The Complexity of Negation-Limited Networks - A Brief Survey

Michael J. Fischer[†]
Massachusetts Institute of Technology
Cambridge, Massachusetts U.S.A.

1. Introduction

Let $B = \{0,1\}$, $F_n = \{f \mid f: B^n \to B\}$, and let $\Omega \subseteq \bigcup_{m\geq 1} F_m$. The combinational complexity $C^{\Omega}(F)$ of a set of Boolean functions $F \subseteq F_n$ is the least size network over the basis Ω which computes each of the functions in F. Combinational complexity provides a meaningful measure of the difficulty of finite functions and has been widely studied. Our definitions are similar to those of Savage [20,21] and are formalized in Section 2.

Combinational complexity is interesting for both practical and theoretical reasons. The practical motivation comes from its correspondence with the cost of actual digital hardware. Theoretical interest derives both from its clean mathemati- structure and its connection with computation time on Turing machines [17]. Namely, if $g: B^* \to B^*$ can be computed in time $T(n)$ on a multitape Turing machine, then the restriction $g_n = g \mid B^n$ of g to length n inputs can be computed by a network over any complete basis of size $O(T(n) \log T(n))$.[††] (Schnorr strengthens this bound to $cp{\cdot}T(n) \log S(n)$ where p is the number of instructions of the Turing machine, and $S(n)$ is the number of storage cells visited [22].)

It follows that a lower bound greater than $cn \log n$ on the combinational complexity of g_n implies a non-linear lower bound on the Turing machine time complexity of g. Such lower bounds on Turing machine time have never been obtained for particular concrete functions g except by diagonal techniques.

[†] This research was supported in part by the National Science Foundation under research grant GJ-43634X to M.I.T. The author did some of this work at the University of Toronto and the University of Frankfurt.

[††] The lengths of $g_n(x)$ may differ for different $x \in B^n$. Since a network has a fixed number of outputs, we must assume appropriate conventions for representing the values of g_n. For example, if $m = \max_{x\in B^n} |g_n(x)|$, then the network might have m pairs of outputs encoding one of the three symbols 0, 1 or blank.

Unfortunately, few techniques exist for proving lower bounds on the combinational complexity of specific functions of interest, even though the following theorem shows that "most" functions are hard.

Theorem 1 (Lupanov [8]). $C(f) \sim \frac{2^n}{n}$ for all but a vanishing fraction of functions $f \in F_n$.[†]

For certain natural problems, diagonal techniques can be applied to obtain large lower bounds [2,25]. A sample of such a result is the following:

Theorem 2 (Meyer [11]). Let D_n be the decision problem for length n sentences of Presburger arithmetic [5,19]. Then $C(D_n) \geq c^n$, where $c > 1$ is independent of n.

These techniques, however, do not apply to concrete problems, for example, to any Boolean function whose truth-table can be generated by a multitape Turing machine in time polynomial in the length of the truth-table. Examples of concrete problems are binary integer multiplication, Boolean matrix product, Boolean convolution product, transitive closure of a Boolean matrix, context-free language recognition, and numerous other problems from automata and language theory, combinatorics, and other branches of discrete mathematics and computer science.

Recently, attention has been directed toward developing new techniques for proving lower bounds on concrete problems [6,15,23,24]. To date, only linear lower bounds have been obtained, but with a coefficient of linearity greater than one. An example of such a theorem is the following:

Theorem 3 (Schnorr [23]). Let $f = (x_1 \wedge x_2 \wedge \ldots \wedge x_n) \vee (\neg x_1 \wedge \neg x_2 \wedge \ldots \wedge \neg x_n)$. Then $C(f) = 2n - 3$.

The largest lower bound of this kind is 2.5n, obtained recently by Paul [15].

2. Basic Definitions

Let $B = \{0,1\}$. For each $n \in \mathbb{N}$, let $F_n = \{f \mid f: B^n \to B\}$. Let $\Phi = \bigcup_{n\geq 1} F_n$. For $f \in F_n$, let $\rho(f) = n$, the number of arguments of f.

A logical network $\mathfrak{n}$ over the basis $\Omega \subseteq \Phi$ and initial functions $A \subseteq F_n$ is a directed acyclic graph $G = (V,E)$ with labelled vertices such that the arcs entering each vertex are ordered. The vertices of indegree zero are called source nodes and are denoted by V_s. The remaining vertices, V_g, are called gates. The labels are specified by a function $\nu: V \to \Omega \cup A$ such that (1) if $v \in V_s$, then $\nu(v) \in A$; and (2) if $v \in V_g$, then $\nu(v) \in \Omega$ and the indegree of $v = \rho(\nu(v))$.

