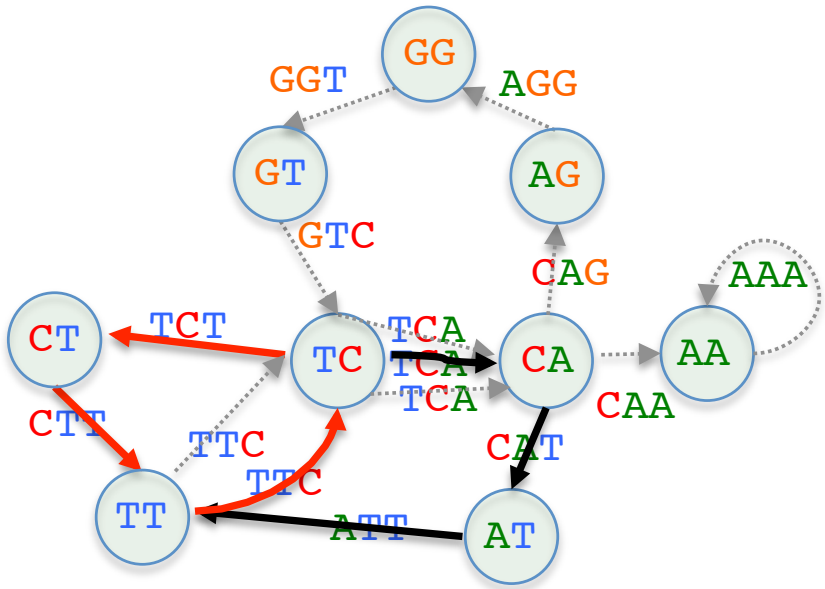
TCATT...

Follow the edges and write down the letters

Traversing a de Bruijn graph

Continuing

TCATTCTTCAGGTCAAA



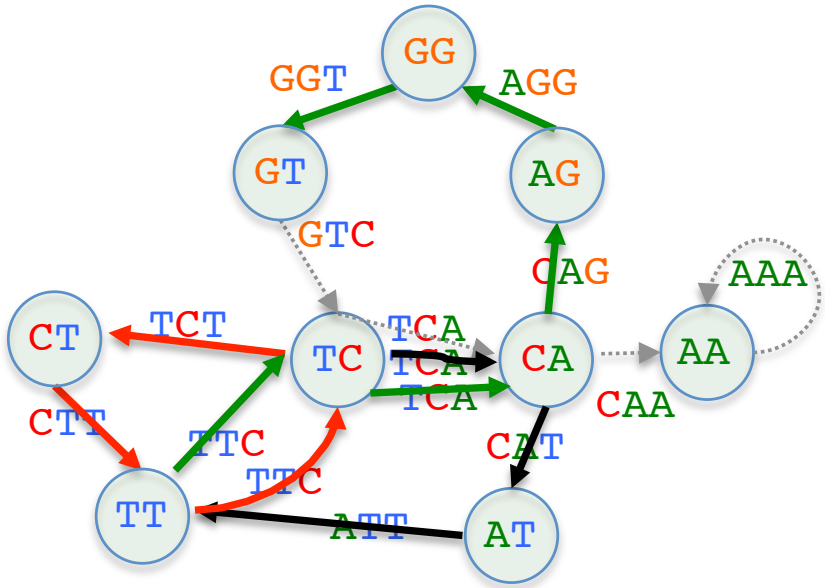
TCATTCTT...

Follow the edges and write down the letters

Traversing a de Bruijn graph

Continuing

TCATTCTTCAGGTCAAA



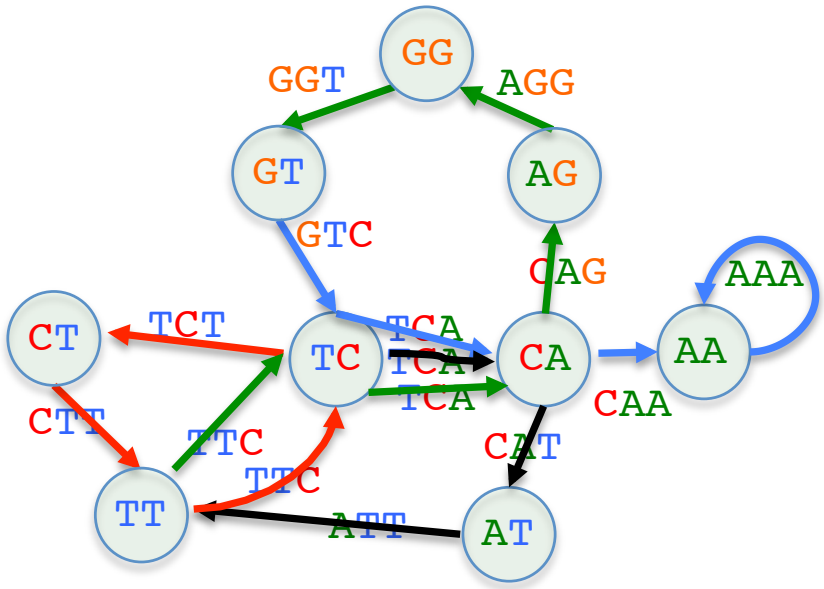
TCATTCTTCAGGT...

