Review of whole genome methods

- Suffix-tree based
  - MUMmer, Mauve, multi-Mauve
- Gene based
  - Mercator, multiple orthology approaches
- Dot plot/clustering based
  - MUMmer 2.0, Pipmaker, LASTZ

Background for yeast study

- Brewing evolved in middle ages Europe to produce ale-type beer via *Saccharomyces cerevisiae*, the same yeast used in wine and leavened bread.

- Lager-brewing arose in 15th century Bavaria, and is the most popular technique

- Lager, however, requires slow, low temperature fermentation by cryotolerant yeast(s).

Saccharomyces pastorianus

- Used to make lager, but never has been found in wild and depends on humans
- Allotetraploid hybrid of *S. cerevisiae* and an unknown yeast species.
- Understanding this unique contribution is important for understanding domestication of this yeast for human use

Table 1. Strains used in this study, and their culture collection aliases

	Strain	Culture collection aliases[a]			Earliest collection entry date	Other information	Collection locale
		CBS	DBVPG[b]	NCYC			
S. pastorianus strains							
Group 1	GSY509	**2440**		398	June 1952		Brewery-Saaz type beer; bottom yeast
	GSY133	1486	**6258**[1]	397	June 1935		Brewery-Saaz type beer
	GSY501	**1174**			June 1931		Brewery-Saaz type beer
	GSY131	1538	**6047**[1]	392	October 1935 (described by Hansen in 1904)	*S. pastorianus*-type strain	Carlsberg Brewery
	GSY137		**6284**			AJL248	Alfred Jorgensen's Laboratorium (now Danbrew)
	GSY129	1513	**6033**[1]	396	October 1947 (original culture 1883, Hansen)	*S. carlsbergensis*-type strain	Carlsberg Brewery; bottom yeast no. I
	GSY134	1503	**6261**		(original culture 1908, Hansen)	*S. monacensis*-type strain	Carlsberg Brewery ; bottom yeast no. II
Group 2	GSY132	1260	**6257**	400	March 1937		Frohberg-type bottom yeast, Netherlands
	GSY138		**6285**[2]			M 1563	Copenhagen
	GSY139		**6560**[2]			C83 1562	Denmark
	GSY135		**6282**[2]		1962	BK 2233	Labatt Brewery, Canada; bottom-fermenting
	GSY136		**6283**[2]		1969	BK 2230	Rainier Brewery, WA; bottom-fermenting
	GSY516	**6903**			September 1976		Brewery, Netherlands
	GSY515	**5832**			December 1967		Brewery, Netherlands
	GSY503	**1483**			July 1927		Brewery-Heineken, Netherlands; bottom yeast
	GSY504	**1484**			February 1925		Cloudy beer—Oranjeboom, Netherlands; bottom yeast
	GSY508	**2156**		457	June 1955		Brewery, Netherlands
S. cerevisiae strains							
Ale strains	GSY161					Wyeast1388	Belgian Strong Ale; probable origin Duvel
	GSY708					Wyeast1056	American Ale Yeast; probable origin Sierra Nevada and/or Ballantine breweries
	GSY934					Leinenkugel Ale	Miller brewery collection, Leinenkugel ale, WI

Results

- Saccharomyes are associated with oak trees in Northern hemisphere.

- This study focused on Patagonia in South America with 123 cryotolerant species and two isolates of *S. cerevisiae*. The fact so many were cryotolerant is unique relative to the northern hemisphere.

- These group with biological assays with the two known contaminants of lager/cider/wine fermentation

Lager paper

- Three cool facts when you get a chance to read
  - Yeast used for lager beer probably arose in ale breweries
  - Two distinct types of lager yeast, referred to as groups 1 and 2
  - Both groups probably arose independently in Europe

Domestication and analysis

- Lager yeast is a mix of at least three yeast species
- Interestingly, all cryotolerant species have the same chunk of S. cervisiae useful for processing maltose
  - Maltose is one of the most abundant sugars in wort used in brewing
- Relationships are contentious as the lager yeast and related yeasts previously were only found in human fermentation efforts- resolved via seq