We associate with each $v \in V$ a function $\xi_v \in F_n$. If $v \in V_s$, then $\xi_v = \nu(v)$.

† $r(n) \sim s(n)$ iff $\lim \frac{r(n)}{s(n)} = 1$, where $r,s: \mathbb{N} \to \mathbb{R}$.

If $v \in V_g$, then

$$\xi_v = \nu(v)(\xi_{w_1}, \ldots, \xi_{w_k})$$

where $k = \rho(\nu(v))$ and $(w_1,v), \ldots, (w_k,v)$ are the arcs incident on v, in order. The fact that $\mathfrak{n}$ is acyclic insures that each ξ_v is well-defined. Let $F \subseteq F_n$. We say $\mathfrak{n}$ computes F if $F \subseteq \{\xi_v \mid v \in V\}$.

The cost $C(\mathfrak{n})$ of a network $\mathfrak{n}$ is the number of gates it contains. The combinational complexity of $F \subseteq F_n$ relative to the basis $\Omega \subseteq \Phi$ and the initial functions $A \subseteq F_n$ is

$$C^{\Omega,A}(F) = \min\{C(\mathfrak{n}) \mid \mathfrak{n} \text{ is a network over basis } \Omega \text{ and initial functions } A, \text{ and } \mathfrak{n} \text{ computes } F\}.$$

When considering n-argument functions, we will always assume the initial functions $A_n = \{x_1, \ldots, x_n, 0, 1\}$, where x_i is the ith projection function of n arguments $\lambda y_1 \ldots y_n \cdot y_i$, and 0 and 1 are the constant functions of n arguments with values 0 and 1, respectively. We generally omit explicit mention of the initial functions. If the basis is also not specified, the full binary basis $F_1 \cup F_2$ is assumed.

3. Monotone Networks

In an effort to understand better the difficulties in proving lower bounds on combinational complexity and hopefully to develop new proof techniques applicable to the general case, various restrictions on networks have been considered which enable non-trivial lower bounds to be proved.

Let $M = \{\wedge, \vee\} \subseteq F_2$. We call a network over M a monotone network, and we denote $C^M(F)$ by $MC(F)$, the monotone complexity of F. A function $f \in F_n$ is monotone increasing (or monotone for short) if for all $x, y \in B^n$, $x \le y \Rightarrow f(x) \le f(y)$, where $x \le y$ iff $x_j \le y_j$ for all j, $1 \le j \le n$. It is clear by induction that a monotone network can compute only monotone increasing functions. Conversely, every monotone function can be computed by a monotone network.

A close relationship between monotone functions and general Boolean functions allows Theorem 1 to be used to establish the existence of hard monotone functions.

Definition. Let $f \in F_n$, $g \in F_{2n}$. g is a monotone cover of f if (i) g is monotone increasing, and (ii) $f(x_1, \ldots, x_n) = g(x_1, \neg x_1, \ldots, x_n, \neg x_n)$.

Lemma 4. Every $f \in F_n$ has a monotone cover $g \in F_{2n}$.

Proof. Take

$$g(x_1,y_1, \ldots, x_n,y_n) = \begin{cases} 1 & \text{if } \Sigma(x_i + y_i) > n; \\ 0 & \text{if } \Sigma(x_i + y_i) < n; \\ f(x_1, \ldots, x_n) & \text{otherwise.} \end{cases}$$ □

Theorem 5. Let $MC^{max}(n) = \max\{MC(g) \mid g \in F_n \text{ and } g \text{ is monotone}\}$. Then $MC^{max}(n) \gtrsim \sqrt{2}\cdot \frac{2^{n/2}}{n}$.[†]

Proof. Let $m = \lfloor n/2 \rfloor$. Choose a function $f \in F_m$ maximizing $C(f)$. Let g be a monotone cover of f. Then using Theorem 1,

$$\frac{2^{(n-1)/2}}{n/2} \le \frac{2^m}{m} \lesssim C^{max}(m) = C(f) \le C(g) + m \le MC^{max}(n) + m,$$

where $C^{max}(m) = \max\{C(f) \mid f \in F_n\}$. The theorem follows since $m = o(\frac{2^{n/2}}{n})$. $\square$

