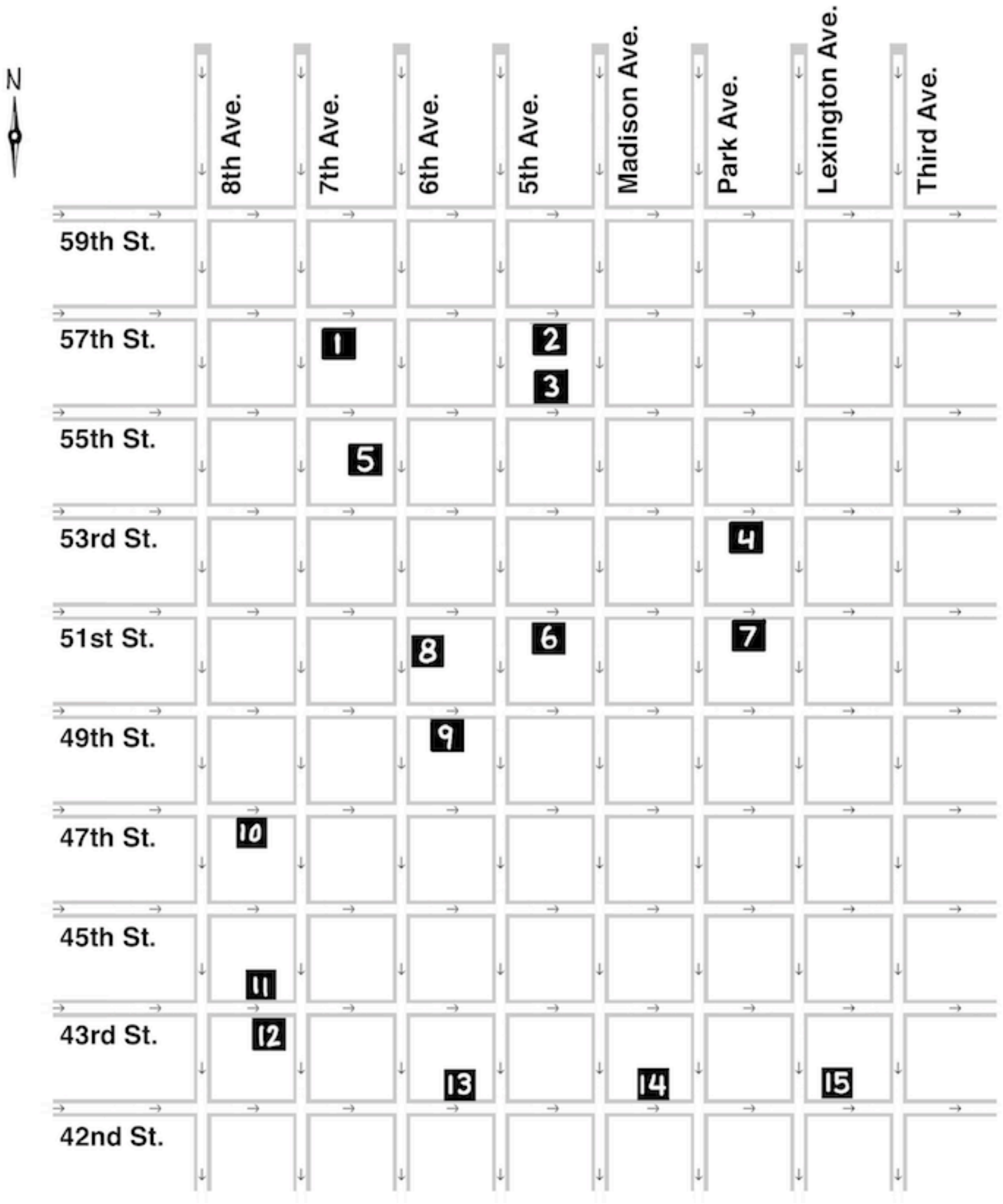
CMSC 423:
Sequence Alignment

Part 2

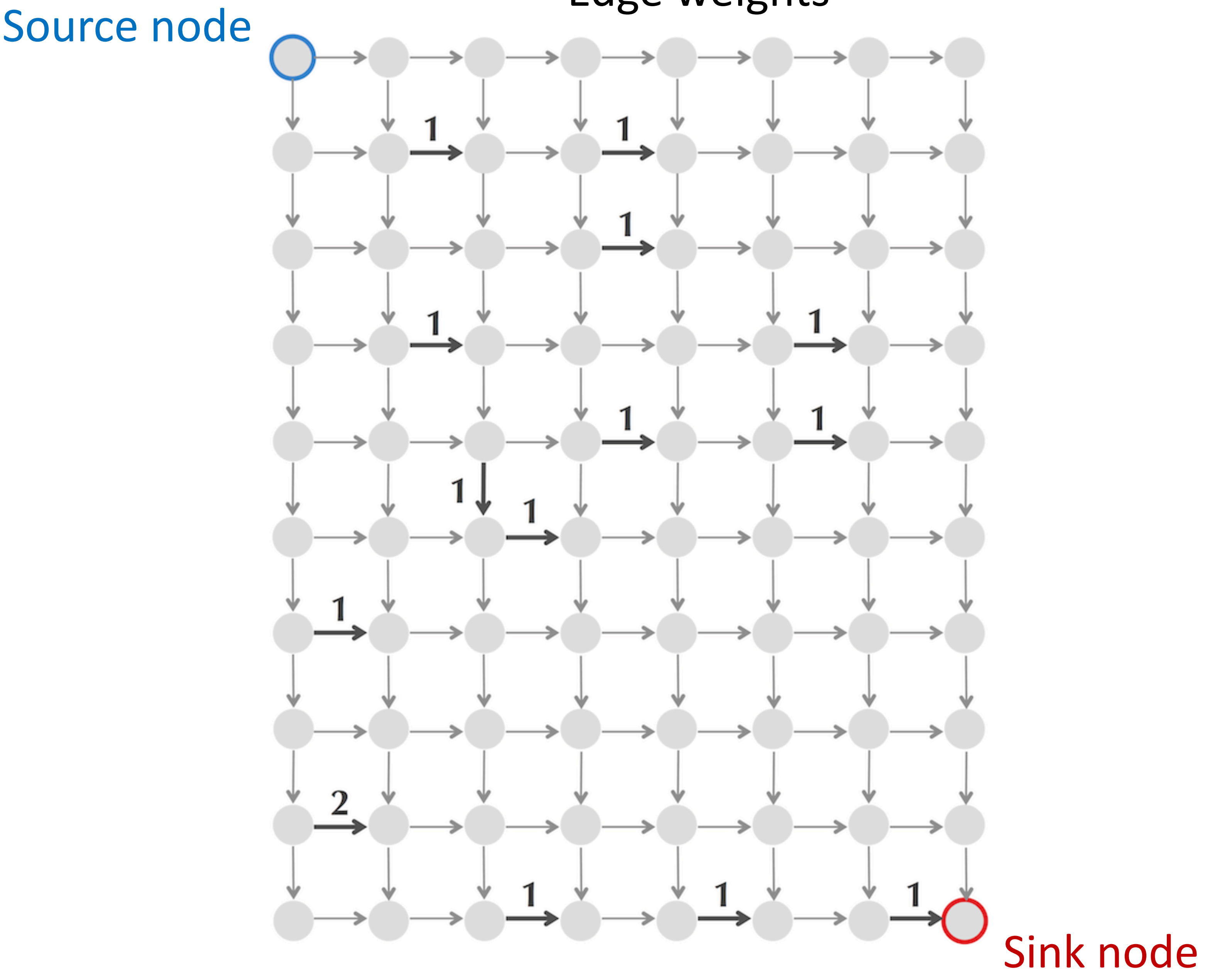
Longest Common Subsequence (LCS)

- An alignment of two string maximizing the number of matches corresponds to the longest common subsequence
- Two strings can have more than one longest common subsequences
- How do we solve this?