Suffix arrays

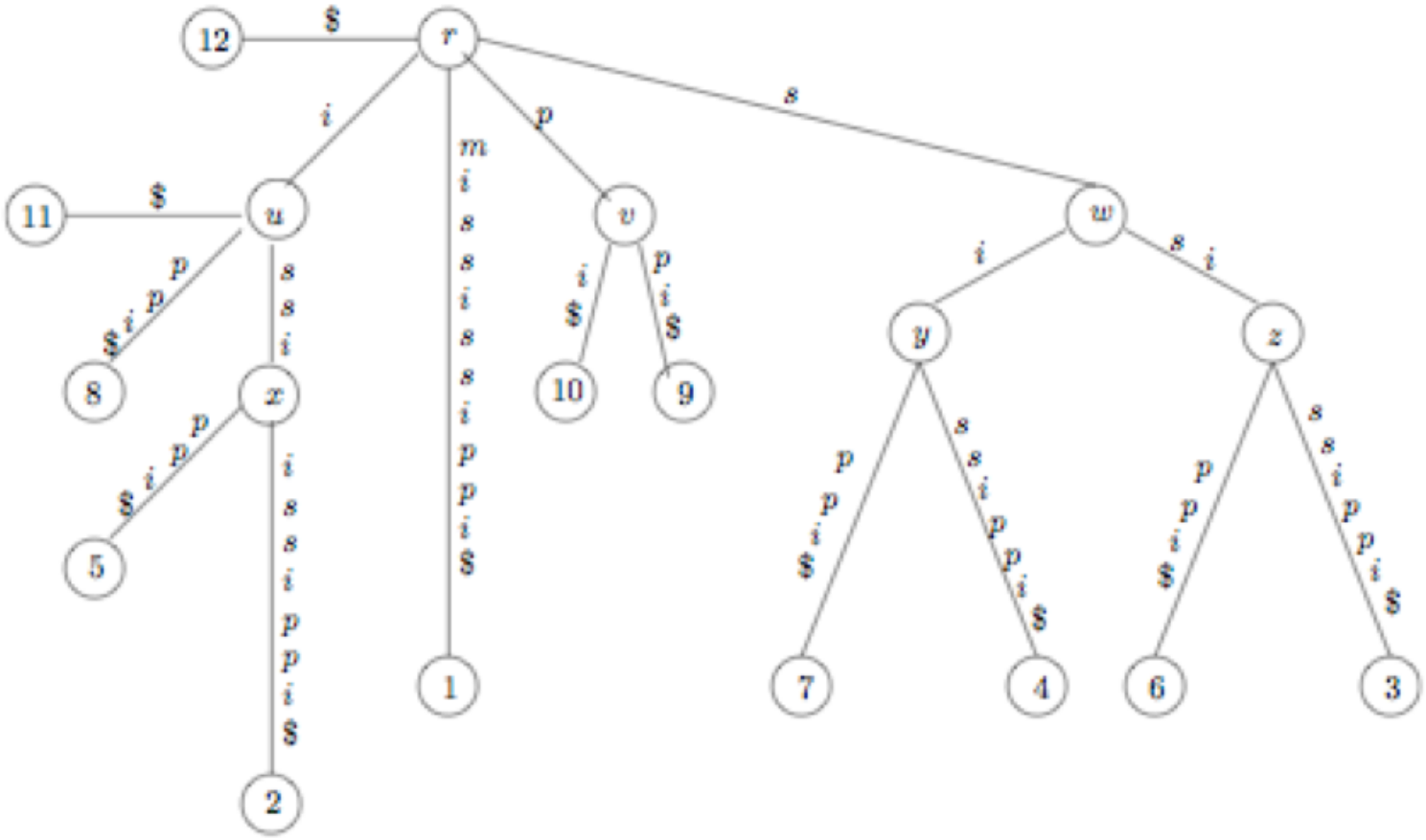
- Suffix arrays require even less space than a suffix tree

- Very simply, it is a sorted list of suffixes
  - Example in the Aluru chapter posted as a resource

Linear time of suffix arrays

- There were three papers in 2002 that solved the old problem of constructing suffix arrays in linear time.

- These were:
  - Ko and Aluru – very interesting, but hard to understand
  - Kim et al. – was based on older parallel suffix tree algorithms
  - Karakkanen and Sanders is the simplest and most elegant.



SA	12	11	8	5	2	1	10	9	7	4	6	3

Lcp	0	1	1	4	0	0	1	0	2	1	3	

FIGURE 1.1: Suffix tree, suffix array and *Lcp* array of the string *mississippi*. The suffix links in the tree are given by $x \to z \to y \to u \to r$, $v \to r$, and $w \to r$.

Try it out (other way)

- Construct the suffix array of the string "BANANA$"
- Construct the LCP array for the suffix array above
- Given the suffix array and LCP array, can you draw a suffix tree?

