Soham Chatterjee
Email: sohamc@cmi.ac.in
Course: Parallel Algorithms and Complexity

Assignment - 1
Roll: BMC202175
Date: August 16, 2023

Problem 1

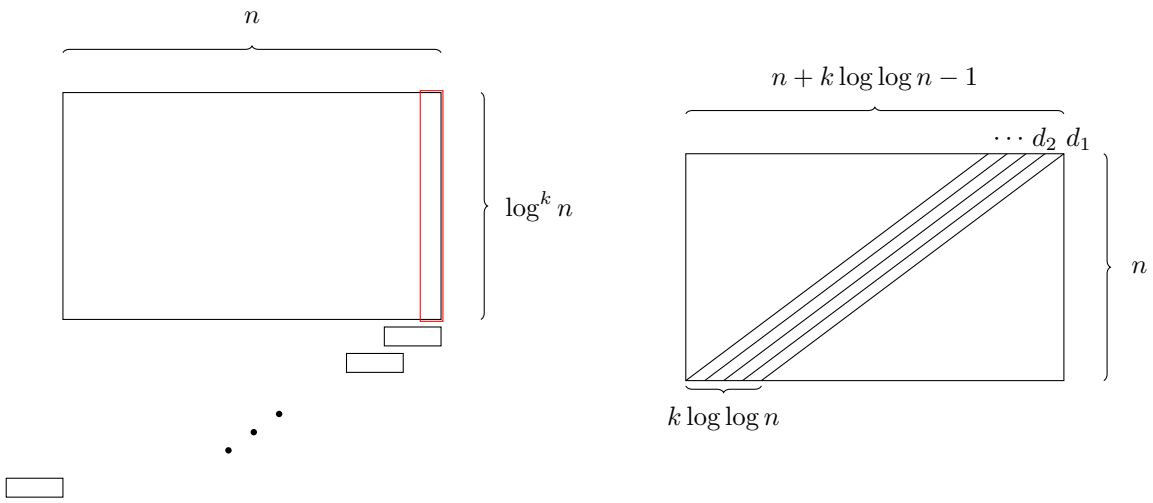
$ITERADD_{poly(\log n),n} \in AC^0$ where $ITERADD_{poly(\log n),n}$ is problem of adding polylog many n-bit numbers.

Solution: We already know $ITERADD_{2,n} \in AC^0$. So we will use this in order to show

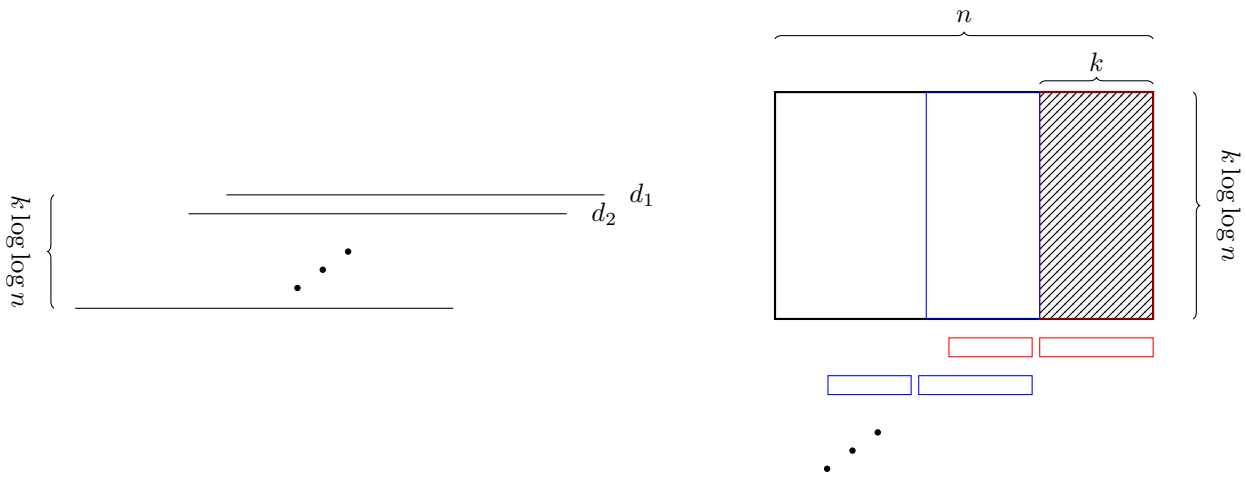
$$ITERADD_{poly(\log),n} \in AC^0$$

Suppose we are given $\log^k n$ many n-bit numbers. Let the numbers are $a_1, \ldots, a_{\log^k n}$ where $a_i = a_{i,n-1} \cdots a_{i,0}$ where $a_{i,j} \in \{0, 1\}$ for all $1 \leq i \leq n$, $0 \leq j \leq n-1$.

Now consider $s_j = \sum_{i=1}^{\log^k n} a_{i,j}$ $\forall$ $0 \leq j \leq n$. It is the sum of all the bits in the i-th column. Since we are adding $\log^k n$ many bits s_j will have $k \log\log n$ many bits.



Consider the diagonals after adding the bits in a column. Since each s_j has $k \log\log n$ many bits there are total $k \log\log n$ many diagonals and since there are total n columns each diagonal has n bits. Lay them out horizontally but keeping the lowermost diagonal at the topmost then the 2nd lower diagonal at the 2nd top position and continue like that. But we have to shift the kth diagonal one place left from the leftmost position of the $k-1$th diagonal.



Now since there are $k \log\log n$ bits in each column in this new arrangement. We divide the arrangement into blocks of k columns. Now we create truth tables for all possible summations and we can do this because since there are n numbers after summing k bits of n numbers there will be $k \log n$ bits.

Now for each such k length block there are two numbers. Ther left part is the sum and the right part is the carry. The sum and carry blocks are positioned in such a way that ith sum is added only with the $i-1$th carry. So now in each k length block we are adding two numbers. This as we stated at the start ie $ITERADD_{2,n} \in AC^0$. So we have shown that $ITERADD_{poly(\log),n} \in AC^0$

□

Problem 2

Defintiion 1: It is the class of constant depth polynomial size circuits with AND, OR, NOT, MAJORITY gates.
Defintiion 2: It is the class of constant depth polynomial size circuits with NOT, MAJORITY gates.

Equivalence of The two Definitions of TC^0

Solution: If we can construct the AND and OR gate with MAJORITY gate in constant depth and polynomial size then we can say these two definitions are equivalent.

Now if $a_0, \ldots, a_{n-1}$ are n bits. Then

$$\bigwedge_{i=0}^{n-1} a_i = MAJORITY\Big(a_0, \ldots, a_{n-1}, \underbrace{0, 0, \ldots, 0}_{n-1 \text{ times}}\Big)$$

Similarly

$$\bigvee_{i=0}^{n-1} a_i = MAJORITY\Big(a_0, \ldots, a_{n-1}, \underbrace{1, 1, \ldots, 1}_{n-1 \text{ times}}\Big)$$

Now the AND and OR gate both can be replaced with $O(1)$ depth $O(n)$ size subcircuits using only MAJORITY gates in a circuit and still the total depth of the circuit will be constant and size is polynomial. Hence the two definitions are equivalent.

□