CSE599s: Extremal Combinatorics November 21, 2011

Lecture 15 An Arithmetic Circuit Lowerbound and Flows in Graphs

Lecturer: Anup Rao

1 An Arithmetic Circuit Lower Bound

An arithmetic circuit is just like a boolean circuit, except that the gates are either multiplication or addition gates, and the inputs are either variables or constants. Formally, it is a directed acyclic graph where every vertex has in-degree (aka fan-in) 2 or 0, each vertex (aka gate) of in-degree 0 is labeled by a variable or a field element, and every other vertex is labeled either $+$ or $\times$. The fan-out of a gate is its out-degree.

We say that a circuit computes the polynomial $p(X_1, \ldots, X_n)$ if there is gate in the circuit whose evaluation gives the polynomial $p(X_1, \ldots, X_n)$. Note that although every function from $F_2^n \to F_2$ can be represented as a polynomial, and multiplication and addition (over $\mathbb{F}_2$) can be used to simulate any boolean function on 2 bits, it is *harder* to design a small arithmetic circuit for computing a particular polynomial than it is to design a boolean circuit that computes the corresponding function. This is because many polynomials can evaluate to the same function on $\mathbb{F}_2^n \to \mathbb{F}_2$. For example, the polynomial $(X+Y)(X+Z)$ evaluates to the same function as the polynomial $X+Y+Z+YZ$, so if you want to compute the function $X+Y+Z+YZ$, you can do it with three gates, even though computing the polynomial requires 4 gates. A boolean circuit can choose to evaluate the "easiest" polynomial, and so be of smaller size. Indeed, this restriction allows us to prove stronger lower bounds on arithmetic circuits, and gives us more techniques to attack them.

Suppose we want to compute the polynomial X^d. This can be done by repeatedly squaring X with a circuit of size $\log d$. It is easy to see that this construction is tight: each additional gate can at most double the degree of the polynomials computed by the circuit, so at least $\log d$ multiplications are needed to get degree d.

What about if we want a circuit that simultaneously computes each of the polynomials $X_1^d, X_2^d, X_3^d, \ldots, X_n^d$? One way to do this is to compute each one separately, for a total size of $n \log d$. Is this the best one can do?

Theorem 1 (Bauer and Strassen)**.** *Any arithmetic circuit computing $X_1^d + X_2^d + \ldots + X_n^d$ must use $\Omega(n \log d)$ wires.*

In particular, we get the following easy corollary:

Corollary 2. *Any arithmetic circuit computing each of $X_1^d, X_2^d, \ldots, X_n^d$ must use $\Omega(n \log d)$ wires.*

There are two parts to the proof of Theorem 1. First we prove Corollary 2. Then we show that any size s circuit that computes $X_1^d + \ldots + X_n^d$ can be used to obtain a size $O(s+n)$ circuit computing $X_1^d, X_2^d, \ldots, X_n^d$. Actually you can show that any circuit that computes a polynomial p can be used to obtain a circuit that computes all the partial derivatives of p in similar size, which gives what we want. Here we shall just prove the first part with a proof due to Smolensky that is based on dimension counting.

Suppose that there is some circuit C with s wires that computes $X_1^d, \ldots, X_n^d$. For a polynomial $r(Y;Z)$ in variables partitioned into two lists Y, Z, we say r has degree $(d-1,1)$ if the degree of any variable in Y is at most $d-1$ and any variable in Z is at most 1. Given two lists of polynomials $p = p_1, \ldots, p_k \in \mathbb{F}[X_1, \ldots, X_n]$ and $q = q_1, \ldots, q_\ell \in \mathbb{F}[X_1, \ldots, X_n]$, define the set of polynomials

$$\tau(p||q) = \{r(p;q) : r \text{ has degree } (d-1,1)\ \}.$$

We have the following claims:

Claim 3. *If $f = g \times h$, then $\tau(f, p_1, \ldots, p_k||q_1, \ldots, q_\ell) \subseteq \tau(g, h, p_1, \ldots, p_k||q_1, \ldots, q_\ell)$.*

Indeed, in any degree $(d-1,1)$ polynomial r, $f^t = g^t h^t$, so we obtain a new polynomial r' in one additional variable that computes the same thing as r.

Claim 4. *If $f = g + h$, then $\tau(f, p_1, \ldots, p_k||q_1 \ldots, q_\ell) \subseteq \tau(g, h, p_1, \ldots, p_k||q_1 \ldots q_\ell)$.*

Again, in any degree $(d-1,1)$ polynomial r, we can replace $f^t = (g+h)^t$ and again obtain a degree $(d-1,1)$ polynomial q' that computes the same thing as q.

