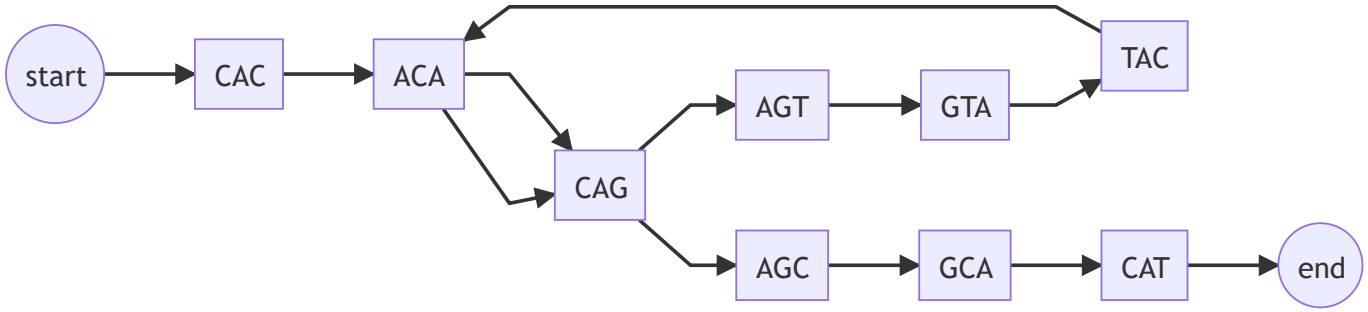
BIOL8706: Dividing and conquering sequence alignment using De Bruijn Graphs



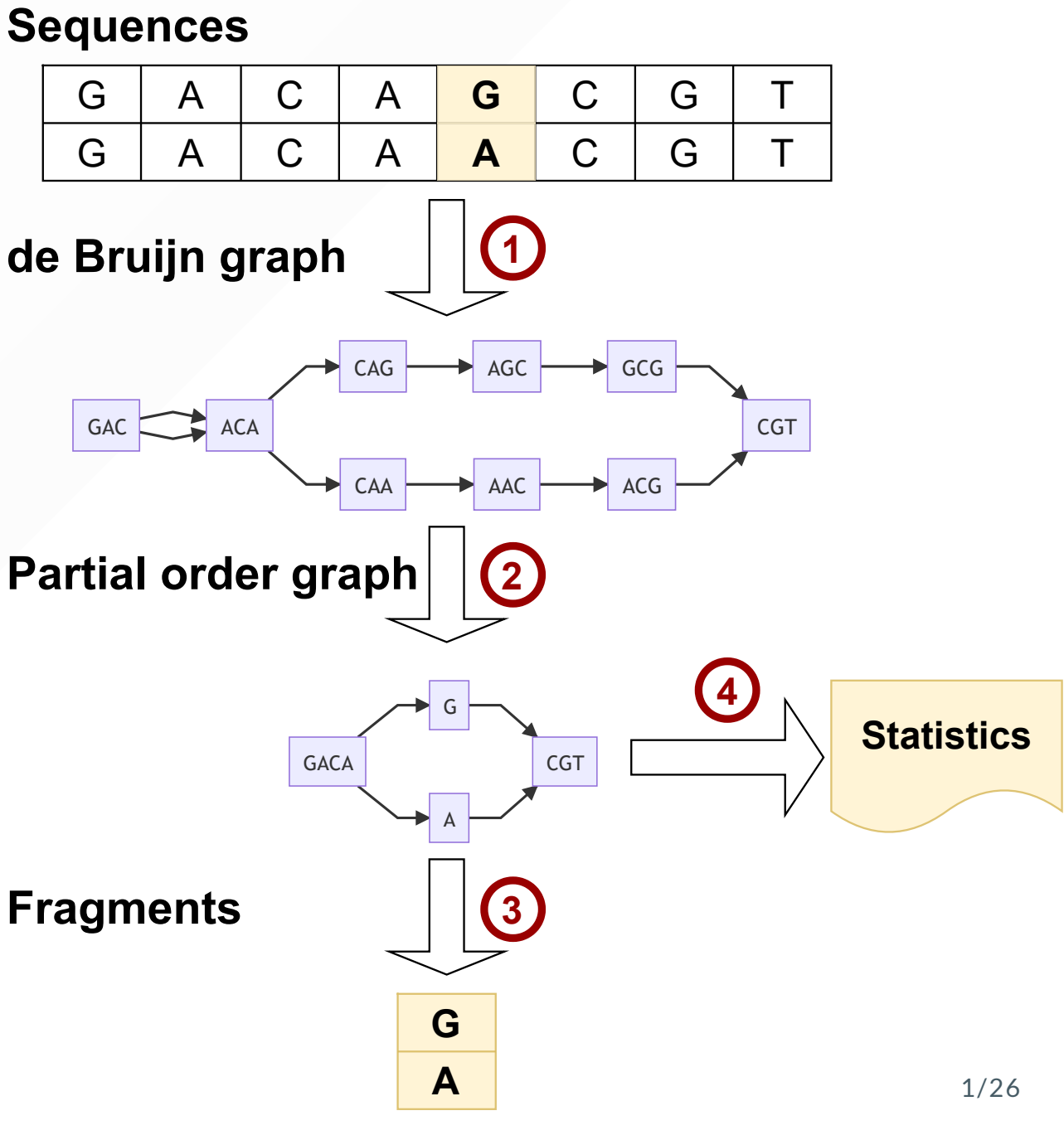
- Student: Richard Morris
- Huttley lab, Australian National University
- Supervisors: Gavin Huttley, Vijini Mallawaarachchi

Australian National University

Project aims

Build prototype

1. **construct** a de Bruijn graph from sequences
2. **projects** de Bruijn graph to a partial order graph
3. **emit** fragments from the partial order graph
4. generate statistics on work required for alignment



BACKGROUND: Sequence alignment

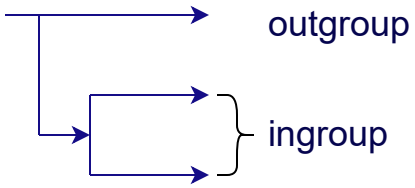
Sequences → **Alignment** → **Phylogenetic relationship**

Sequence A ACAGTACTGACAT

Sequence B ACAGCTGCAT

Sequence C ACAGCTACAT

A	C	A	G	T	A	C	T	G	A	C	A	T
A	C	A	G	-	-	C	T	G	-	C	A	T
A	C	A	G	-	-	C	T	-	A	C	A	T



BACKGROUND: Sequence alignment

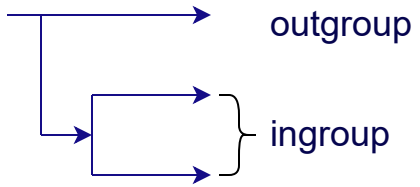
Sequences → **Alignment** → **Phylogenetic relationship**

Sequence A ACAGTACTGACAT

Sequence B ACAGCTGCAT

Sequence C ACAGCTACAT

A	C	A	G	T	A	C	T	G	A	C	A	T
A	C	A	G	-	-	C	T	G	-	C	A	T
A	C	A	G	-	-	C	T	-	A	C	A	T



We start with a set of DNA sequences to be aligned

BACKGROUND: Sequence alignment

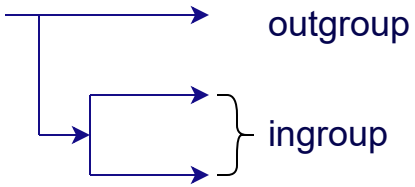
Sequences → **Alignment** → **Phylogenetic relationship**

