AMRITA
VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY

22BIO211: Intelligence
of Biological Systems - 2

ALIGNMENT GRAPHS

Dr. Manjusha Nair M
Amrita School of Computing, Amritapuri

Email : manjushanair@am.amrita.edu
Contact No: 9447745519

Alignment Graphs

- To create Alignment Graphs,
  - *Add two arrays of integers to an alignment of ATGTTATA and ATCGTCC.*

```
0   1   2   2   3   4   5   6   7   8
  A   T   -   G   T   T   A   T   A
  A   T   C   G   T   -   C   -   C
0   1   2   3   4   5   5   6   6   7
```

  - *The array [0 1 2 2 3 4 5 6 7 8] holds the number of symbols of ATGTTATA used up to a given column in the alignment.*
  - *The array [0 1 2 3 4 5 5 6 6 7] holds the number of symbols of ATCGTCC used up to a given column in the alignment.*

Alignment Graphs

- We add a third array [↘ ↘ → ↘ ↘ ↓ ↘ ↓ ↘]
  - *whether each column represents a match/mismatch (↘/↘), an insertion (→), or a deletion (↓).*

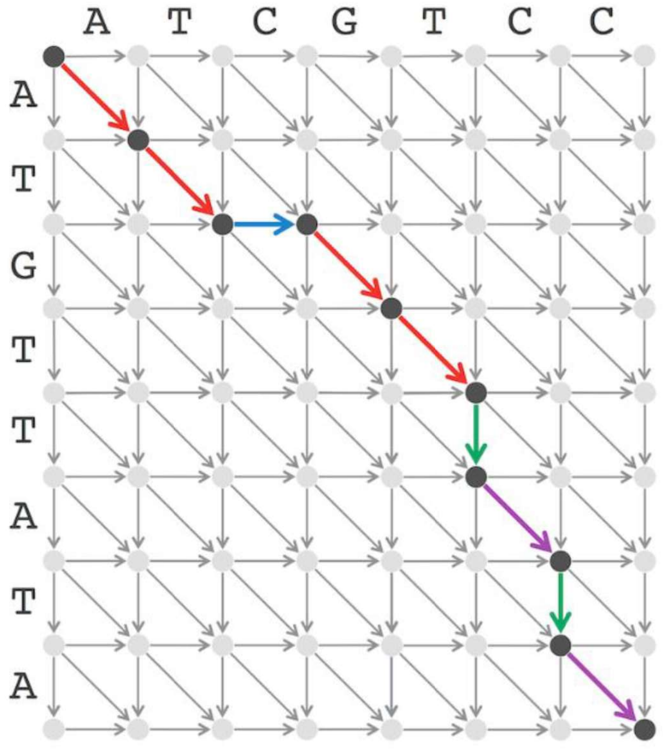
0		1		2		2		3		4		5		6		7		8
	A		T		-		G		T		T		A		T		A	
	A		T		C		G		T		-		C		-		C	
0		1		2		3		4		5		5		6		6		7
	↘		↘		→		↘		↘		↓		↘		↓		↘	

- This third array corresponds to a path from source to sink in an 8 × 7 rectangular grid.
- The i-th node of this path is made up of the i-th element of [0 1 2 2 3 4 5 6 7 8] and the i-th element of [0 1 2 3 4 5 5 6 6 7]:

(0. 0) ↘ (1, 1) ↘ (2, 2) → (2, 3) ↘ (3, 4) ↘ (4, 5) ↓ (5, 5) ↘ (6, 6) ↓ (7, 6) ↘ (8, 7)

Alignment Graphs and Alignment Paths

- This path is shown in the figure below.
  - *Note that in addition to horizontal and vertical edges, we have added diagonal edges connecting (i, j) to (i + 1, j + 1).*
  - *We call this DAG the* ***alignment graph*** *of strings v and w, denoted AlignmentGraph(v, w)*
  - *we call a path from source to sink in this DAG an* ***alignment path.***



(0. 0) ↘ (1, 1) ↘ (2, 2) → (2, 3) ↘ (3, 4) ↘ (4, 5) ↓ (5, 5) ↘ (6, 6) ↓ (7, 6) ↘ (8, 7)

