Sequence Assembly-III

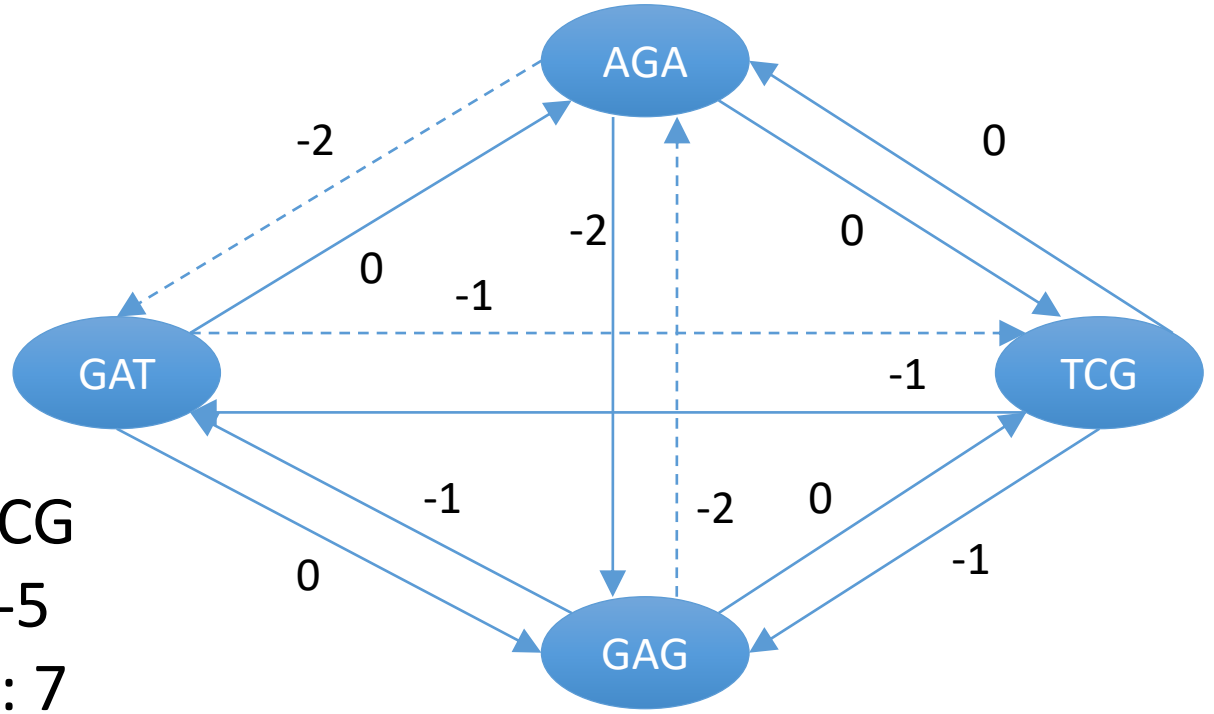
Hammad Naveed

hammad.naveed@nu.edu.pk

The Greedy Algorithm

- While *G* is disconnected
  - Pop the next possible edge *e* = (*u*,*v*) off of Q
  - If *outdegree(u) = 0* and *indegree(v) = 0* and *e* does not create a cycle
    - Add *e* to *G*

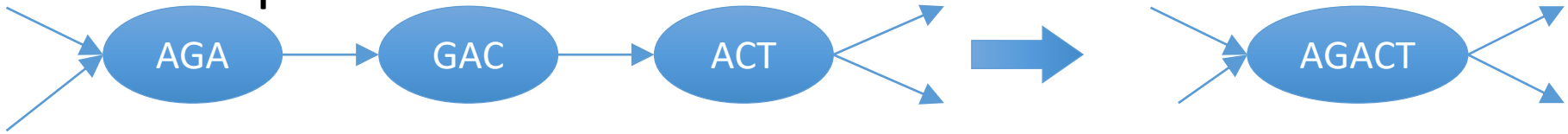
- GAG -> AGA -2
- AGA -> GAG -2
- AGA -> GAT -2
- GAG -> GAT -1
- TCG -> GAT -1
- TCG -> GAG -1
- GAT -> TCG -1