Sequence A ACAGTACTGACAT

Sequence B ACAGCTGCAT

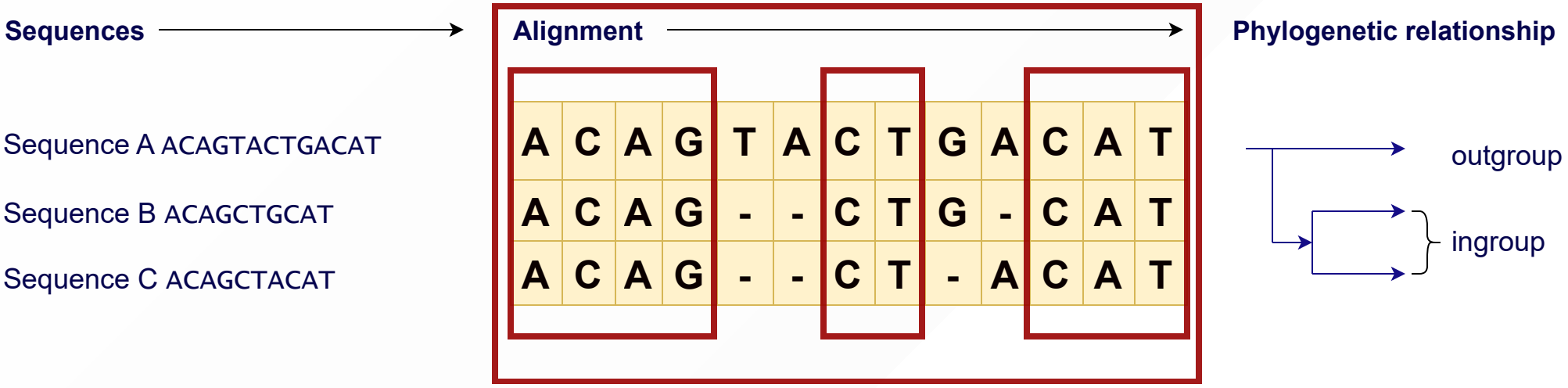
Sequence C ACAGCTACAT

A	C	A	G	T	A	C	T	G	A	C	A	T
A	C	A	G	-	-	C	T	G	-	C	A	T
A	C	A	G	-	-	C	T	-	A	C	A	T



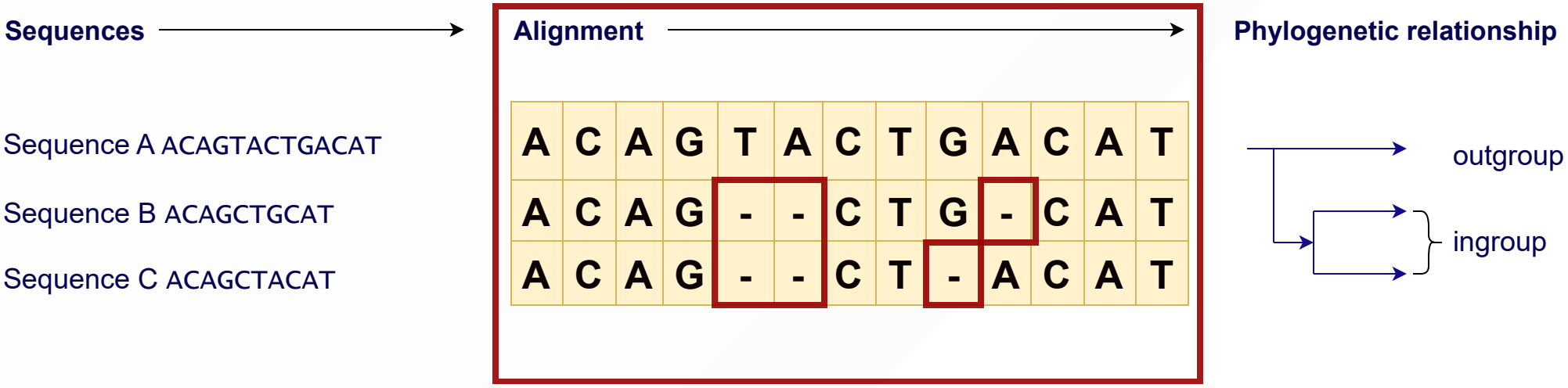
We align those sequences

BACKGROUND: Sequence alignment



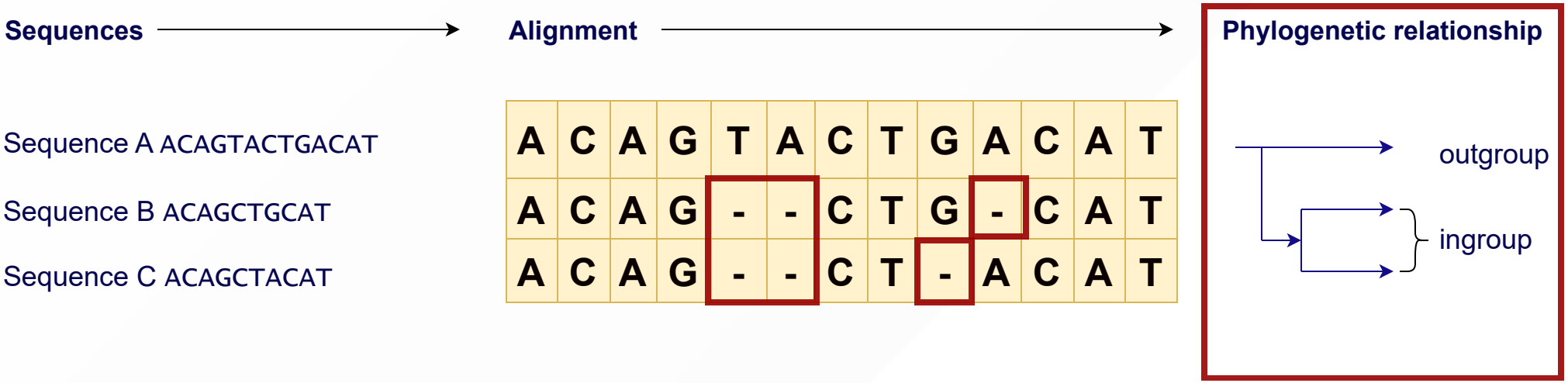
By lining up regions that are similar

BACKGROUND: Sequence alignment



Noting those that are different

BACKGROUND: Sequence alignment



And we can infer evolutionary relationships between those sequences

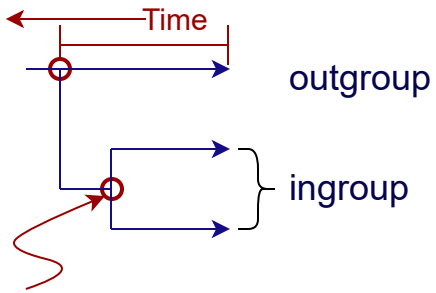
- ingroup (1 letter different)
- outgroup (3 letters different)

BACKGROUND: Sequence alignment

Sequences → **Alignment** →

Sequence A ACAGTACTGACAT	A	C	A	G	T	A	C	T	G	A	C	A	T
Sequence B ACAGCTGCAT	A	C	A	G	-	-	C	T	G	-	C	A	T
Sequence C ACAGCTACAT	A	C	A	G	-	-	C	T	-	A	C	A	T

Phylogenetic relationship



And we can infer evolutionary relationships between those sequences

- ingroup (1 letter different)
- outgroup (3 letters different)
- likely unobserved ancestor sequence
- how long ago sequences likely diverged

BACKGROUND: Sequence alignment

Sequences → **Alignment** →

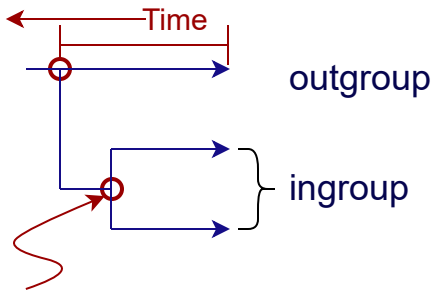
Sequence A ACAGTACTGACAT

Sequence B ACAGCTGCAT

Sequence C ACAGCTACAT

A	C	A	G	T	A	C	T	G	A	C	A	T
A	C	A	G	-	-	C	T	G	-	C	A	T
A	C	A	G	-	-	C	T	-	A	C	A	T

Phylogenetic relationship



Sequence alignment + phylogeny is a time machine for homologous sequences

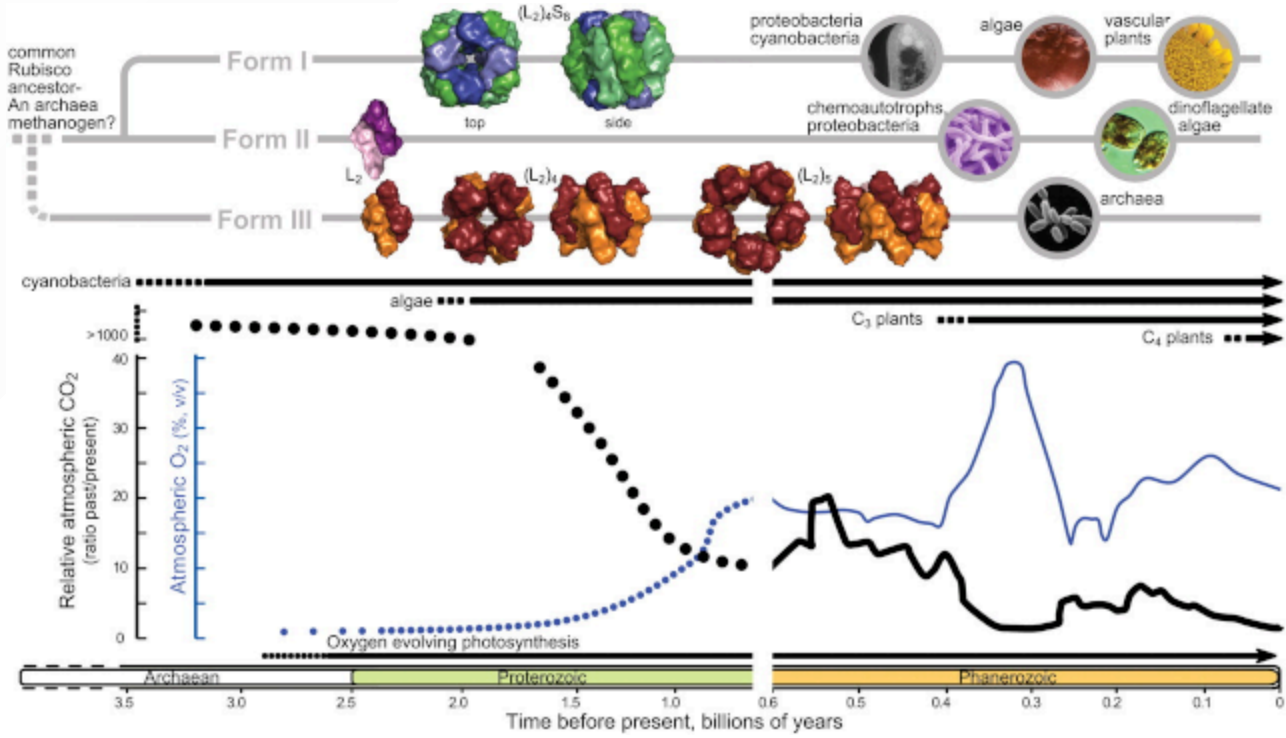
3 instructive cases of sequence alignment

- Evolution of RuBisCO
- Trajectory of the SARS-CoV-2 virus
- Our own Family tree

CASE: Evolution of RuBisCO

- Enzyme that converts CO_2 to organic carbon during photosynthesis
- Sequence alignment can infer it's evolutionary history
- Compare with a geological understanding of the atmosphere at that time

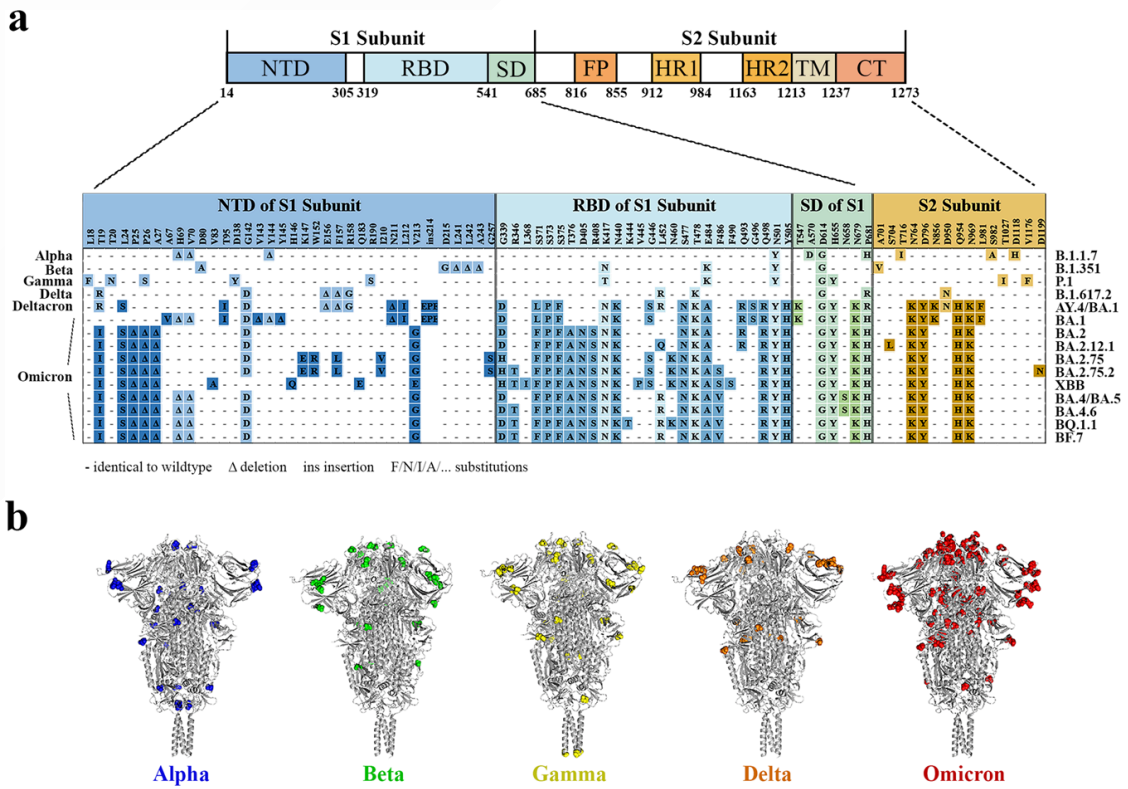
We can associate **features** appearing in the protein with the **environment** in which it evolved?