LCS is similar to the Manhattan Tourist Problem



Edge weights



Manhattan Tourist Problem: *Find a longest path in a rectangular city.*

- **Input**: A weighted $n \times m$ rectangular grid with $n + 1$ rows and $m + 1$ columns.
- **Output**: A longest path from source (0,0) to sink (n, m) in the grid.

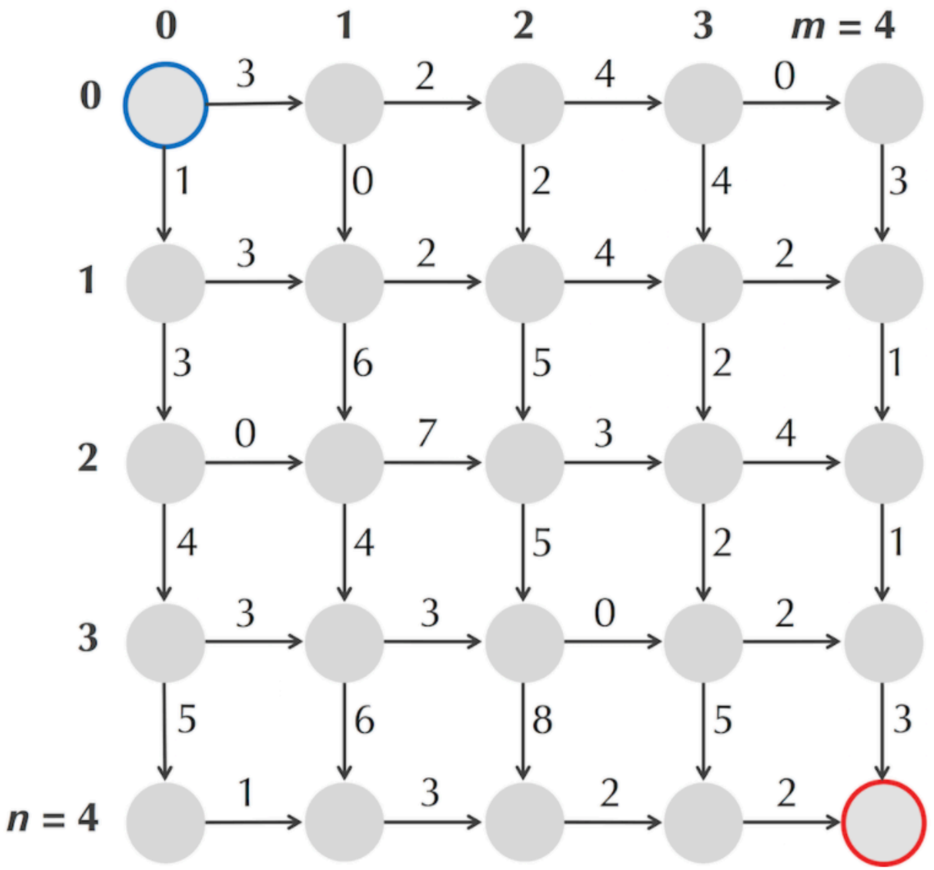
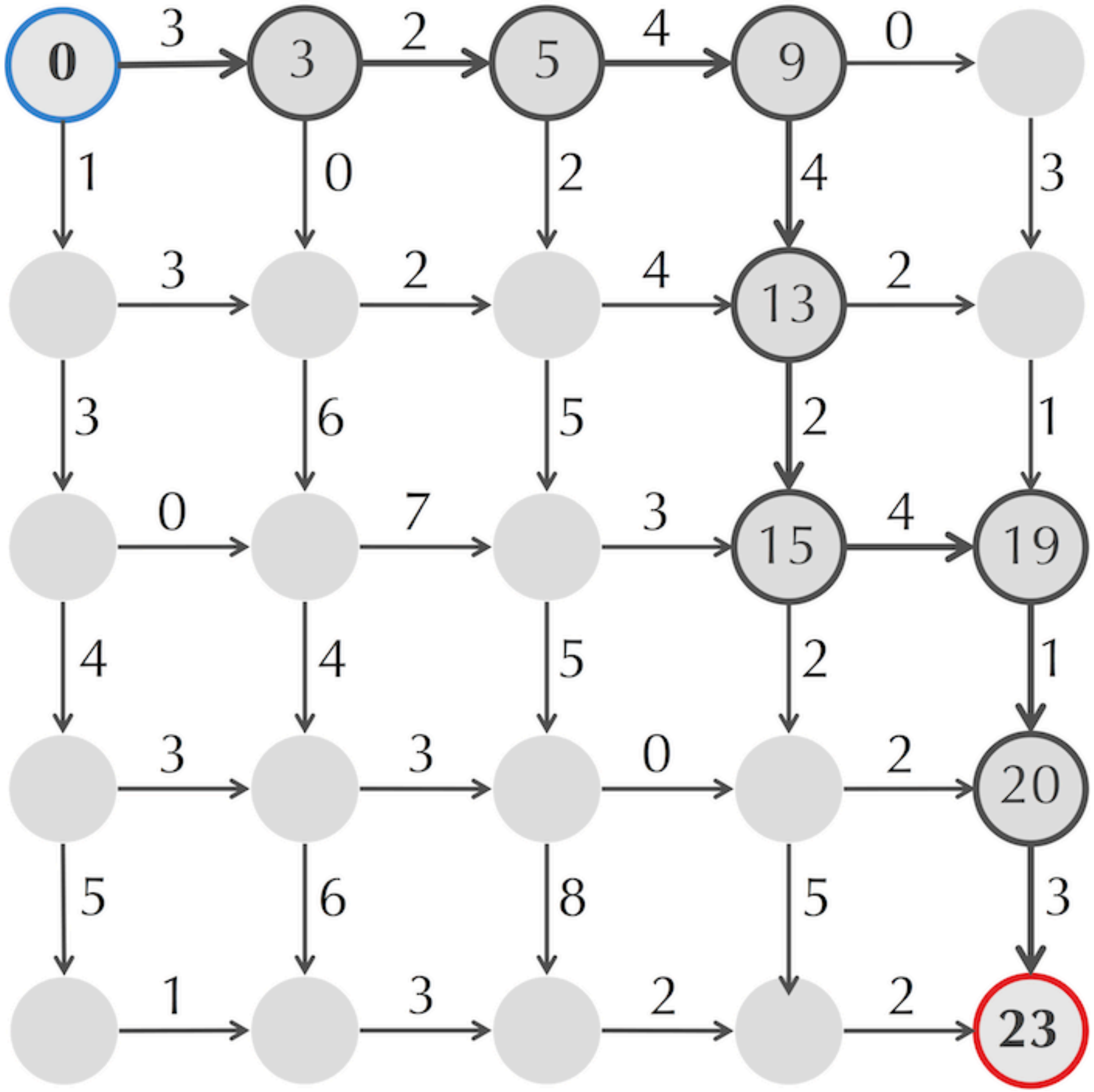
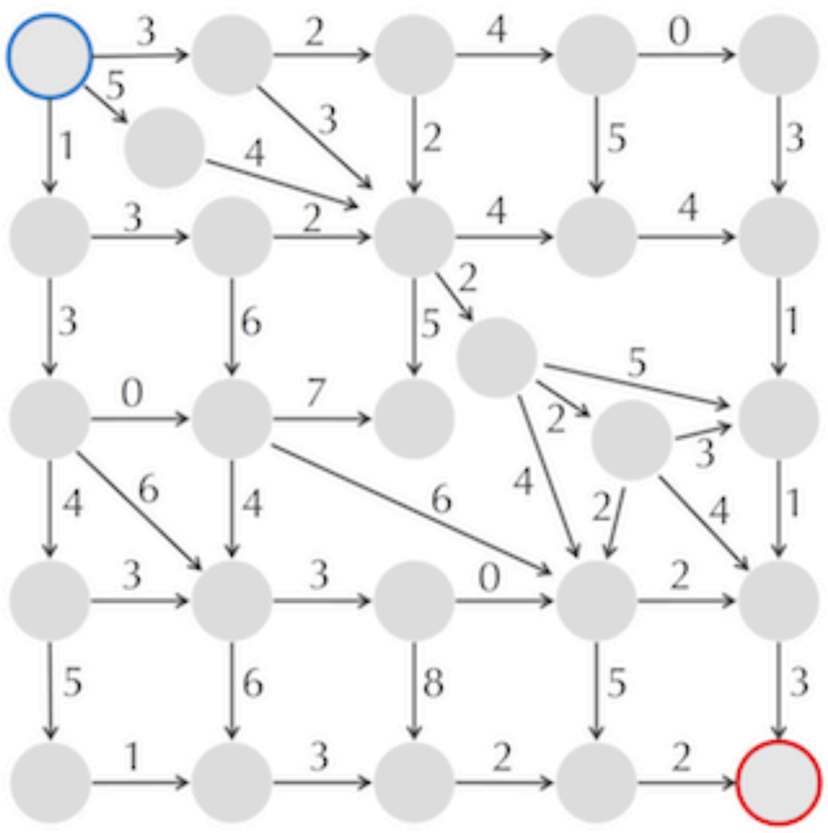


