**ECE374 Assignment 5**

Due 03/20/2023

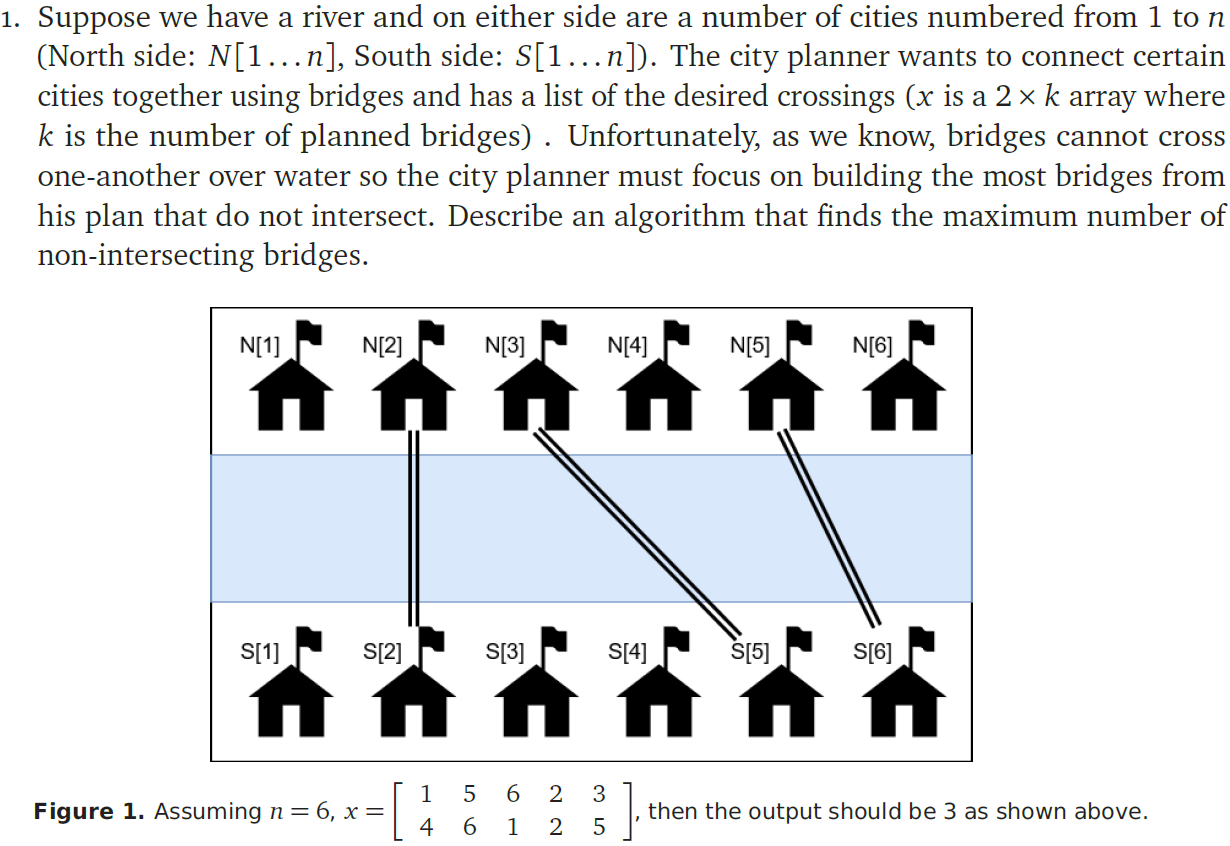
**Group & netid**

*Chen Si chensi3*

*Jie Wang jiew5*

*Shitian Yang sy39*

**Problem 1**



Solution:

Intuition:

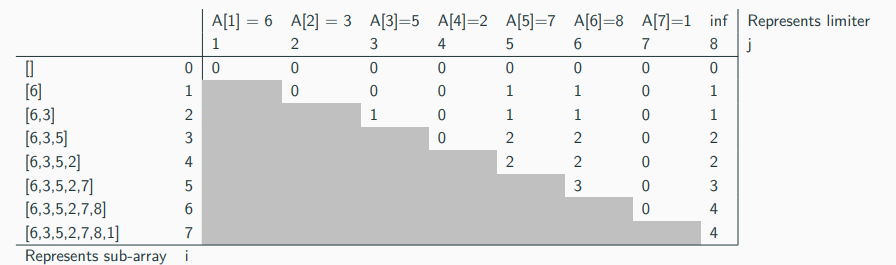
(1) We could solve this problem by the following steps:

i. sort x by the first row. e.g.

ii. then, to avoid bridge crossing, we build the bridges from left to right, and each column of a bridge must have the second-row number larger than the last one before building. Therefore, we could find the number of the longest increasing subsequence (LIS) of the second row with backtracking skills that would be the max number of bridges if we consider in the first-row perspective.

(2) In the LIS algorithm we have the recurrence relation as

(3) Therefore, we could use a n×(n+1) table A to store the intermediate values, with form



, and the value that we would like to find as result is A[n, n+1].

(4) As LIS algorithm with dynamic programming has run time O(n^2), our algorithm is O(logn + n^2) = O(n^2).

Therefore, we could describe the algorithm as

**BuildBridge**(x[1…N,1…N],n):

sortedMatrix = **sort**(x, row=1)

count = **LIS**(sortedMatrix[2,:])

return count

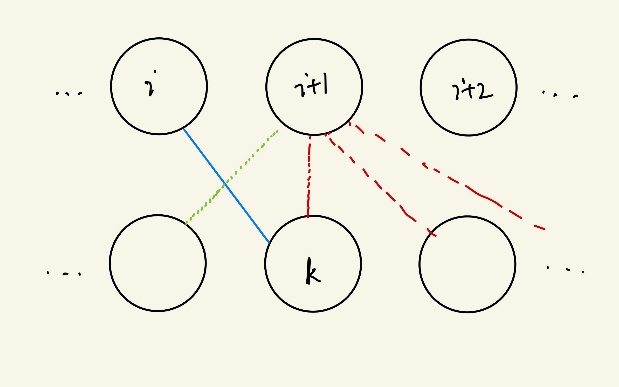
Some Explanations:

1. Why soring the rows at first?

Answer: Sorting the rows would allow us to determine which next bridge are we discussing on whether we should add it or not in the dynamic programming process, enabling a “sequence” to appear within a set of unordered bridges.

2. Why using LIS?

Answer:

When building bridges, take the left chart as an example. If we previously built a bridge linking the ith node to the kth node (shown in blue line), we then could only build bridges that starts from node i+1 and goes to nodes k and nodes to the right of k (node >= k), as shown with red lines. We can’t connect node i+1 with any node less than k below, as shown with green lines. Therefore, we are actually requiring a non-decreasing subsequence in the row below, and the maximum number of bridges is the length of this non-decreasing subsequence.