A stronger result can be obtained by looking into the proof of Theorem 1. The lower bound of Theorem 1 is established by counting the number N_q of n-input networks of size at most q and showing that $N_q \le (cq)^q$ for some constant c. (Cf. Fischer [3] or Strassen [26].) Comparing this number with 2^{2^n}, the number of functions in F_n, yields the bound. For monotone functions we need only compare N_q with the number of monotone functions in F_n. While this number is not known exactly, it is at least $2^{\binom{n}{\lfloor n/2 \rfloor}}$, for the set $S = \{s \in B^n \mid \# \text{ 1's in } S = \lfloor n/2 \rfloor\}$ has cardinality $\binom{n}{\lfloor n/2 \rfloor}$, and a monotone function can assume arbitrary values on S. Taking $q = (1-\varepsilon)\sqrt{\frac{2}{\pi}} \cdot \frac{2^n}{n^{3/2}}$ it follows that $\frac{N_q}{\# \text{ monotone functions in } F_n} \to 0$ as $n \to \infty$.

This proves the following:

Theorem 6. All but a vanishing fraction of monotone functions f in F_n have $MC(f) \ge C(f) \gtrsim \sqrt{\frac{2}{\pi}} \cdot \frac{2^n}{n^{3/2}}$.

As with general networks, no non-trivial lower bounds are known on the monotone complexity of particular concrete single functions. However, for <u>sets</u> of monotone functions, considerable success has been achieved. Several such results follow from a general theorem about graphs due to Pippenger and Valiant [16,27].

Theorem 7. Let $G = (V,E)$ be an undirected graph with vertices V and edges E. Let $X,Y \subseteq V$, $X \cap Y = \emptyset$, and let $\mathcal{P} = \{P_1, \ldots, P_k\}$, where each P_i is a set of vertex-disjoint paths from X to Y. Assume further that for every $x \in X$, $y \in Y$, there is a path from x to y in $P = \bigcup_i P_i$. Then $|E| \ge \frac{c|X| \cdot |Y|}{|\mathcal{P}|} \log(|X| + |Y|)$.

To apply this theorem, we must show the existence of sets of disjoint paths with the required properties. This is where we make use of the assumption of monotonicity.

† $r(n) \lesssim s(n)$ $(s(n) \gtrsim r(n))$ iff $\limsup_{n\to\infty} \frac{r(n)}{s(n)} \le 1$, where $r,s: \mathbb{N} \to \mathbb{R}$.

Definition. Let $\tau_k^n(x_1, \ldots, x_n) = \begin{cases} 1 \text{ if } \sum_i x_i \,(= |\{x_i \mid x_i = 1\}|) \ge k; \\ 0 \text{ otherwise.} \end{cases}$

τ_k^n is the threshold-k function of n arguments. Boolean sorting of n arguments is the set

$$BS_n = \{\tau_k^n \mid 1 \le k \le n\} \subseteq F_n.$$

The following corollary to Theorem 7 was originally proved directly by Lamagna and Savage [7].

Corollary 8. $MC(BS_n) \ge cn \log n$ for some $c > 0$.

Proof. Let $\mathfrak{n}$ compute BS_n. Let $X = \{x_0, \ldots, x_{n-1}\}$ be the source vertices of $\mathfrak{n}$ and let $Y = \{y_0, \ldots, y_{n-1}\}$ be vertices of $\mathfrak{n}$ such that $\xi_{y_k} = \tau_{k+1}^n$, $0 \le k \le n - 1$.

We define a family $\mathcal{P} = \{P_0, \ldots, P_{n-1}\}$ of sets of vertex-disjoint paths. Intuitively, each set P_k is obtained by initially setting all inputs to zero and then turning them on one at a time. Since $\mathfrak{n}$ computes BS_n, the outputs also turn on one at a time. Clearly, if an output changes as a result of a change in a single input, there must be a path in the network connecting the input to the output on which every vertex changes value. By monotonicity, every vertex on that path changes from zero to one when the input is turned on. Thus, each such path is distinct from those previously obtained. These paths comprise the set P_k. By varying the order of turning on the inputs, we obtain in this way each of the sets in $\mathcal{P}$.

More precisely, for $0 \le k \le n-1$, $0 \le j \le n$, let $\alpha_{k,j} \in B^n$ be the input vector whose ith component is given by

$$(\alpha_{k,j})_i = \begin{cases} 1 & \text{if } i = k + \ell (\text{mod } n) \text{ for some } \ell,\ 0 \le \ell < j; \\ 0 & \text{otherwise.} \end{cases}$$