Follow the edges and write down the letters

Traversing a de Bruijn graph

Done!

TCATTCTTCAGGTCAAA

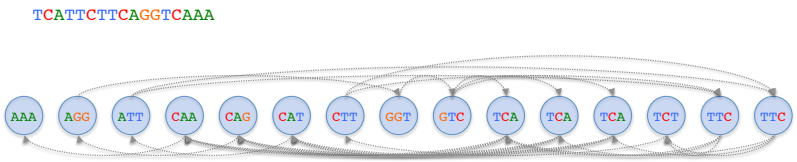


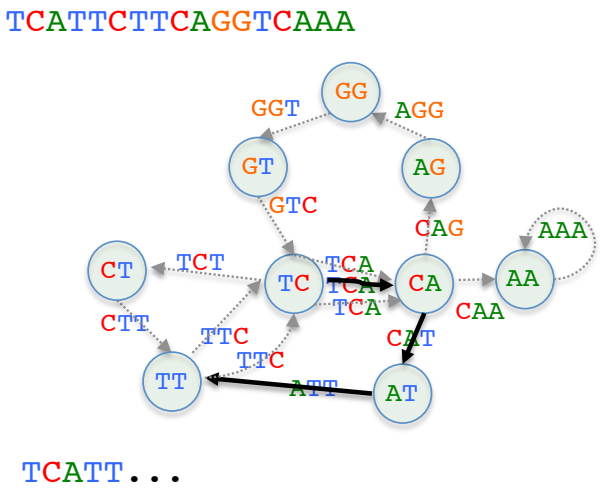
TCATTCTTCAGGTCAAA

Follow the edges and write down the letters

Hamilton and Euler

So we have seen two potential methodologies for traversing a graph to reconstruct a sequence based on the Hamiltonian and the Eulerian path problem.





- Which do we take?

Hamilton and Euler

The **Eulerian path problem** (Is there a path that visits every edge exactly once), and the **Hamiltonian path problem** (is there a path that visits every edge exactly once?) are superficially similar.

It turns out that[2]

- the Eulerian path problem has efificient algorithms to solve it
- the Hamiltonian path problem is NP complete

[2]We will not review proofs here, they are standard

de Bruijn Graph of k-mers

The de Bruijn graph of a collection of k-mers is

- A representation of every k-mer as an edge between its prefix and its suffix
- The nodes of the graph are thus the (k-1)-mer suffices and prefices
- All nodes with identical labels are merged, preserving edges

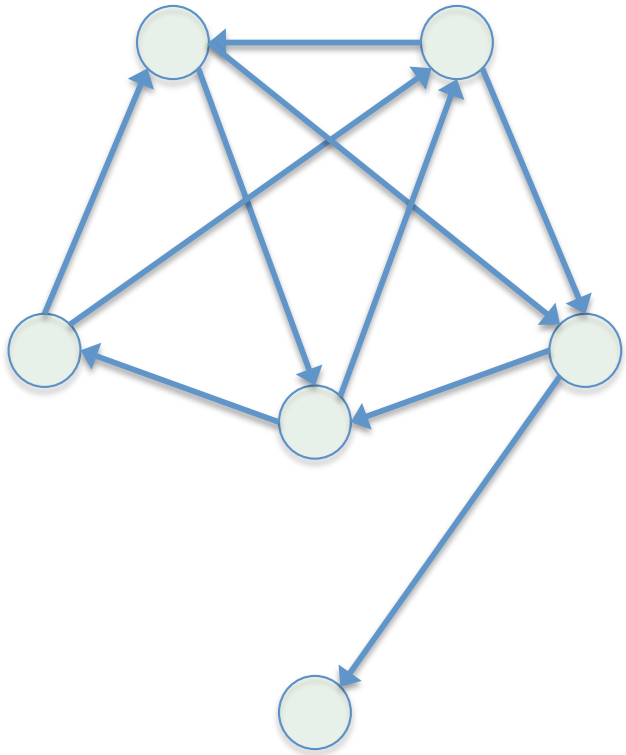
de Bruijn Graph of k-mers

Algorithm 1 Create annotation

```
1: Form a unique node for each k-mer in kmers
2: for each k-mer ∈ kmers do
3:     Connect prefix node and suffix node with edge
4: end for
```