CASE: Trajectory of the spike protein of SARS-CoV-2

Sequence alignment

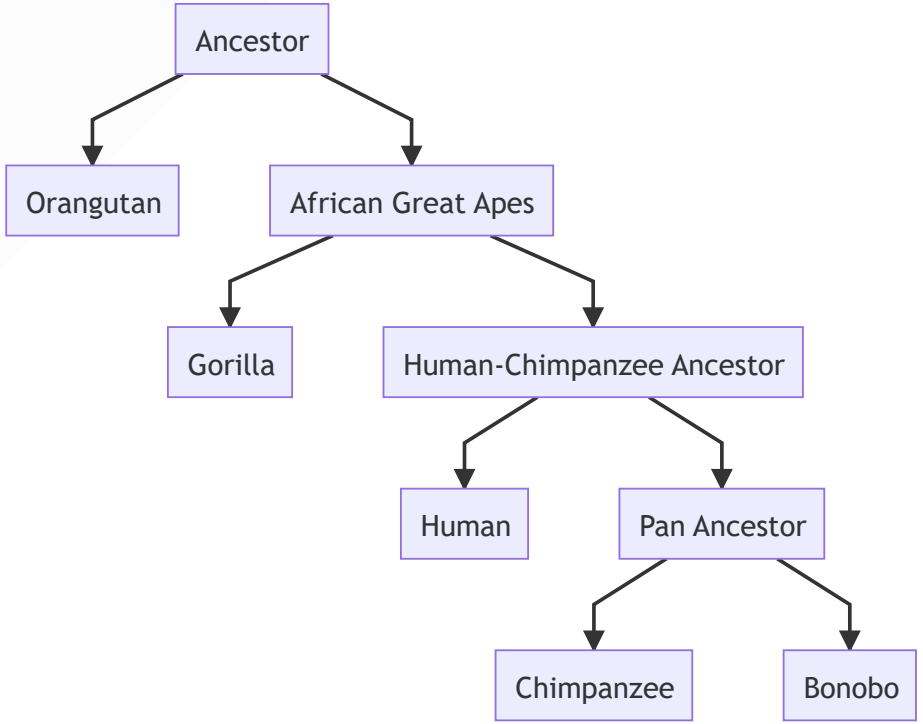
- allows us to identify conserved regions to inform vaccine/drug development
- can help us predict the virus's trajectory
  - where it came from
  - where it is going to



Alignment of S mutation points of SARS-CoV-2 variants

CASE: Our immediate family tree

- How do we differ from other great apes
- How are we the same
- This has direct applications in biomedical science



The family tree of great apes

PROBLEM: Sequence alignment is a big job

- Historically sequence alignment was done manually, like a really big evil jigsaw puzzle
- Since 1970[1] it has become a computational problem
- The task is to compare **each** letter in **each** sequence with **all** the letters of **every** other sequence.

- The terms: **each**, **all** and **every** should tell you that it will be a big job for computers too.

Needleman & Wunsch, 1970 doi.org/10.1016/0022-2836(70)90057-4

PROBLEM: Exhaustive sequence alignment takes time

A computational scientist might say that the asymptotic complexity of an exhaustive alignment is given by the following big-O notation

$$O(L^n)$$

Where:

- L is the average length of the sequence
- n is the number of sequences

"Big-O tells you how code **slows** as data **grows**" *Ned Batchelder*

REFRAME: Work increases as data grows

Let's rephrase this big-O notation as the order of $Work(L^n)$

Average length (L)	Number of sequences (n)	$Work$ required (comparisons)
1,000	3	1,000,000,000
2,000	3	8,000,000,000
3,000	3	27,000,000,000
4,000	3	64,000,000,000
5,000	3	125,000,000,000
6,000	3	216,000,000,000
7,000	3	343,000,000,000
8,000	3	512,000,000,000
9,000	3	729,000,000,000
10,000	3	1,000,000,000,000

REFRAME: Work increases as data grows

Let's rephrase this big-O notation as the order of $Work(L^n)$

Average length (L)	Number of sequences (n)	$Work$ required (comparisons)
1,000	2	1,000,000
1,000	3	1,000,000,000
1,000	4	1,000,000,000,000
1,000	5	1,000,000,000,000,000
1,000	6	1,000,000,000,000,000,000
1,000	7	1,000,000,000,000,000,000,000
1,000	8	1,000,000,000,000,000,000,000,000
1,000	9	1,000,000,000,000,000,000,000,000,000
1,000	10	1,000,000,000,000,000,000,000,000,000,000

PROBLEM: The scale of our 3 cases

Case	Average length (L)	Number of sequences (n)	$Work$ required (L^n)
RuBisCO	2 kbp	350,000	$2{,}000^{350{,}000}$
SARS-CoV-2	29 kbp	5,000,000*	$29{,}000^{5{,}000{,}000}$
Great apes	3 gbp	5	$3\ \text{billion}^5$

Large computation problems take

- **Time** ⌚
- **Money** $
- **Energy** 💡

* GISAID had 5.1M copies of SARS-CoV-2 sequences as of Oct 2021 www.nature.com/articles/s41588-022-01033-y


WE'RE GONNA NEED
MORE HAMSTERS
imgflip.com

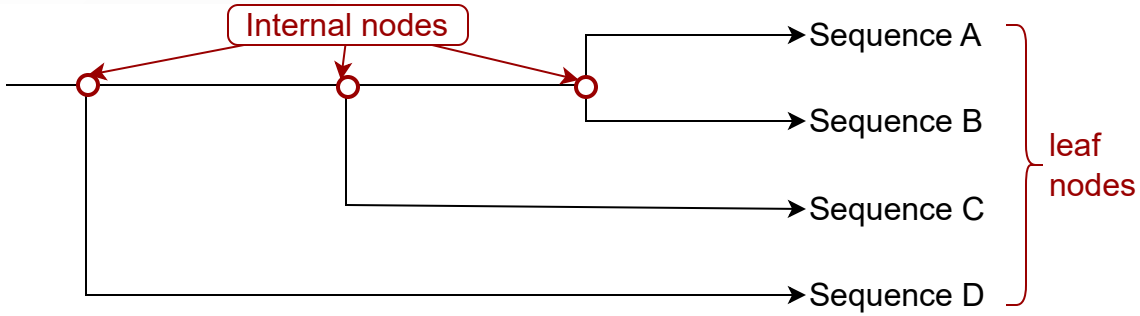
STATE OF THE ART: Progressive alignment

Progressive alignment is a method that reduces the work required

Strategy:

1. start with a phylogeny

Phylogenetic tree

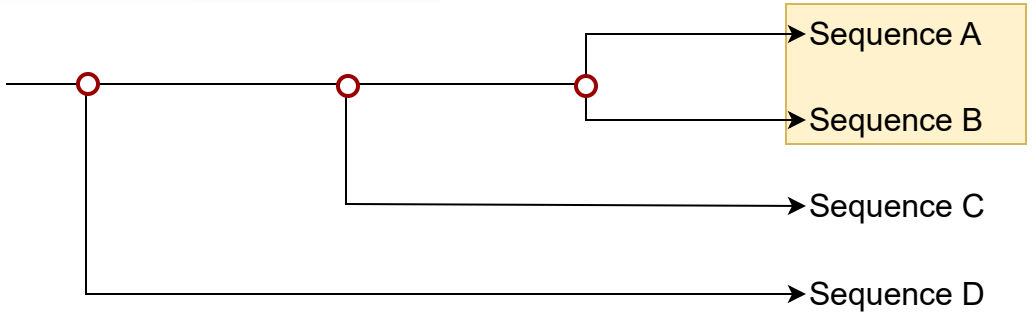


STATE OF THE ART: Progressive alignment

Progressive alignment is a method that reduces the work required

Strategy:

1. start with a phylogeny
2. align the **most closely related** sequences into a statistical model called a profile



STATE OF THE ART: Progressive alignment

Progressive alignment is a method that reduces the work required

Strategy:

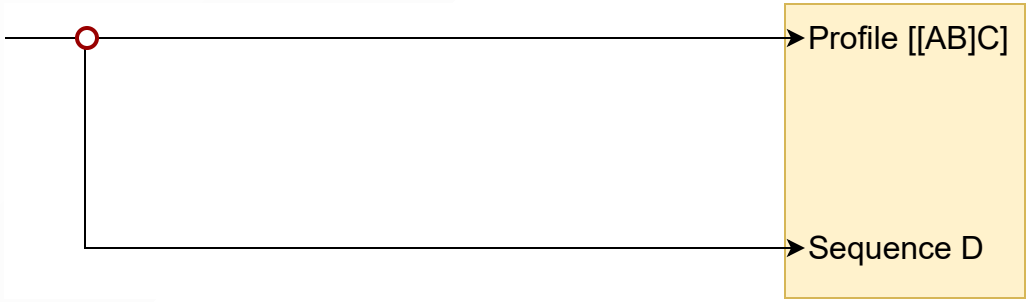
1. start with a phylogeny
2. align the most closely related sequences into a statistical model called a profile
3. align that profile with the **next** most closely related sequence

STATE OF THE ART: Progressive alignment

Progressive alignment is a method that reduces the work required

Strategy:

1. start with a phylogeny
2. align the most closely related sequences into a statistical model called a profile
3. align that profile with the next most closely related sequence
4. REPEAT until you have finished aligning **all the sequences**

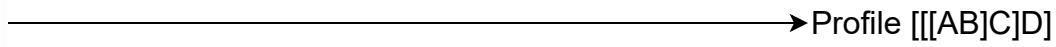


STATE OF THE ART: Progressive alignment

Progressive alignment is a method that reduces the work required

Strategy:

1. start with a phylogeny
2. align the most closely related sequences into a statistical model called a profile
3. align that profile with the next most closely related sequence
4. REPEAT until you have finished aligning all the sequences



This reduces the order of $Work(L^n) \rightarrow Work(i.L^2)$

- Where i is the number of internal nodes originally in the tree
  - binary tree: $i = (n-1)$
  - non-binary tree: $1 \geq i \leq (n-1)$

... That is a lot less $Work$

Progressive multiple sequence alignment (MSA) ...


PHYLOGENY IS NEEDED FOR ALIGNMENT
ALIGNMENT IS NEEDED FOR PHYLOGENY
imgflip.com

The problem space

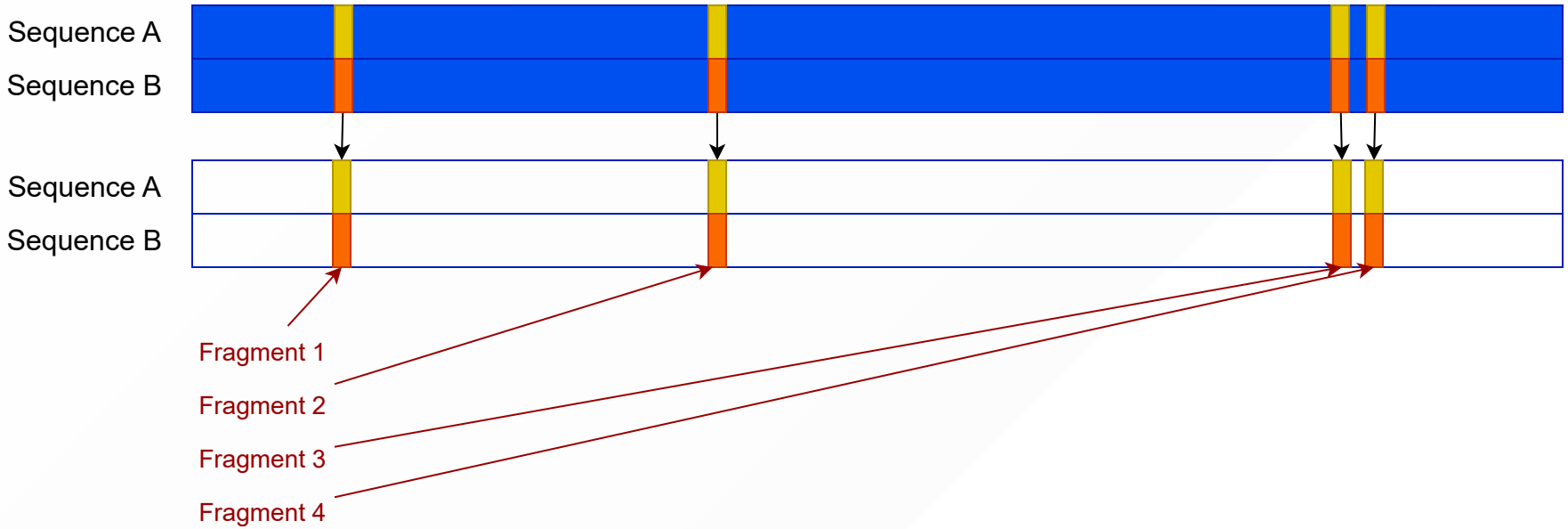
Sequence alignment is sensitive to

- The **length** of sequences to be aligned
- The **number** of sequences to be aligned
- the “ Chicken and Egg ” problem

An ideal strategy would reduce

- The **length** of sequences to be aligned
- The **number** of sequences to be aligned
- Dependence on knowing the phylogeny in advance

What if we could quickly remove similar regions?

