Boolean circuits

An n-ary *Boolean function* $f(x_1, \ldots, x_n)$ is a function from $\{0,1\}^n$ with values in $\{0,1\}$. The values $0,1$ are called *Boolean values*, or bits, or truth values if they are identified with False and True. Elements of $\{0,1\}^n$ are *Boolean strings* of length n (or bit strings). B_n is the set of all 2^{2^n} n-ary Boolean functions.

A basis of connectives is any non-empty set Ω of Boolean functions, not necessarily of the same arity. An Ω*-formula* for $f(x_1, \ldots, x_n)$ is a formula built from Boolean variables $x_1, \ldots, x_n$ using connectives from Ω. Basis Ω is *complete* if all Boolean functions are definable by an Ω-formula. Examples of complete bases are: B_2, the set of all binary Boolean functions, or the DeMorgan basis $\{0, 1, \neg, \vee, \wedge\}$.

A more general way to define (equivalently, compute) Boolean functions is by circuits (straight line programs) that are simplified mathematical models of integrated circuits. A *Boolean circuit* over basis Ω for $f \in B_n$ is a sequence $g_1, \ldots, g_k$ of functions from B_n such that $g_k = f$ and such that any g_i is either one of the projections $x_1, \ldots, x_n$ or it equals to $h(g_{j_1}, \ldots, g_{j_r})$, for some $h \in \Omega$ and $j_1, \ldots, j_r < i$. Drawing directed edges from all such $j_1, \ldots, j_r$ to i, and labelling i by h, defines a labelled directed acyclic graph. *The size* of the circuit is k. The maximum length of a path from a vertex corresponding to one of the inputs $x_1, \ldots, x_n$ to g_k is *the depth* of the circuit. Note that formulas are circuits whose graphs are trees.

There are three basic ways to measure complexity of a Boolean function, given a basis Ω: the minimum size $L_\Omega(f)$ of an Ω-formula defining f, the minimum size $C_\Omega(f)$ of a circuit over Ω defining f, and the minimum depth $D_\Omega(f)$ of a circuit over Ω defining f. For two complete bases Ω, Ω' measures $C_\Omega(f), C_{\Omega'}(f)$ and $D_\Omega(f), D_{\Omega'}(f)$ are proportional to each other, and $L_\Omega(f), L_{\Omega'}(f)$ are polynomially related. A relation exists between the formula size and the circuit depth (see [13]): $\log(L_\Omega(f))$ and $D_\Omega(f)$ are proportional to each other, provided Ω is finite.

By estimating the number of circuits of a bounded size C. E. Shannon [11] proved (cf.[14] for an optimal computation) that for almost all $f \in B_n$, $C_{B_2}(f) \geq \frac{2^n}{n}$. On the other hand, $C_{B_2}(f) \leq \frac{2^n}{n}(1 + o(1))$, for all $f \in B_n$, see [8].

For a **(formal) language** $L \subseteq \{0,1\}^*$ let L_n be the characteristic function of $L \cap \{0,1\}^n$. The main open problem in Boolean complexity (cf.[4] is whether there is a language L accepted by a **non-deterministic polynomial-time**

Turing machine such that $C_\Omega(L_n)$ is not bounded by any polynomial in n. The importance of this question stems from the fact that the affirmative answer implies that the **computational complexity classes** $\mathcal{P}$ and $\mathcal{NP}$ are different, as $C_{B_2}(L_n)$ is bounded by a polynomial for any L decidable in polynomial-time, cf.[10].

No super-linear lower bounds on $C_{B_2}(L_n)$ are known for any $L \in \mathcal{NP}$. However, if additional restrictions are put on the circuits then strong lower bounds are known. Most notable are two results: any depth $d \geq 2$ circuit over a DeMorgan-like basis $\{0, 1, \neg, \bigvee, \bigwedge\}$ allowing $\bigvee, \bigwedge$ of unbounded arity computing the parity function $\bigoplus_n$ must have the size at least $\exp(\Omega(n^{\frac{1}{d-1}}))$ (see [1], [5], [15], [6]), and any circuit in monotone basis $\{0, 1, \vee, \wedge\}$ computing whether or not a graph on n vertices contains a clique of size at least $k \leq n^{\frac{1}{4}}$ must have size at least $n^{\Omega(\sqrt{k})}$ (see [9], [3]).

Also interesting upper bounds are known. Let us mention just two: a size $O(n \log n \log \log n)$ DeMorgan circuit for multiplication of two n-bit long numbers by [12] (it is open if there are linear size circuits), and a monotone DeMorgan **sorting network** of simultaneous size $O(n \log n)$ and depth $O(\log n)$, see [2].

Various parts of **computational complexity** are directly related to lower bound problems for Boolean functions; for example, **communication complexity**, cf.[7].

References

[1] Ajtai, M. (1983) Σ_1^1 - formulae on finite structures, *Annals of Pure and Applied Logic*, **24**, pp.1-48.

[2] Ajtai, M., Komlós, J., and Szemerédi, E. (1983) An $O(n \log n)$ sorting network, *Combinatorica*, **3**, pp.1-19.

[3] Alon, N., and Boppana, R. (1987) The monotone circuit complexity of Boolean functions, *Combinatorica*, **7(1)**, pp.1-22.

[4] Boppana, R., and Sipser, M. (1990) Complexity of finite functions. in: *Handbook of Theoretical Computer Science*, **Vol.A**, Chtp.14, ed.J. van Leeuwen, pp.758-804.

[5] Furst, M., Saxe, J., B., and Sipser, M. (1984) Parity, circuits and the polynomial-time hierarchy, *Math. Systems Theory*, **17**, pp.13–27.

[6] Hastad, J. (1989) Almost optimal lower bounds for small depth circuits. in: *Randomness and Computation*, ed. S.Micali, Ser.Adv.Comp.Res. **Vol.5**, pp.143-170. JAI Press.

[7] Kushilevitz, E., and Nisan, N. (1996) *Communication Complexity*, Cambridge University Press.

[8] Lupanov, O. B. (1958) A method of circuit synthesis (in Russian), *Izvestia V.U.Z. (Radiofizika)*, **1 (1)**, pp.120-140.

[9] Razborov, A. A. (1985) Lower bounds on the monotone complexity of some Boolean functions, *Soviet Mathem. Doklady*, **31**, pp.354-357.

[10] Savage, J. E. (1972) Computational Work and Time on Finite Machines, *J. of the ACM*, **19(4)**, pp.660-674.

[11] Shannon, C. E. (1949) The synthesis of two-terminal switching circuits, *Bell Systems Techn. J,*, **28(1)**, pp.59-98.

[12] Schonhage, A., and Strassen, V. (1971) Schnelle Multiplikation Grosser Zahlen, *Computing*, **7**, pp.281-292.

[13] Spira, P. M. (1971) On time-hardware complexity of tradeoffs for Boolean functions, in: *Proc.* 4^{th} *Hawaii Symp. System Sciences*, pp. 525-527. North Hollywood, Western Periodicals Co..

[14] Wegener, I. (1987) *The Complexity of Boolean Functions*, John Wiley&Sons and B. G. Teubner, N. York and Stuttgart.

[15] Yao, Y. (1985) Separating the polynomial-time hierarchy by oracles, in: *Proc. 26th Ann. IEEE Symp. on Found. of Comp. Sci.*, pp. 1-10.

AMS 1991 Subject Classification: 68Q05