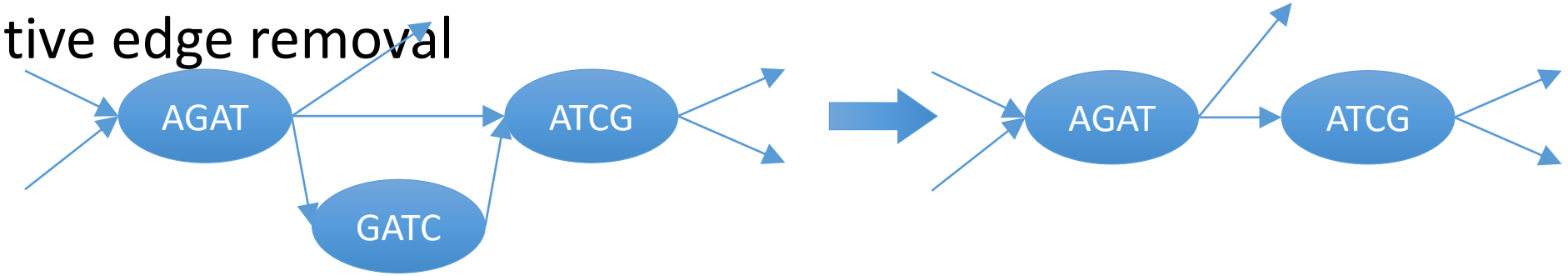
Path: GAGATCG
Path length: -5
String length: 7

