Math 262A Lecture Notes
Counting, Threshold, Vector Addition

Lecturer: Samuel Buss
Scribe: Udbhav Singh

January 15, 2014

Let $f_n = \{f_n\}_n$ be an n-ary function.

Definition 1 (AC^0)**.** *$f \in AC^0 \iff f_n$ has poly size, constant depth, unbounded fan in, $\vee, \wedge, X, \bar{X}$ circuits.*

Definition 2 (NC^1)**.** *$f \in NC^1 \iff \forall n, f_n$ has poly size, $\mathcal{O}(\log n)$ depth, fan in 2, $\vee, \wedge, X, \bar{X}$ circuits.*

Definition 3 (TC^0)**.** *$f \in TC^0 \iff \forall n, f_n$ has poly size, constant depth, unbounded fan in, $\vee, \wedge, Majority, X, \bar{X}$ circuits.*

Also we have
NC - Nick's Class
SC - Steve's Class
SC^k - simultaneously polynomial time and log space.

When defining circuit size, we count the number of wires or edges in the circuit (because a symbol can be fed multiple times).
Recall

$$Majority^n(X_1, \ldots, X_n) = \begin{cases} 1 & \text{if } \sum_{i=1}^{n} X_i \geq n/2 \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$Threshold_k^n(X_1, \ldots, X_n) = \begin{cases} 1 & \text{if } \sum_{i=1}^{n} X_i \geq k \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$Exact_k^n(X_1, \ldots, X_n) = \begin{cases} 1 & \text{if } \sum_{i=1}^{n} X_i = k \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

All the above three functions are examples of symmetric functions.

Definition 4 (Symmetric Functions). *A function $f(X_1, \ldots, X_n)$ is symmetric if $f(X_1, \ldots, X_n) = g(\sum_{i=1}^{n} X_i)$ for some function g.*

The definition implies that order doesn't matter in symmetric functions which is the source of the name symmetric.

Proposition 1. *$Threshold_k^n$ can be expressed with a TC^0 circuit.*

Proof. Trivially $Majority^n(X_1, \ldots, X_n) = Threshold^n_{\lceil \frac{n}{2} \rceil}(X_1, \ldots, x_n)$. In the other direction, we can write

$$Threshold_k^n(X_1, \ldots, X_n) = \begin{cases} Majority^{2k}(X_1, \ldots, X_n, 0, \ldots, 0) & \text{if } k \geq n/2 \\ Majority^{2(n-k)}(X_1, \ldots, X_n, 1, \ldots, 1) & \text{if } k \leq n/2 \end{cases}$$

We could equivalently define other TC^0 gates using $Threshold_k^n$ gates.

$$Exact_k^n(X_1, \ldots, X_n) \equiv Threshold_k^n(X_1, \ldots, X_n) \wedge \neg Threshold_{k+1}^n(X_1, \ldots, X_n)$$
$$Threshold_k^n(X_1, \ldots, X_n) \equiv \vee_{j=k}^{n} Exact_j^n(X_1, \ldots, X_n)$$

Hence all these gates can be defined in terms of each other. □

Theorem 1. *$Majority^n$ is in NC^1. thus $TC^0 \subseteq NC^1$*

Part 1: Proof of conclusion. We will first prove the conclusion that if $Majority^n$ is in NC^1, then $TC^0 \subseteq NC^1$.
We will convert a given circuit C of fixed depth d, of size $S = n^{\mathcal{O}(1)}$ using $\vee, \wedge, Majority$ gates into an NC^1 circuit.
Consider a particular gate g in C. g is either $\vee, \wedge$ or $Majority$. g has less than S many inputs. If g is $\vee$ or $\wedge$, we can "balance" it by replacing it by a tree of fan in 2 $\vee, \wedge$s of depth $\mathcal{O}(\log S) = \mathcal{O}(\log n)$
If g is $Majority^m, m = n^{\mathcal{O}(1)}$, replace it by a NC^1 circuit for $Majority^m$.
The overall circuit is thus $\mathcal{O}(d \log n) = \mathcal{O}(\log n)$ (d is a constant) □

Before showing that $Majority^n$ is in NC^1, we will present a few definitions and lemmas to guide us with the proof.