Let $G_{k,j} = \{v \in \text{vertices } (\mathfrak{n}) \mid \xi_v(\alpha_{k,j}) = 1\}$. Since $\alpha_{k,j+1} \ge \alpha_{k,j}$, then $G_{k,j+1} \supseteq G_{k,j}$ by the monotonicity of $\mathfrak{n}$. Let $D_{k,j} = G_{k,j+1} - G_{k,j}$, $0 \le j < n$. Clearly, the sets $D_{k,0}, \ldots, D_{k,n-1}$ are pairwise disjoint. Since $\xi_{y_j}(\alpha_{k,j+1}) = 1$ and $\xi_{y_j}(\alpha_{k,j}) = 0$, $y_j \in D_{k,j}$. Also, $x_{k+j (\text{mod } n)} \in D_{k,j}$. An easy induction shows that to every node in $D_{k,j}$ there is a path from $x_{k+j (\text{mod } n)}$ consisting entirely of nodes in $D_{k,j}$. Let $p_{k,j}$ be such a path from $x_{k+j(\text{mod } n)}$ to y_j, and let $P_k = \{p_{k,j} \mid 0 \le j < n\}$. Then P_k is the desired set of vertex-disjoint paths from X to Y.

It is easily verified that $\mathcal{P}$ satisfies the hypotheses of Theorem 7, and the lower bound immediately follows. $\square$

For another application of this theorem, we consider networks which rotate a subset of their inputs under control of the remaining variables.

Definition. Let $R = \{g_0, \ldots, g_{n-1}\} \subseteq F_{n+p}$, $\beta \in B^p$. β causes an (n,r)-rotation in R if

$$g_j(x_0, \ldots, x_{n-1}, \beta_0, \ldots, \beta_{p-1}) = x_{j+r(\mathrm{mod}\ n)}$$

for all $x_0, \ldots, x_{n-1} \in B$ and all $0 \le j \le n - 1$, where $(\beta_0, \ldots, \beta_{p-1}) = \beta$. R is an n-rotation set if for all r, $0 \le r \le n - 1$, there exists $\beta_r \in B^p$ which causes an (n,r)-rotation in R.

Corollary 9. There exists $c > 0$ such that for all n-rotation sets R, $MC(R) > cn \log n$.

Proof. The proof is identical to that of Corollary 8 except that we define

$$(\alpha_{k,j})_i = \begin{cases} 1 & \text{if } 0 \le i < n \text{ and } i = k + \ell(\mathrm{mod}\ n) \text{ for some } \ell,\ 0 \le \ell < j; \\ 0 & \text{if } 0 \le i < n \text{ and } i \ne k + \ell(\mathrm{mod}\ n) \text{ for any } \ell,\ 0 \le \ell < j; \\ \beta_{k,i-n} & \text{if } n \le i < n + p \end{cases}$$

where $\beta_r = (\beta_{r,0}, \ldots, \beta_{r,p-1})$ causes an (n,r)-rotation. □

Still a third application of Theorem 7 is to Boolean matrix product.

Definition. The Boolean matrix product of two $n \times n$ matrices $A = (a_{ij})$ and $B = (b_{ij})$ is the set $MP_n = \{c_{ij} \mid c_{ij} = \bigvee_{k=1}^{n} a_{ik} \wedge b_{kj},\ 1 \le i,j \le n\} \subseteq F_{2n^2}$.

Corollary 10. $C(MP_n) > cn^2 \log n$ for some $c > 0$ independent of n.

This bound was greatly improved using more refined techniques developed originally by Pratt [18].

Theorem 11 (Mehlhorn [10] and Paterson [14]). $MC(MP_n) = 2n^3 - n^2$. Moreover, the straightforward network obtained from the definition of MP_n is uniquely optimal to within the associativity and commutativity of the basic operations in M.

The lower bound techniques we have discussed so far depend critically on the monotone restriction, for asymptotically smaller networks are known using a full basis.

Theorem 12 (Muller and Preparata [12]). $C(BS_n) = O(n)$.

Theorem 13 (Fischer and Meyer [4]). $C(MP_n) = O(n^{\log_2 7} \cdot (\log n)^{1+\epsilon})$, for any $\epsilon > 0$.

The combinations of Corollary 8 with Theorem 12, and Theorem 11 with Theorem 13, establish that a considerable savings in the complexity of a network for a set of monotone functions can be realized by using negations. To discuss the extent of such savings, we define the gap between monotone and general combinational complexity.

Definition. Let $F \subseteq F_n$, F monotone. Then $GAP(F) = MC(F)/C(F)$. (If $C(F) = 0$, let $GAP(F) = 0$).

Question. How big is $\max\{\mathrm{GAP}(F) \mid F \subseteq F_n,\ F \text{ monotone}\}$ as a function of n?