Eulerian Graphs

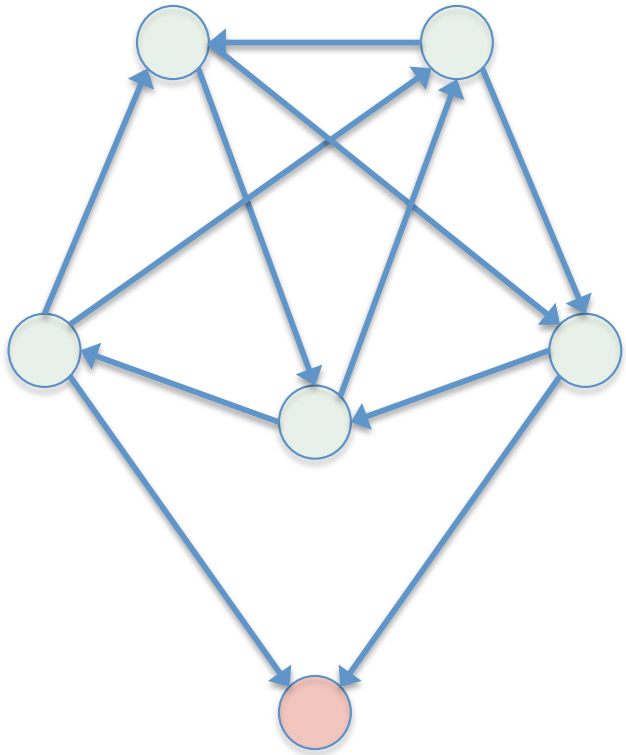
A **Eulerian cycle** is a path that traverses every edge exactly once and returns at the end of the traversal to the start node.



- Does this graph contain a Eulerian cycle?

Eulerian Graphs

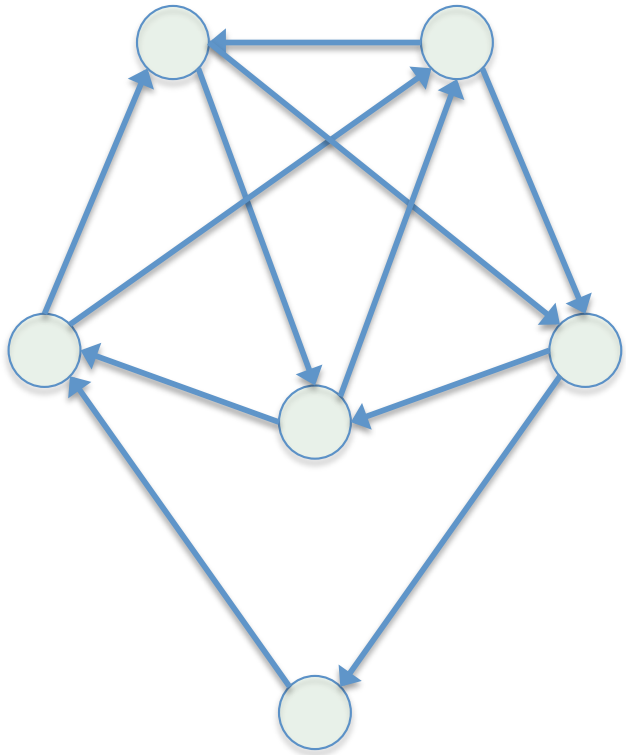
A **Eulerian cycle** is a path that traverses every edge exactly once and returns at the end of the traversal to the start node.



- Does this graph contain a Eulerian cycle?

Eulerian Graphs

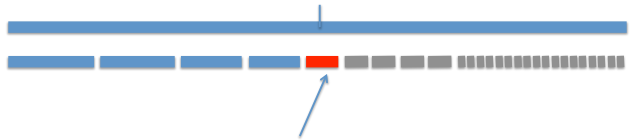
A **Eulerian cycle** is a path that traverses every edge exactly once and returns at the end of the traversal to the start node.



- Does this graph contain a Eulerian cycle?

N50

The N50 measure is used to estimate the quality of a genome assembly



- Arrange the contigs from largest to smallest
- Find the position where the contigs cover 50% of the total genome size
- The length of the contig at this position is defined as the **N50**
- The longer the N50 is, the better the assembly

Finally

- Email: peter.robinson@charite.de
- Office hours by appointment

Further reading

- Miller JR (2010) Assembly algorithms for next-generation sequencing data.
  Genomics **95**:315–327
- Flicek P, Birney E (2009) Sense from sequence reads: methods for alignment and assembly.
  Nature Methods **6**:S7–S11
- Li Z (2011) Comparison of the two major classes of assembly algorithms: overlap-layout-consensus and de-bruijn-graph.
  Brief Funct Genomics **11**:25-37.