Claim 5. *$\tau(f, f, p_1, \ldots, p_k||q_1, \ldots, q_\ell) \subseteq \tau(f, p_1, \ldots, p_k||f^d, q_1, \ldots, q_\ell)$.*

To prove this claim, note that since r has access to two copies of f, it can actually compute f^t for any $t \leq 2(d-1)$. To simulate this new computation, it is enough to have access to f^d with degree up to 1 and f with degree up to $d-1$.

Claim 6. *If a circuit C uses s wires to compute polynomials $p_1, \ldots, p_k$ at k distinct gates, then there exist at most s polynomials $q_1, \ldots, q_s$ such that $\tau(p_1, \ldots, p_k||) \subseteq \tau(X_1, \ldots, X_n||q_1, \ldots, q_s)$.*

Proof We prove the claim inductively. Suppose we are working with the space $\tau(p_1, \ldots, p_k||q_1, \ldots, q_r)$, where each p_i is a polynomial computed at a distinct gate of the circuit. Suppose p_1 is a polynomial of maximal depth in the circuit (namely it corresponds to the gate that is farthest away from an input variable). If p_1 is equal to X_i for some i, then we are done, since all p_i's must be at depth 0. Otherwise, by Claims 3 or 4, we get that $\tau(p_1, \ldots, p_k||q_1, \ldots, q_r) \subseteq \tau(g, h, p_2, \ldots, p_k||q_1, \ldots, q_r)$, where g, h are polynomials computed at gates of lower depth in the circuit. g, h may correspond to the same gate as one of the p_i's, in which case we apply Claim 5 (possibly twice) to take care of this duplication. If we repeatedly apply this argument, note that we can apply Claim 5 at most s times. This proves the claim. ■

All that remains is to count dimensions. Note that $\tau(X_1^d, \ldots, X_n^d, X_1, \ldots, X_n||)$ is simply the set of all polynomials in $X_1, \ldots, X_n$ whose degree in each variable is less than d^2. The dimension of this space is thus $(d^2)^n = d^{2n}$. On the other hand, $\tau(X_1, \ldots, X_n||q_1, \ldots, q_s)$ is spanned by a set of at most $2^s \cdot d^n$ polynomials. Thus $2^s \cdot d^n \geq d^{2n} \Rightarrow s \geq n \log d$.

2 Schwartz Zippel Lemma

Lemma 7. *Let $0 \neq p \in \mathbb{F}[X_1, \ldots, X_n]$ be a polynomial of total degree at most d. Then p has at most $d \cdot |\mathbb{F}|^{n-1}$ roots.*

Proof We proceed by induction on n. The base case is that a univariate polynomial of degree d can have at most d roots, which we have shown.

If $n > 1$, let $p = g_t(X_2, \ldots, X_n)X_1^t + g_{t-1}(X_2, \ldots, X_n)X_1^{t-1} + \cdots + g_0(X_2, \ldots, X_n)$. Now consider what happens when we evaluate p at a random point in $(a_1, \ldots, a_n) \in \mathbb{F}^n$, by first sampling $X_2, \ldots, X_n$ and then sampling X_1. By induction, $\Pr[g_t(a_2, \ldots, a_n) = 0] \leq (d-t)/|\mathbb{F}|$, since it has total degree $d-t$. If g_t does not vanish, we are left with a univariate non-zero polynomial of degree t. Thus $\Pr[p(a_1, \ldots, a_n) = 0 | g_t(a_2, \ldots, a_n) \neq 0] \leq t/|\mathbb{F}|$. The union bound then completes the argument. ■

3 Computing Edge Connectivity

Given a directed graph G and two vertices s, t, and $s \to t$ *cut* in the graph is a subset of vertices S such that $s \in S, t \notin S$. The *size* of the cut is the number of edges that go from S to the complement of S. The following is a well known theorem:

Theorem 8 (Max Flow-Min Cut)**.** *The maximum number of edge disjoint paths from s to t is the same the minimum size of an $s-t$ cut.*

Proof Suppose the maximum number of edge disjoint paths is k, and the minimum size of an $s \to t$ cut is c. Then clearly $k \leq c$, since each edge disjoint path must cross the cut using a different edge. Let P be a set of k edge disjoint paths. Then consider the graph G_P which is obtained from G by reversing the direction of each edge involved in a path of P.

If there is a path $s \to t$ path p in G_P, then this path can be used to obtain $k+1$ edge disjoint paths from s to t in G as follows. Let E_P denote the set of edges involved in the paths P, R_p denote the set of edges of E_P that were used in p in the reverse direction, and U_p denote the set of edges of p that did not come from P. Then it is easy to check that $E_P \cup U_p - R_p$ is a set of $k+1$ edge disjoint paths.

