# Next-generation sequencing

Lecture 4

# NGS

- Introduction to the background
- NGS workflow and accuracy
- Data format, quality control, data management
- Assembly
- RNA-seq
  - Aligner
  - Analysis tools
  - Applications, such as MiRNA
- Chip-seq
  - Applications

# Data format: fastq

Example of one read (Illumina):

```
@HWI-EAS209_0006_FC706VJ:5:58:5894:21141#ATCACG/1
TTAATTGGTAAATAAATCTCCTAATAGCTTAGATNTTACCTTNNNNNNNNNNTAGTTTCTTGAGATTTGTT
+HWI-EAS209_0006_FC706VJ:5:58:5894:21141#ATCACG/1
efcfffffcfeefffcffffffddf`feed]`]_Ba_^__[YBBBBBBBBBBRTT\]][]dddd`ddd^dd
```

Line 1: "@" + identifier
Line 2: sequence
Line 3: "+" + identifier (optional)
Line 4: phred-based quality scores

# Quality Score

- A quality value *Q* is an integer mapping of *p* (i.e., the error probability that the corresponding base call is incorrect. p is the small, the better).
- Two different equations have been in use:
  - Phred quality score

$$p = 10^{\frac{-Q}{10}}$$

  - Solexa quality score

$$\frac{p}{1-p} = 10^{\frac{-Q}{10}}$$

Ewing B, Green P. (1998) Genome Res. 8(3):186-194.

## Quality control

- remove reads from problematic tiles that may not be reliable due to sequencing chip quality
- remove reads with low quality, such as mean Q score < 20 for illumina RNA-seq.
- remove low quality bases at two ends of the reads until the quality score reaches a given threshold, such as mean Q score = 20.
- remove short reads.
  - FastQC tool (http://www.bioinformatics.bbsrc.ac.uk/projects/fastqc/)
  - NGSQC: Cross-Platform Quality Analysis Pipeline for Deep Sequencing Data.
    - http://brainarray.mbni.med.umich.edu/brainarray/ngsqc/
  - HTQC: a fast quality control toolkit for Illumina sequencing data
    - https://sourceforge.net/projects/htqc

# NGS

- Introduction to the background
- NGS workflow and accuracy
- Data format, quality control, data management
- Assembly: **sequence assembly** refers to aligning and merging fragments of a much longer DNA sequence in order to reconstruct the original sequence.
- RNA-seq
  - Aligner
  - Analysis tools
  - Applications, such as MiRNA
- Chip-seq
  - Applications

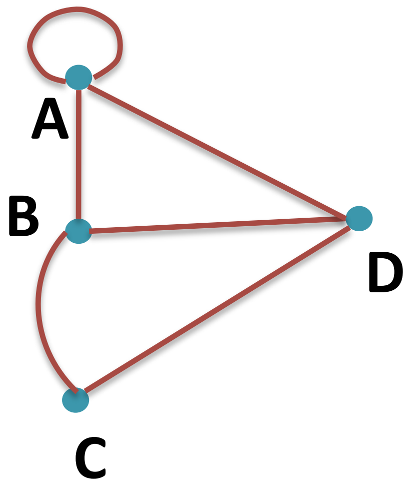
## Assembly

- Assembly algorithms
- De novo whole genome assembling strategies
- Mapping assembling strategies

## Assembly algorithms

- Assembly, finding an optimal path that connect all short reads (or part of short reads) once time, is NP-hard problem.
- No efficient solution.
- We need to use some approximation algorithm.