Algorithm

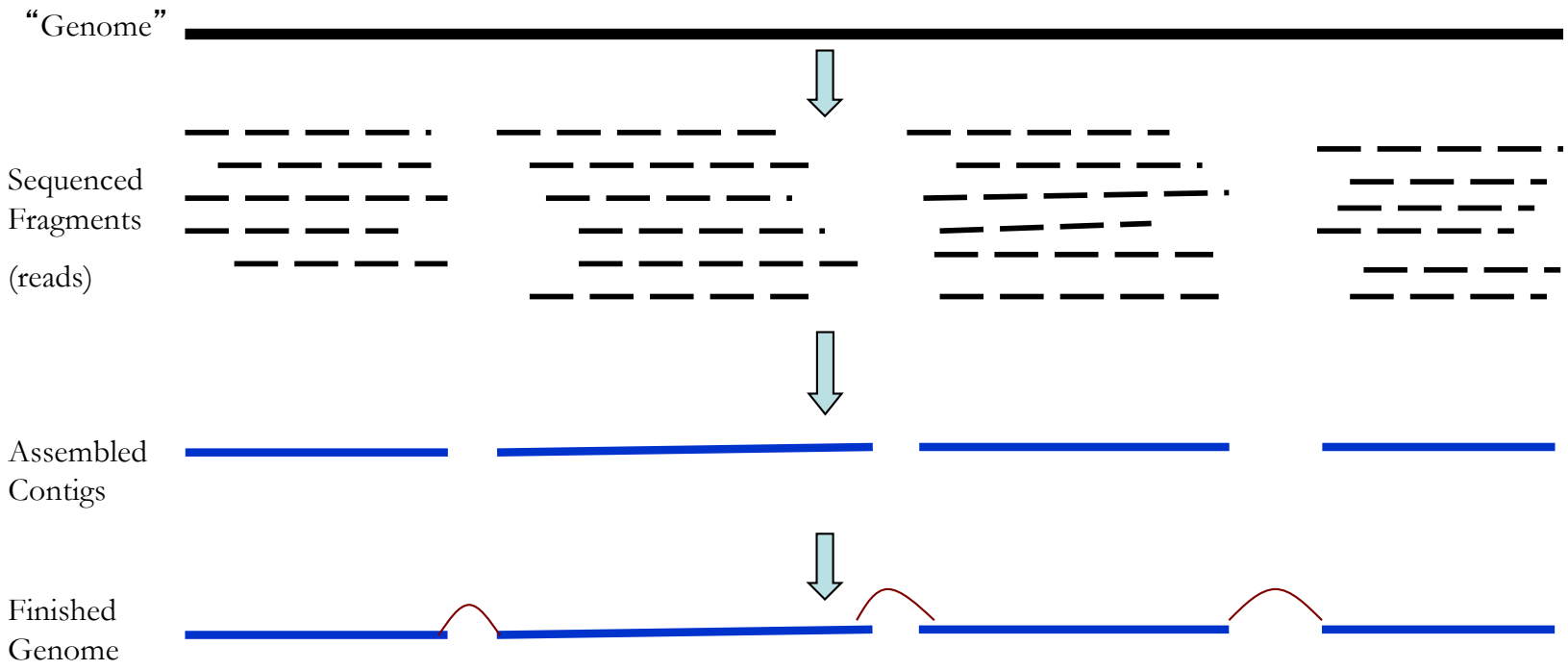
- Recursively sort the 2/3n suffixes with i mod 3 != 0
- Sort the 1/3n suffixes with i mod 3 == 0 using the previous result.
- Merge the two sorted arrays.

Some thoughts

- The sorting can be done using Radix sort and the relative ranks of suffixes used for the ordering.
- The 1/3 and 2/3 split makes the merging much easier; other ½ ½ approaches (e.g. Kim et al.) use this with clever tricks.
- Similar to the odd and even suffix technique of Farach.


PHASE TWO : INTERPRETATION
I THINK I FOUND A CORNER PIECE.
3 BILLION PIECES
GENOME

Sequence Assembly



Greedy solution is bounded

$$G_o = (V,E,o)$$

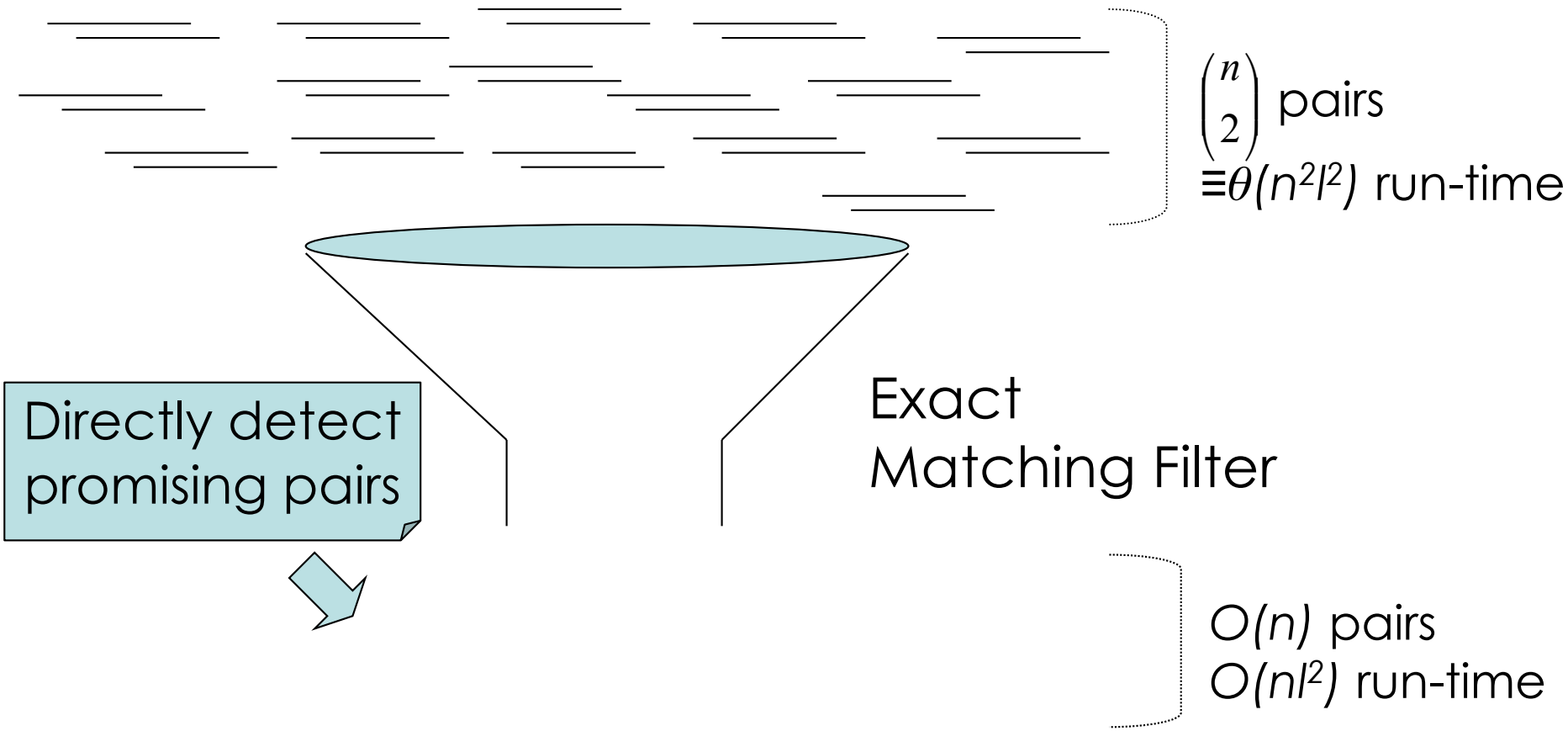
$$\overset{s3}{\text{CCACC}} \xrightarrow{4} \overset{s1}{\text{CACCC}} \xrightarrow{2} \overset{s2}{\text{CCGGGTGC}}$$