Simplifications of overlap graph

- Require minimum length for overlap
- Linear chain compression

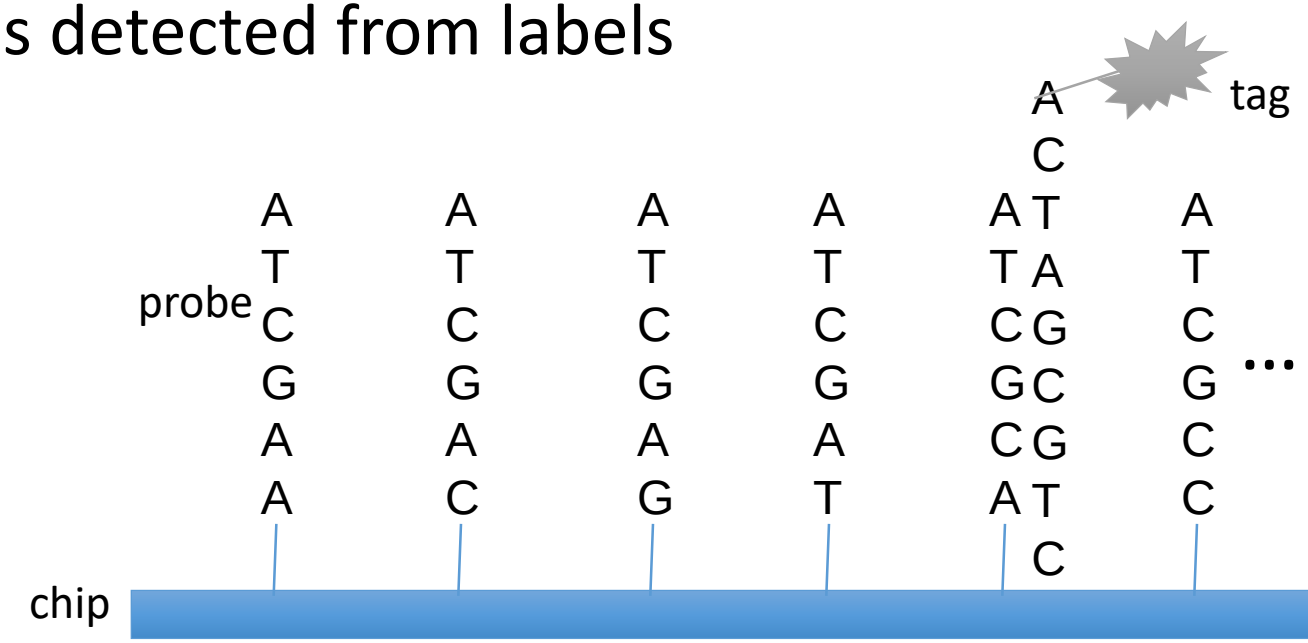


- Transitive edge removal

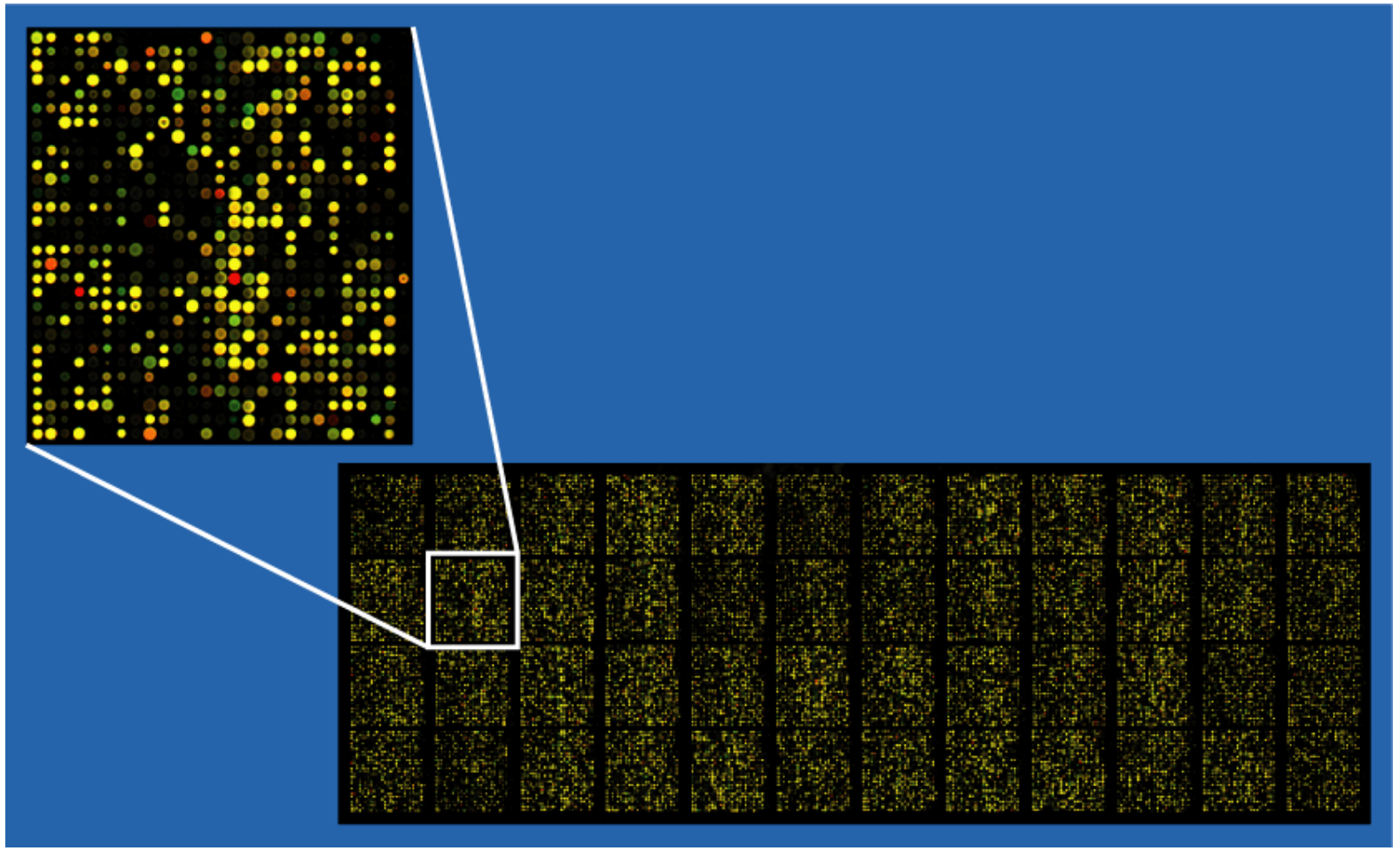


Universal DNA arrays

- Array with all possible oligonucleotides (short DNA sequence) of a certain length as probes
- Sample is labeled and then washed over array
- Hybridization is detected from labels