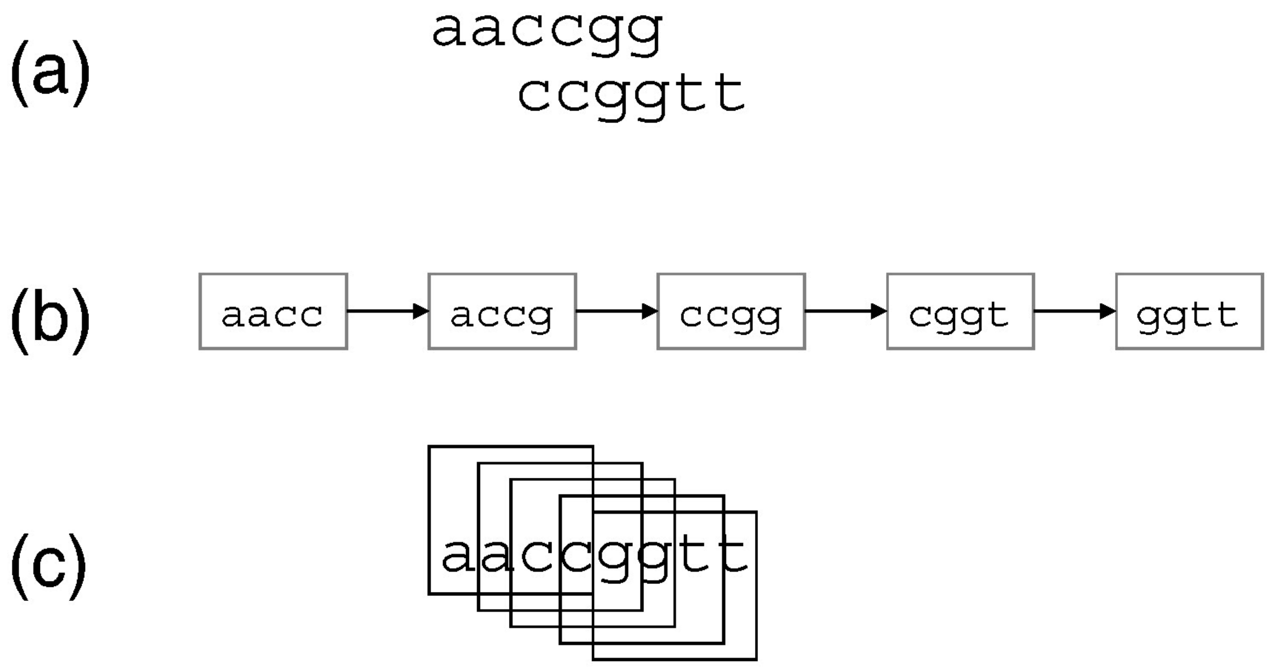
# Graph model for Assembly



**Graphs are made up by vertices (nodes) and edges (links).**
***G*=(*V*,*E*)**
***V*: a finite set of vertices.**
***E*: edges of the graph**

**Graph for Assembly:**

- **Graph model. A node is a read or a k-mer subsequence**
- **Edge: if there are over lap between two k-mer subsequence**



Miller, 2010

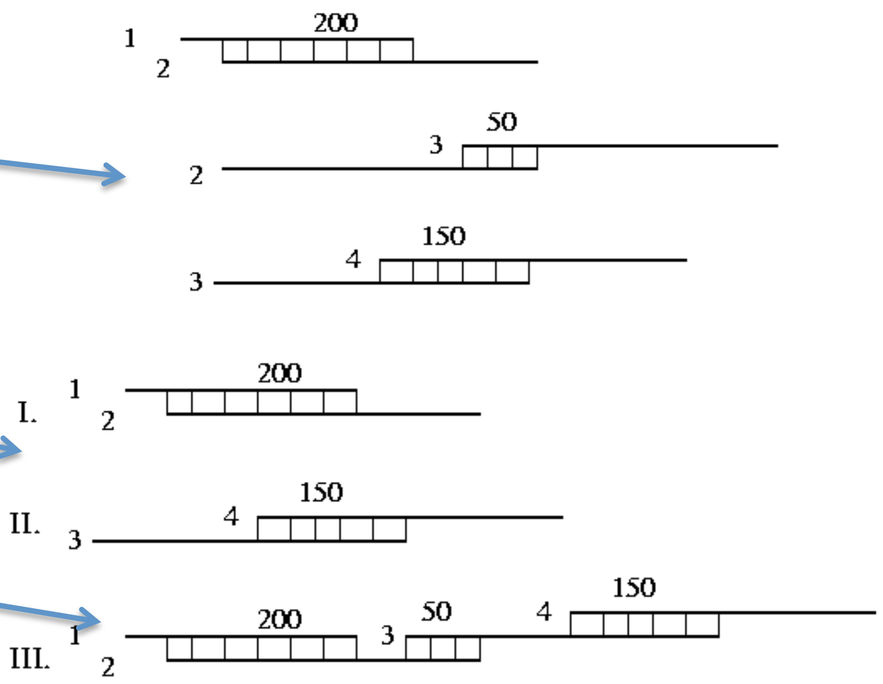
## Assembly Algorithms

Main algorithm used:

- Greedy algorithms
- Overlap Layout Consensus
- De brujin graphs

## Greedy Assembly

- Build a rough map of fragment overlaps (pairwise alignment)
- Pick the largest scoring overlap
- Merge the two fragments
- Repeat until no more merges can be done