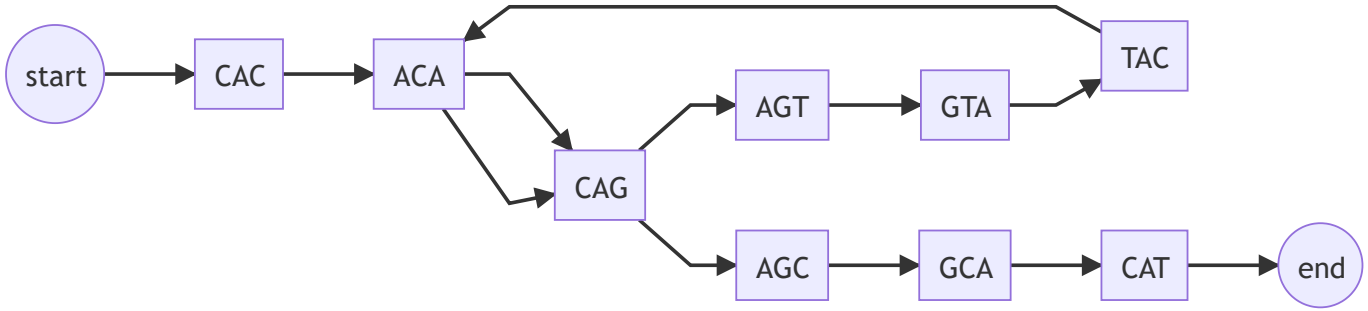


We could pass the alignment method, just the fragments that differ

Sequence alignment using De Bruijn Graphs

My work builds upon the work by **Xingjian Leng**[1] , under the supervision of **Dr. Yu Lin** and **Prof. Gavin Huttley**.

Xingjian tackled the length problem using de Bruijn graphs



de Bruijn graphs can be used for sequence assembly from reads

De Bruijn graphs can also be used for sequence alignment

[1] Leng, Xingjian. 'Sequence Alignment Using De Bruijn Graphs'. Australian National University, 2022

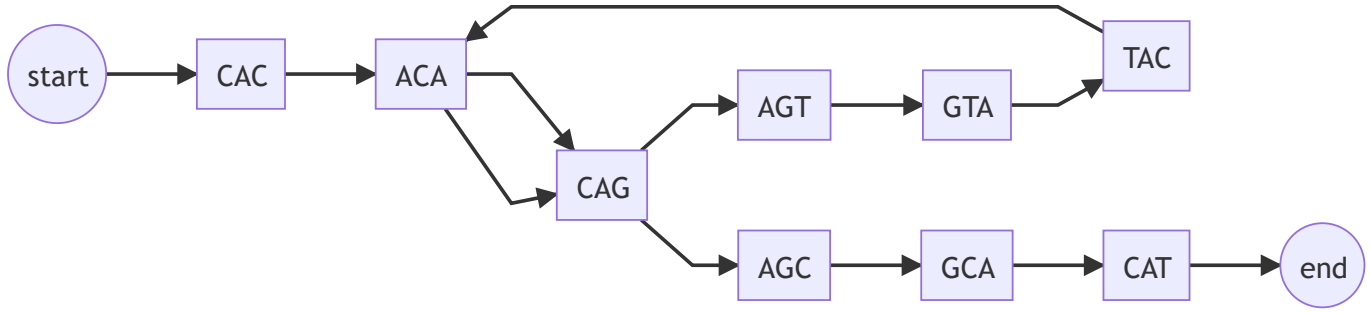
De Bruijn graphs

Building a De Bruijn graph is $Work(nL)$

This “Work” scales linearly not exponentially.

Consider the following sequence as a de Bruijn graph of order 3 (nodes overlap by 2 characters):

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



De Bruijn graphs

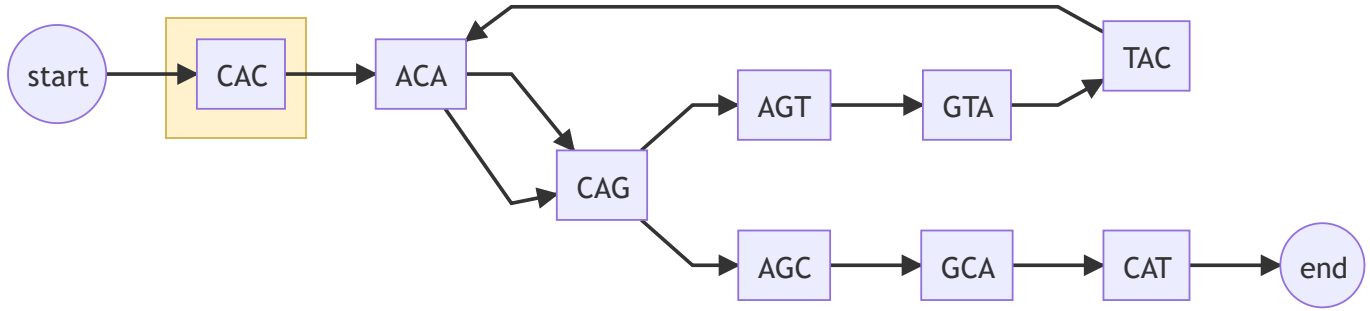
Building a De Bruijn graph is $Work(nL)$

This “Work” scales linearly not exponentially.

Consider the following sequence as a de Bruijn graph of order 3 (nodes overlap by 2 characters):

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

C	A	C



De Bruijn graphs

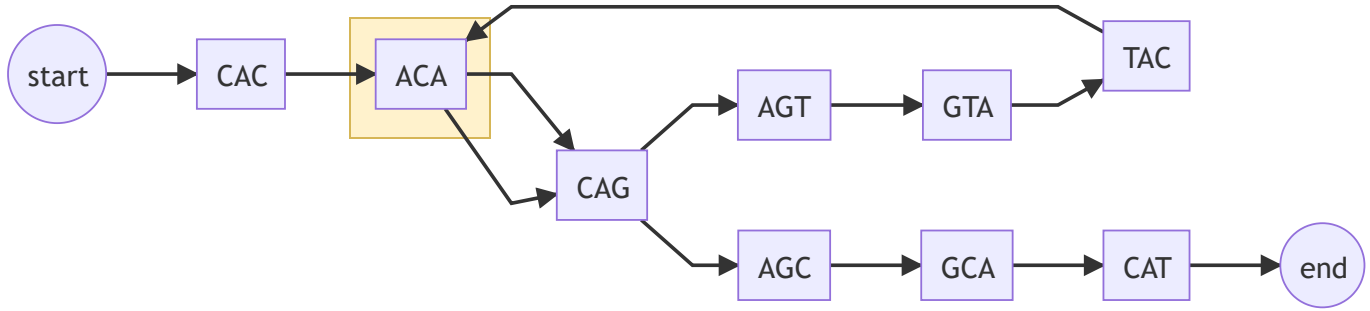
Building a De Bruijn graph is $Work(nL)$

This “Work” scales linearly not exponentially.

Consider the following sequence as a de Bruijn graph of order 3 (nodes overlap by 2 characters):

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

A	C	A

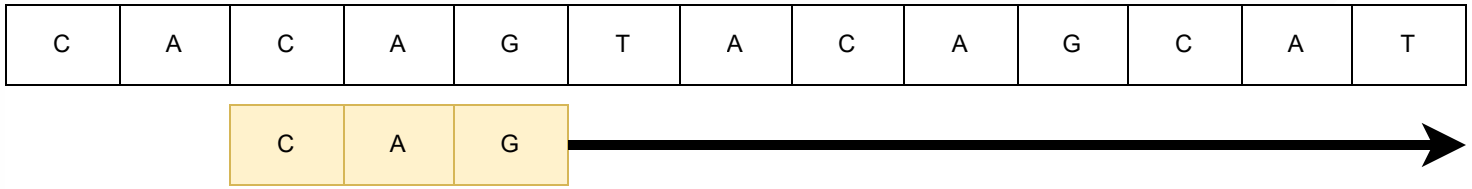


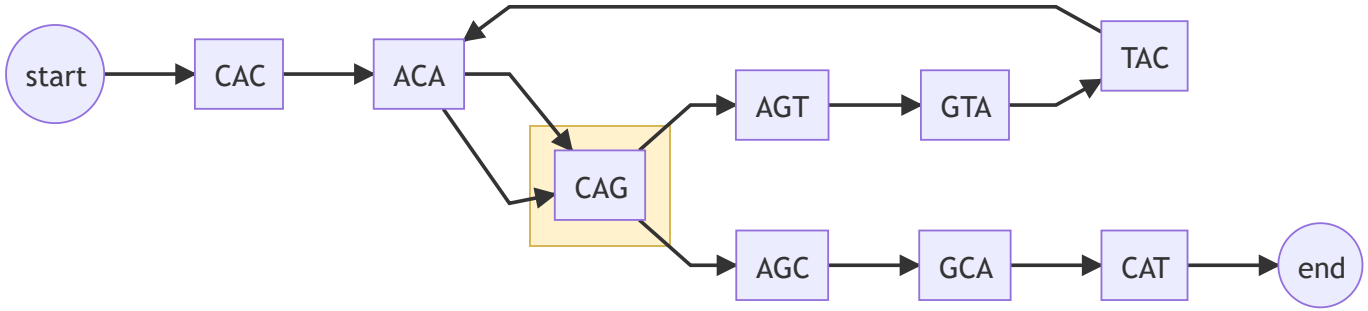
De Bruijn graphs

Building a De Bruijn graph is $Work(nL)$

This "Work" scales linearly not exponentially.