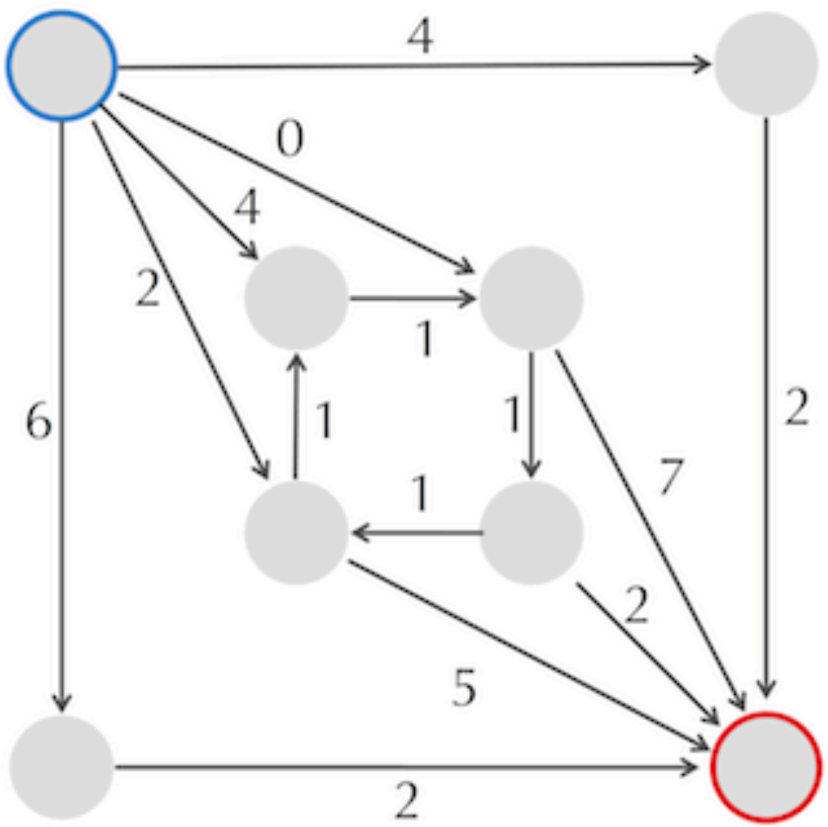
Figure: An $n \times m$ city grid represented as a graph with weighted edges for $n = m = 4$. The bottom left node is indexed as (4, 0), and the upper right node is indexed as (0, 4).