Reading a DNA array



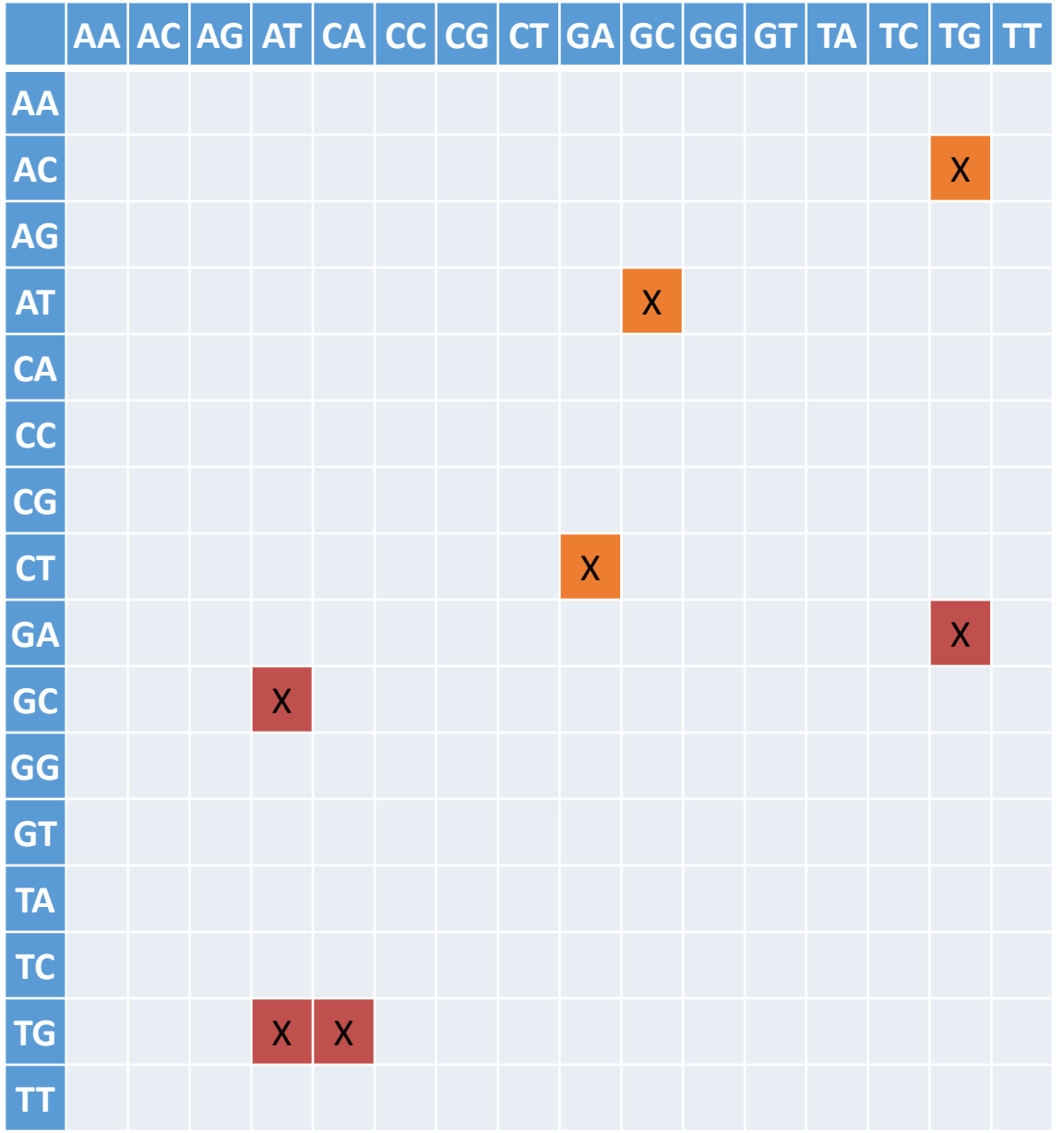
Sequencing by Hybridization (SBH)

- SBH array has probes for all possible *k*-mers
- For a given DNA sample, array tells us whether each *k*-mer is *PRESENT* or *ABSENT* in the sample
- The set of all *k*-mers present in a string *s* is called its *spectrum*
- Example:
  - s = ACTGATGCAT
  - *spectrum(s, 3)* = {ACT, ATG, CAT, CTG, GAT, GCA, TGA, TGC}

Example DNA Array

Sample:
ACTGATGCAT

Spectrum (k=4):
{ACTG, ATGC,
CTGA,GATG,
GCAT,TGAT,
TGCA}



SBH Problem

- Given: A set *S* of *k*-mers
- Do: Find a string *s*, such that *spectrum(s,k) = S*

{ACT, ATG, CAT, CTG, GAT, GCA, TGA, TGC}

?