Thus G_P contains no $s \to t$ path. Let S denote the set of vertices reachable from s in G_P. There is no edge leaving S in G_P. None of the paths in P can enter S from the outside, since otherwise there would be an edge leaving S in G_P. Every edge leaving S in G must be used by a path of P, or else again there would be an edge leaving S in G_P. Thus there must be at most k edges leaving S, since each path can leave S at most once. This implies that $k \geq c$. ■

We remark that the above theorem can easily be strengthened to the case of weighted directed graphs, where the weights are viewed as capacities. Then we can talk about *flows* from s to t, where a flow is an assignment of non-negative values to edges, such that in every vertex except for s and t, the flow in is equal to the flow out. Similarly, the size of a cut can be defined to be the total weight of out going edges that are cut. Essentially the same proof as above can be used to show that the maximum flow is equal to the size of the minimum cut.

Here we give a beautiful algorithm by Cheung, Lau and Leung for computing the number of edge disjoint paths. A major selling point of the algorithm is that given a vertex s, it simultaneously computes the number of edge disjoint paths to all vertices t, with a running time that is faster than previously known algorithms for doing this. Another advantage is that the algorithm is extremely simple.

Suppose the graph has n vertices and m edges. We work with a vector space over a finite field $\mathbb{F}^t$. For every pair of edges e_1, e_2 in the graph, we sample a uniformly random element $c_{e_1,e_2} \in \mathbb{F}$.

For every edge e in the graph, we shall attempt to associate a vector $a_e \in \mathbb{F}^t$, in such a way that the following linear constraints are satisfied:

- For the j'th edge $e = (s, v)$ leaving s, if $g_1, \ldots, g_\ell$ are the edges that come in to s, then

$$a_e = \sum_i c_{g_i,e} \cdot a_{g_i} + a_j,$$

  where a_j denotes the j'th unit vector.

- For every other edge $e = (u, v)$, if $g_1, \ldots, g_\ell$ are the edges that come in to u, then

$$a_e = \sum_i c_{g_i,e} \cdot a_{g_i}$$

We can easily express the above linear constraints using matrices. Let C be the matrix whose rows and columns are both indexed by the edges such that $C_{e,e'} = c_{e',e}$. Let A denote the matrix whose rows are indexed by edges, such that the e'th row is a_e. Then the above linear constraints can be written

$$A = C \cdot A + H,$$

where H is matrix that captures the contributions to the constraints that come from the unit vectors (it is independent of the $c_{e,e'}$'s). If $I - C$ is invertible, then $A = (I - C)^{-1}H$.

Lemma 9. *$I - C$ is invertible except with probability $m/|\mathbb{F}|$.*

Proof Since C is 0 on the diagonal, $I-C$ has ones on the diagonal. Every off-diagonal coordinate is either 0 or a variable. In other words, when all the constants are chosen to be 0, this matrix is the identity. Thus the determinant $\det(I - C)$ is a non-zero polynomial in the $c_{e,e'}$'s of degree at most m. By the Schwartz-Zippel lemma, the probability that the determinant vanishes is at most $m/|\mathbb{F}|$, so it has an inverse except with probability $m/|\mathbb{F}|$. ■

So with high probability, we get a unique solution $A = (I - C)^{-1}H$. Next we shall argue that the number of edge disjoint paths from s to t is equal to the rank of the vectors coming in to t with high probability. Let A_t denote the matrix whose rows are the vectors a_e for edges e coming in to t.

Theorem 10. *The rank $\mathsf{rk}(A_t)$ is equal to the number of edge disjoint paths from s to t, except with probability $m^2/|\mathbb{F}|$.*

Proof Let k be the size of a minimum $s - t$ cut S. Then observe that the vectors for edges that are outside S can be expressed as linear combinations of vectors associated with the edges leaving S. Thus, the $\mathsf{rk}(A_t) \leq k$.

Consider any set of k edge disjoint paths P from s to t. If we set $c_{e,e'} = 1$ for each pair of edges that are adjacent in P, and set all other constants to 0, then the constraints are satisfied by the vectors that send a distinct unit vector along each path to t. Thus, in this case A_t contains k distinct unit vectors, as rows and so must contain a $k \times k$ submatrix of rank k. Call this submatrix A'_t. We shall argue that $\mathsf{rk}(A'_t) = k$ with high probability.

Indeed, each entry of $A = (I - C)^{-1}H$ is a polynomial of degree $m - 1$, divided by $\det(I - C)$. Thus $\det(A'_t)$ is a polynomial of degree at most $k(m-1) \leq m(m-1)$, divided by $\det(I-C)^k$. These

polynomials do have some setting that gives them a non-zero value, as we saw by considering the k disjoint paths. By the union bound, the probability that either the numerator or denominators vanish is at most $m(m-1)/|\mathbb{F}| + m/|\mathbb{F}|$. ■

The algorithm involves computing $A = (I - C)^{-1}H$, and then computing the rank of A_t for each t.