# Identifiability of a string from its substrings

Esko Ukkonen
University of Helsinki
IWOCA 2015 Verona 5-7 October

# Identifiability of a string

- Given a sample of substrings (’reads’) of unknown target string T, reconstruct T

**cctcgagtttaa**
**tacttaactcgag**
**cgggcagtacttaa**
**aagtactgcccgcg**
**gcccgcggcttcaacggat**
**cccgcggcttcaacggatctgtg**
**cccgacacagat**
**tgtgtcgggagtcg**

# Identifiability of a string

- Given a sample of substrings (’reads’) of unknown target string T, reconstruct T

**cctcgagtttaa**
**tacttaactcgag**
**cgggcagtacttaa**
**aagtactgcccgcg**
**gcccgcggcttcaacggat**
**cccgcggcttcaacggatctgtg**
**cccgacacagat**
**tgtgtcgggagtcg**

# Identifiability of a string

- Given a sample of substrings (’reads’) of unknown target string T, reconstruct T

```
cctcgagtttaa
tacttaactcgag
cgggcagtacttaa
aagtactgcccgcg
gcccgcggcttcaacggat
cccgcggcttcaacggatctgtg
cccgacacagat
tgtgtcgggagtcg
```

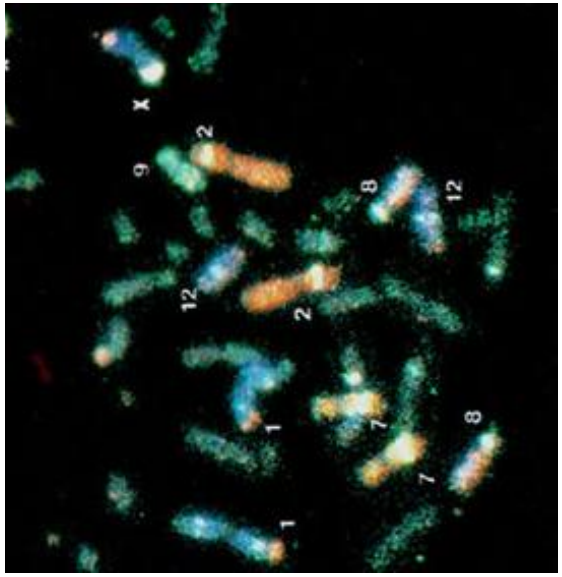
**cctcgagttaagtactgcccgcggcttcaacggatctgtcgggagtcg**

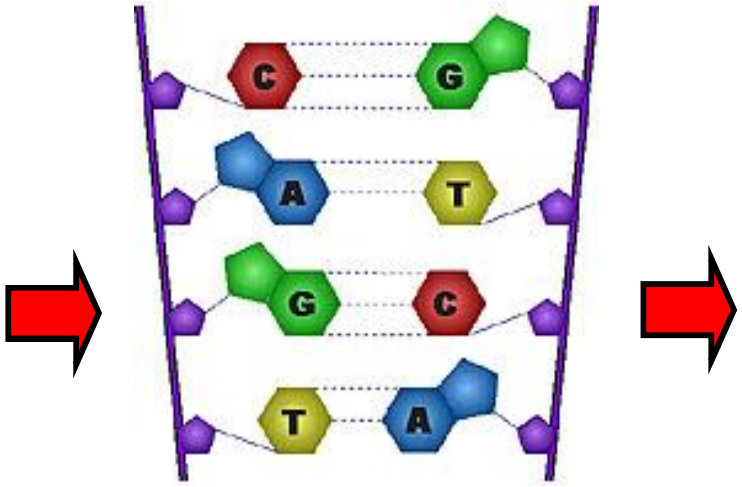
# Example repeat

Sites in HG: 28395980, 28401554r  Lenght: 2559

ttagggtacatgtgcacaacgtgcaggtttgttacatatgtatacacgtgccatgatggtgtgctgcacccattaactcgtcatttagcgttaggtatatctccgaatgctatccctccccccctcccc
ccaccccacaacagtccccggtgtgtgatgttccccttcctgtgtccatgtgttctcattgttcaattcccacctatgagtgagaacatgcggtgtttggttttttgtccttgcgaaagtttgctgaga
atgatggtttccagcttcatccatatccctacaaaggacatgaactcatcatttttttatggctgcatagtattccatggtgtatatgtgccacattttcttaacccagtctacccttgttggacatctg
ggttggttccaagtctttgctattgtgaatagtgccgcaataaacatacgtgtgcatgtgtctttatagcagcatgatttataatcctttgggtatatacccagtaatgggatggctgggtcaaatg
gtatttctagttctagatccctgaggaatcaccacactgacttccacaatggttgaactagtttacagtcccagcaacagttcctatttctccacatcctctccagcacctgttgtttcctgacttttta
atgatcgccattctaactggtgtgagatggtatctcattgtggttttgatttgcatttctctgatggccagtgatgatgagcatttttttcatgtgttttttggctgcataaatgtcttctttttgagaagtg
tctgttcatatccttcgcccacttttgatggggttgtttgttttttttcttgtaaatttgttggagttcattgtagattctgggtattagccctttgtcagatgagtaggttgcaaaaattttctcccattct
gtaggttgcctgttcactctgatggtggtttcttctgctgtgcagaagctctttagtttaattagatcccatttgtcaattttggcttttgttgccatagcttttggtgttttagacatgaagtccttgccc
atgcctatgtcctgaatggtattgcctaggttttcttctagggtttttatggttttaggtctaacatgtaagtctttaatccatcttgaattaattataaggtgtatattataaggtgtaattataaggt
gtataattatatattaattataaggtgtatattaattataaggtgtaaggaagggatccagtttcagctttctacatatggctagccagttttccctgcaccatttattaaatagggaatcctttccc
cattgcttgtttttgtcaggtttgtcaaagatcagatagttgtagatatgcggcattatttctgagggctctgttctgttccattggtctatatctctgttttggtaccagtaccatgctgttttggttac
tgtagccttgtagtatagtttgaagtcaggtagcgtgatggttccagctttgttcttttggcttaggattgacttggcaatgtgggctctttttggttccatatgaactttaaagtagtttttttccaatt
ctgtgaagaaattcattggtagcttgatggggatggcattgaatctataaattaccctgggcagtatggccattttcacaatattgaatcttcctacccatgagcgtgtactgttcttccatttgttt
gtatcctcttttatttcattgagcagtggtttgtagttctccttgaagaggtccttcacatcccttgtaagttggattcctaggtattttattctctttgaagcaattgtgaatgggagttcactcatgat
ttgactctctgtttgtctgttattggtgtataagaatgcttgtgatttttgcacattgattttgtatcctgagactttgctgaagttgcttatcagcttaaggagattttgggctgagacgatggggttt
tctagatatacaatcatgtcatctgcaaacagggacaatttgacttcctcttttcctaattgaatacccgttatttccctctcctgcctgattgccctggccagaacttccaacactatgttgaatagg
agtggtgagagagggcatccctgtcttgtgccagttttcaaagggaatgcttccagtttttgtccattcagtatgatattggctgtgggtttgtcatagatagctcttattattttgagatacatccc
atcaatacctaatttattgagagtttttagcatgaagagttcttgaattttgtcaaaggccttttctgcatcttttgagataatcatgtggtttctgtctttggttctgtttatatgctggagtacgttta
ttgattttcgtatgttgaaccagccttgcatcccagggatgaagcccacttgatcatggtggataagctttttgatgtgctgctggattcggtttgccagtattttattgaggatttctgcatcgatgt
tcatcaaggatattggtctaaaattctctttttttgttgtgtctctgtcaggctttggtatcaggatgatgctggcctcataaaatgagttagg

# DNA sequencing and genome projects



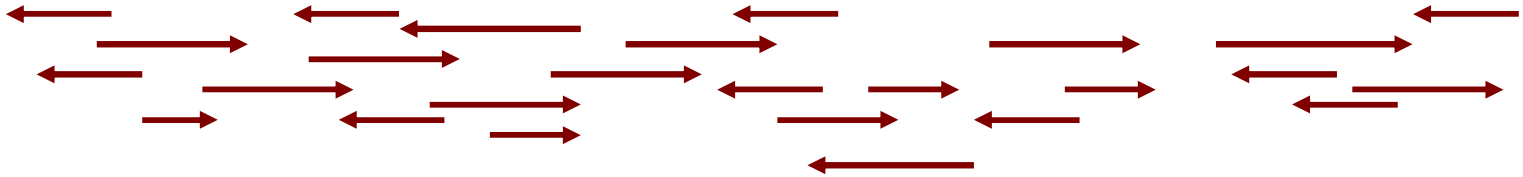



cgccgagtgacagaga
gctaatcaggctgtgtt
tcaggatgcgtaccgag
tgggagacagcagcac
accag...