It follows from Theorem 11 and 13 that $\mathrm{GAP}(\mathrm{MP}_n) \gtrsim n^{\alpha}$ for any $\alpha < 3 - \log_2 7 \approx .19$. Just how much larger the maximum can be is not known -- it is not even known if the maximum grows exponentially in n. If a good upper bound on GAP(F) could be obtained, then a lower bound on MC(F) would translate into a lower bound on C(F), giving a new technique for obtaining lower bounds on combinational complexity.

4. Inversion Complexity

An obvious way to generalize the class of monotone functions is to consider those functions that can be realized by a network over the complete basis $\Delta = \{\wedge, \vee, \neg\}$ but using only a limited number of negations. The monotone functions are the extreme case in which no negations are permitted.

For a network η over Δ, let $I(\eta)$ = the number of negations in η. For $F \subseteq F_n$, define $I(F) = \min\{I(\eta) \mid \eta \text{ computes } F\}$.

I(F) can be characterized quite simply. Let $|F| = m$ and treat F as a function $B^n \to B^m$. A sequence $C = (\zeta_1, \zeta_2, \ldots, \zeta_k)$ of vectors $\zeta_i \in B^n$ is called a chain of length k. For such a chain, define

$$\mathrm{alt}_F(C) = |\{i \mid 1 \le i < k \text{ and } F(\zeta_i) \not\le F(\zeta_{i+1})\}|.$$

Let $A(F) = \max\{\mathrm{alt}_F(C) \mid C \text{ is a chain}\}$, and let $b(n) = \lceil \log_2(n+1) \rceil$.

Theorem 14. $I(F) = \lceil \log_2(A(F) + 1) \rceil$.

Corollary 15.

(a) $\max\limits_{f \in F_n} I(f) = \lfloor \log_2(n+1) \rfloor$;

(b) $\max\limits_{F \subseteq F_n} I(F) = \lceil \log_2(n+1) \rceil = b(n)$.

Theorem 14, stated for singleton sets F, and Corollary 15 are due to Markov and appear in [9]. Nakamura, Tokura and Kasami [13] and this author [3] give algorithms for finding a network for F with only $\lceil \log_2(A(F) + 1) \rceil$ negations. A proof that $I(F) \ge \lceil \log_2(A(F) + 1) \rceil$ also appears in [3].

We now define the negation-restricted complexity of a set $F \subseteq F_n$ such that $I(F) \le k$ to be

$$NC_k(F) = \min\{C^{\Delta}(\eta) \mid \eta \text{ is a network over } \Delta,\ \eta \text{ computes } F, \text{ and } I(\eta) \le k\}.$$

Note that $NC_0 = MC$.

As mentioned in the last section, little is known about the behavior of GAP(F), or even if it is bounded by a polynomial in the number of arguments of F. A natural generalization of GAP is to let $\mathrm{GAP}_k(F) = NC_k(F)/C(F)$. (If $C(F) = 0$, let $\mathrm{GAP}_k(F) = 0$.) Thus, $\mathrm{GAP}_k(F)$ is defined only for those F which can be realized with at most k negations and is an expression of the increase in network size when the number of negations is restricted to k. Note that $GAP_0(F)$ agrees with our previous definition of $GAP(F)$.

In view of our inability to bound $GAP(F)$ nontrivially from above, the following theorem and its corollary come as somewhat of a surprise.

Theorem 16. $NC_{b(n)}(F) \le 2 \cdot C^{\Delta}(F) + O(n^2 \log^2 n) \le 6 \cdot C(F) + O(n^2 \log^2 n)$ for all $F \subseteq F_n$.

Corollary 17. $GAP_{b(n)}(F) \le O(n \log^2 n)$ for all $F \subseteq F_n$.

Proof of Corollary 17. We say a function $f \in F_n$ depends on its i^{th} argument if there exist $a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n \in B$ such that

$$f(a_1, \ldots, a_{i-1}, 0, a_{i+1}, \ldots, a_n) \ne f(a_1, \ldots, a_{i-1}, 1, a_{i+1}, \ldots, a_n).$$

Let $F \subseteq F_n$, let X be the set of argument positions upon which some function in $F-A_n$ depends, and let $m = |X| \le n$. (Recall that A_n is the set of initial functions, so the functions in $F-A_n$ must be computed by gates.) Let $F' \subseteq F_m$ be the restriction of $F-A_n$ to the argument positions in X. It is easily shown that $C(F') = C(F)$ and $NC_k(F') = NC_k(F)$. Also, $C(F') \ge m/2$ since we permit only gates of at most two inputs, and F' depends on all of its arguments. By Theorem 16,

$$GAP_{b(n)}(F') \le 6 + \frac{cm^2 \log^2 m}{C(F')}$$

for a fixed constant c. Hence,

$$GAP_{b(n)}(F) = GAP_{b(n)}(F') \le 6 + 2cm \log^2 m \le 6 + 2cn \log^2 n = O(n \log^2 n).$$

Note that $GAP_{b(n)}(F)$ can only be as large as $O(n \log^2 n)$ for F of low complexity; if $o(C(F)) = n^2 \log^2 n$, then $GAP_{b(n)}(F) \lesssim 6$. The behavior of $GAP_k(F)$ is not well-understood, however, for any $k < b(n)$.

