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

Assignment - 2
Roll: BMC202175
Date: August 18, 2023

Problem 1

Prove that in NC^1 circuits and formulas are equivalent.

Solution: Let $C \in NC^1$. Let depth of C is denoted by d and size is denoted by s. Since the circuit is in NC^1 we can assume the circuit has fanin 2. We will show that we can fit the circuit C into a complete binary tree of depth d which has size 2^d.

We will unveil the circuit from the top and go towards bottom. We assume that till depth i we have filled the binary tree with depth $i-1$ and every gate with depth from 1 to i have fanout 1. For any depth level i if for any k many gates there is a common child gate then there are k many child gates are less at depth $i+1$. We make k many copies of that child gate, one for each gate at depth level i and each of that child gate remains at depth $i+1$, so that each child gate has fanout 1. With this at max at depth $i+1$ we have 2^{i+1} gates. Now since every gate has fanin 2 even we made k copies of the same gate all the gates under that gate in the subcircuit rooted at gate is copied at depth j number of gates can not grow more than 2^j. Now We have converted every gate at depth $i+1$ fanout 1 and till depth $i+1$ it fits into the binary tree with depth $i+1$.

We continue this and go downwards and after the process the whole circuit is converted to a formula which has at most the structure of complete binary tree with depth d. So the size of the formula is $O(2^d)$. Since $d = O(\log n)$ we have size of the formula $O(2^{O(\log n)}) = O(n^{O(1)})$ which is $poly(n)$. So in NC^1 the circuits and formulas are equivalent.

$\square$

Problem 2

Prove that $TC^0 \subseteq NC^1$ using Redundant Algebra

Solution: We will first shoe using Redundant Algebra $ADD_{2n} \in NC^0$. In Redundant Algebra with base 4 while adding two digits the result can be at most in normal integers $-6, \dots, 6$. We only have to check for $\pm 6, \pm 5, \pm 3$. Other case we dont have to watch out.

$$6 = 12 \qquad\qquad 5 = 11 \qquad\qquad 3 = 1\overline{1}$$
$$\overline{6} = \overline{12} \qquad\qquad \overline{5} = \overline{11} \qquad\qquad \overline{3} = \overline{1}1$$

For ± 4 it is the normal representation in this system ie 10 and $\overline{1}0$ respectively. Now whenever we add two digits in this system in the sum result can be at most 2 digits. Among them we call the right most digit the sum digit or s and the left digit the carry digit c becasue it becomes the carry for the addition. Now see in all of the numbers the sum digit $|s| \leq 2$ and carry digit $|c| \leq 1$ So whenever we add a carry generated before to the current sum no new carry is generated becasue of the carry. So We dont have to look for carry generation and propagation. The carry generated at the previous position will add to the sum digit in the current position after getting added to that it will not further propagate after getting added the final digit at the current place will be between 3 and $\overline{3}$ So we proved that addition of two n-bit numbers using Redundant algebra is in NC^0

Now we will show converting a number n from base 2 to base 4 is in NC^0. let

$$n = \sum_{k=0}^{m} a_k 2^k = \sum_{k=0}^{\lfloor \frac{m}{2} \rfloor} (a_{2k+1} \times 2 + a_{2k})\, 4^k \qquad \text{if } m \text{ is odd then } a_{m+1} = 0$$

So we just take two bits in binary multiply the left one with 2 and add to the right one and we have the digit at base 4 in that position. So we can change base in NC^0. From now on we will by default assume all the addition is done using Redundant Algebra.

Then we are done in showing $TC^0 \subseteq NC^1$. We will first show that adding n bits is in NC^1 ie $BCOUNT \in NC^1$. So we have n bits $a_n, a_{n-1}, \ldots, a_1$. We will group the bits into groups of two bits and add the two bits in each group. We can do the addition for all groups in parallel. And since we have already shown addition of two k-bit numbers is in NC^0 this takes constant depth and size since we are adding two bits. Now the number of bits are halved. We do tha same process again and again. Each time it takes $poly(n)$ size and constant depth and at each iteration the number of numbers becomes half. So this whole process takes $O(\log n)$ many iterations. So adding n bits takes $poly(n)$ size and depth $O(\log n)$. So $BCOUNT \in NC^1$

Now We will show that $MAJORITY \in NC^1$. Let the addition of n bits we got is a and the Redundant Algebra representation of $\lceil \frac{n}{2} \rceil$ is b. Now we can calculate $-b$ with reversing the sign of every digit in b ie if $b = \sum_{i=0}^{m} a_k \times 4^k$ then $-b = \sum_{i=0}^{m} \overline{a_k} \times 4^k$. Now we add a and $-b$ which can be done in NC^0. And now we will look for the left most negative digit. If there is nonw we output 1 i.e. majority of the bits are 1 and if there exists a negative bit then output is 0. Hence we have $MAJORITY \in TC^0$. So $TC^0 \subseteq NC^1$.

$\square$