# Shotgun sequencing (1980 ->)

Original DNA:

Reads:



reads: sequence fragments (substrings) from random locations and with random direction & with reading errors & gaps in coverage

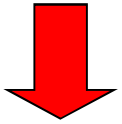
# DNA puzzle

**cctcgagttaagtactgcccgcggcttcaacggatctgtcgggagtcg**


**cctcgagtttaa**
**tacttaactcgag**
**cgggcagtacttaa**
**aagtactgcccgcg**
**gcccgcggcttcaacggat**
**cccgcggcttcaacggatctgtg**
**cccgacacagat**
**tgtgtcgggagtcg**

```
cctcgagtttaa
tacttaactcgag
cgggcagtacttaa
aagtactgcccgcg
gcccgcggcttcaacggat
cccgcggcttcaacggatctgtg
cccgacacagat
tgtgtcgggagtcg
```

```
cctcgagtttaa
 ctcgagt-taagta
        t-taagtactgcccg
           aagtactgcccgcg
                  gcccgcggcttcaacggat
                   cccgcggcttcaacggatctgtg
                                    atctgtgtcggg
                                       tgtgtcgggagtcg
```



```
cctcgagt-taagtactgcccgcggcttcaacggatctgtgtcgggagtcg
```

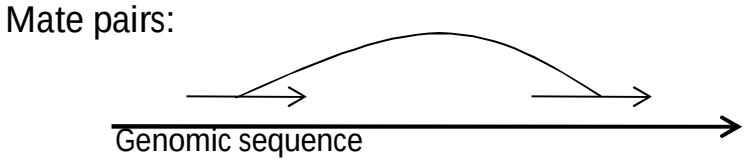
# Glanville fritillary butterfly (*Melitaea cinxia*) genome

V. Ahola et al.: The Glanville fritillary genome retains an ancient karyotype and reveals selective chromosomal fusions in Lepidoptera, *Nature Communications* (Sept 2014)

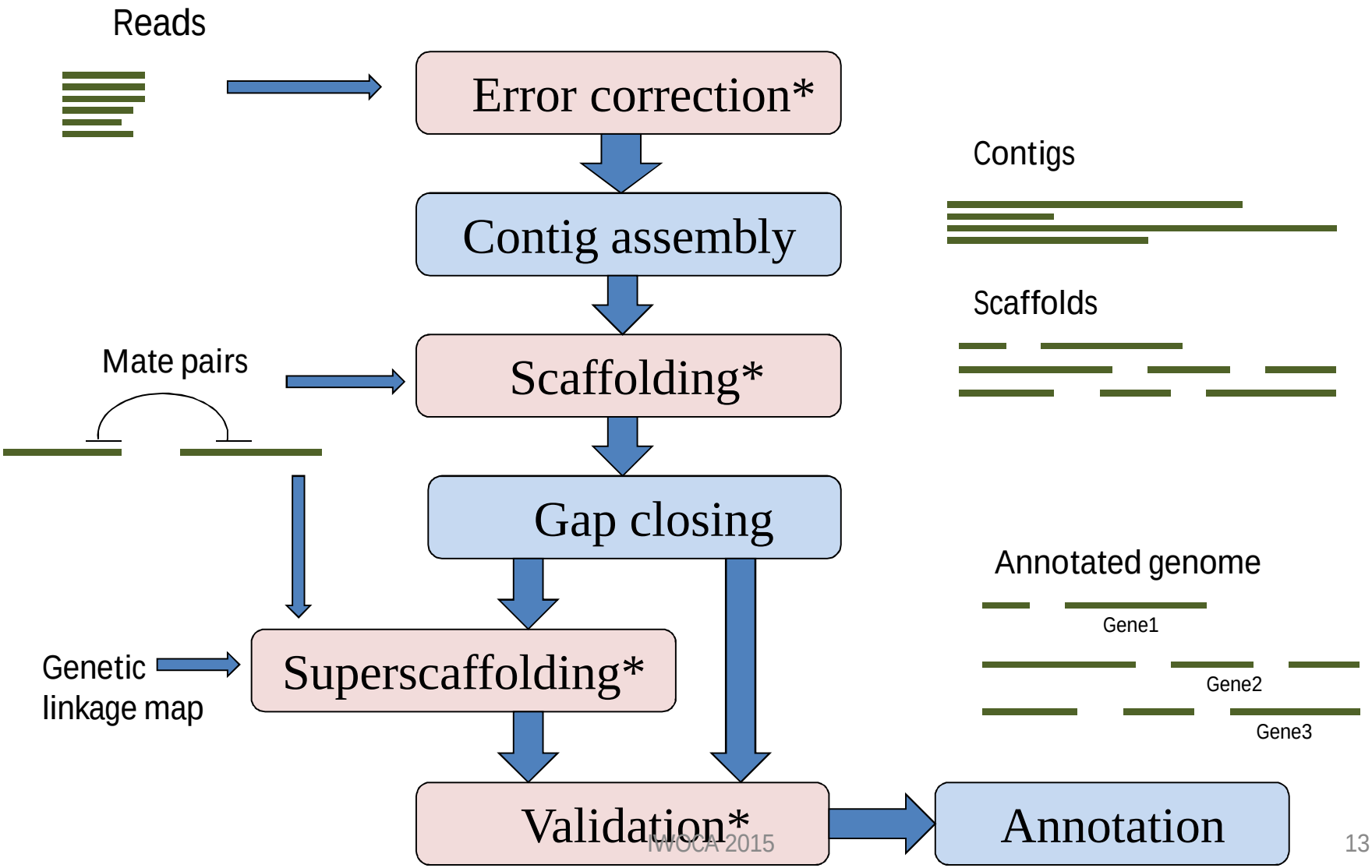
# Overview of the data

| | 454 | SOLiD | Illumina | PacBio |
|---|---|---|---|---|
| Reads | 12 million | 210 million | 349 million | 2.7 million |
| Read length | 400-800 bp | 50 bp | 75-150 bp | Avg 2700 bp<br>Max 23 kbp |
| Errors | Indels | Mismatches | Mismatches | Indels |
| Paired end | - | - | 460 bp,710 bp | - |
| Mate pairs | 7 kbp, 17 kbp | 2-5 kbp | 1-3 kbp | - |
| Other | Mostly single end | Color coded | - | High error rate |

Total size of data: 50 Gbp (91 Gbp before filtering)
Estimated length of the genome: 350 Mbp  Av. coverage: 140 (260)



# Genome assembly workflow



# Statistics of the draft genome

| | Number of contigs/scaffolds | N50 | Total length |
|---|---|---|---|
| Contigs | 49,851 | 13,489 | 360,975,554 |
| Scaffolds | 8,262 | 119,328 | 389,896,394 |
| Superscaffolds | 1,453 | 330,752 | 282,503,348 |

N50:
The total length of contigs longer than the N50 statistic is at least half the length of the whole contig collection.