Before proving Theorem 16, we generalize and strengthen Lemma 4. A monotone cover of a set of functions $F \subseteq F_n$ is a set $G \subseteq F_{2n}$ that contains a monotone cover of each function in F.

Lemma 18. Every $F \subseteq F_n$ has a monotone cover $G \subseteq F_{2n}$ such that $MC(G) \le 2 \cdot C^{\Delta}(F)$.

Proof. We proceed by induction on $C^{\Delta}(F)$.

Base: If $C^{\Delta}(F) = 0$, then F is monotone, so G is trivially constructed.
Induction: Let $s > 0$ and assume the lemma holds for all F' such that $C^{\Delta}(F') < s$. Let $C^{\Delta}(F) = s$ and let η be a network over Δ of cost s which computes F. By choosing an initial gate of η, F may be decomposed in one of three ways, depending on the label of the gate, for some $F' \subseteq F_{n+1}$:

1. $F(x_1, \ldots, x_n) = F'(x_i \vee x_j, x_1, \ldots, x_n)$;
2. $F(x_1, \ldots, x_n) = F'(x_i \wedge x_j, x_1, \ldots, x_n)$;
3. $F(x_1, \ldots, x_n) = F'(\neg x_i, x_1, \ldots, x_n)$.

$C^{\Delta}(F') \le s - 1$ since a network for F' is obtained from $\mathfrak{N}$ by deleting the chosen gate. By the induction hypothesis, there is a monotone cover $G'(y,y',x_1,x_1',\ldots,x_n,x_n')$ of F' such that $MC(G') \le 2\cdot C^{\Delta}(F')$. Define G according to the case that obtained above:

1. $G(x_1,x_1', \ldots, x_n,x_n') = G'(x_i \vee x_j, x_1' \wedge x_j', x_1,x_1', \ldots, x_n,x_n')$;
2. $G(x_1,x_1', \ldots, x_n,x_n') = G'(x_i \wedge x_j, x_i' \vee x_j', x_1,x_1', \ldots, x_n,x_n')$;
3. $G(x_1,x_1', \ldots, x_n,x_n') = G'(x_i',x_i,x_1,x_1', \ldots, x_n,x_n')$.

It follows easily using DeMorgan's law that G is a monotone cover for F. Also, $MC(G) \le 2 + MC(G') \le 2 + 2\cdot C^{\Delta}(F') \le 2\cdot C^{\Delta}(F)$, proving the lemma for $C^{\Delta}(F) = s$. That the lemma holds for all s follows by induction. □

From Lemma 18 we see that any F can be realized with no negations and with only a factor 2 increase in complexity if the negations of the variables are available as inputs.

Definition. $V_n = \{x_1, \neg x_1, \ldots, x_n, \neg x_n\}$.

Lemma 19. $NC_{b(n)}(V_n) \le O(n^2\log^2 n)$.

Proof. Let

$$\tau^n_{k,i}(x_1,\ldots,x_n) = \begin{cases} 1 & \text{if } \sum\limits_{j\ne i} x_j \ge k; \\ 0 & \text{otherwise.} \end{cases}$$

Note that $\tau^n_{k,0}$ is the function τ^n_k defined previously in connection with Boolean sorting. We use the fact that

$$\begin{aligned} \neg x_i = 1 &\Leftrightarrow x_i = 0 \\ &\Leftrightarrow \forall k[\tau^n_{k,0}(x) = \tau^n_{k,i}(x)] \\ &\Leftrightarrow \forall k[\tau^n_{k,0}(x) \rightarrow \tau^n_{k,i}(x)] \\ &\Leftrightarrow \bigwedge_k [\neg \tau^n_{k,0}(x) \vee \tau^n_{k,i}(x)]. \end{aligned}$$

Thus, $\neg x_i$ can be computed from the functions $\neg \tau^n_{k,0}$ and $\tau^n_{k,i}$ using only $2n-1$ $\wedge$- and $\vee$-gates. It remains to compute $U^n_i = \{\tau^n_{k,i} \mid 1 \le k \le n\}$, $0 \le i \le n$, and $V^n = \{\neg \tau^n_{k,0} \mid 1 \le k \le n\}$.