$$\text{GREEDY}(S) \leq 2.5\ \text{OPT}(S)$$

$$\text{Runtime } O(\binom{n}{2}\, l^2)$$

SUPERSTRING is MAX SNP-hard, so one of the best approximation algorithms possible.

Typical assembly strategy



“Traditional” Assemblers

- TIGR Assembler
- CAP3/PCAP
- PHRAP
- Celera Assembler
- ARACHNE
- JAZZ
- PHUSION
- ATLAS

Advantages

- Effective heuristics to solve this NPC problem
- Brute-force parallelization is easy to implement

Limitations

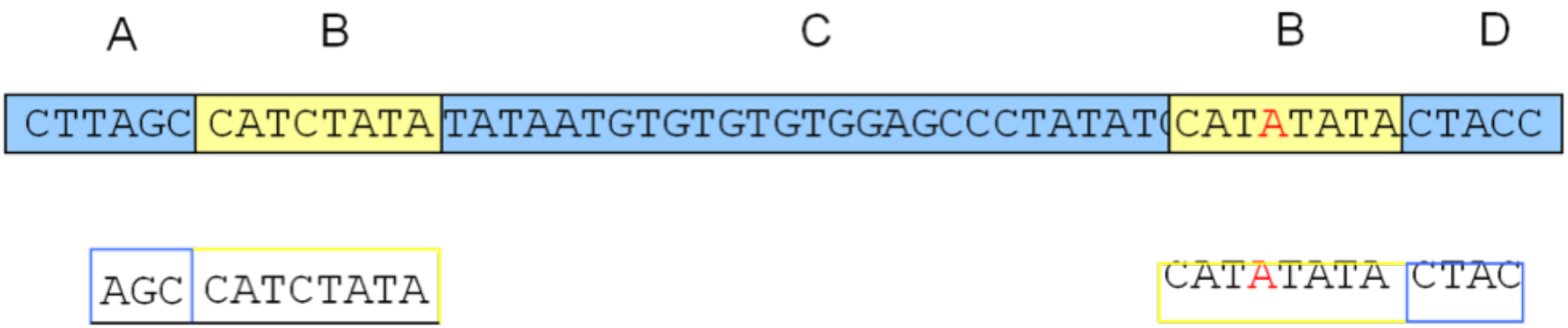
- $\theta(n^2)$ space required in the worst case
- Limited scaling as a result of using disk

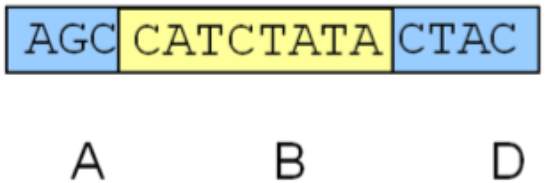
A Look at the maize genome

“Repeats”