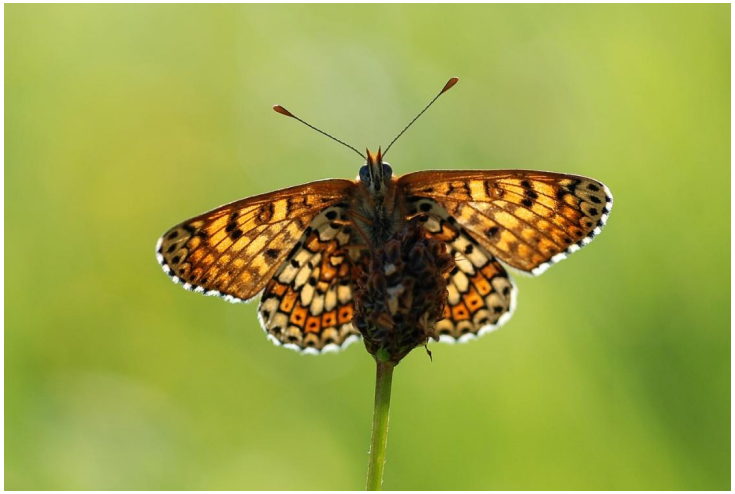
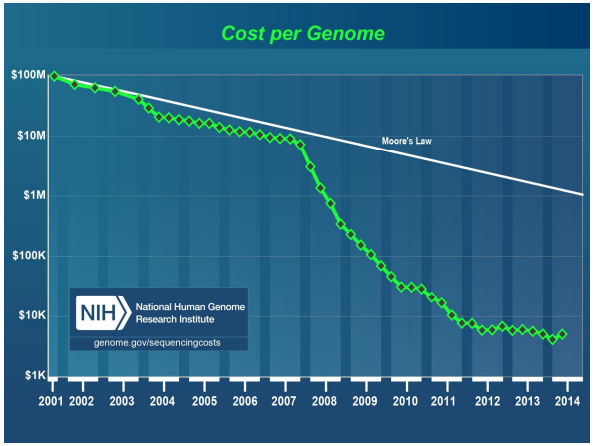
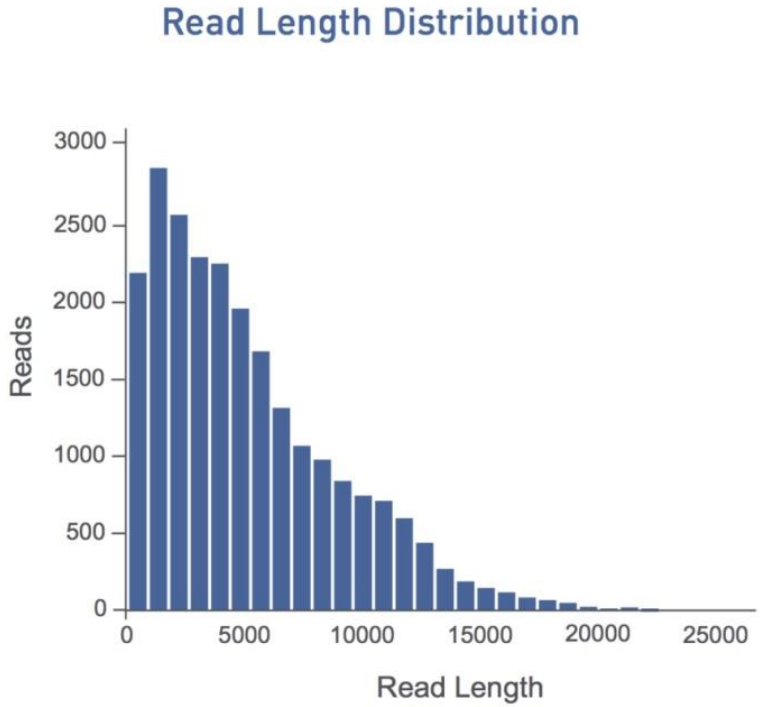


Photo: Gilles San Martin

# New sequencing – longer reads

- Read length up to tens of kb
- Uniform noise (but quite high level)
- Uniform sampling
- High coverace (may be hundreds), 500 Gbp per week per machine
- Cost goes down strongly




Typical PacBio C2XI raw read length distribution. From http://pacificbiosciences.com/brochure (February 2013)

# Complications in fragment assembly

- Main difficulty: Long repeated regions
- Sequencing errors
- Unknown orientation
- Incomplete coverage
- . . .

# The view of this talk: identifiability

- Given (exact) reads F from some unknown target string, is the solution unique?
- Not in general, but perhaps under some natural conditions?
- What happens when the length of the overlap between reads, that are adjcent in the target, grows?
- Length of required overlap vs the length of longest repeated substring of the target

# Approaches to read assembly

i. Build overlap graph for the reads & report Hamiltonian path in the overlap graph

ii. Build de Bruijn graph from the k-mers of the reads & report Eulerian path in the de Bruijn graph (Pevzner et al 2001)

iii. Weighted string graph & maximum flow (Myers 2005)

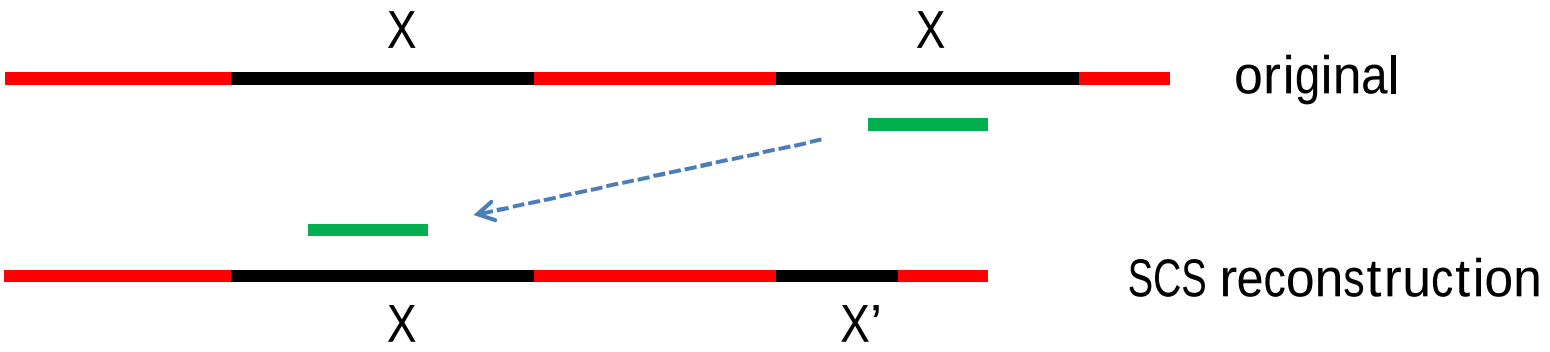
iv. Shortest Common Superstring

v. Probabilistic modeling: Given the reads F, what is the most likely target string that could generate F (Medvedev & Brudno 2009)

vi. Interval graph: the overlap graph should have the structure of an interval graph as the reads are intervals from the same target (Peltola et al 1983)

# SCS is not a good model for fragment assembly

- SCS may collapse repeated substrings:




- BUT: the Greedy algorithm for SCS works fine for fragment assembly if the overlaps between adjacent fragments are longer than any repeat of the target, i.e., overlaps are unique

# The Greedy Algorithm for SCS

Algorithm: Repeatedly merge two maximum overlapping strings into one, until there is only one string left (or there are no non-empty overlaps between the remaining strings)

# The Greedy Algorithm for SCS

Algorithm: Repeatedly merge two maximum overlapping strings into one, until there is only one string left (or there are no non-empty overlaps between the remaining strings)

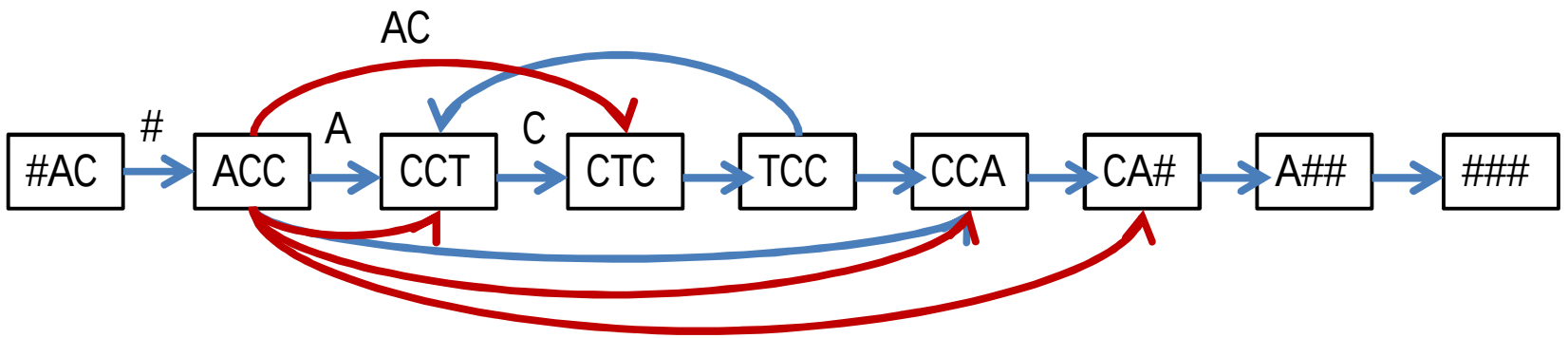
- Tarhio & Ukkonen 1986:
  - compression ratio $r_c(\text{Greedy}) \leq 2$
  - Approximation ratio conjecture: $r_l(\text{Greedy}) \leq 2$
- Blum at al 1991: $r_l(\text{Greedy}) \leq 4$
- Romero et al 2004: $r_l(\text{Greedy}) \leq 1.014$ on average on simulated data
- Bin Ma 2008 (c.f. Plociennik 2009): the average $r_l(\text{Greedy})$ on small perturbations of the instance is 1+o(1), i.e., arbitrarily good smoothed performance ratio
- Kulikov et al 2015: The conjecture $r_l(\text{Greedy}) \leq 2$ true for reads of length 4