Fact (Batcher [1]). $MC(U^n_i) \le O(n \log^2 n)$.

It follows immediately that $MC(\bigcup\limits_{i=0}^{n} U^n_i) \le O(n^2\log^2 n)$.

To compute V^n, it suffices to find a network $\mathfrak{M}_n$ for any set of functions $\Gamma_n = \{\gamma^n_1, \ldots, \gamma^n_n\} \subseteq F_n$ with the property that

$$\gamma^n_i(\tau^n_{1,0}(x), \ldots, \tau^n_{n,0}(x)) = \neg \tau^n_{i,0}(x);$$

that is, when the inputs are sorted in decreasing order, γ_i^n is the complement of the i^{th} input.

Let $n = 2^r - 1$. We define Γ_n inductively on r.

$r = 1$: $\gamma_1^1(x_1) = \neg x_1$.

$r > 1$: Let $m = 2^{r-1}$. For $1 \le j \le m - 1$, let

$$\delta_j^n(x_1, \ldots, x_n) = (x_j \wedge \neg x_m) \vee x_{j+m}.$$

Now define

$$\gamma_i^n(x_1, \ldots, x_n) = \begin{cases} \gamma_i^{m-1}(\delta_1^n(x), \ldots, \delta_{m-1}^n(x)) \wedge \neg x_m & \text{if } 1 \le i \le m-1; \\ \neg x_m & \text{if } i = m; \\ \gamma_{i-m}^{m-1}(\delta_1^n(x), \ldots, \delta_{m-1}^n(x)) \vee \neg x_m & \text{if } m+1 \le i \le n. \end{cases}$$

A network $\mathfrak{M}_n$ for Γ_n is pictured in Figure 1.

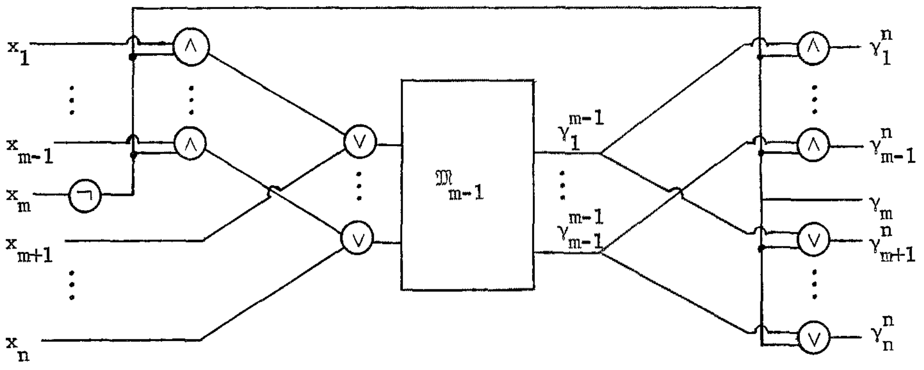

Figure 1. A network for Γ_n.

To see that Γ_n has the desired properties, we consider separately the two cases $x_m = 0$ and $x_m = 1$. Note that if the inputs are sorted in decreasing order, $x_m = 0$ implies that $x_k = 0$ for all $k > m$, and $x_m = 1$ implies $x_k = 1$ for all $k < m$. The remaining details are left to the reader. □

Proof of Theorem 16. Let $F \subseteq F_n$. By Lemma 18, there is a monotone cover G of F such that $MC(G) \le 2 \cdot C^{\Delta}(F)$. By Lemma 19, $NC_{b(n)}(V_n) \le O(n^2 \log^2 n)$. Since F can be realized as the composition of G with V_n, $NC_{b(n)}(F) \le 2 \cdot C^{\Delta}(F) + O(n^2 \log^2 n)$.

To complete the proof, we must show that $C^{\Delta}(F) \le 3 \cdot C(F)$. But that is immediate since for every $f \in F_1 \cup F_2$, $C^{\Delta}(f) \le 3$. Hence, each gate in a general network for F can be replaced by a sub-network of at most three gates from Δ.

5. Conclusion

Proving lower bounds on the combinational complexity of concrete functions is a difficult and challenging problem. Previous successes in establishing lower bounds for monotone networks and the known gaps between the monotone and general combinational complexity indicate the key role that negations play in determining combinational complexity.