Greedy strategy doesn't guarantee longest path



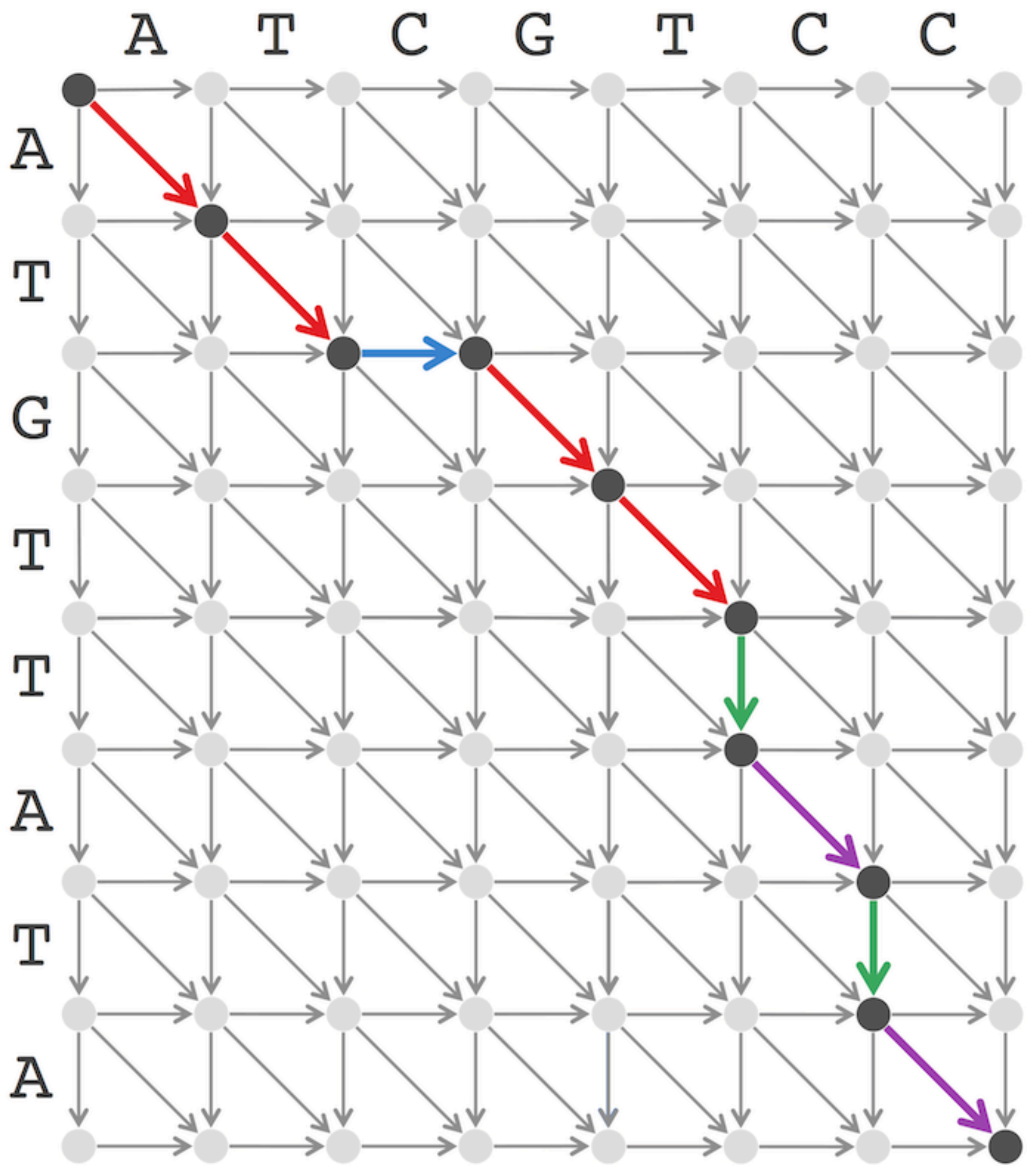
Cycles could be