# Representing possible assemblies I : overlap graph

- Overlap graph (string graph):
  - vertices = F
  - edges: for reads f, g in F, if f = uv and g = vw, then the overlap graph has an edge (f,g), labeled with u

f = AGTTTTGAA
g = TTGAACTC

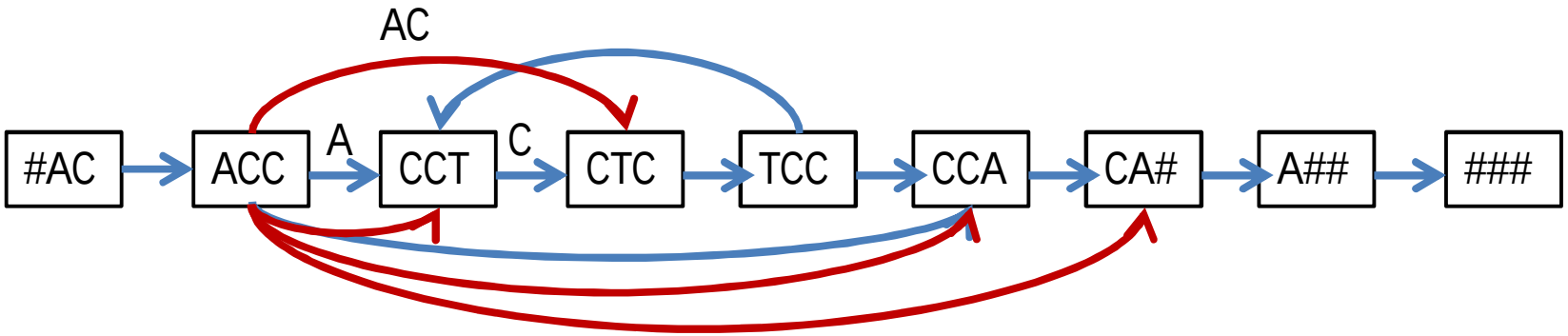
ACC
A##
CA#
CCA
CCT
CTC
TCC
#AC
###




*T* = #ACCTCCA###

# Assembly = Hamiltonian path

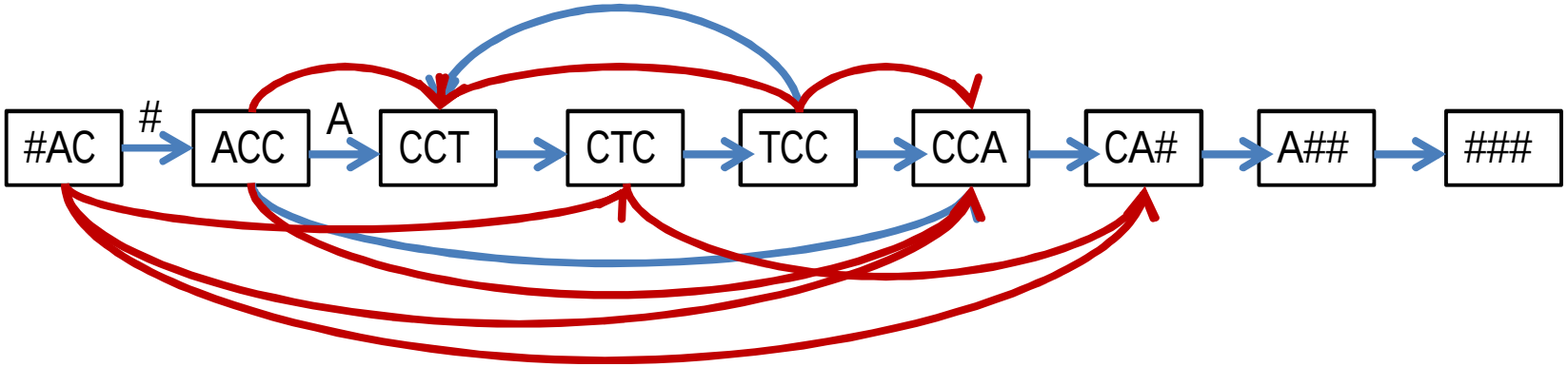
- Hamiltonian path
  - NP-hard
  - Which alternative is the correct one? Unitig = non-branching path.
- Transitive reduction: if (f,g,v), (g,h,u), and (f,h,vu) are edges, remove (f,h,vu) as it it is implied by the other two edges




#ACCTCCA###

# Assembly = Hamiltonian path

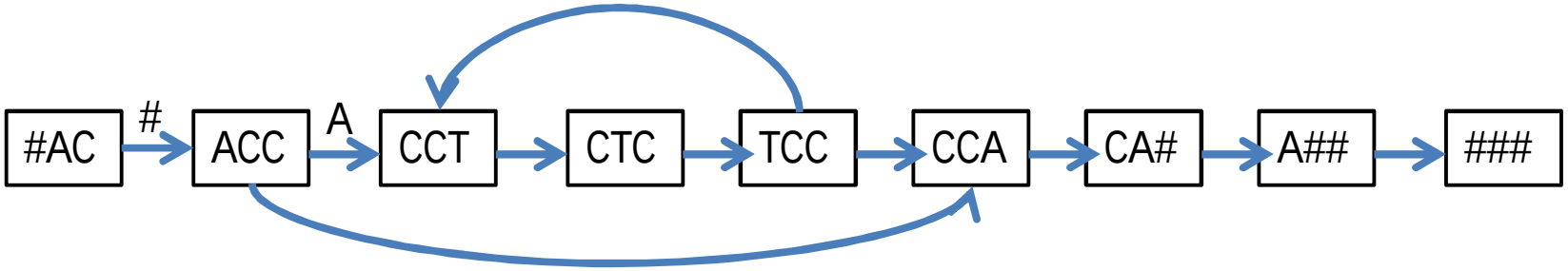
- Hamiltonian path
  - NP-hard
  - Which alternative is the correct one? Unitig = non-branching path.
- Transitive reduction: if (f,g,v), (g,h,u), and (f,h,vu) are edges, remove (f,h,vu) as it it is implied by the other two edges




#ACCTCCA###

# Assembly = Hamiltonian path

- Hamiltonian path
  - NP-hard
  - Which alternative is the correct one? Unitig = non-branching path.
- Transitive reduction: if (f,g,v), (g,h,u), and (f,h,vu) are edges, remove (f,h,vu) as it it is implied by the other two edges