Consider the following sequence as a de Bruijn graph of order 3 (nodes overlap by 2 characters):

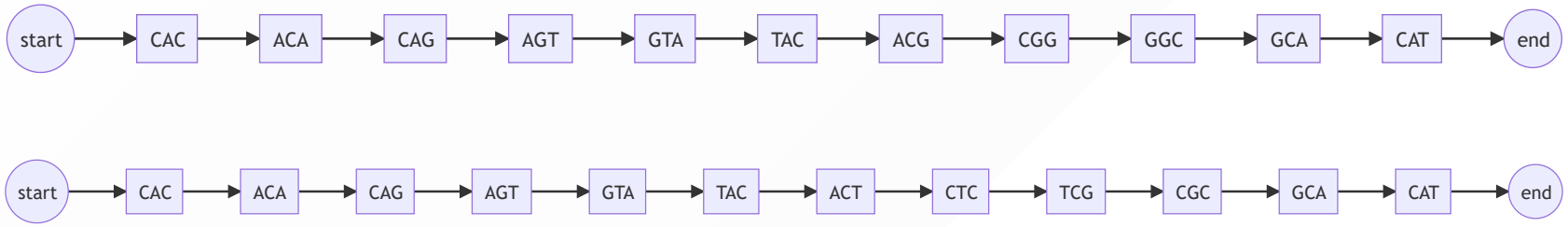




Reducing the length of fragments to be aligned

Seqence A	C	A	C	A	G	T	A	C	**G**	G	C	A	T
Seqence B	C	A	C	A	G	T	A	C	**T**	G	C	A	T

Produces the following de Bruijn graphs

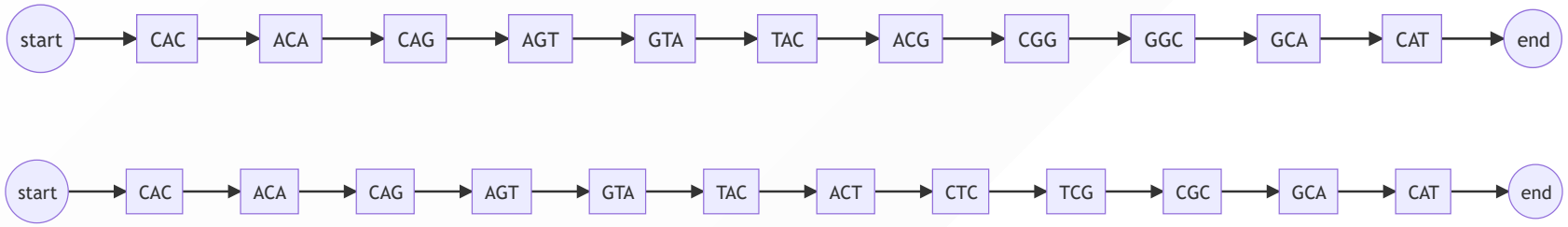


If we combine both sequences into a single de Bruijn graph, it will develop **“bubbles”** where regions are different.

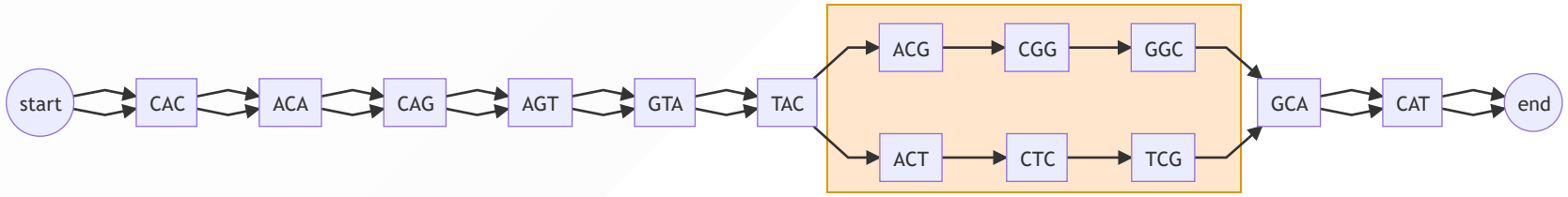
Reducing the **length** of fragments to be aligned

Seqence A	C	A	C	A	G	T	A	C	**G**	G	C	A	T
Seqence B	C	A	C	A	G	T	A	C	**T**	G	C	A	T

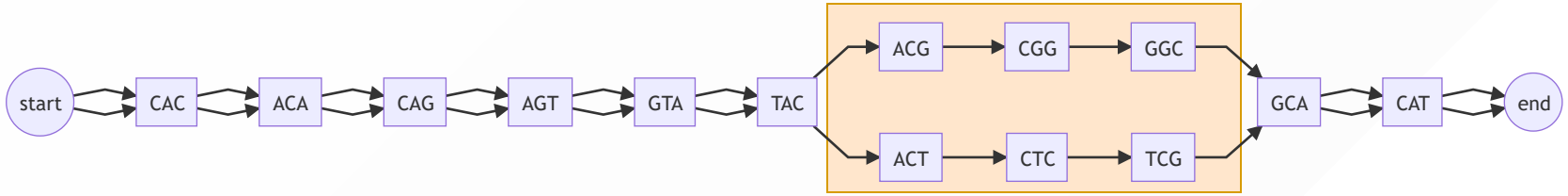
Produces the following de Bruijn graphs



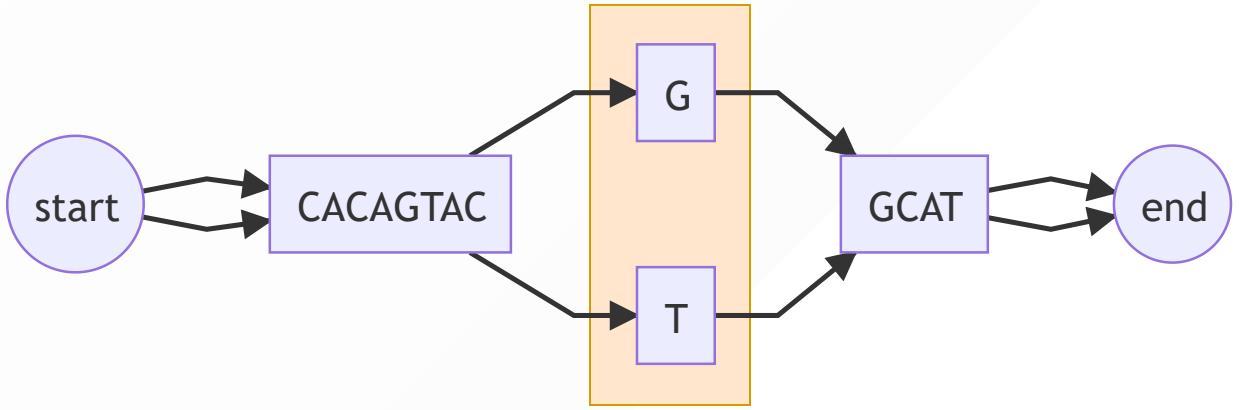
If we combine both sequences into a single de Bruijn graph, it will develop **“bubbles”** where regions are different.



Reducing the **length** of fragments to be aligned



can be transformed to the partial order graph

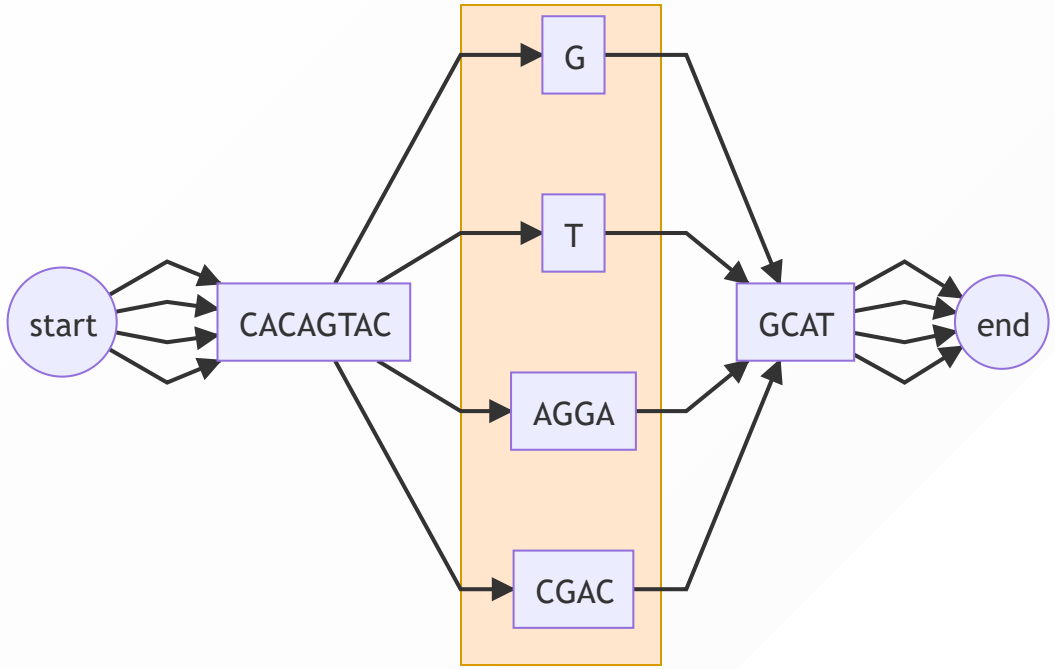


We have reduced $Work(14 \times 14) = 196$
to $Work(1 \times 1) = 1$

196x less “work”.

De Bruijn multiple sequence alignment

Consider aligning 4 sequences



We have reduced $Work(13 \times 13 + 13 \times 16 + 16 \times 16) = 633$ for a progressive alignment to $Work(1 \times 1 + 1 \times 4 + 4 \times 5) = 24$

26x less “work” than a progressive alignment

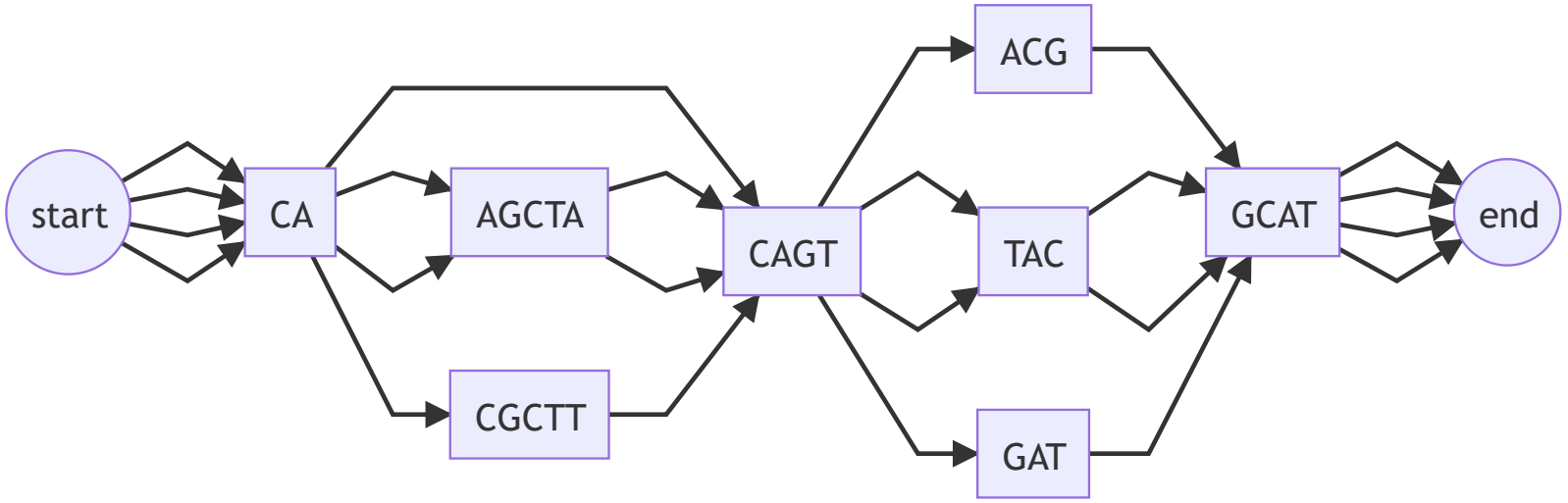
Taking the de Bruijn graph to the next level

We have changed the length of fragments

Can we change the number of fragments to align?