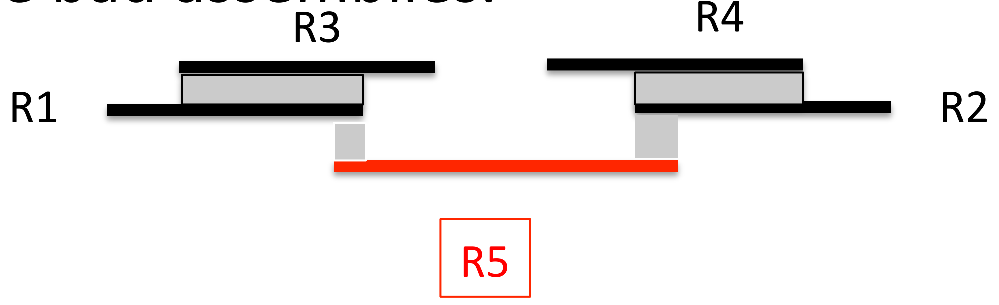


## Alternative Greedy Algorithm

- Instead of calculating overlaps of all reads:
  1. Start with a random read as an initial contig (seed)
  2. Go over all unassembled reads, pick the one that best fits the 3' end of the contig  and elongate the contig by this read.
  3. Repeat step 2 until no elongation is possible anymore.
  4. Repeat step 2 for the 5' end of the reverse complement of the contig.
  5. Stop in case of a conflict (fork)
     - if two reads that do not overlap with each other could elongate the contig equally well.

# Greedy Assembly

- Advantages:
  - Simple and easy to implement
  - effective
- Disadvantages
  - Since local information is considered at each step, the assembler can be easily confused by complex repeats, leading to mis-assemblies.
  - Local approach. Easy to be trapped into a local optimal solution (local minimum).
  - Early mistakes create bad assemblies.



# Greedy Assemblers

- TIGR Assembler: ftp://ftp.jcvi.org/pub/software/assembler/, Sanger, 2003
- SSAKE: http://www.bcgsc.ca/platform/bioinfo/software/ssake , small genomes, Solexa/Illumina, 2007
- SHARCGS: http://sharcgs.molgen.mpg.de/ , small genomes, Solexa/Illumina, SOLiD, Sanger, 454, 2007
- VCAKE: http://sourceforge.net/projects/vcake , small genomes, Solexa/Illumina, 2007
- Phrap: http://www.phrap.org/ , Sanger, 454, Solexa, 1995-2008

## Assembly Algorithms

Main algorithm used:

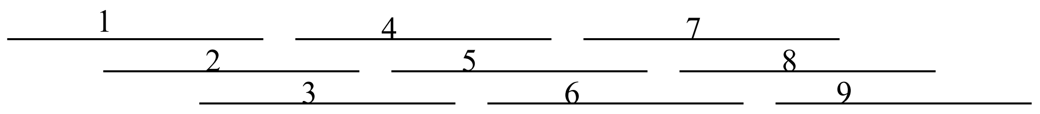
- Greedy algorithms
- Overlap Layout Consensus
- De brujin graphs

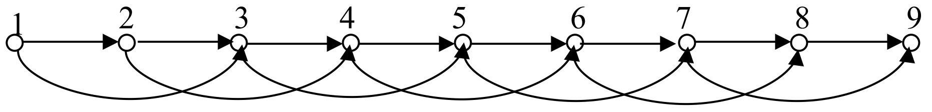
# Overlap-layout-consensus

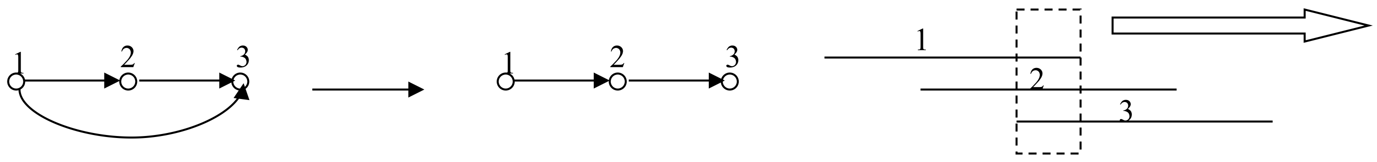
Main entity: read
Relationship between reads: overlap