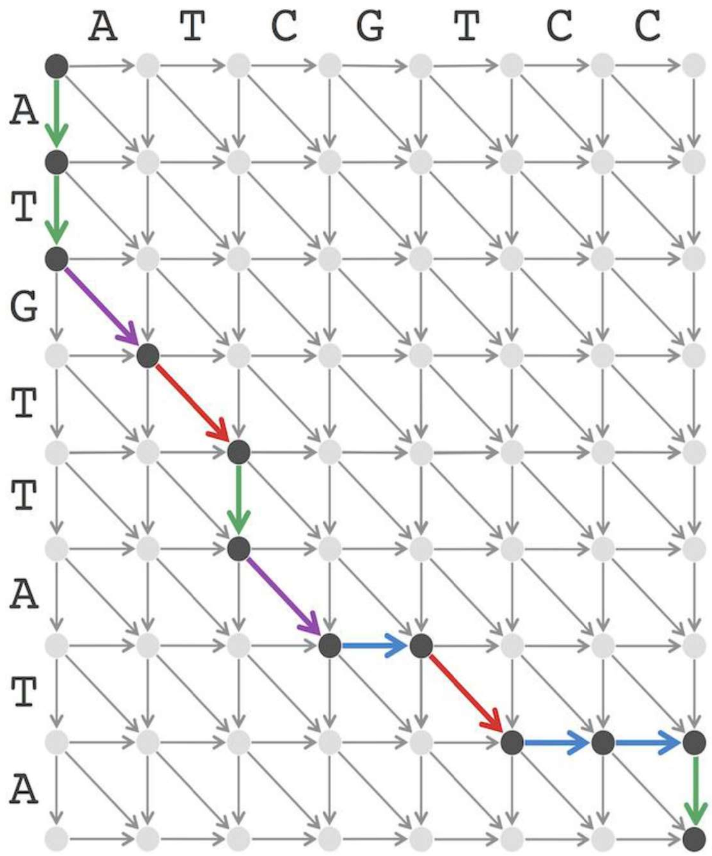
Alignment Graphs

- Every alignment of v and w can be viewed as a set of instructions to construct a unique alignment path in AlignmentGraph(v, w)
  - *where each match/mismatch, insertion, and deletion corresponds to an edge* ↘/↘, →, *and* ↓, *respectively.*

Can we use the alignment graph to find a longest common subsequence of two strings?

Longest Common Subsequence (LCS)

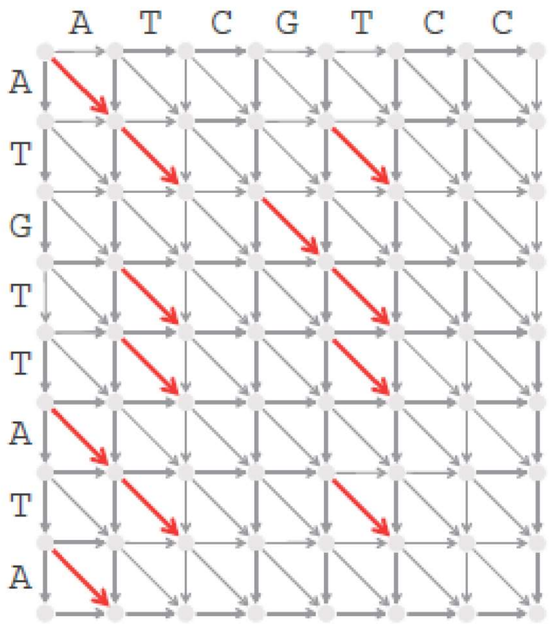
- Consider another alignment of string (ATGTTATA, ATCGTCC) .
- Finding a longest common subsequence of two strings is equivalent to finding an alignment of these strings maximizing the number of matches.



Longest Common Subsequence (LCS)

- If we assign a weight of 1 to all diagonal matching edges and 0 to all other edges, then the LCS Problem is equivalent to finding a longest path in this weighted DAG!
- In the given figure, the red edges correspond to potential matched symbols in an alignment of the two strings.

We need to design an algorithm for the Longest Path in a DAG Problem



AlignmentGraph(ATGTTATA, ATCGTCC) with all edges of weight 1 colored red (all other edges have weight 0).

Summary

- Alignment Graphs
- Alignment Paths
- Longest Common Subsequence (LCS)