We have investigated the way in which the complexity of a set of functions F decreases with the use of additional negations beyond the minimum number necessary to realize F. For sets F of maximum inversion complexity, at most a factor of 2 and an additive term of order $n^2 \log^2 n$ is saved. However, for sets of lower inversion complexity, no interesting bounds are known on the amount of savings possible. Good upper bounds on the amount of such savings would enable lower bounds on combinational complexity to be concluded from lower bounds on the negation-restricted complexity.

Acknowledgment

The author is grateful to A.R. Meyer, M.S. Paterson, and N. Pippenger for several helpful discussions.

References

1. Batcher, K.E., Sorting networks and their applications, Proc. AFIPS Spring Joint Computer Conference, Vol. 32, AFIPS Press, Montvale, N.J.,(1968), 296-291.

2. Ehrenfeucht, A., Practical decidability, Report CU-CS-008-72, Dept. of Computer Science, Univ. of Colorado, Boulder, Colo., (1972), 14 pp.

3. Fischer, M.J., Lectures on network complexity, University of Frankfurt, Germany, June 1974, 25 pp.

4. Fischer M.J., and A.R. Meyer, Boolean matrix multiplication and transitive closure, Proc. 12th IEEE Symp. on Switching and Automata Theory (1971), 129-131.

5. Fischer, M.J. and M.O. Rabin, Super-exponential complexity of Presburger arithmetic. In Complexity of Computation, SIAM-AMS Proceedings, Vol. 7 (1974), 27-41.

6. Hsieh, W.N., L.H. Harper and J.E. Savage, A class of Boolean functions with linear combinational complexity, MAC Technical Memorandum 55, M.I.T. Project MAC, Cambridge, Mass. (1974), 38 pp.

7. Lamagna, E.A. and J.E. Savage, Combinational complexity of some monotone functions, Proc. 15th IEEE Symp. on Switching and Automata Theory (1974), 140-144.

8. Lupanov, O.B., A method of circuit synthesis, Izvestia v.u.z. Radiafizike, No. 1 (1958), 120-140.

9. Markov, A.A., On the inversion complexity of a system of functions, J. ACM 5, 4 (1958), 331-334.

10. Mehlhorn, K., On the complexity of monotone realizations of matrix multiplication, Technical report A74-11, Fachbereich Angewandte Mathematik und Informatik, Universität des Saarlandes, Saarbrucken, Germany (1974), 17 pp.

11. Meyer, A.R., Private communication.

12. Muller, D.E. and F.P. Preparata, Bounds to complexities of networks for sorting and for switching, J. ACM 22, 2 (1975), 195-201.

13. Nakamura, K., N. Tokura, and T. Kasami, Minimal negative gate networks, IEEE Trans. Comp., Vol. C-21, No. 1 (1972), 5-11.

14. Paterson, M.S., Complexity of monotone networks for Boolean matrix product, Theoretical Computer Science 1, 1 (1975), to appear.

15. Paul, W.J., A 2.5 N-lower bound on the combinational complexity of Boolean functions, Proc. 7th ACM Symp. on Theory of Computing (1975), 27-36.

16. Pippenger, N., Private communication.

17. Pippenger, N., and M.J. Fischer, Relationships among complexity measures, in preparation.

18. Pratt, V.R., The power of negative thinking in multiplying Boolean matrices, Proc. 6th ACM Symp. on Theory of Computing (1974), 80-83.

19. Presburger, M., Über die Vollständigkeit eines gewissen Systems der Arithmetic ganzer Zahlen in welchem die Addition als einzige Operation hervortritt. Comptes-rendus du I Congrès des Mathématiciens des Pays Slaves, Warsaw (1930), 92-101, 395.

20. Savage, J.E., Computational work and time on finite machines, J. ACM 19, 4 (1972), 660-674.

21. Savage, J.E., The Complexity of Computing, manuscript, 1974.

22. Schnorr, C.P., The network complexity and the Turing machine complexity of finite functions, manuscript, University of Frankfurt, Germany (1975), 18 pp.

23. Schnorr, C.P., The combinational complexity of equivalence, Theoretical Computer Science, to appear.

24. Schnorr, C.P., Zwei lineare untere Schranken fur die Komplexität Boolescher Funktionen, Computing 13 (1974), 155-171.

25. Stockmeyer, L.J., The complexity of decision problems in automata theory and logic, Project MAC Technical Report 133, M.I.T., Cambridge, Mass. (1974), 224 pp.

26. Strassen, V., Berechnungen in partiellen Algebren endlichen Typs, Computing 11 (1973), 181-196.

27. Valiant, L.G., On non-linear lower bounds in computational complexity, Proc. 7th ACM Symposium on Theory of Computing (1975), 45-52.