Different sequences – the same spectrum

- Different sequences may have the same spectrum:

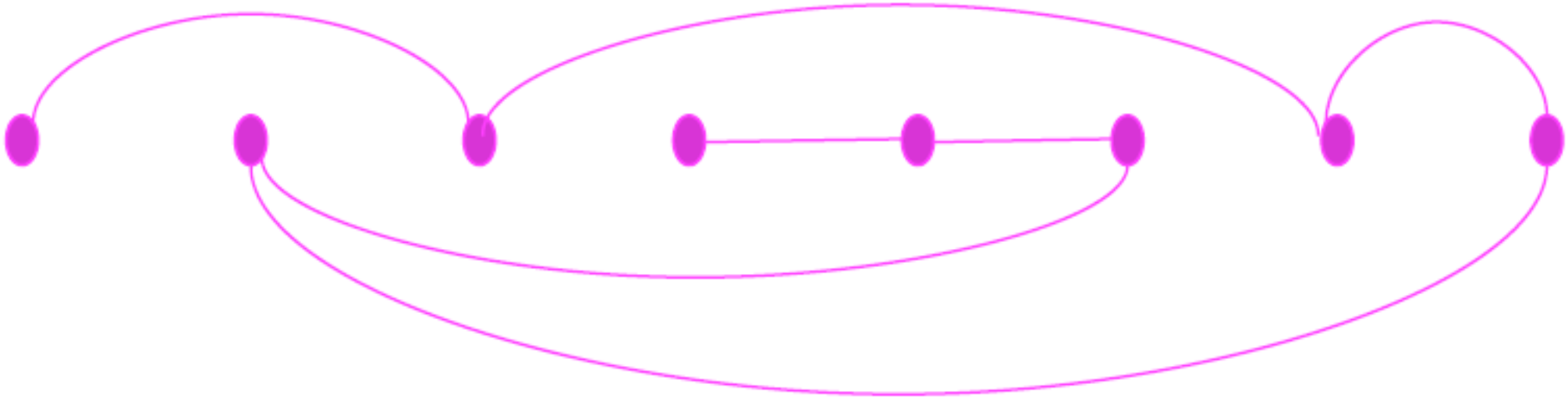
  Spectrum(GTATCT,2)=

  Spectrum(GTCTAT,2)=

  {AT, CT, GT, TA, TC}

Hamiltonian Path Approach

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

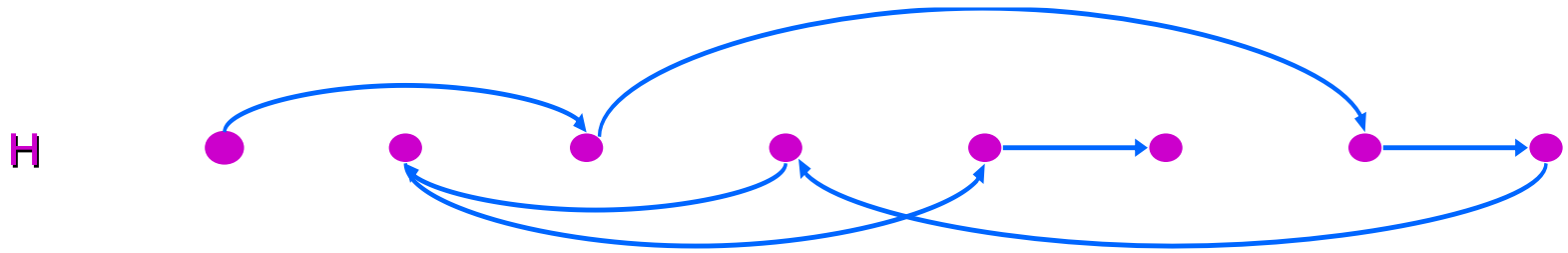


ATGCAGGTCC

Path visited every VERTEX once

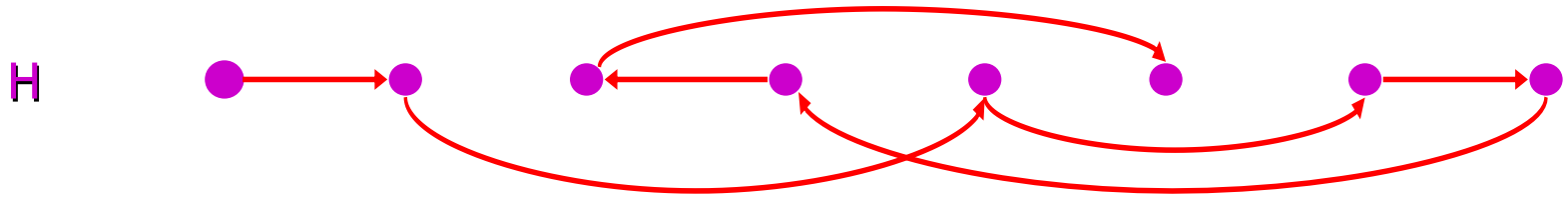
Hamiltonian Path Approach

- A more complicated graph:
- *S* = { ATG TGG TGC GTG GGC GCA GCG CGT }
- Path 1:



