On the number of ANDs versus the number of ORs in monotone Boolean circuits

Uri Zwick *

October 27, 1995
Revised: April 17, 1996

Abstract

Alon and Boppana showed that if a monotone Boolean function f of n variables can be computed by a monotone circuit containing k AND gates, where $k > 1$, then it can also be computed using a monotone circuit containing k AND gates and $O(k(n+k))$ OR gates. They note that their result is tight up to a logarithmic factor. Here we show that under the same assumption the function f can be computed using a monotone circuit containing k AND gates and $O(k(n+k)/\log k)$ OR gates. This result is tight up to a constant factor. By duality the same result holds when the roles of the AND and OR gates are interchanged.

Keywords: Boolean complexity, circuit complexity, monotone complexity, AND and OR gates

Let f be a monotone Boolean function of n variables. Alon and Boppana [2] showed that if f can be computed using a monotone circuit containing k AND gates, where $k > 1$, then it can also be computed by a monotone circuit containing k AND gates and $O(k(n+k))$ OR gates. They use this simple fact as the last step in a proof that every monotone circuit for $CLIQUE_s(n)$, the function of $\binom{n}{2}$ Boolean variables that correspond to the edges of an undirected graph on n vertices that returns the value 1 if and only if the represented graph has a clique of size s, contains $\Omega((n/\log n)^s)$ AND gates.

Alon and Boppana note that their result is tight up to a logarithmic factor. This follows from the fact that the monotone complexity of almost all quadratic monotone Boolean functions on n variables, i.e., functions of the form $\bigvee_{i,j} a_{i,j} \wedge x_i \wedge x_j$, is $\Omega(n^2/\log n)$, as shown by Bloniarz [4], while each such function can be easily computed using a monotone circuit that contains only $n-1$ AND gates using the form $\bigvee_{1\le i<n} x_i \wedge (\vee_{j|a_{i,j}=1} x_j)$.

In this short note we show that the result of Alon and Boppana can be improved, making it tight up to a constant factor. Though this does not lead, at present, to any improved lower bounds, it does settle the relationship between the number of AND and OR gates in monotone Boolean circuits.

Let β be a monotone circuit containing k AND gates that computes the function f of n Boolean variables $x_1,\ldots,x_n$. Let y_i be the output of the i-th AND gate in β and let $z_{i,1}$ and $z_{i,2}$ be the two inputs to this gate. As no additional AND gates may be used, each of $z_{i,1}$ and $z_{i,2}$ must be an OR of a subset of $x_1,\ldots,x_n,y_1,\ldots,y_{i-1}$. Thus, each of $z_{i,1}$ and $z_{i,2}$, and in fact both of them together, can be computed from $x_1,\ldots,x_n,y_1,\ldots,y_{i-1}$ using at most $n+i-2$ fanin-2 OR gates. Similarly, the output of the circuit f must be the OR of a subset of $x_1,\ldots,x_n,y_1,\ldots,y_k$ and can be computed from them using at most $n+k-1$ OR gates. The result of Alon and Boppana follows easily.

The transformation described above amounts to the computation of $2k+1$ Boolean sums, i.e., ORs, of subsets of $x_1,\ldots,x_n,y_1,\ldots,y_k$. Although each such sum on its own may require $n+k-1$ OR gates,

*Department of Computer Science, School of Mathematical Sciences, Raymond and Beverly Sackler Faculty of Exact Sciences, Tel Aviv University, Tel Aviv 69978, Israel. E-mail address : `zwick@math.tau.ac.il`.

we can compute all these ORs together more efficiently. This is done using a well known trick, used among others by Lupanov [6], Arlazarov, Dinic, Kronrod and Faradzev (the "Four Russians") [3] (see also [1], pp. 245–246) and by Savage [7].

In the following discussion we assume that $k \geq 2$. We also assume for simplicity that k is a power of 2. We divide the $n+k$ elements $x_1, \ldots, x_n, y_1, \ldots, y_k$ into blocks of size $\log k$. For each block $b_1, \ldots, b_{\log k}$ we compute all its $2^{\log k} = k$ subset sums. It is easy to see that this can be done using less than k OR gates ($k - \log k - 1$ OR gates are enough). This is done by computing all the subset sums of b_1, then of b_1, b_2 and so on. The computation of each new subset sum requires at most one new OR gate. It is important to note here that the subset sums of $b_1, \ldots, b_i$ can be produced in this way even if b_{i+1} is not yet available. The computation of all subset sums of the blocks of size $\log k$ requires therefore at most $k(n+k)/\log k$ OR gates.

Each one of the $2k+1$ Boolean sums can now be computed using at most $(n+k)/\log k$ OR gates. All the $2k+1$ Boolean sums can therefore be computed using at most $O(k(n+k)/\log k)$ OR gates.

We obtain, as promised, the following result:

Theorem 1 *Let f be a monotone Boolean function in n variables. If f can be computed using a monotone circuit that contains k AND gates, then f can also be computed using a monotone circuit that contains k AND gates and $O(k(n+k)/\log k)$ OR gates. The same statement holds if the roles of AND and OR are interchanged.*

As follows from the result of Bloniarz, this bound is tight up to a constant factor. As a more general example, suggested by Noga Alon, consider the collection of *r-homogeneous* Boolean functions of n variables, i.e., all functions of the form $\bigvee_{i_1<i_2<\ldots<i_r} a_{i_1,\ldots,i_r} \wedge x_{i_1} \wedge \ldots \wedge x_{i_r}$. It follows from a simple counting argument that the monotone complexity of almost all such functions is $\Omega(n^r/\log n)$ (this fact is mentioned in [5], p. 262 and in [8], p. 107). On the other hand, it is not difficult to see that every r-homogeneous function of n variables can be computed using a monotone circuit containing $O(n^{\lceil r/2 \rceil})$ AND gates. For every *even* r, we get therefore a collection of functions for which Theorem 1 is tight.

References

[1] A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *The design and analysis of computer algorithms.* Addison-Wesley, 1974.

[2] N. Alon and R.B. Boppana. The monotone circuit complexity of Boolean functions. *Combinatorica*, 7:1–22, 1987.

[3] V.L. Arlazarov, E.C. Dinic, M.A. Kronrod, and I.A. Faradzev. On economical construction of the transitive closure of a directed graph. *Dokl. Akad. Nauk SSSR*, 194:487–488, 1970. English translation in *Soviet Math. Dokl.*, 11:1209–1210, 1970.

[4] P.A. Bloniarz. The complexity of monotone Boolean functions and an algorithm for finding shortest paths in a graph. Technical Report 238, Ph.D. Dissertation, Laboratory for Computer Science, Massachusetts Institute of Technology, 1979.

[5] P.E. Dunne. *The complexity of Boolean networks.* Academic Press, 1988.

[6] O.B. Lupanov. O ventilnykh i kontaktno-ventilnykh skhemakh. *Dokl. Akad. Nauk SSSR*, 111:1171–1174, 1956.

[7] J.E. Savage. An algorithm for the computation of linear forms. *SIAM Journal on Computing*, 3:150–158, 1974.

[8] I. Wegener. *The Complexity of Boolean Functions.* John Wiley & Sons, 1987.