traversed indefinitely





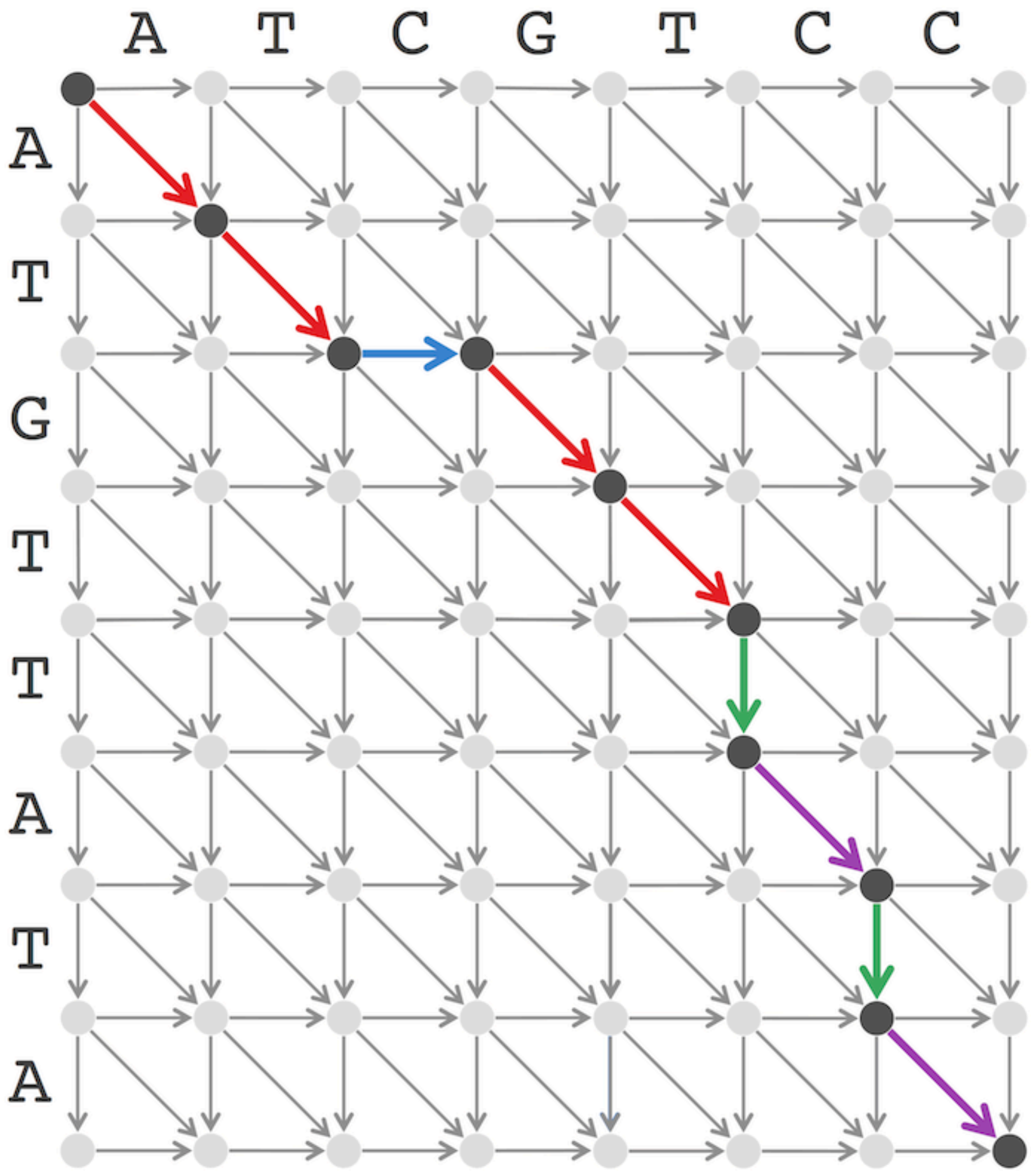
We are using Directed Acyclic Graphs (DAGs)