ATGCGTGGCA

- Path 2:



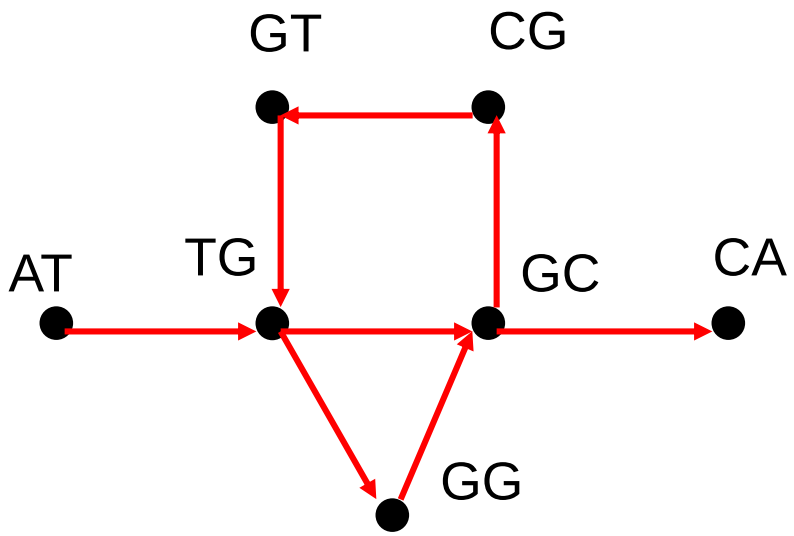
ATGGCGTGCA

SBH as Eulerian path

- Could use Hamiltonian path approach, but not useful due to *NP*-completeness
- Instead, use *Eulerian* path approach
- *Eulerian path*: A path through a graph that traverses every edge exactly once
- Construct graph with all *(k-1)-mers* as vertices
- For each *k*-mer in spectrum, add edge from vertex representing *first k-1* characters to vertex representing *last k-1* characters

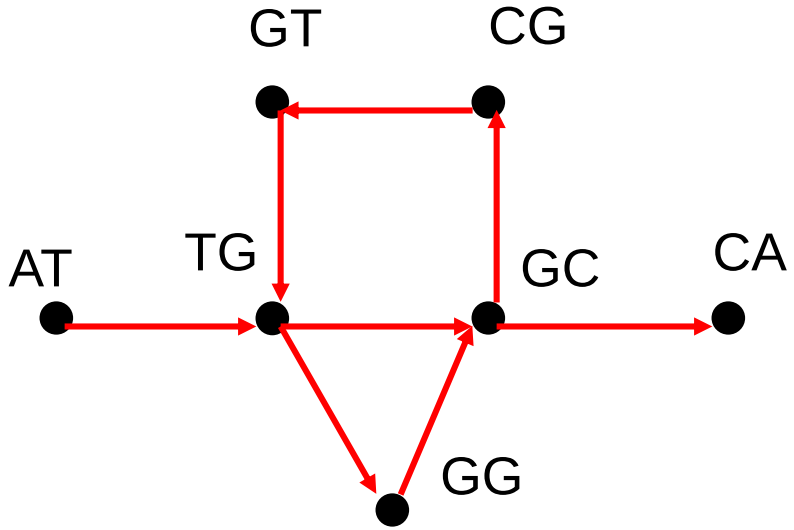
Eulerian Path Approach

- = { ATG, TGC, GTG, GGC, GCA, GCG, CGT }
- Vertices correspond to (*k*– 1)–mers : { AT, TG, GC, GG, GT, CA, CG }
- Edges correspond to *k*–mers from *S*



Path visited every EDGE once

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



ATGGCGTGCA

ATGCGTGGCA

Properties of Eulerian graphs

- It will be easier to consider *Eulerian cycles*: Eulerian paths that form a cycle
- Graphs that have an *Eulerian cycle* are simply called *Eulerian*
- ***Theorem:*** A connected directed graph is *Eulerian* if and only if each of its vertices are *balanced*
- A vertex *v* is *balanced* if *indegree(v) = outdegree(v)*
- There is a polynomial-time algorithm for finding Eulerian cycles!