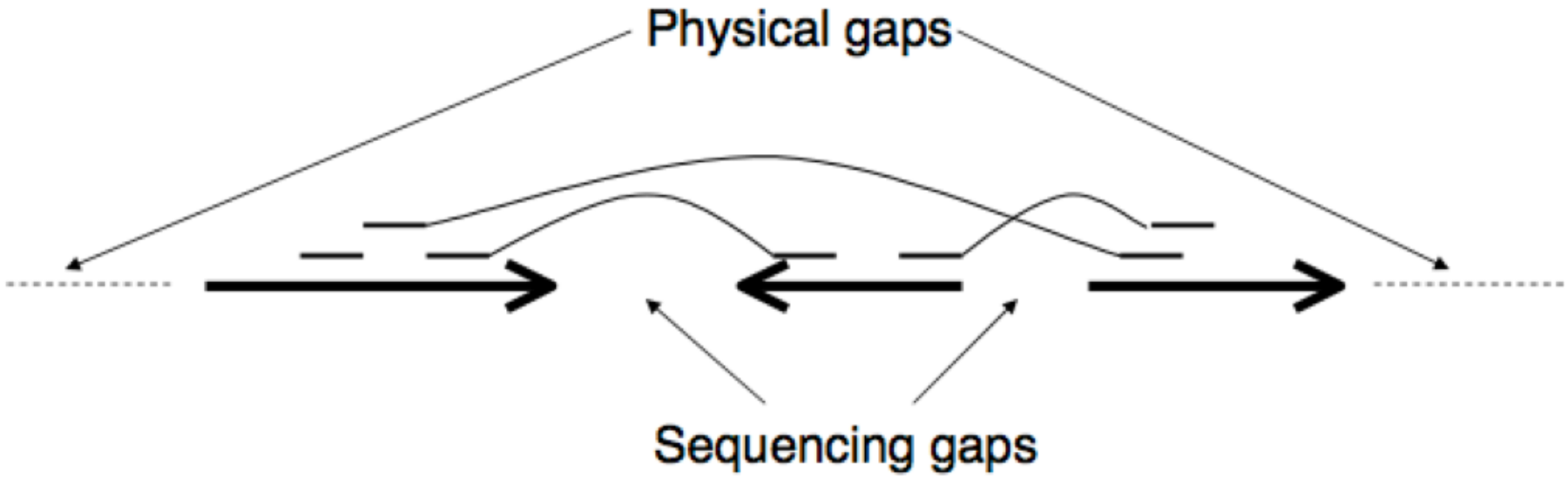
“Gene islands”

Problems due to repeats





Types of sequencing gaps



sequencing gap - we know the order and orientation of the contigs and have at least one clone spanning the gap

physical gap - no information known about the adjacent contigs, nor about the DNA spanning the gap

Slide from Mihai Pop and Michael Schatz

Modern assembler: de Bruijn graphs

- G = (V, E) where V is the set of all length k subfragments and E are directed edges if nodes overlap by k-1 characters.

- Relevant papers:
  - De Bruijn, 1946; Idury and Waterman, 1995; Pevzner, Tang, Waterman, 2001

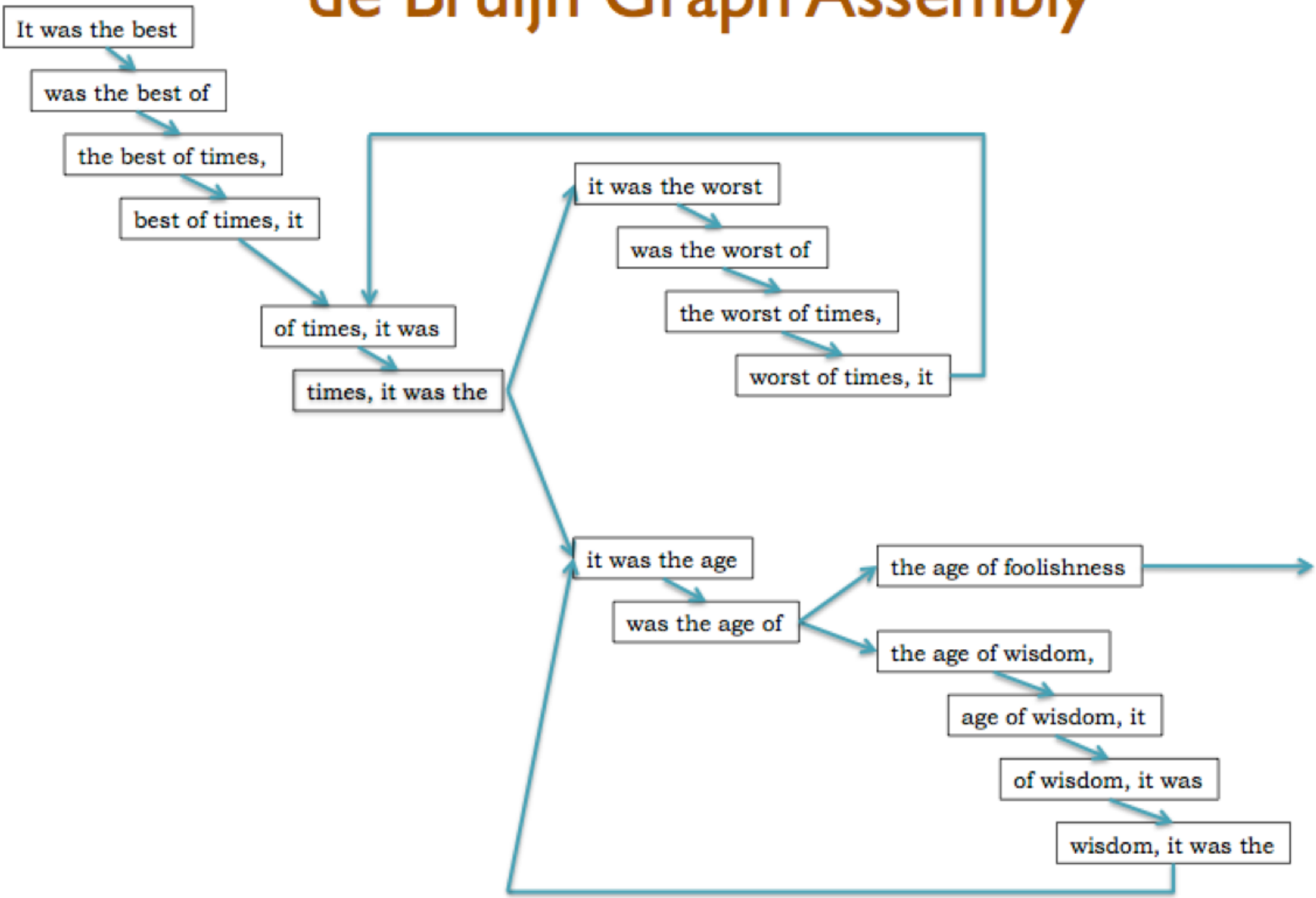
- Good news: the correct assembly exists as a path through G
- Bad news: there are many such paths!