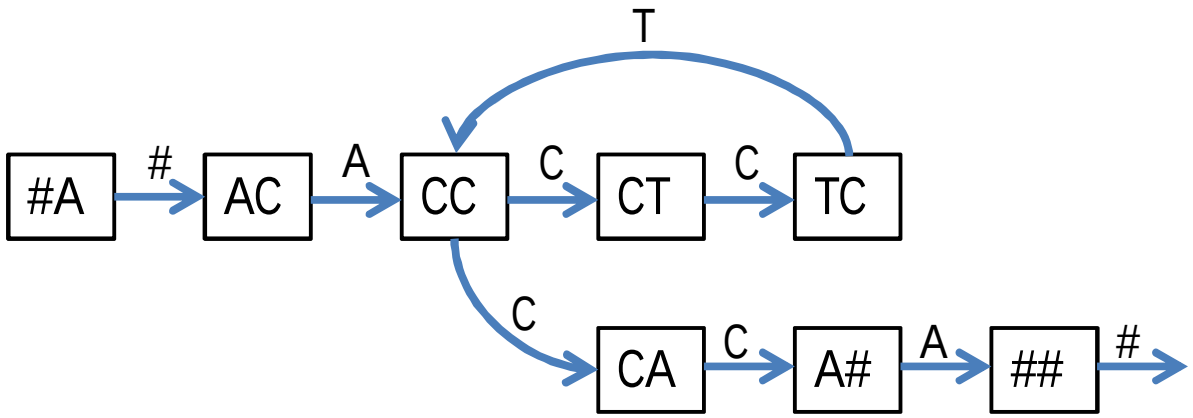

#ACCTCCA###

# Representing possible assemblies II: de Bruijn graph

- De Bruijn graph:
  - all reads of fixed length k (k-mers, k-grams)
  - edges = F
  - vertices <-> overlaps of length k-1: the edge for k-mer f is (u,v), if f = av = ub for some alphabet symbols a, b and strings u, v of length k-1. Edge (u,v) is labeled by a.

f = ACGGT
k = 5

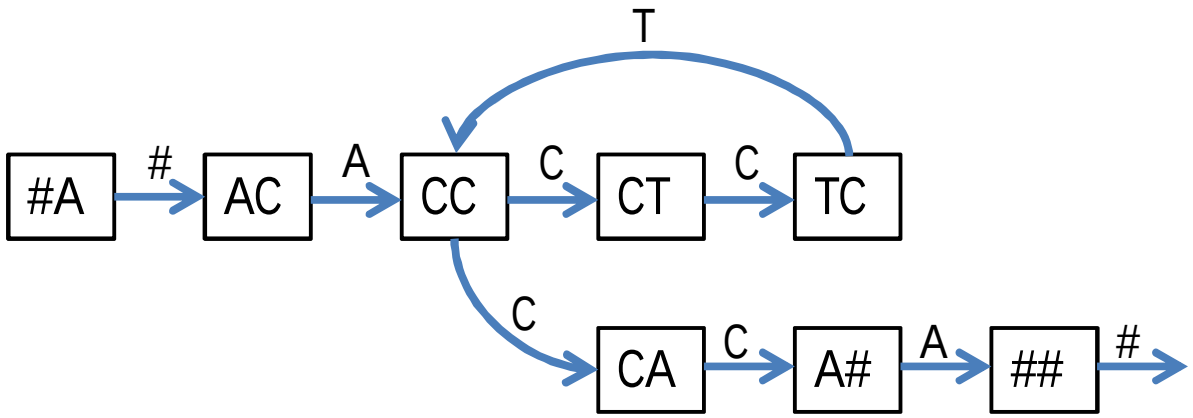
ACC
A##
CA#
CCA
CCT
CTC
TCC
#AC
###




#ACCTCCA###
k = 3

# Assembly = Eulerian path

- Eulerian path visits every edge exactly once
- Eulerian path <-> a string with the same k-mers
- Eulerian path exists <-> indegree = outdegree at every vertex (- start vertex, - end vertex)
- Easy to find (Fleury's alg, Hierholzer's alg)




#ACCTCCA###

k = 3

# k-mer spectrum of target T

- Target string $T = t_1 \ldots t_n$
- The *k-mer spectrum* of T is the multiset

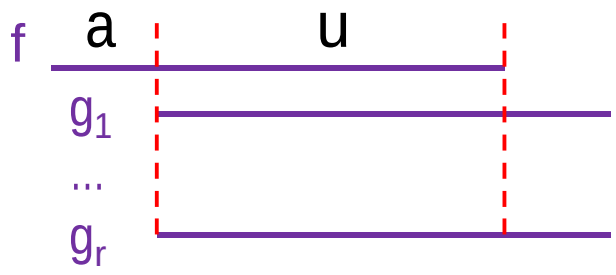
$$F_k(T) = (t_i \ldots t_{i+k-1} \mid i = 1, \ldots, n)$$

*T* = #ACCTCCA###
k = 3

ACC
A##
CA#
CCA
CCT
CTC
TCC
#AC
###

# De Bruijn graph $B_k(T)$ of the k-mer spectrum

- $B_k(T)$ = de Bruijn graph of k-mers of T
- Vertices of $B_k(T)$ can be represented as a subset of k-mers:




- The sink vertex for f is subset $E(f) = \{g_1, \ldots, g_r\}$ of $F_k(T)$
  - Elements of E(f) have common prefix u
  - Each k-mer g belongs to exactly one set E(f)
  - There is an edge from E(f) for each $g_i \in E(f)$ => edges have multiplicites as weights

# Example

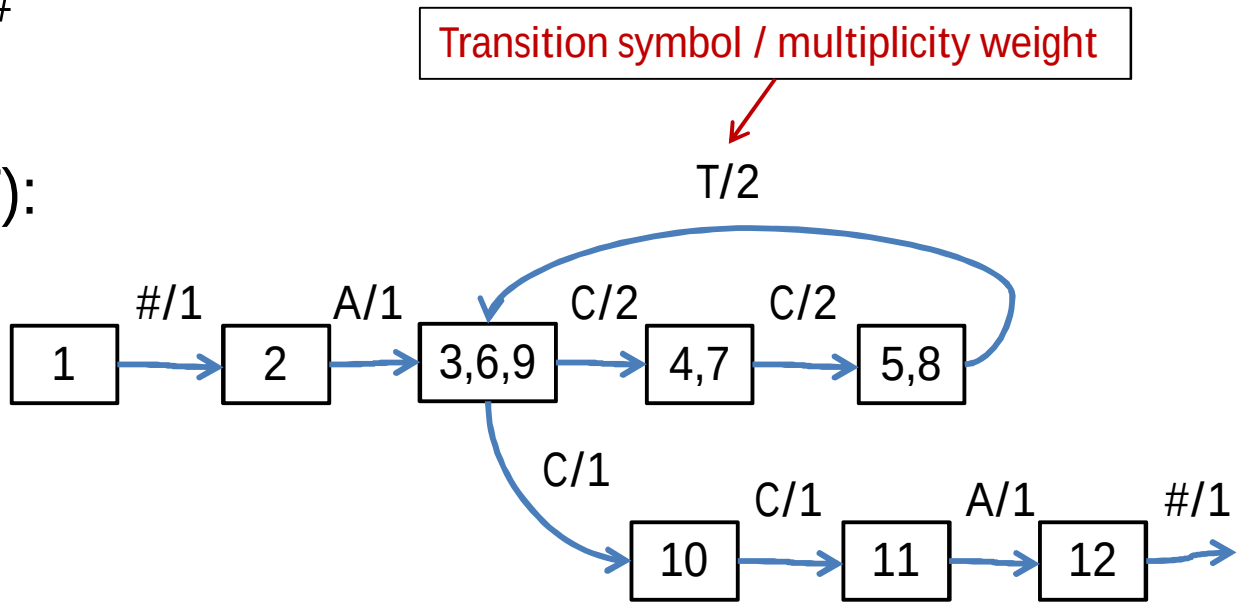
T = #ACCTCCTCCA###

k = 3

$F_3(T)$:

1. #AC
2. ACC
3. CCT
4. CTC
5. TCC
6. CCT
7. CTC
8. TCC
9. CCA
10. CA#
11. A##
12. ###

$B_3(T)$:



# Identifiability of T from k-mer spectrum

- Def: String T is identifiable from k-mer spectrum $F_k(T)$ if T is the only string with that spectrum
- Strings with spectrum $F_k(T)$ < - > Eulerian paths of de Bruijn graph $B_k(T)$
- Thm: T identifiable from k-mer spectrum <-> $B_k(T)$ has only one Eulerian path