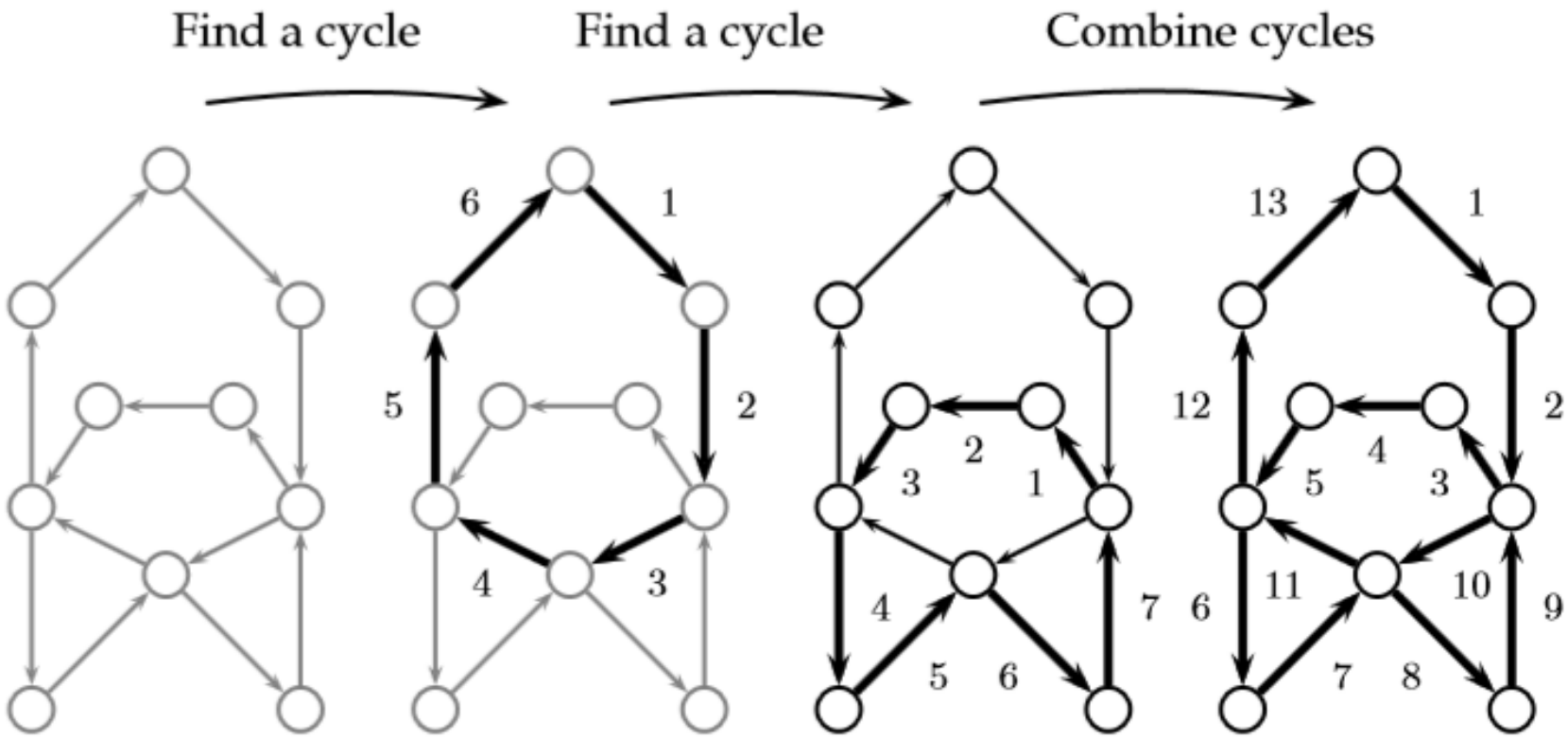
Seven Bridges of Königsberg

Euler answered the question: “Is there a walk through the city that traverses each bridge exactly once?”

Eulerian cycle algorithm

- Start at any vertex *v*, traverse unused edges until returning to *v*
- While the cycle is not Eulerian
  - Pick a vertex *w* along the cycle for which there are untraversed outgoing edges
  - Traverse unused edges until ending up back at *w*
  - Join two cycles into one cycle

