COMP 531 – Advanced Theory of Computation
Assignment #3

Guillaume Labranche (260585371)

due on 7 March 2016

1. Let $r = \sqrt{\log n}$. Following the hint, we divide the input as $r \times r$ blocks containing $\sqrt{\log n} \times \sqrt{\log n} = \log n$ bits each. Define the addition function $f_i(x_1, \ldots, x_r)$ that computes the ith bit of the sum of r r-bit numbers. Because any formula s with k inputs can be put in CNF or DNF, any function of k inputs can be computed by a circuit of depth 2 with unbounded gates and with size 2^k. So we can create a circuit for f_i of depth 2 and size $2^{r \times r} = 2^{\log n} = n$. We then need one f_i for each output bit of our block sums. When adding r r-bit numbers, the output will be at most $r + \log r$ long. Therefore we need $r + \log r < 2r$ circuits computing f_i. This all takes constant depth and polynomial size $(2r \cdot n)$.

   For each row of blocks, we compute the sum of each block and arrange the results into two rows. The first one concatenating all the sums of even-numbered blocks without shifting the position of the least significant bit of any block sum by padding with 0's. The other row for the odd-numbered blocks. What remains is $2\frac{\log n}{r} = 2\frac{\log n}{\sqrt{\log n}} = 2\sqrt{\log n} = 2r$ rows of n-bit integers.

   We again separate the input into $r \times r$ blocks. Note that we now have only two rows of blocks and $\frac{n}{r}$ blocks per row. Then we once again compute their sum with parallel AC^0 circuits and arrange the results as done above. What remains is 4 rows of n-bit numbers. Since addition of two n-bit numbers is in AC^0, we can perform addition on our 4 numbers and obtain the final sum.

2. We show that this problem of multiplying two n-bit integers can be AC^0-reduced from $PARITY$. If this problem was in AC^0 then $PARITY$ would also be. But we showed in class that it is not.

   For any number n, define n_1 as the least significant bit of n and n_i as the ith bit.

   Let x be the n-sized input to the $PARITY$ problem and assume integer multiplication is doable in AC^0. From x we insert $\log n$ 0's between every x_i to create a new variable y. From y we replace every variable corresponding to an x_i with 1 to create a new variable z. Now compute $w = y \cdot z$. Note that $|y| = |z| = n + (n-1)\log n$

   Consider the naive way of computing w:

$$w = \sum_{i \in \{j : z_j = 1\}} y \cdot 2^{i-1}$$

   In other words, we start with $w = 0$ and for each i such that $z_i = 1$, we add y shifted left by $i - 1$ bits (i.e. padded with i 0's on the right). Consider the $(n-1+(n-1)\log n)$th bit of w. This bit is essentially $\sum_i x_i + c_{n-1+(n-1)\log n}$ modulus 2 where c_i is the carry at the ith bit in the addition. Therefore $w_{n-1+(n-1)\log n} = PARITY(x_1, \ldots, x_n, c_{n-1+(n-1)\log n})$. But since we have added $\log n$ 0's between each x_i, the carries will get propagated to the $\log n$ 0-columns right of $n - 1 + (n-1)\log n$ but not further, and $c_{n-1+(n-1)\log n} = 0$. Therefore $w_{n-1+(n-1)\log n} = PARITY(x_1, \ldots, x_n)$ and we have computed $PARITY$ using integer multiplication in AC^0, which is in contradiction to the result seen in class. Therefore integer multiplication cannot be computed in AC^0.

3. No time

4. We first make a few assumptions about any PARITY gate:

   - It does not contain both x and $\bar{x}$ since this always has parity 1 and can be replaced by a constant.
   - It does not contain x or $\bar{x}$ more than once, since any pair of the same variable has parity 0, and so can be removed without affecting the gate.

   Let n be the number of variables that are available in the circuit. For $i = 0, \ldots, n$, a PARITY gate can either be fed x, $\bar{x}$ or none of those. Therefore we start with a circuit which has at most 2^{3n} PARITY gates. As mentioned in the hint by email, a PARITY gate corresponds to a linear equation of our variables modulus 2. We can build a matrix A from the starting circuit, where each row corresponds to a PARITY gate in the circuit and each column corresponds to a variable. A has $2n$ columns and at most 2^{3n} rows. Because all PARITY gates feed into one AND gate, this corresponds to solving the equation $Ax = (1, \ldots, 1)$.

   Since the rank of the matrix can never be more than the number of columns, in this case the rank of A is $2n$ or less. We can perform Gaussian elimination on the matrix (without changing the solution set) and end up with at most $2n$ non-zero rows at the top. We can get rid of the all-zero rows. We can then transform A back into a circuit using the scheme described above in reverse.