# Number of Eulerian paths – BEST Theorem

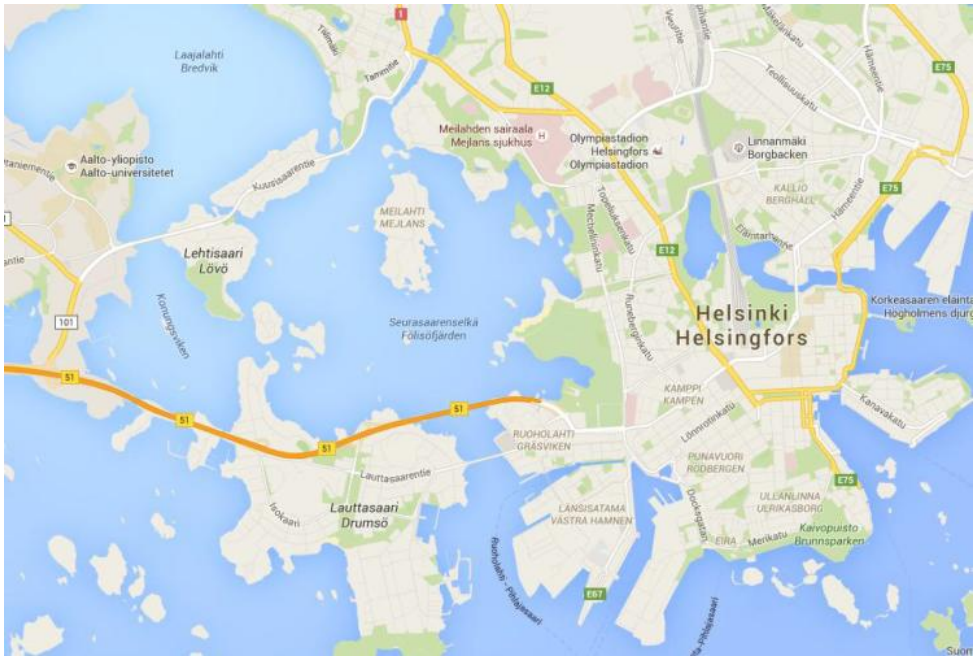
- The number *e(G)* of Eulerian circuits in a connected Eulerian graph *G* is given by

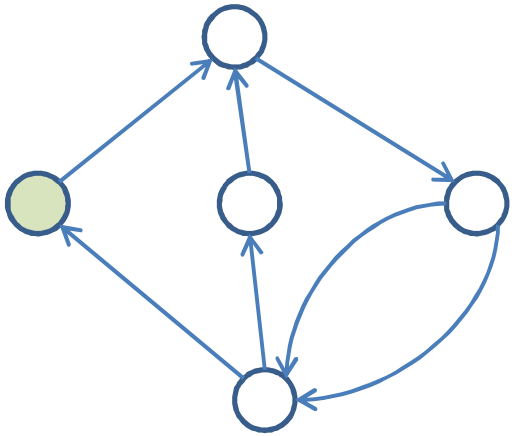
$$e(G) = t(u) \prod_{v \in V} (\deg(v) - 1)!$$

- t(u) = number of directed spanning trees of G directed towards a vertex u
- One path <=> only one spanning tree and all vertices have (in & out)degree ≤ 2

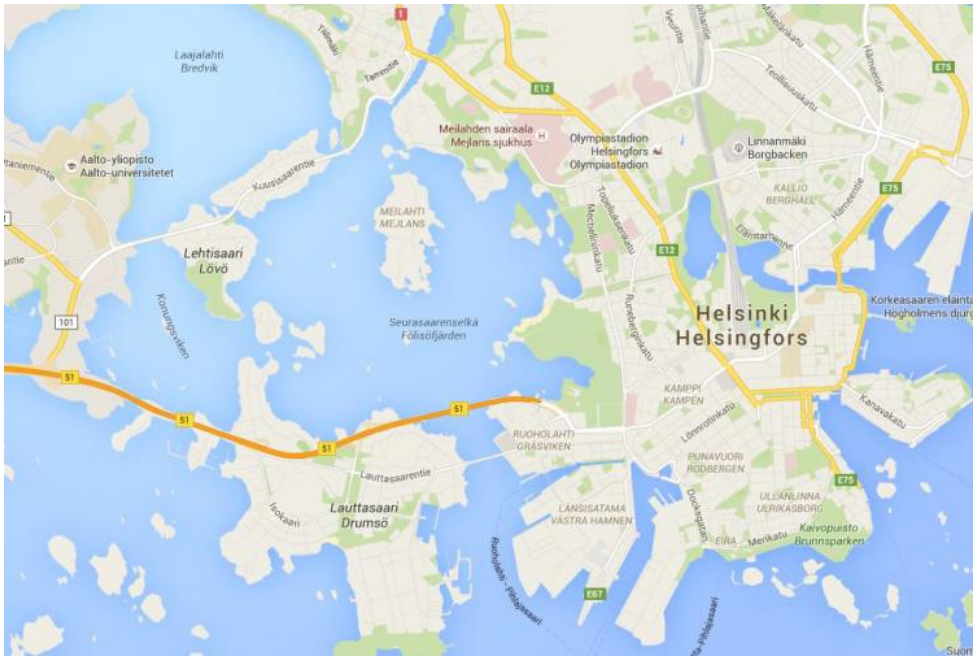
- BEST = de Bruijn, van Aardenne-Ehrenfest (1951), Smith and Tutte (1941)