Joining cycles

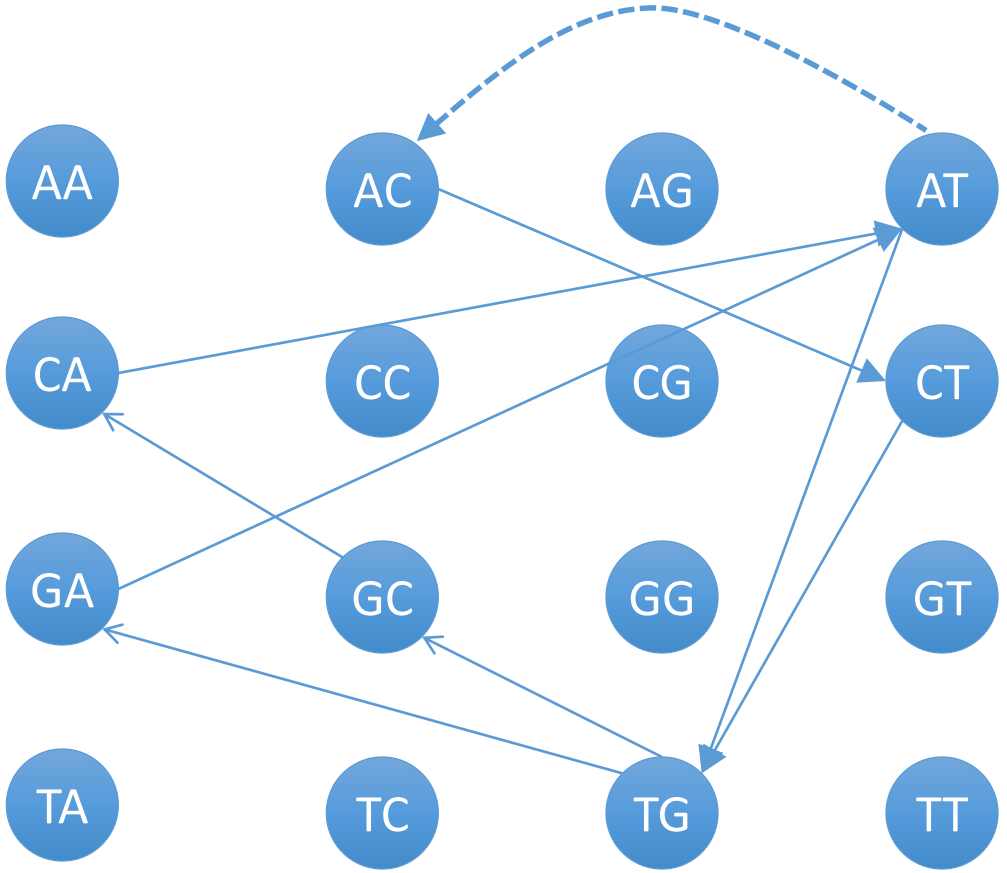


Eulerian Path -> Eulerian Cycle

- If a graph has an Eulerian Path starting at *s* and ending at *t* then
  - All vertices must be balanced, except for *s* and *t* which may have $|indegree(v) - outdegree(v)| = 1$
  - If and *s* and *t* are not balanced, add an edge between them to balance
    - Graph now has an Eulerian cycle which can be converted to an Eulerian path by removal of the added edge

SBH graph example

{ACT, ATG, CAT, CTG, GAT, GCA, TGA, TGC}



SBH difficulties

- In practice, sequencing by hybridization is hard
  - Arrays are often inaccurate -> incorrect spectra
    - False positives/negatives
  - Need long probes to deal with repetitive sequence
    - But the number of probes needed is exponential in the length of the probes!
    - There is a limit to the number of probes per array (currently between 1-10 million probes / array)

Fragment assembly challenges

- Read errors
  - Complicates computing read overlaps
- Repeats
  - Roughly half of the human genome is composed of repetitive elements
  - Repetitive elements can be long (1000s of bp)
  - Human genome
    - 1 million Alu repeats (~300 bp)
    - 200,000 *LINE* repeats (~1000 bp)

Overlap-Layout-Consensus

- Most common assembler strategy for long reads

1. *Overlap*: Find all significant overlaps between reads, allowing for errors
2. *Layout*: Determine path through overlapping reads representing assembled sequence
3. *Consensus*: Correct for errors in reads using layout

Consensus

Layout

GTATCGTAGCTGACTGCGCTGC
ATCGTCTCGTAGCTGACTGCGCTGC
ATCGTATCGAATCGTAG
TGACTGCGCTGCATCGTATCGTATC

Consensus TGACTGCGCTGCATCGTATCGTATCGTAGCTGACTGCGCTGC

References

- Lecture notes of Colin Dewey @ University of Wisconsin-Madison
- Lecture notes of Arne Elofsson @ Stockholm University
- Lecture notes of Yuzhen Ye @ Indiana University