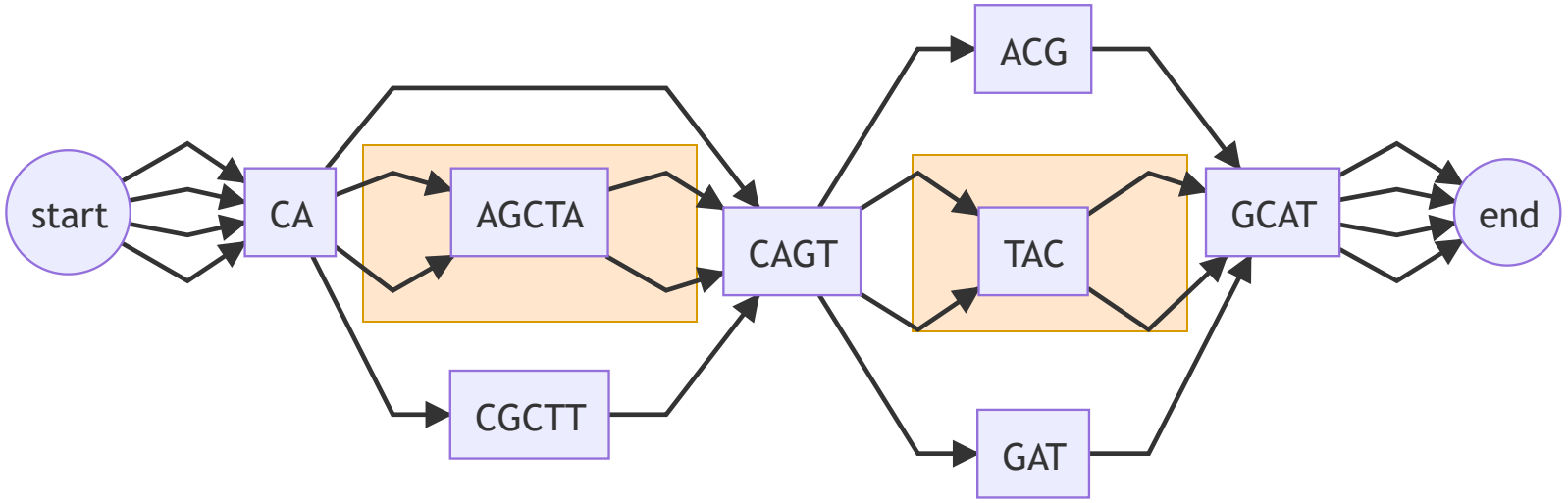
RESULT: Reducing the number of fragments to be aligned

Consider this partial order graph containing 4 sequences



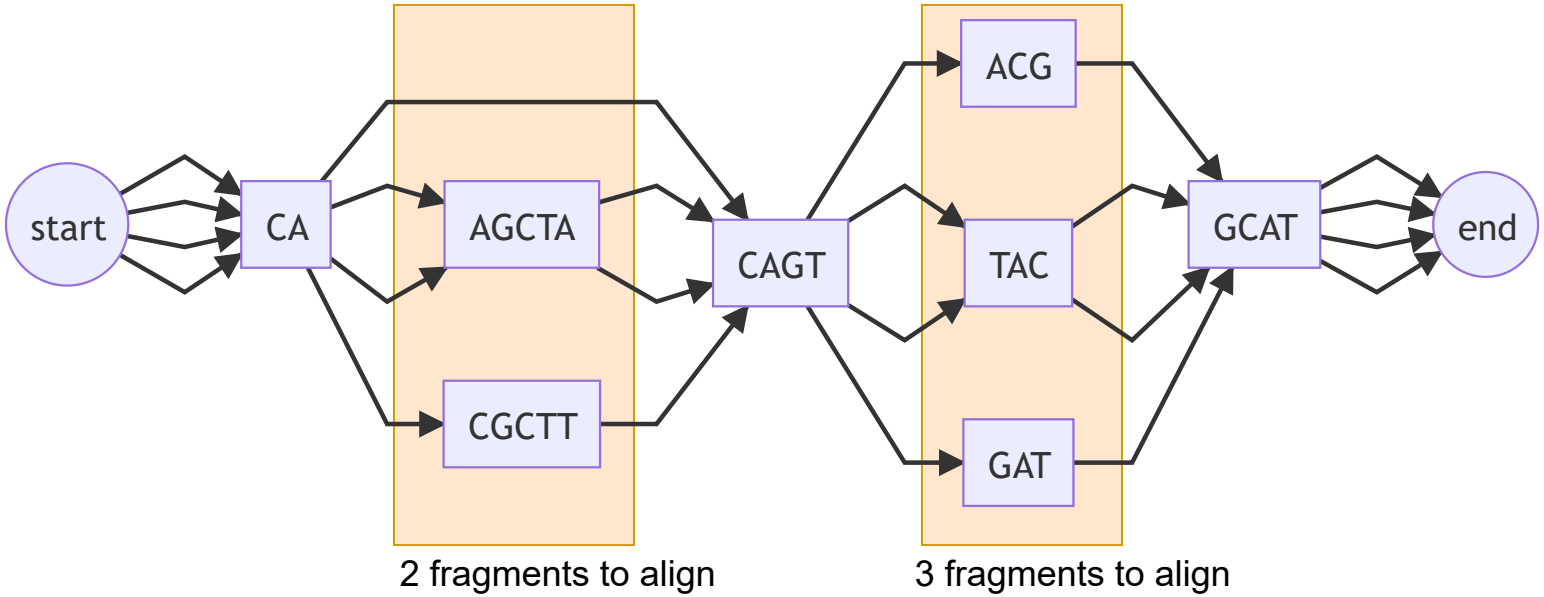
RESULT: Reducing the number of fragments to be aligned

Consider this partial order graph containing 4 sequences



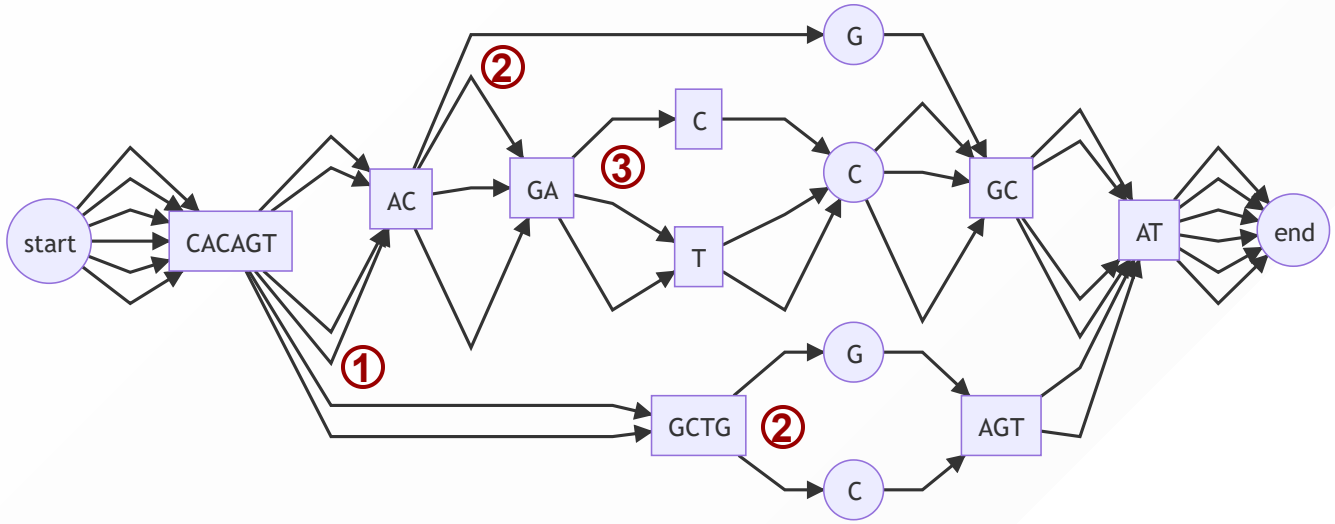
Sequences in bubbles can **braid** together

RESULT: Reducing the number of fragments to be aligned



progressive alignment	reduce length	reduce number
$(14 \times 18) + 2(18 \times 18)$	$3(5^2) + 3(3^2)$	$5^2 + 2(3^2)$
900	102 (**>8x** less work)	43 (**>20x** less work)

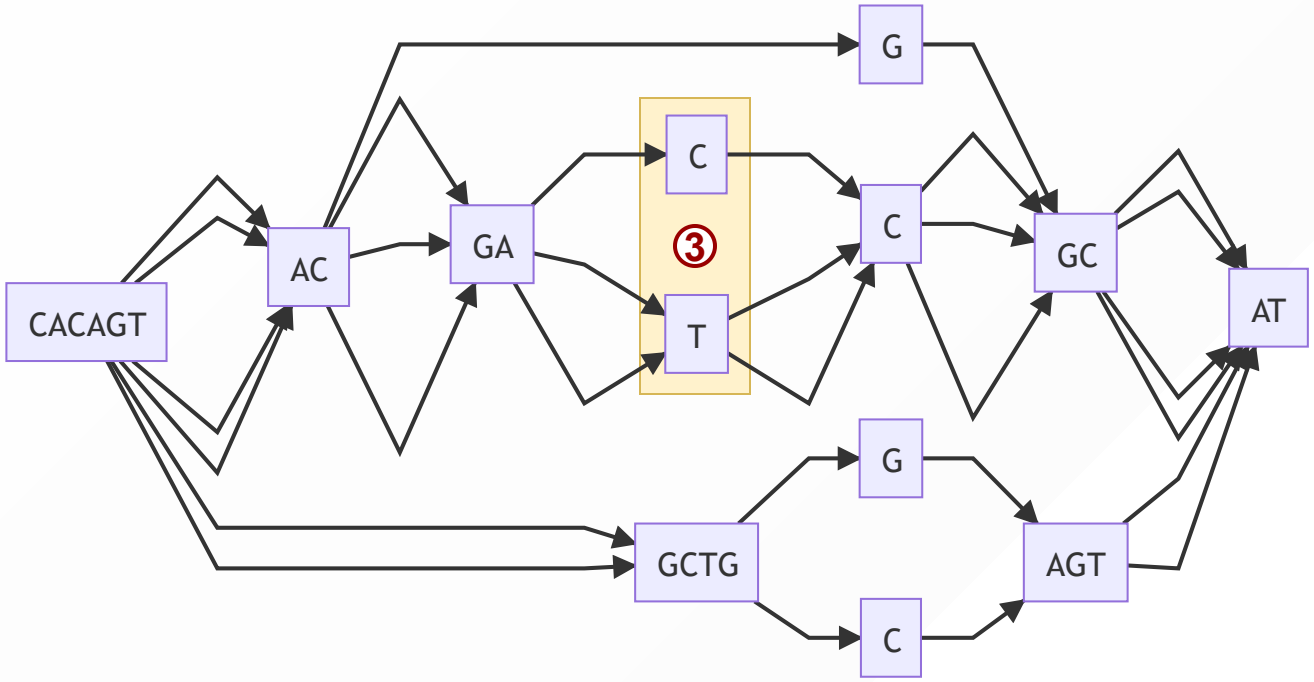
RESULT: Reduce the dependence on the phylogeny