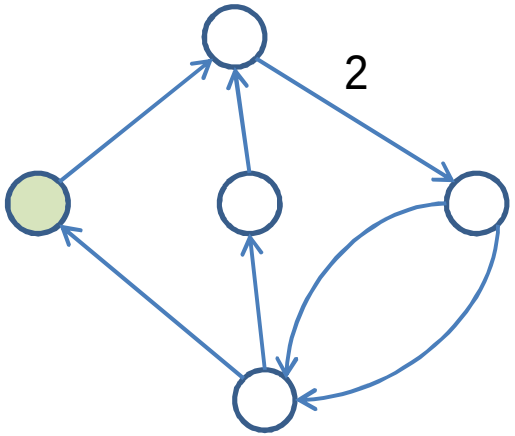
# Helsinki bridges



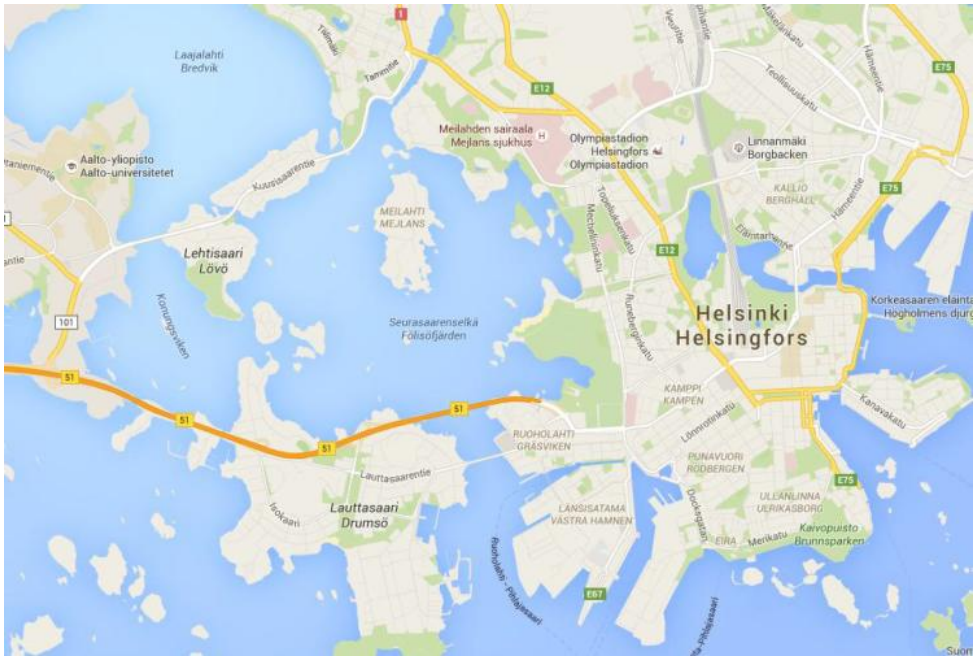



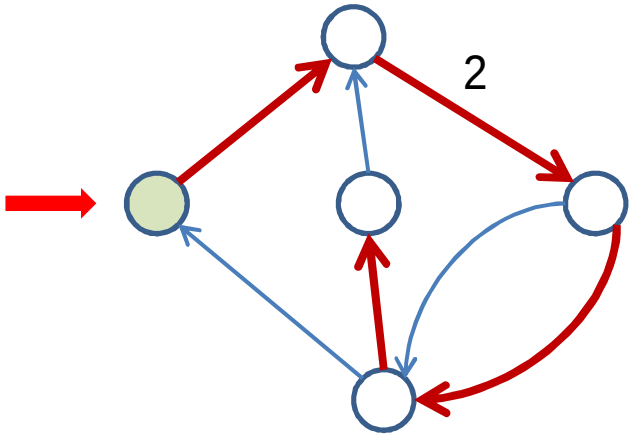
# Helsinki bridges



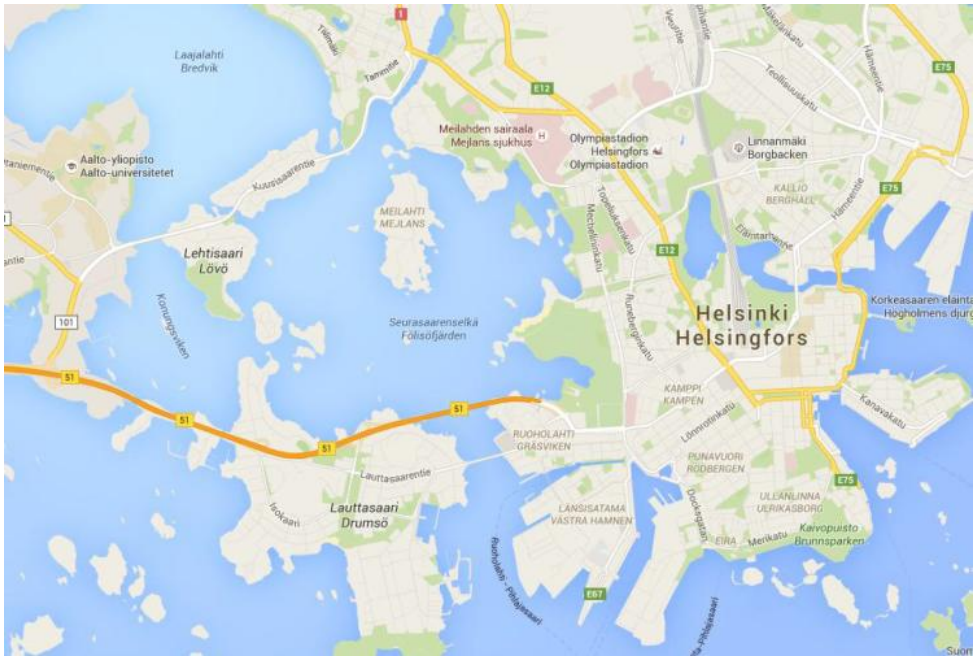



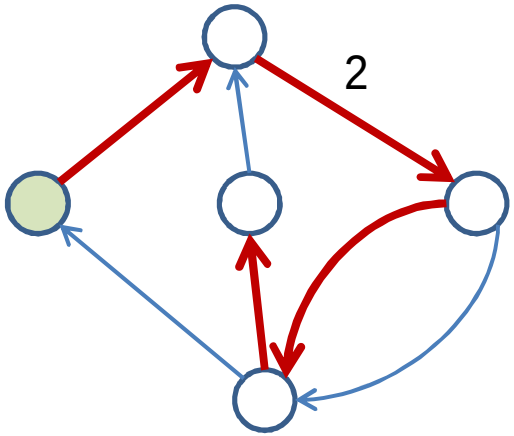
# Helsinki bridges



# Helsinki bridges




2 spanning trees => more than one Eulerian cycle

# Testing $B_k(T)$ for unique path

- Only one directed spanning tree: depth-first search
- Degree (sum of weights) of vertices is ≤ 2 but the last loop in each branch of the spanning tree can have multiplicity (weight) higher than 2: Test that the spanning tree reaches each loop along an edge with weight 1

# Unidentifiable T

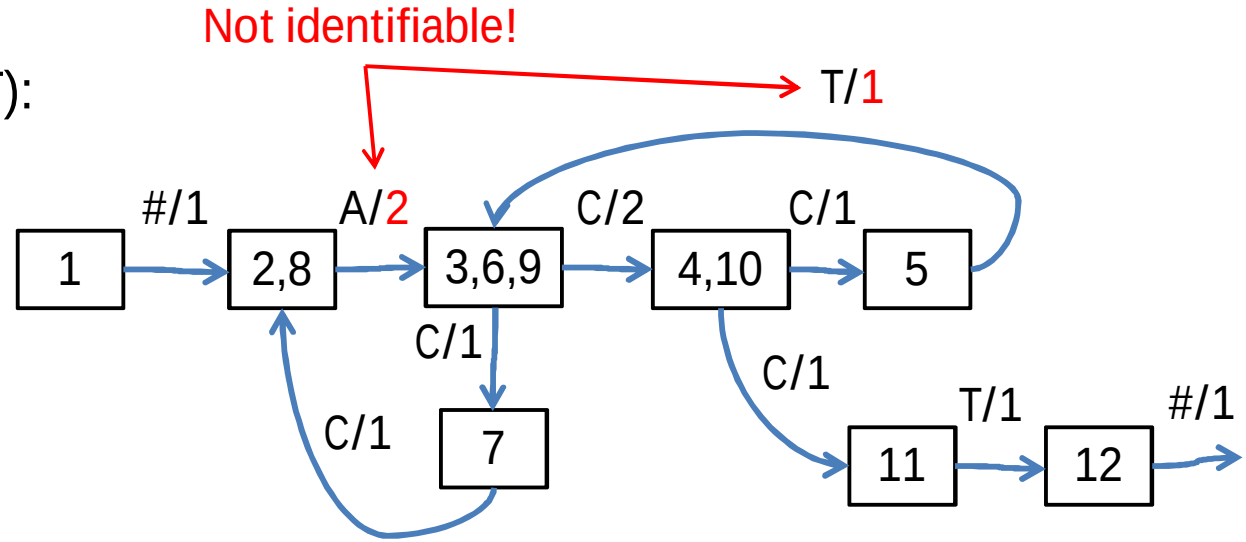
$T$ = #ACCTCCACCT###

k = 3

$F_3(T)$:

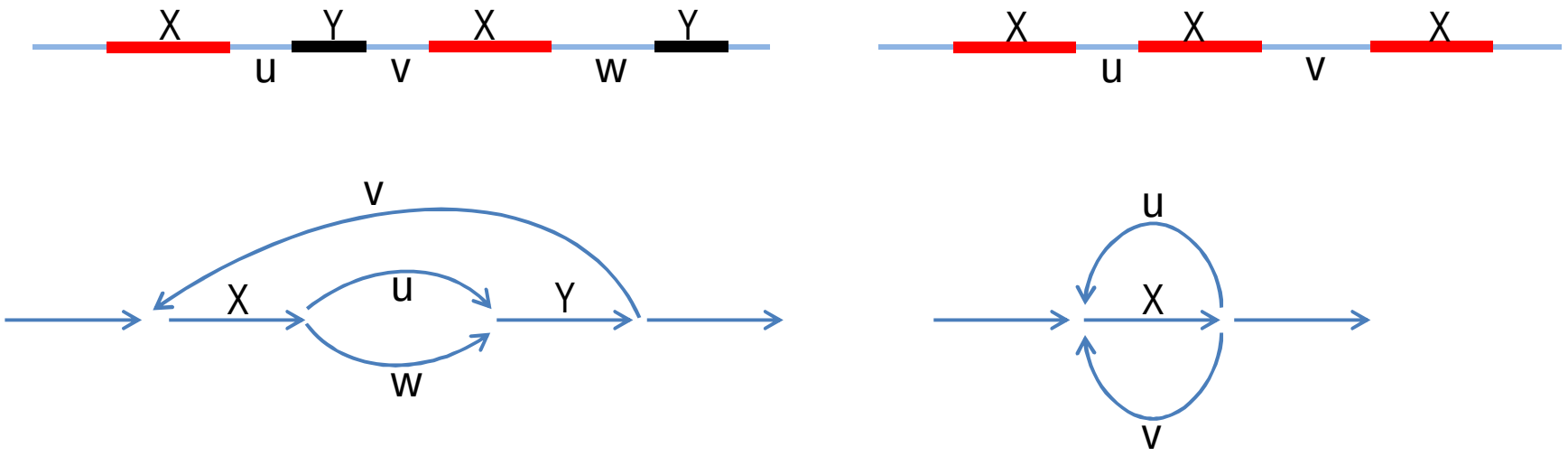
1. #AC
2. ACC
3. CCT
4. CTC
5. TCC
6. CCA
7. CAC
8. ACC
9. CCT
10. CT#
11. T##
12. ###