Definition 5. *The function $Addition^m(X_0, \ldots, X_{m-1}, Y_0, \ldots, Y_{m-1}) = Z_0, \ldots, Z_m$ defines the binary addition operation on two m-digit binary numbers and outputs their sum in binary i.e $(Z_0, \ldots, Z_m)_2 = (X_0, \ldots, X_{m-1})_2 + (Y_0, \ldots, Y_{m-1})_2$ or $(\vec{Z})_2 = (\vec{X})_2 + (\vec{Y})_2$ in vector notation.*

We can define the vector addition function analogously.

Definition 6. *The function $VecAdd^m(X_{00}, \ldots, X_{0,l-1}, X_{10}, \ldots, X_{1,l-1}, \ldots, X_{m-1,l-1}) = Z_0, \ldots, Z_{l+\log m}$ performs binary addition of m many l bit numbers and gives their result in binary.*

Lemma 1. *$Addition^m$ has circuits of size $\mathcal{O}(m)$ and depth $\mathcal{O}(m)$*

Proof. We can make use of Half-Adders ad Full-Adders for building such circuits. Each of these have size $\mathcal{O}(1)$ and depth $\mathcal{O}(1)$. to add we simply build the following circuit. □

Lemma 2. *$Addition^m \in AC^0$ (and hence in NC^1)*

Proof. We can modify the circuit of the previous lemma to get a constant depth circuit. The only problem is the carry bit which *ripples* through the circuit to give linear depth. We can instead get the carry bit at each stage by using a circuit of depth 3 to get to constant depth circuit. This is called carry lookahead. We can define the ith carry bit to be

$$C_i = \bigvee_{j=0}^{i} \left(\left(X_j \wedge Y_j \right) \wedge \left(\bigwedge_{k=j+1}^{i} \left(X_k \vee Y_k \right) \right) \right)$$

Using the carry bit from this circuit at each stage, we get a circuit for $Addition^m$ of depth 3. Thus $Addition^3$ is in AC^0. □

Lemma 3. $VecAdd^m \in NC^1$

Proof. We use carry save addition [Ofman '62][Wallace '64]
A carry save operation on three numbers $(\vec{X})_2, (\vec{Y})_2, (\vec{Z}))_2$ gives two numbers $(\vec{U})_2, (\vec{V})_2$ such that $(\vec{X})_2 + (\vec{Y}_2 + (\vec{Z}))_2 = (\vec{U})_2 + (\vec{V})_2$. In function form, this is written as

$$CSA^m(X_0, \dots, X_{m-1}, Y_0, \dots, Y_{m-1}, Z_0, \dots, Z_{m-1}) = U_0 \dots U_{m-1}, V_0 \dots V_{m-1} \quad (4)$$

The implementation of the function is as follows.
For an NC^1 circuit for $VecAdd_l^m$, we apply carry save addition operations in parallel (in rounds). There are $\lceil \log_{3/2} m \rceil$ many rounds each of depth $\mathcal{O}(1)$ and at the end we get two numbers $(\vec{U})_2$ and $(\vec{V})_2$ which we can sum using AC^0 circuits. □

Remark: This suffices to prove that $Majority^n \in NC^1$ and hence the first part of Theorem 1
$Majority^n$ can be expressed using $VecAdd_1^n$ as follows

$$Majority^n(X_1, \dots, X_n) = 1 \leftrightarrow VecAdd_1^n(X_1, \dots, X_n) \geq k = n/2 \quad (5)$$