Try it out!

- Consider the text:
  - **It was the best of times it was the worst of times it was the age of wisdom it was the age of foolishness**

- Nodes in the graph are overlapping phrases of length 4, aka “It was the best” and “was the best of”
- Draw an edge between nodes if the last three words of one node match the first three of another.

de Bruijn Graph Assembly



Try it out! (part 2)

- Consider the text:
  - **It was the best of times it was the worst of times it was the age of wisdom it was the age of foolishness**

- How could you construct an “assembly” based on this graph? Are there multiple answers?

- How many possible answers are correct

Compressed de Bruijn

- Non-branching paths replaced by single nodes
- A Eulerian/Chinese postman traversal can reconstruct the text
- More importantly, different sequences may have the same string graph constructed as previously discussed.

Implementations

- There are multiple assemblers:
  - ALLPATHS
  - Abyss
  - Velvet
  - SOAP-denovo
  - SPADEs

- Michael Schatz has a map-reduce formulation, we are interested in grid-based tools.

EULER - A New Approach to Fragment Assembly

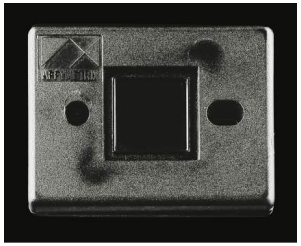
- Traditional “overlap-layout-consensus” technique has a high rate of mis-assembly
- EULER uses the Eulerian Path approach borrowed from “sequencing by hybridization” (SBH)
- Fragment assembly without repeat masking can be done in linear time with greater accuracy

Sequencing by Hybridization (SBH): History

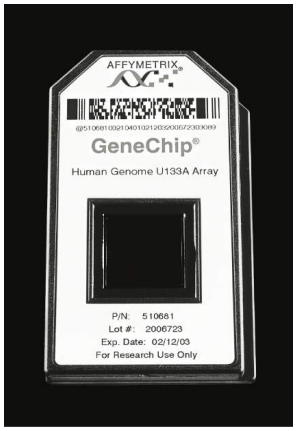
- 1988: SBH suggested as an an alternative sequencing method. Nobody believed it will ever work
- 1991: Light directed polymer synthesis developed by Steve Fodor and colleagues.
- 1994: Affymetrix develops first 64-kb DNA microarray

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

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



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



How SBH Works

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

How SBH Works (cont' d)

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

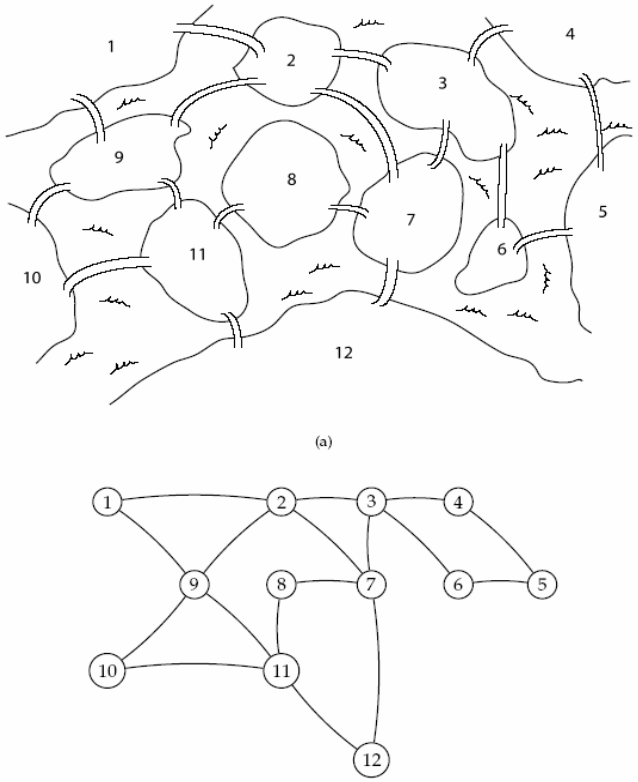
- Apply the combinatorial algorithm (previous) to reconstruct the sequence of the target DNA fragment from the *l* – mer composition.