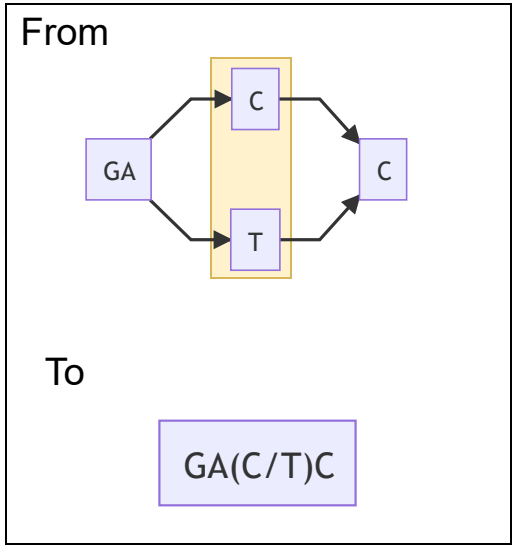


1. order fragments by depth of bubble
2. start with deepest set of fragments
3. align progressively

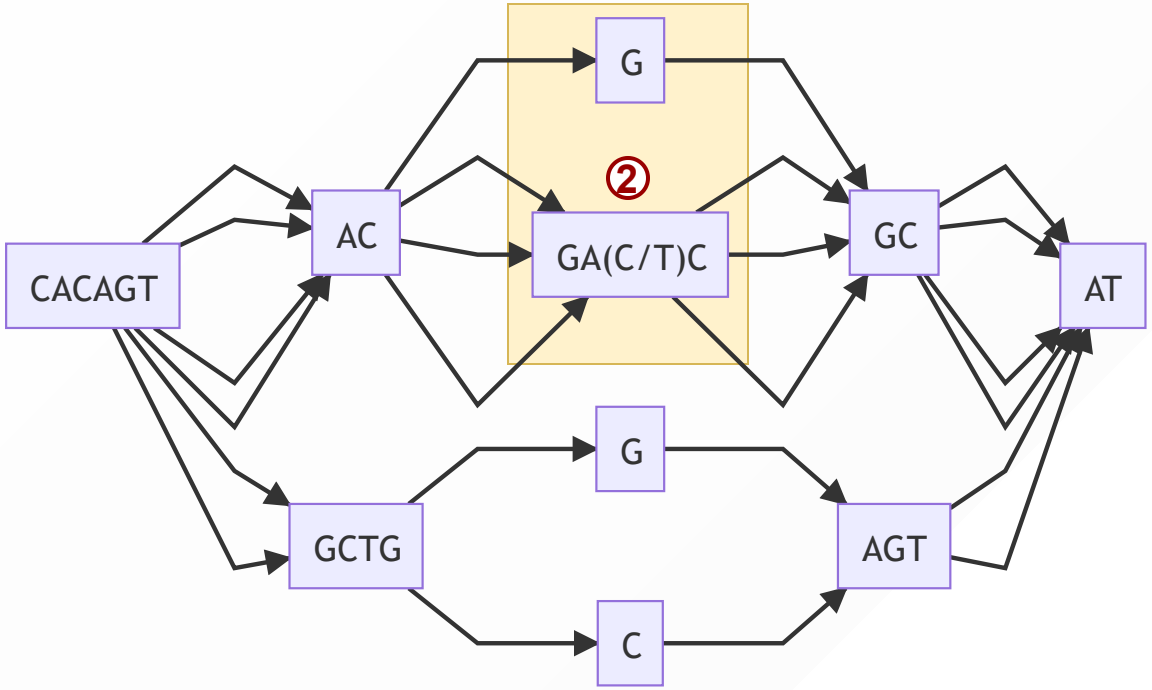
Align without needing to know in advance the phylogeny

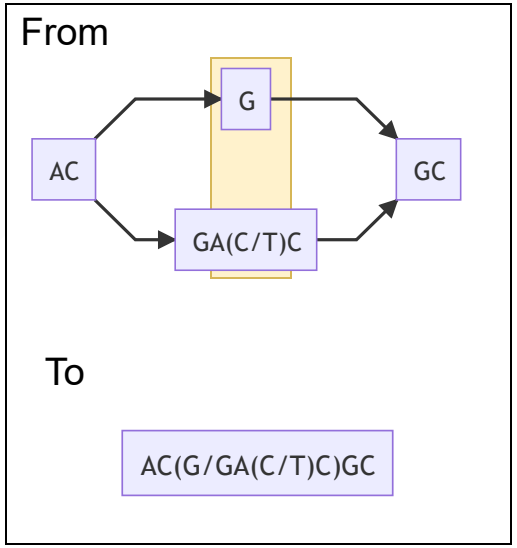
RESULT: Reduce the dependence on the phylogeny



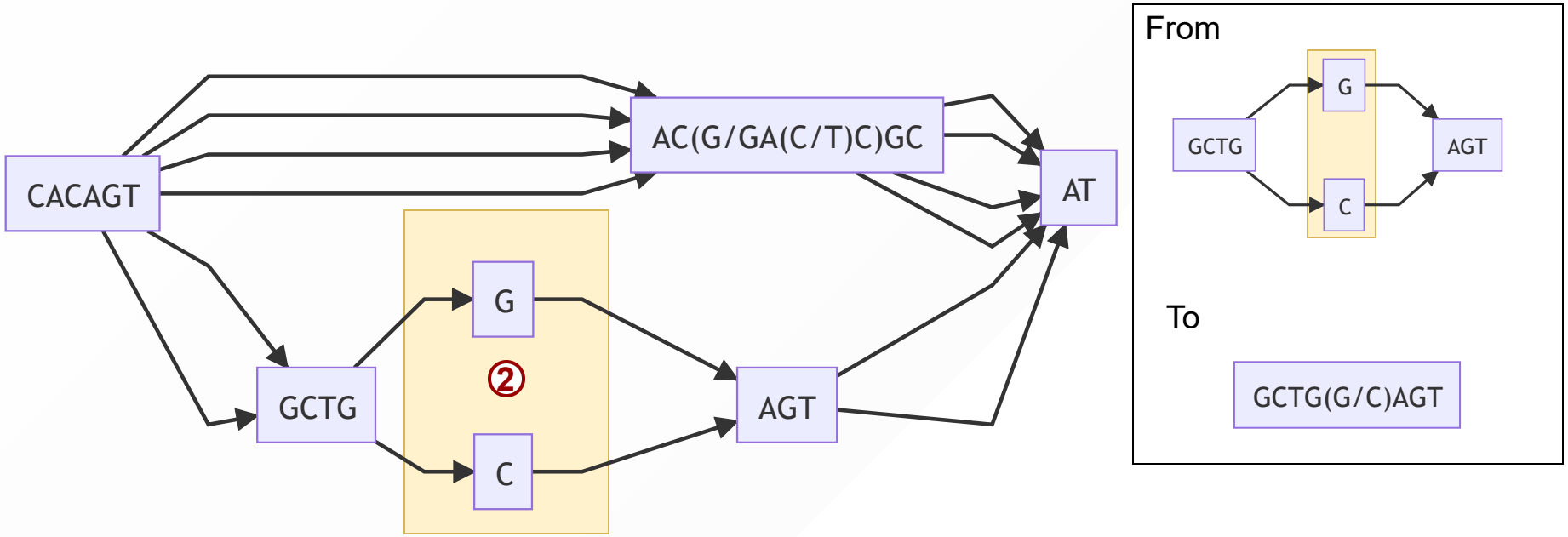


RESULT: Reduce the dependence on the phylogeny

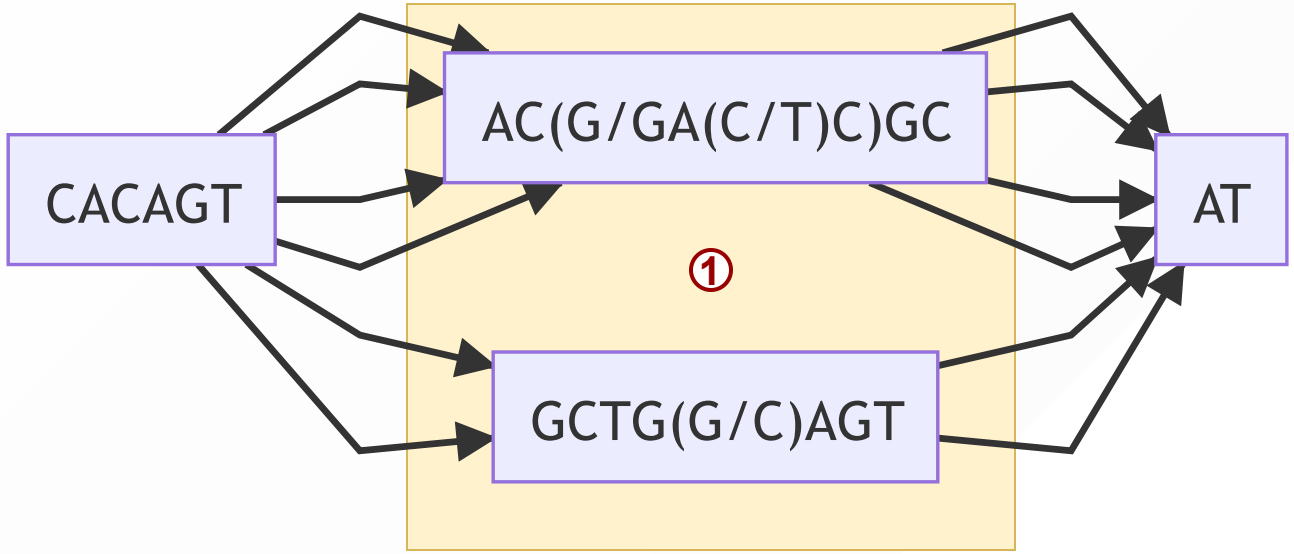


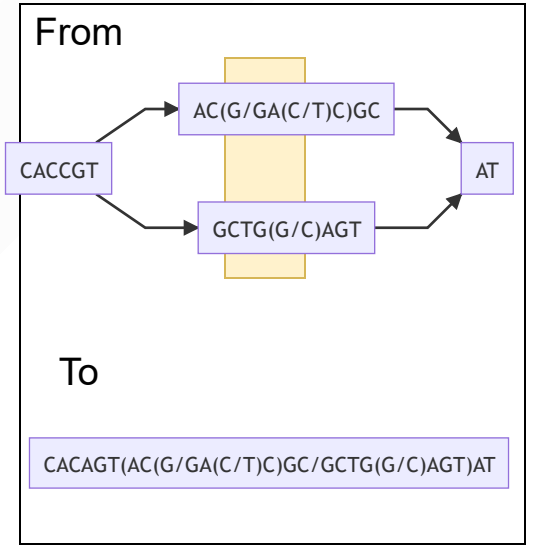


RESULT: Reduce the dependence on the phylogeny



RESULT: Reduce the dependence on the phylogeny





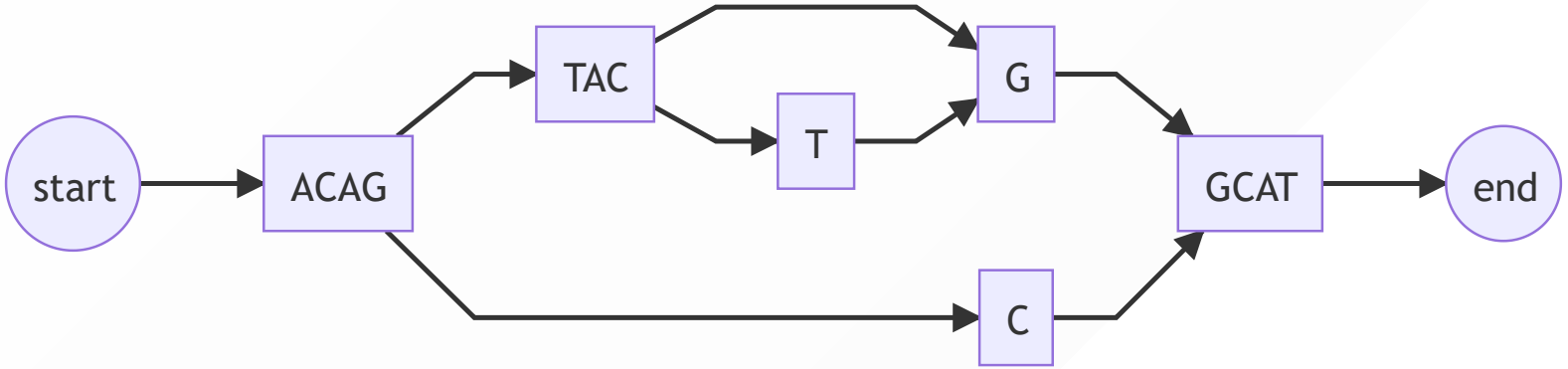
RESULT: Reduce the dependence on the phylogeny

