MATH 523 NOTES ON FORMULA AND CIRCUIT LOWER BOUNDS

JOEL FRIEDMAN

ABSTRACT. These are my notes for Math 523 (equally well for a complexity theory course) on formula and circuit lower bounds. As of January 2014, there has been some recent progress on algebraic lower bounds, namely some progress on "Permanent Versus Determinant." Furthermore, the Mulmuley-Sohoni approach to complexity theory (which they call "Geometric Complexity Theory") is still a possible approach to lower bounds in complexity theory. Here I will outline some of the basic results; details will be given in class and/or may be found in the text by Arora and Barak. These notes are a work in progress; use at your own risk: the material is probably incomplete, and may contain errors, jokes, inaccuracies, and worse. For example, at this point there are almost no references given, although some of this can be found in Arora and Barak. All of this material will be covered in class.

0. Basic Definitions

The idea of finding lower bounds on circuit/formula size/depth is older than the formalization of the question P versus NP. Let us summarize the results.

A *Boolean formula* is a formula involving (Boolean) variables $x_1, \ldots, x_n$, parentheses, and the logical operations $\vee$, $\wedge$, and $\neg$. We will alternatively view a Boolean formula as a rooted tree, whose leaves are variables (sometimes variables and their negations), and whose interior vertices are operations. The *size* of the formula is usually defined to be the number of variables (i.e., number of leaves), or this number minus one. The *depth* of a formula is the maximum length of a path from the root to a leaf. We may propogate the negations to the leaves (via De Morgan's laws), so that the formula has leaves $x_1, \ldots, x_n, \neg x_1, \ldots, \neg x_n$ and is a binary tree whose interior vertices are labelled either $\vee$ or $\wedge$.

An *algebraic formula* over a ring or field is defined similarly, where the logical operations $\vee, \wedge, \neg$ are replaced with $+, \times$ (we sometimes allow $-$ and/or $\div$), and the leaves may also contain any element of the field or ring. The notions of *size* and *depth* are defined similarly.

Any Boolean formula can be expressed as an algebraic formula over the field $\mathbb{Z}/2\mathbb{Z}$.

A *Boolean circuit* is a directed acyclic graph (DAG) with a unique target, whose sources are variables (and sometimes their negations), and whose other vertices are the above mentioned Boolean operations. The fundamental difference between circuits and formulas is that the circuits allow an arbitrary out-degree to their vertices, unlike a formula where the out-degree is always one. The *size* of a circuit

Date: Wednesday 12th February, 2014, at 09:03(remove currenttime eventually).
Research supported in part by an NSERC grant.

is the total number of vertices of the DAG, and its *depth* is the maximum length of a path from the (unique) target to any of its sources.

An *algebraic circuit* is defined similarly.

1. Motivation

Probably the most compelling motivation for studying formulas and circuits is the following remark.

Remark 1.1. Consider any function (language) as a function $f\colon \{0,1\}^* \to \{0,1\}$. If for any positive integer c one can show that for sufficiently large n we have that the restriction of f to $\{0,1\}^n$ requires circuits of size larger than n^c, then f does not lie in P. In particular, if this holds of some function (language) in NP, then P $\neq$ NP.

The following is usually referred to as the *trivial bound*.

Remark 1.2. If $f = f(x_1, \ldots, x_n)$ merely depends on all of its variables, then the size of a formula or circuit to compute f is at least n, and its depth at least $\log_2(n)$.

Remark 1.3. There is a constant C such that most Boolean functions on n variables cannot be computed with a circuit of size at most $C2^n/n$, and therefore of depth at most $n + \log_2(C/n)$.

The above remark follows by a routine (and easy) calculation showing that the total number of circuits of size at most $2^n/Cn$ and showing that it is less than one half of the number of Boolean functions on n variables, i.e., 2^{2^n}.

Remark 1.4. Any Boolean function on n variables has a formula of size at most $n2^n$, and of depth at most $n + \log_2 n$.

This above can be done as a disjuction (repeated "or") of the at most 2^n assignments of variables that yield a value 1 (true).

2. Lower Bounds for Boolean Functions

As far as I know, it is known that for numerous functions, including the XOR function

$$\mathrm{XOR}(x_1, \ldots, x_n) = (x_1 + \cdots + x_n) \bmod 2$$

its minimal formula size is $n^{3-\epsilon}$ (Hastad, building on many articles beforehand, method of random restrictions and the "shrinkage exponent"). This method per se, as other formula size methods (e.g., Neciporuk's Theorem), has no hope of settling P versus NP.

All other Boolean function (circuit size and depth) lower bounds of which I am aware prove lower bounds within a constant of the trivial bound. They, too, have little hope of resolving P versus NP.

3. Remarks on Depth

Remark 3.1. Any circuit of depth d can be "expanded" to get a formula of the same depth.

Hence the minimum formula depth and circuit depth are essentially the same.

A standard "balancing" argument shows the following theorem.