Some Difficulties with SBH

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

Eulerian Cycle Problem

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



More complicated Königsberg

Euler Theorem

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

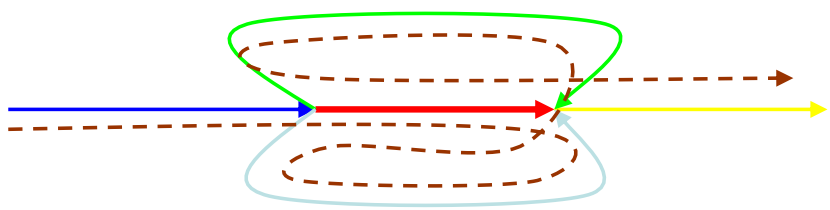
$$in(v)=out(v)$$

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

Approaches to Fragment Assembly (cont' d)

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

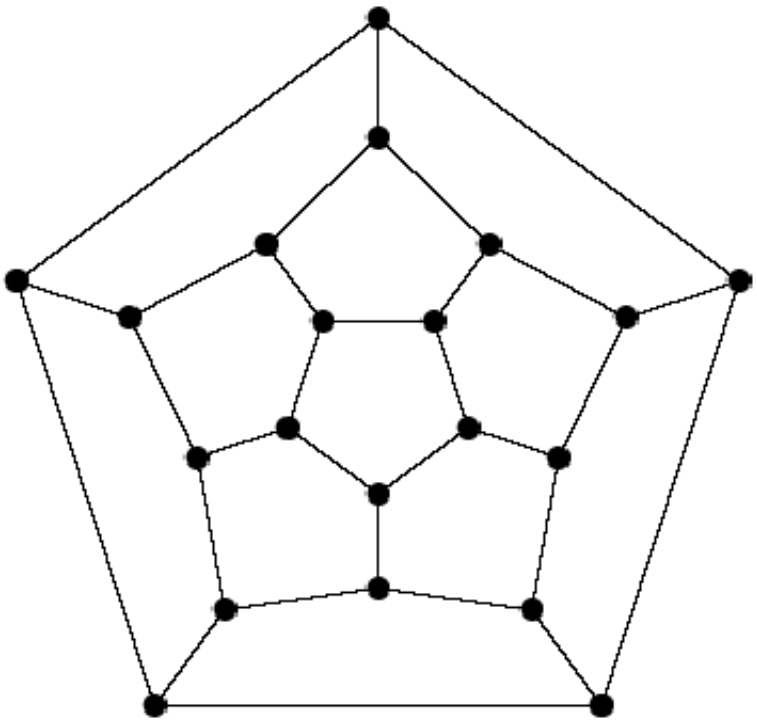
Eulerian path problem



Linear time algorithms are known

Hamiltonian Cycle Problem

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



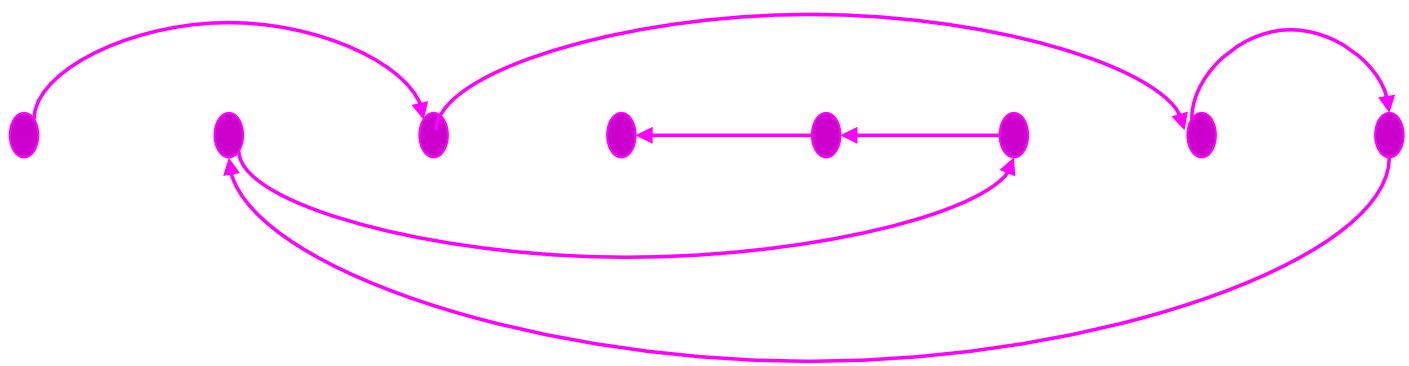
Game invented by Sir William Hamilton in 1857