Sequence Alignment is the Manhattan Tourist Problem in Disguise



STOP and Think

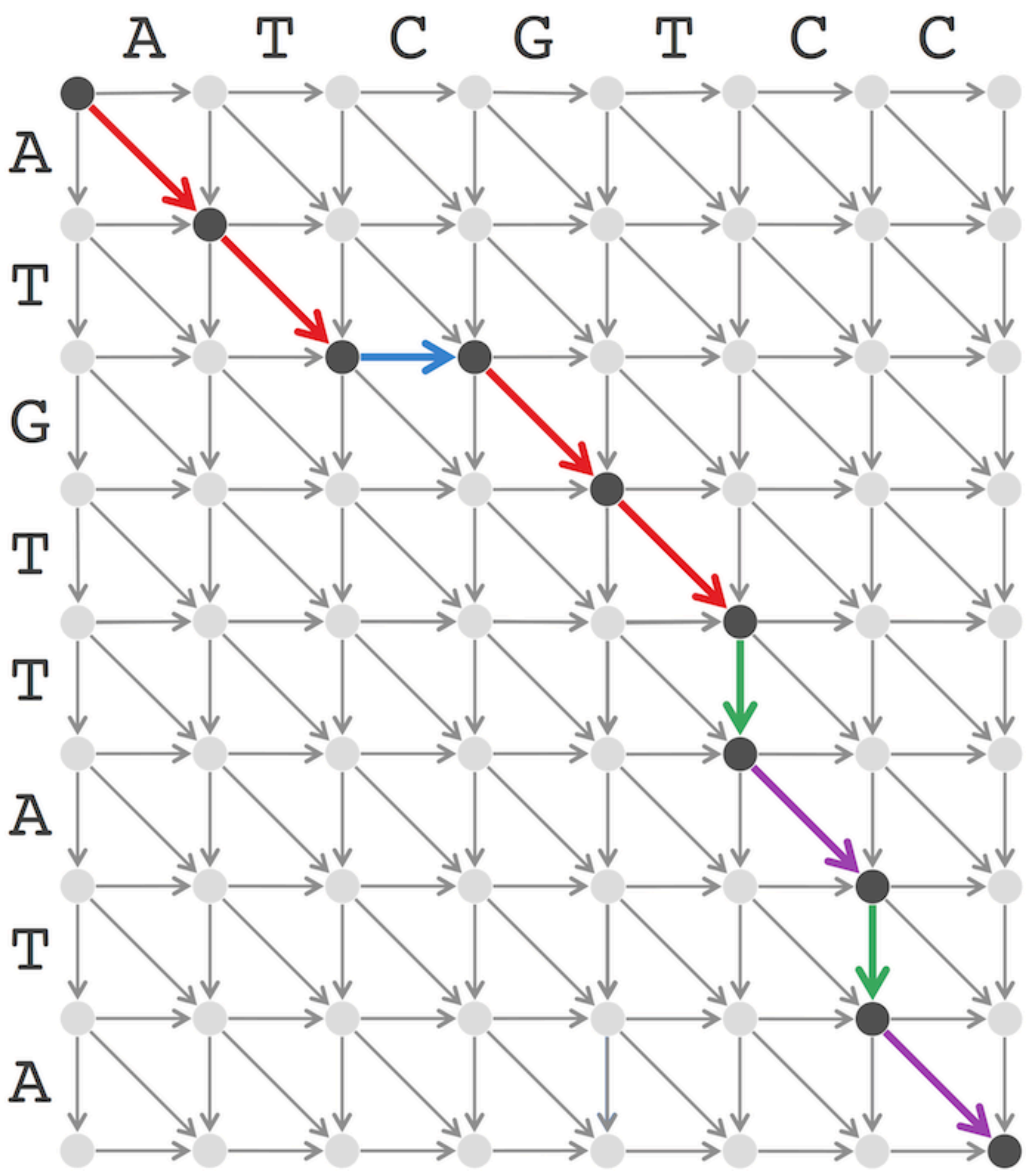
What alignment is produced by this alignment graph?



and Think

What alignment is produced by this alignment graph?

```
AT-GTTATA
ATCGT-C-C
```



and Think

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

and Think

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

Yes, with dynamic programming!