Theorem 3.2. *There is a constant, $c \in \mathbb{R}$, such that any Boolean formula of size s can be "balanced" to yield a formula of depth at most $c \log s$. Similarly for algebraic formulae.*

Proof. Let n be the number of variables of the formula, let m be its size, and let $g(x_1, \ldots, x_n)$ be the function computed by the formula. Consider the number of descendant leaves of each interior vertex. Since each interior vertex has at most two children, there must exist an interior vertex that has between $1/3$ and $2/3$'s of the leaves. Choose some such interior vertex, v, and let $f(x_1, \ldots, x_n)$ be the function computed at v. We have

$$g(x) = h(x; f),$$

where h can be expressed as a formula of size at most $1 + 2m/3$. But we may also write

$$g(x) = (f(x) \wedge h(x; 0)) \vee (\neg f(x) \wedge h(x; 1))$$

in the Boolean case, or

$$g(x) = h(x; 0) + f(x)h(x; 1)$$

in the algebraic case (without the operations of minus and $\div$). This means that a formula of size m can be expressed, by the above procedure, as a formula of depth

$$2 + \tau(2m/3),$$

where $\tau(k)$ denotes the maximum depth required to write a formula of size k. By recursion, we see that $\tau(k)$ is bounded by roughly

$$2 \log_{3/2}(k).$$

□

It follows that for any Boolean or algebraic function (where the algebraic function is computed using only $+$ and $\times$) we have that the following are equivalent to within a constant factor:

(1) minimum formula depth,
(2) minimum circuit depth (which equals miminum formula depth), and
(3) the logarithm of the miminum formula size.

4. Avoiding Divisions

Generally speaking, it seems easier to study algebraic formulae and circuits because we have more tools to apply. Usually we only consider formulae/circuits without allowing the division operation. Let us explain why.

Strassen [Str73] (see also the Mathematical Reviews entry for this article for a summary) showed that over an infinite field, if $L[f]$ is the minimum circuit size to compute a polynomial, $f = f(x_1, \ldots, x_n)$, of degree d (i.e., each monomial of f has total degree—summing the powers of all variables involved—being at most d) using the operations $+, \times$, and $L(f)$ is the same using $+, \times, \div$ (minus can be written as multiplying by the constant -1), then

$$L(f) \leq L[f] 4d \log_2 d.$$

The essential idea is that if f is any polynomial in x with a non-zero constant term,

$$f(x) = k_1(1 + g(x)),$$

with $g(0) = 0$, then we have

$$1/f = (1/k_1)(1 - g + g^2 - \cdots)$$

as a power series in x. In particular we may replace a division by f with the evaluation of a power series; if the function to be evaluated is of degree at most d, then we may truncate this power series by d. Furthermore, since any circuit for f involving division only divides by a finite number of functions, there is a $\beta = (\beta_1, \ldots, \beta_n)$ on which all of these functions do not vanish; hence the substitution $x_i - \beta_i$ for x_i allows us to replace divisions by multiplication by a power series as above. By a process of converting all functions computed into homogenous functions, we can truncate this power series after finitely many terms.

Similarly, one sees that the depth of the circuit increases by a factor of at most $\log d$ by replacing each division by a power series evaluation. Since one is usually interested in computing functions of n variables whose degree is polynomial in n, avoiding divisions changes the circuit size by at most a factor of a polynomial in n, and the depth by at most a factor of $\log n$.

5. Permanent versus Determinant

In Section 16.1 of Arora and Barak the permanent and determinant of an $n \times n$ matrix are defined. It is conjectured that the $n \times n$ permanent cannot be written as an $m \times m$ permanent for $m \le f(n)$, where $f(n)$ is $2^{O(\log n)^2}$, or sometimes, $2^{p(\log n)}$ where p is any polynomial (assuming that $\mathbb{F}$ is a field of characteristic different than two).

For a fixed field (or ring), $\mathcal{F}$, the text defines the classes **AlgP**/poly (also known as **VP**, Valiant's algebraic analogue of **P**) and **AlgNP**/poly (a.k.a. **VNP**); the former is the set of sequences of functions $f_n \colon \mathcal{F}^n \to \mathcal{F}$ that are computed by polynomial sized algebraic circuits (with no divisions), and the latter are those sequences f_n which are of degree bounded by a polynomial in n that can be written as

$$f_n(x_1, \ldots, x_n) = \sum_{e_1, \ldots, e_{m-n} = 0,1} g_n(x_1, \ldots, x_n, e_1, \ldots, e_{m-n}),$$

where $m = m(n)$ is a polynomial in n and g_n is in **AlgP**/poly.

We make the following remarks:

(1) The determinant has a $2^{O(\log n)^2}$ size formula.
(2) Any algebraic formula of size m can be written as a determinant of size $m + 2$ (Valiant, [Val79]); this is easy if we replace $m + 2$ with a polynomial in m.
(3) Most NP-complete problems can be written as a function in **VNP**, and a Boolean function with polynomial sized circuits can be written as functions in **NP**; hence if we can show $\mathbf{P} \neq \mathbf{NP}$ by showing that, say, **SAT** has no polynomial sized circuits, we also have that $\mathbf{VP} \neq \mathbf{VNP}$.
(4) Skyum and Valiant [SV81] defined **pF** to be those sequences of algebraic functions with polynomial sized formulae.
(5) the Permanent Versus Determinant Conjecture is an anologue of comparing Boolean functions computable in polynomial size circuits with poly-log depth (**NC**) with **NP**.

References

[Str73] Volker Strassen. Vermeidung von Divisionen. *J. Reine Angew. Math.*, 264:184–202, 1973.

[SV81] Sven Skyum and Leslie G. Valiant. A complexity theory based on boolean algebra. In *FOCS*, pages 244–253. IEEE Computer Society, 1981.

[Val79] L. G. Valiant. Completeness classes in algebra. In *Proceedings of the eleventh annual ACM symposium on Theory of computing*, STOC '79, pages 249–261, New York, NY, USA, 1979. ACM.

Department of Computer Science, University of British Columbia, Vancouver, BC V6T 1Z4, CANADA, and Department of Mathematics, University of British Columbia, Vancouver, BC V6T 1Z2, CANADA.

E-mail address: jf@cs.ubc.ca or jf@math.ubc.ca