SBH: Hamiltonian Path Approach

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

H

ATG AGG TGC TCC GTC GGT GCA CAG



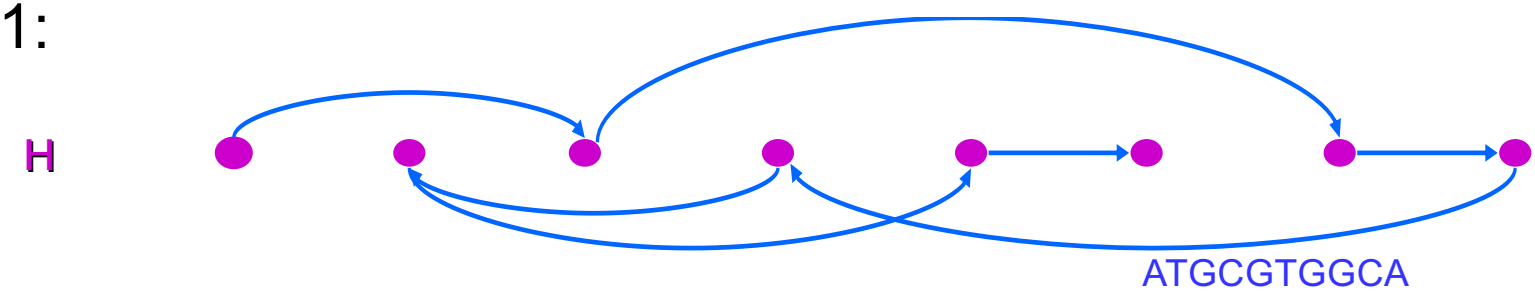
ATG C A G G T C C

Path visited every VERTEX once

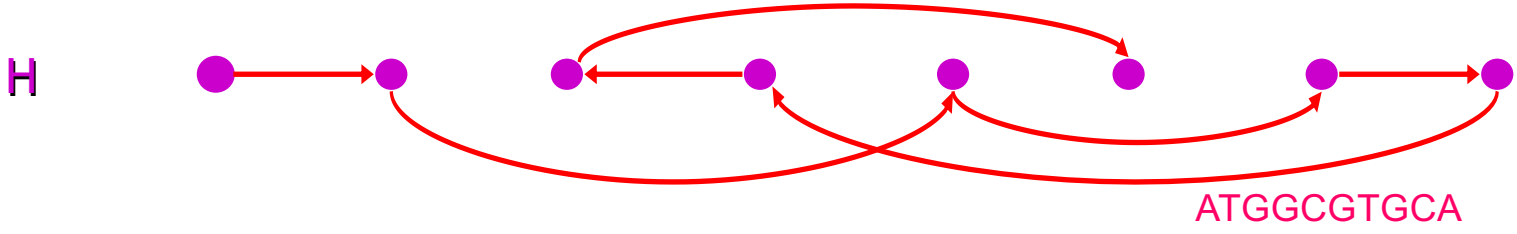
SBH: Hamiltonian Path Approach

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

Path 1:



Path 2:



Hybridization on DNA Array

Universal DNA Array

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

DNA target TATCCGTTT (complement of ATAGGCAAA)

hybridizes to the array of all 4-mers:

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

l-mer composition

Def: Given string s, the ***Spectrum (s, l)*** is *unordered multiset* of all possible $(n - l + 1)$ *l*-mers in a string *s* of length *n*

- The order of individual elements in *Spectrum (s, l)* does not matter
- For *s* = TATGGTGC all of the following are equivalent representations of

  Spectrum (s, 3):

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

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

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

The SBH Problem

- Goal: Reconstruct a string from its *l*-mer composition

- Input: A multiset *S*, representing all *l*-mers from an (unknown) string *s*

- Output: String *s* such that *Spectrum (s,l)* = *S*

Different sequences – the same spectrum!

- Different sequences may have the same spectrum:

  Spectrum(GTATCT,2)=

  Spectrum(GTCTAT,2)=

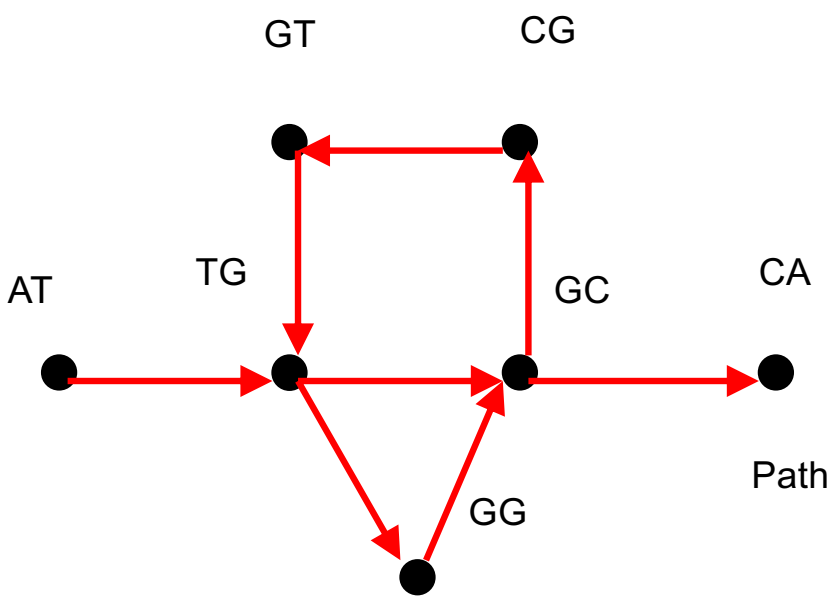
  {AT, CT, GT, TA, TC}

SBH: Eulerian Path Approach

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

Vertices correspond to (*l* – 1) – mers : { AT, TG, GC, GG, GT, CA, CG }

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



Path visited every EDGE once

de Bruijn Graph Assembly