$B_3(T)$:



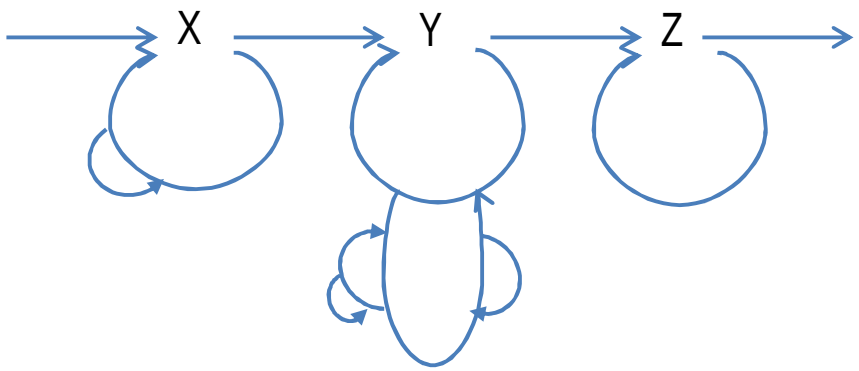
# Conditions on T for identifiability

- No interleaved pairs of repeats of length ≥ k-1
- No 3-repeats of length ≥ k-1



# How does a good $B_k(T)$ look like?

- If all repeats of T are bridged by k-mers, then there is no loops in $B_k(T)$
- If all interleaved 2-repeats and all 3-repeats are bridged, there still may be adjacent and nested 2-repeats that are unbridged
- If k ≥ length of any 3-repeat and any interleaved 2-repeat, then $B_k(T)$ has easy tree-shaped loop structure, i.e., only one Eulerian path



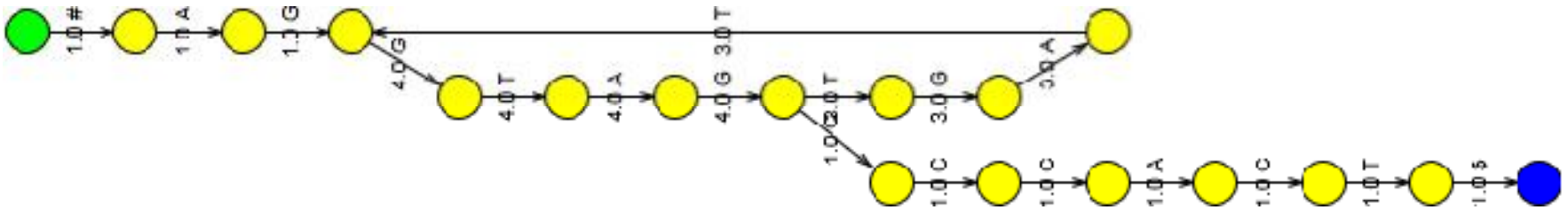
# Monotonicity

- If $B_k(T)$ has unique Eulerian path then $B_{k+1}(T)$ has unique Eulerian path

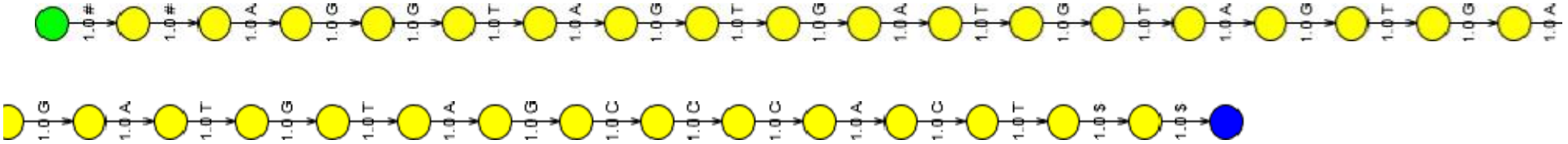
# Monotonicity

*T* = #AGGTAGTGATGTAGTGATGTAGTGATGTAGCCCACT$

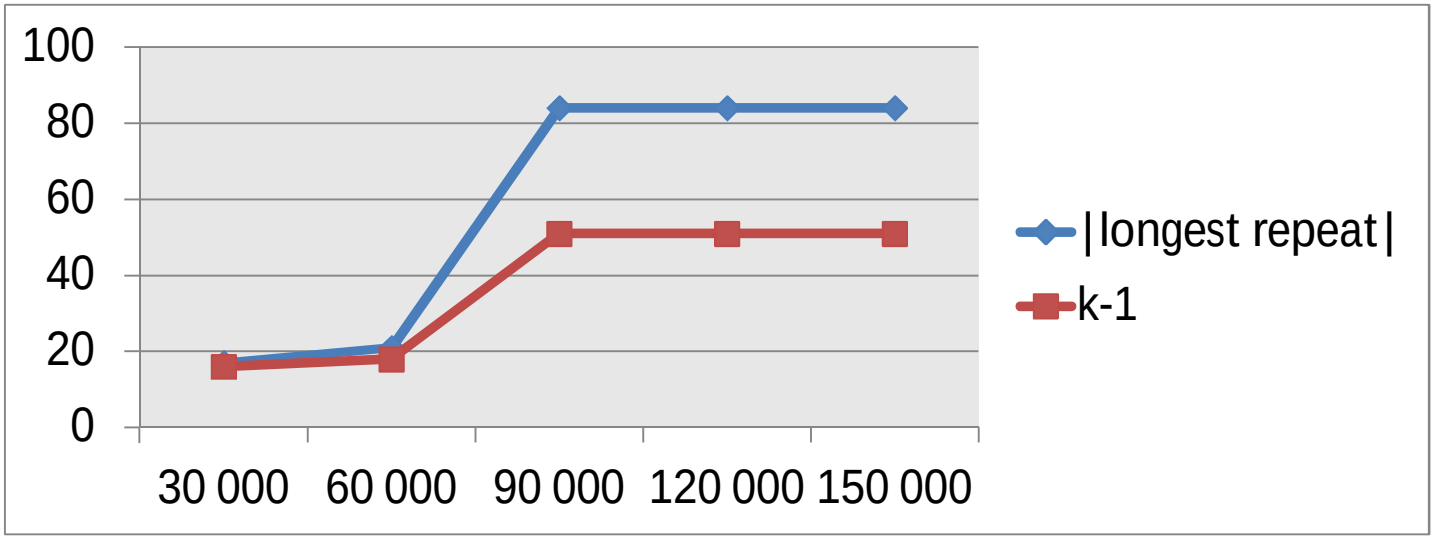
k = 5



k = 33



# Analysis of E. Coli genome




Substring [260 000, 293 000]:

- Length of longest repeat: 770
- k-1 required for identifiability: 176

# (More) general read set F

- *Read set F with overlap length k-1*: length of the reads in F is ≥ k and the reads, that are adjacent in the target, overlap at least by k-1 symbols
- *Substring graph* $B_k(F)$: generalization of de Bruijn graph for variable length reads

# Generalization

- Open:
  - Assume that T is identifiable from k-mers, i.e., $B_k(T)$ has only one Eulerian path. Let F be a read set of T with overlap length k-1 that covers the entire T.
  - Has $B_k(F)$ only one Eulerian path?

# All is linear time

- Construction of the de Bruijn / substring graph $B_k$: linear time by k-truncated suffix-tree / Aho-Corasick
- Identifiability test: linear time by df search etc
- Construction of the Eulerian path: linear time

# To conclude

- Modest observation: reads need not be longer than all repeats (but not very much shorter either)
  - Example (yeast): longest 2-repeat 8375, longest 3–repeat 6466
- Gaps in coverage: substring graph not connected; apply on Eulerian components
- Noiseless reads: error correction combined with selection of consistent subset of reads?

Thank you!