**Graph model:**

- **A node is a**
- **Edge: if there are overlap between two reads**







ACCTGA

ACCTGA

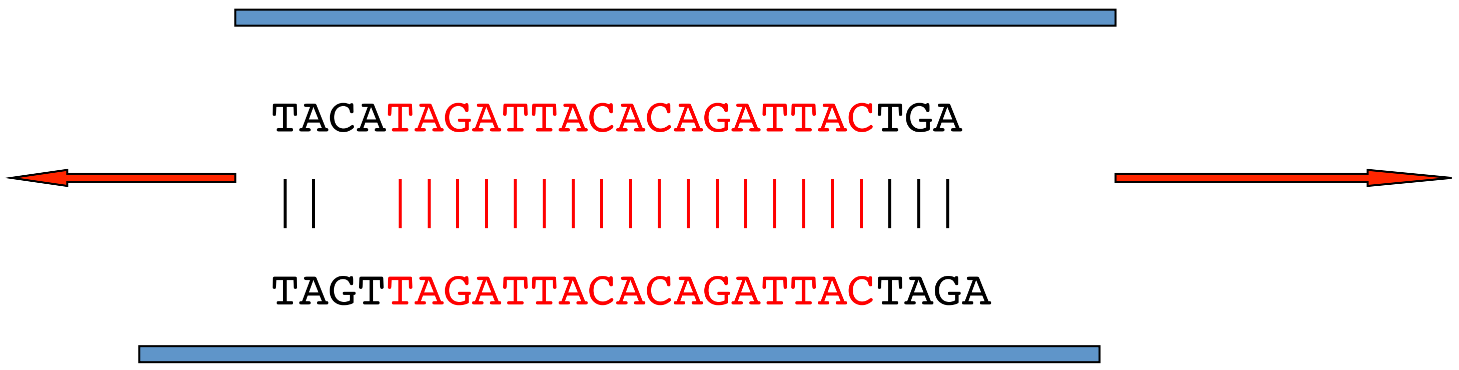
AGCTGA

ACCAGA

# Overlap-layout-consensus

## Step 1: Find Overlapping Reads

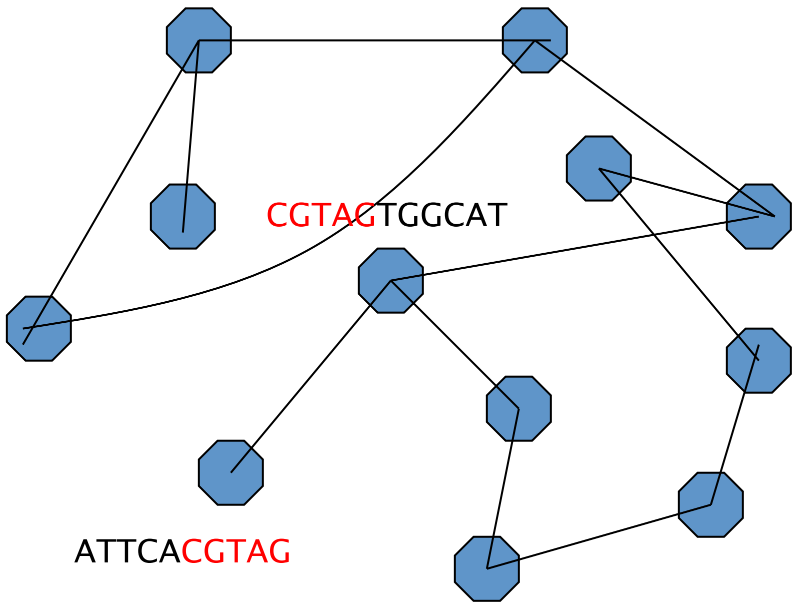
- What reads are intersecting ?
- Issues:
  1. Need efficient alignment algorithm (parallelization and index based strategies)
  2. Doesn't scale well when number of read is high
  3. Use seed based alignment with extension



# Overlap-layout-consensus

## Step 2: Construct overlap graph

- A graph is constructed:
  - Nodes are reads
  - Edges represent overlapping reads
- Need to simplify graph, such as remove redundant nodes and edges.