For this we define the following functions

$$\begin{aligned}
Equal^m(X_0, \dots, X_{m-1}, Y_0, \dots, Y_{m-1}) &\equiv \bigwedge_{i=0}^{m-1} X_i = Y_i \\
GT^m(X_0, \dots, X_{m-1}, Y_0, \dots, Y_{m-1}) &\equiv (\vec{X})_2 > (\vec{Y})_2 \\
&\equiv \bigvee_{i=0}^{m} (X_i \wedge \bar{Y}_i \wedge (\bigwedge_{j=i+1}^{m-1} (X_j \equiv Y_j))) \\
GE^m(\vec{X}, \vec{Y}) &= Equal^m(\vec{X}, \vec{Y}) \vee GT^m(\vec{X}, \vec{Y}) \\
Majority^n(X_1, \dots, X_n) &= GE^m(\vec{a}, VecAdd^n(X_1, \dots, X_n)) \quad (\vec{a}\text{=constant bits of } n/2)
\end{aligned}$$

Thus if $VecAdd^n \in NC^1$, then so is $Majority^n$. A corollary of the lemma is the following

Corollary 1. *Multiplication is in NC^1*

Lemma 4. *$VecAdd \in TC^0$ [Chandra, Stockmeyer, Vishkin 1984]*

Proof. We will build a TC^0 circuit for $VecAdd_l^m$.
Out idea is to to use $VecAdd_1^m$ to add up separate coloumns. We add the ith bits of the m numbers to get S_i which is a m' bit number where $m' = \lceil \log(m+1) \rceil$

$$S_i = Addition_1^m(X_{0,i}, X_{1,i}, \ldots, X_{m-1,i}) \tag{6}$$

Doing this for each i, we get l m'-bit numbers such that

$$\sum_{i=0}^{l-1} 2^i S_i = VecAdd_l^m(X_{0,0}, X_{0,1}, \ldots, X_{0,l-1}, \ldots, X_{m-1,0}, \ldots, X_{m-1,l-1}) \tag{7}$$

We now reblock the bits to get numbers $T_0, T_1, \ldots, T_{m'-1}$, each with $\leq m + m'$ many bits such that

$$\sum_{i=0}^{m'-1} T_i = VecAdd_l^m(X_{0,0}, X_{0,1}, \ldots, X_{0,l-1}, \ldots, X_{m-1,0}, \ldots, X_{m-1,l-1}) \tag{8}$$

Now we repeat the above process with these new numbers to get m'' many $m + m' + m''$-bit numbers $U_0, \ldots, U_{m''-1}$ such that

$$\sum_{i=0}^{m''-1} U_i = VecAdd_l^m(X_{0,0}, X_{0,1}, \ldots, X_{0,l-1}, \ldots, X_{m-1,0}, \ldots, X_{m-1,l-1}) \tag{9}$$

where $m''' = \lceil \log m' \rceil$. The U_is can be computed by a CNF or a DNF of polynomial size ($\mathcal{O}(m)$). $Addition_1^{m'}()$ has only $m' \leq \log m + 2$ many inputs and hence has size $\mathcal{O}(m)$. So finally, we have m'' many $m + m' + m''$-bit numbers which we will sum as follows.

We list the numbers $U_0, \ldots, U_{m'-1}$ as rows and block the bits into groups of $\frac{m'}{m''}$ many contiguous coloumns. Now we pick the even blocks and zero out the other blocks and take the sum of what results. There is no carry propagation from one non-zero block to the next non-zero block. thus each output bit depends on only one block of bits and there are $(m'/m'')m'' = m' \leq \log m + 2$ many bits in each block.
Thus we can use a CNF or DNF formula of size $\mathcal{O}(m)$ to compute each output bit. This gives the binary representation of the number V_1. Now we repeat the same procedure for odd blocks to get the number V_2. Finally we get $V_1 + V_2 = VecAdd_l^m(X_{0,0}, X_{0,1}, \ldots, X_{0,l-1}, \ldots, X_{m-1,0}, \ldots, X_{m-1,l-1})$ using an AC^0 circuit to get the desired result.

□

An immediate corollary is the following

Corollary 2. *Multiplication is in TC^0 (using binary representation)*