CACAGT(AC(G/GA(C/T)C)GC/GCTG(G/C)AGT)AT

Alignment completed without requiring a phylogeny

RESULT: Work statistic from partial order graphs

Consider the same partial order graph



- Work calculates the order of alignment work using 4 strategies
  - Exact = $13 \times 12 \times 9 = 1404$
  - Progressive = $12 \times 13 + 13 \times 9 = 273$
  - DBG_L = $4 \times 5 + 5 \times 1 = 25$ (simplification of sequence length)
  - DBG_LN = $0 \times 1 + 5 \times 1 = 5$ (simplification of sequence length and count)

RESULT: Calculated order of Work - TBD

Sample Sequence sets

Genomes	Avg. length (bp)	number of sequences	Exact	Progressive	dBG(3)_L	dBG(3)_LN
BRCA1_divergent	~3k	7				
BRCA1_hominae	~3k	4				
SARS-CoV-2	30k	22				
IBD_phage	40k	60				
Ocean_phage	40k	130				

Summary

It's quite clear that this method of using de Bruijn graphs to reduce both length and number of fragments to align in these synthetic examples offers a significant performance improvement over reducing just the length, and further significant improvements over the state of the art; progressive alignment.

de Bruijn graphs appear to break the 40 year tautology at the heart of **sequence alignment**, and **phylogenetic reconstruction**.

It is worth perservering with this method to see if it can be applied to evolved sequences.

It looks like this method will make some very big questions tractable

Future directions

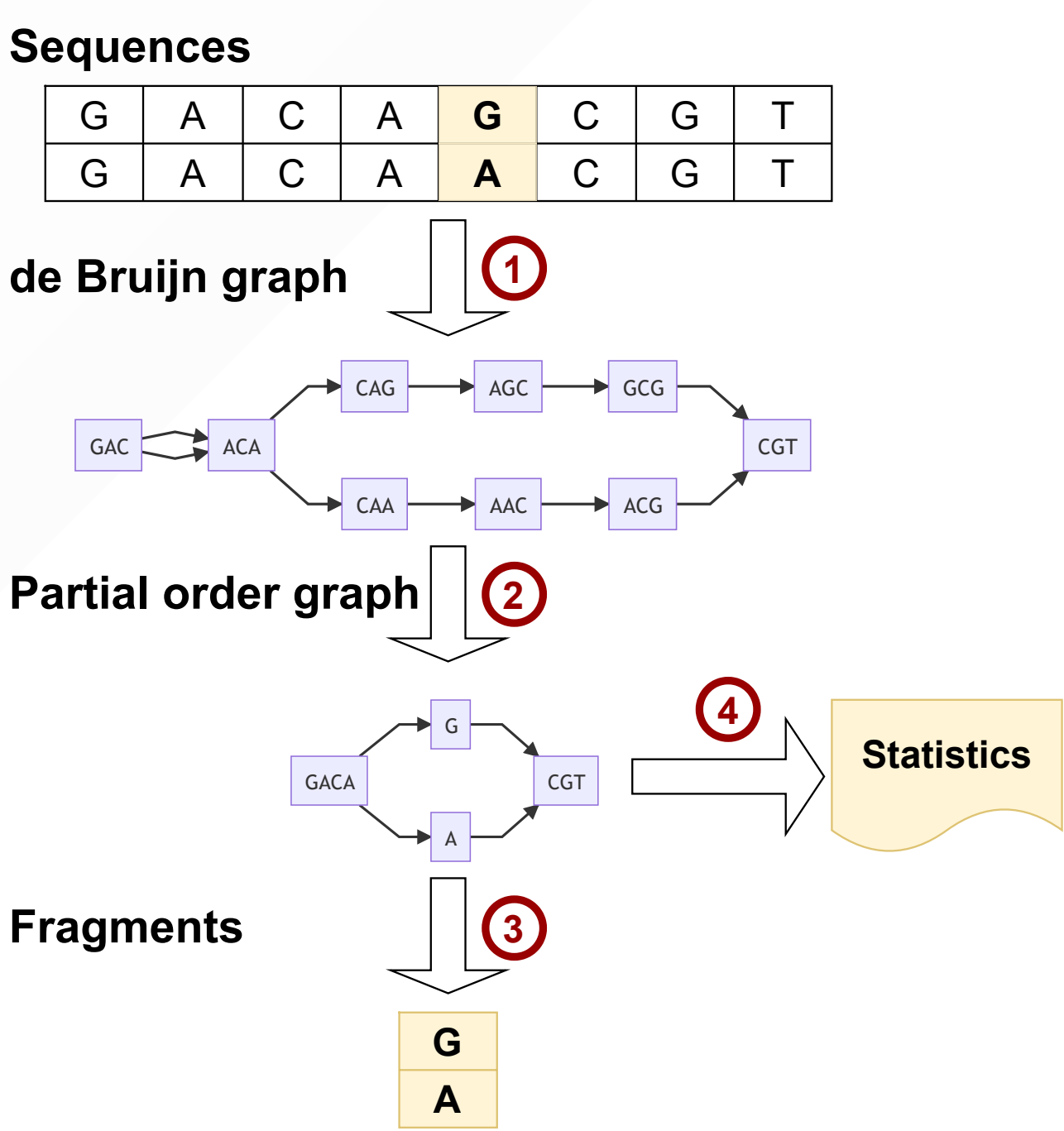
- From first principals, in sequences evolved in an order consistent with data from a progressive tree, **to show that the bubbles in the graph correspond to nodes in a tree** and are similarly ordered
- Using data with known topologies and unambiguous evolution
  - show that the algorithm has **statistical performance** consistent with progressive alignment
  - show that the algorithm has **superior computational performance wrt time and memory** to progressive alignment
- Investigate sequences in species subject to **lateral gene flow** which progressive alignment struggles with

Thanks

- Gavin Huttley
- Vijini Mallawaarachchi
- Yu Lin
- Xinjian Leng

… and the Huttleylab

Questions & Answers



Citations

- Leng, Xingjian (2023), 'Sequence Alignment Using De Bruijn Graphs'. Australian National University
- Needleman & Wunsch (1970), 'A general method applicable to the search for similarities in the amino acid sequence of two proteins' doi.org/10.1016/0022-2836(70)90057-4, 2010
- Whitney, Houtz, and Alonso (2010), 'Advancing Our Understanding and Capacity to Engineer Nature's CO2-Sequestering Enzyme, Rubisco' DOI: 10.1104/pp.110.164814

Sample data sources

- BRCA1_divergent: BRCA1 gene divergent sample of 7 chosen from among 56 mammal species
- BRCA1_hominae: BRCA1 gene from 4 hominae
- SARS-CoV-2: 22 SARS-CoV-2 genomes
- IBD_phage: IBD phage components (https://doi.org/10.1016/j.cell.2015.01.002) |
- Ocean_phage: Tara oceans phage components (https://doi.org/10.1126/science.

Supplementary

> Abandon all hope ye who pass this point
>
> *Tolkein ... probably*

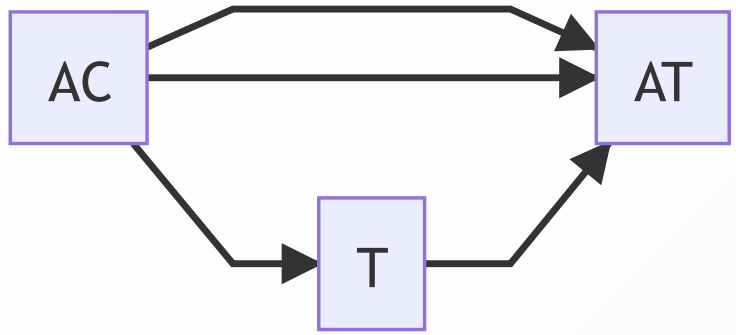
- sample data sources
- Bubbles in real data denote phylogenetic nodes
- unit tests against edge case sequence alignments
  - long sequences
  - numerous sequences
  - cyclic sequences
  - bubbles within bubbles
  - sequential bubbles

Bubbles in real data denote phylogenetic nodes

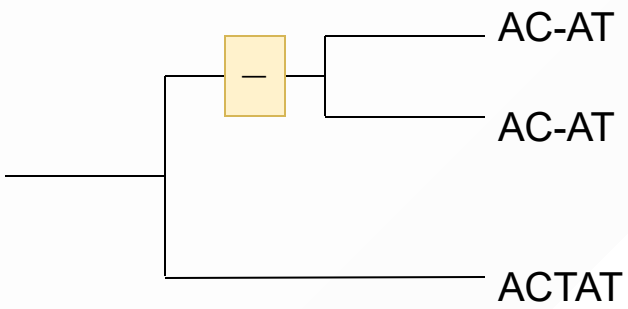
<<Back to Supplementary

We can show in a simple case this is true, but we need to show it is true in general

Partial order graph



Phylogeny



HYPOTHESIS: One side of a bubble is a clade

Unit tests

<<Back to Supplementary

library against edge case sequence alignments
* long sequences
* numerous sequences
* cyclic sequences
* bubbles within bubbles
* sequential bubbles

cyclic sequences

```
def test_pog_cycle(output_dir: Path):
    dbg = dbg_align.DeBrujinGraph(3,cogent3.DNA)
    dbg.add_sequence({
        "seq1": "ACAGTACGGCAT",
        "seq2": "ACAGTACTGGCAT",
        "seq3":"ACAGCGCGCAT" # contains cycle
        })
    with open(output_dir / "cycle.md", "w") as f:
        f.write("```mermaid\n")
        f.write(dbg.to_mermaid())
        f.write("```")
    assert dbg.has_cycles()
    assert len(dbg) == 3
    assert dbg.names() == ["seq1", "seq2", "seq3"]
    assert dbg["seq3"] == "ACAGCGCGCAT" # contains cycle

    dbg.to_pog()
    # write mermaid out to testout folder
    with open(output_dir / "cycle_compressed.md", "w") as f:
        f.write("```mermaid\n")
        f.write(dbg.to_mermaid())
        f.write("```")
```