Homework 13

Adam Karl

November 17, 2020

1 Problem 41 (8 points)

1.1 Motivation

During class on Monday 11/16, we created a CREW PRAM algorithm that solves shortest path in $lg^2(n)$ time using $n^3/lg(n)$ processors. Although I will reference this algorithm and use it in my solution, I will not be re-explaining it.

With this in mind, our goal is to reduce Longest Common Subsequence (LCS) $\leq$ ShortestPath such that the input and output transformations are both $O(lg^2(n))$ with a polynomial number of processors. If this is possible (and we have at least $n^3/lg(n)$ processors as is needed to solve ShortestPath in $O(lg^2(n))$ time), then we can combine the reduction with our solution from class in order to solve LCS in $O(lg^2(n))$ time.

1.2 Transformation Description

To solve LCS(a,b) using an algorithm for ShortestPath, first:

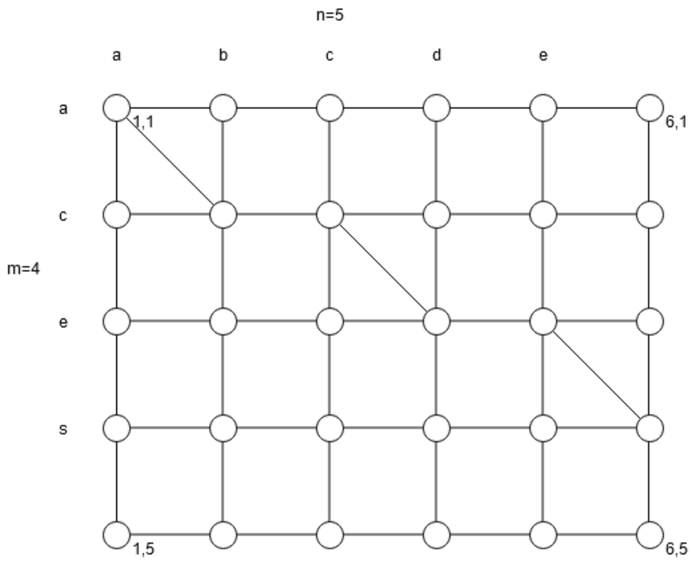
- Let sequences a and b have lengths n and m, such that n>m.
- Assume at least nm processors.

then:

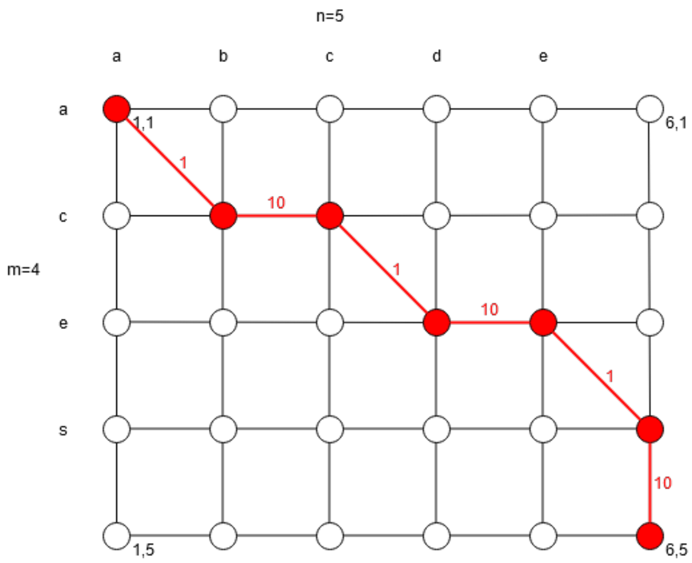
- Create G with nm vertices, labeled $v_{1\to n+1, 1\to m+1}$
  - (constant time with each of nm processors creating precisely 1 vertex)
- Add edges from each vertex to the one with a higher first coordinate, if any (e.g. from $V_{i,j}$ to $V_{i+1,j}$) with a weight of L
  - Choose L such that it is larger than n. This will make it easy to extract the LCS solution from what ShortestPath returns
  - (again, constant time since 1 processor per vertex, each processor adds 1 edge)
- Add edges from each vertex to the one with a higher second coordinate, if any (e.g. from $V_{i,j}$ to $V_{i,j+1}$) with a weight of L
  - (again, constant time since 1 processor per vertex, each processor adds 1 edge)
- At each vertex $V_{i,j}$, compare characters a[i] to b[j] and create an edge to $V_{i+1,j+1}$ if a[i] = b[j]
  - (again, constant time since 1 processor per vertex, each processor makes 2 reads (unimpeded since CR), then creates 0 or 1 edges)
- Run the algorithm for ShortestPath on G with start point $V_{1,1}$ and endpoint $V_{n+1,m+1}$ and return what it returns mod L.

1.3 Visualization

If LCS is run on "abcde" and "aces", then the corresponding graph would look like:



where each horizontal and vertical edge has a weight of, for instance 10, and each diagonal has a weight of 1. We then run Shortest path on the graph using $V_{1,1}$ (top left) and $V_{6,5}$ (bottom right) as the endpoints. On this input, Shortest path would return 33 as shown:



and LCS would return 3 (33 mod 10).

LCS returns precisely the number of "diagonals" taken in the ShortestPath grid, which corresponds to the number of characters in the longest common subsequence.

1.4 LCS $\leq$ ShortestPath

With at least n^2 processors, we have shown that the input and output transformations from LCS to ShortestPath are constant time, and the result from ShortestPath can correctly solve LCS. Therefore, with at least n^2 processors, LCS $\leq_{constanttime}$ ShortestPath. Since our ShortestPath algorithm from class runs in $lg^2(n)$ time using $n^3/lg(n)$ processors, we can utilize that algorithm along with the described input/output transformations to solve LCS in $lg^2(n)$ time using $n^3/lg(n)$ processors.