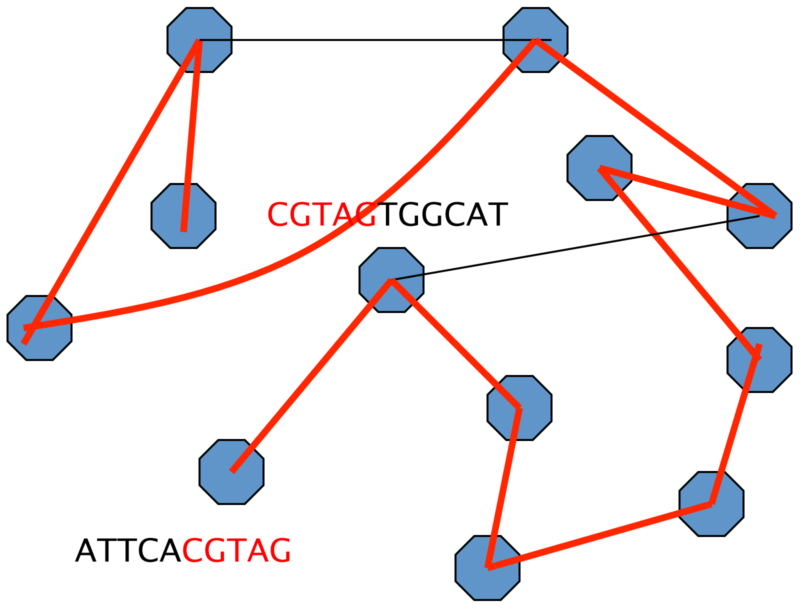
Overlap graph

# Overlap-layout-consensus

## Step 3: Consensus stage, Find Contigs

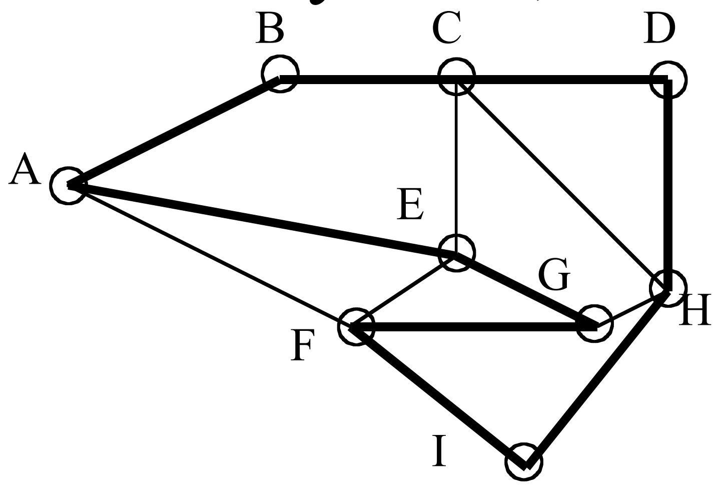
Try to find the Hamiltonian path:

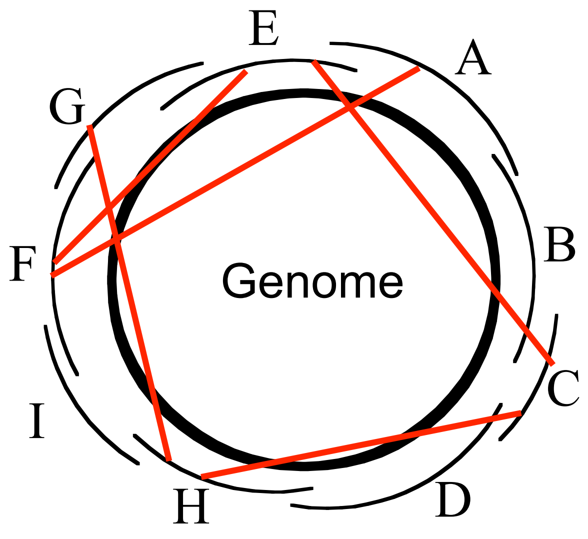
- A path in the graph contains each node exactly once.
- Following the Hamiltonian path, combine the overlapping sequences in the nodes into the sequence of the genome
- Computationally expensive (NP-hard problem)



# Circular genome

- Hamiltonian circuit: visit each node (city) exactly once, returning to the start





## Overlap-layout-consensus

- Better than Greedy algorithm. It can generate correct order of contigs that the Greedy algorithms may have errors.
- No efficient algorithm to find the Hamiltonian path
- Short fragment length = very small overlap therefore many false overlaps
- Overlap discovery is sensitive to settings of K-mer size, minimum overlap length, and minimum percent identity required for an overlap
- Large number of reads + short overlap + higher error are challenging for the overlap - layout - consensus approach
- Can’t assemble repeat longer than read length
- It is mostly used with Sanger or 454 data.

## Overlap-layout-consensus

- Celera (CABOG): http://www.jcvi.org/cms/research/projects/celera-assembler/overview/ , large genome, Sanger, 454,Solexa, 2004/2010
- Newbler: http://www.454.com/ , 454, Sanger, 2009
- Mira: http://sourceforge.net/apps/mediawiki/mira-assembler/ Sanger, 454, Solexa, 1998/2011
- Edena: http://www.genomic.ch/edena.php Snger, 454, 2008/2013