Open access • Journal Article • DOI:10.1109/TC.1977.1674864

Reduction of Depth of Boolean Networks with a Fan-In Constraint — Source link

Preparata, Mulller, Barak

Institutions: University of Illinois at Urbana–Champaign

Published on: 01 May 1977 - IEEE Transactions on Computers (IEEE)

Topics: Boolean expression, Boolean algebra, Fan-in and Reduction (complexity)

REPORT R-712 DECEMBER, 1975 UILU-ENG 75-2248

CSL COORDINATED SCIENCE LABORATORY

REDUCTION OF DEPTH OF BOOLEAN NETWORKS WITH A FAN-IN CONSTRAINT

F. P. PREPARATA
D. E. MULLER
A. B. BARAK

UNIVERSITY OF ILLINOIS – URBANA, ILLINOIS

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* REDUCTION OF DEPTH OF BOOLEAN NETWORKS WITH A FAN-IN CONSTRAINT		5. TYPE OF REPORT & PERIOD COVERED Technical Report
		6. PERFORMING ORG. REPORT NUMBER R-712; UILU ENG 75-2248
7. AUTHOR(s) F. P. Preparata, D. E. Muller and A. B. Barak		8. CONTRACT OR GRANT NUMBER(s) DAAB-07-72-C-0259
9. PERFORMING ORGANIZATION NAME AND ADDRESS Coordinated Science Laboratory University of Illinois at Urbana-Champaign Urbana, Illinois 61801		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Joint Services Electronics Program		12. REPORT DATE December, 1975
		13. NUMBER OF PAGES 24
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

boolean expressions
combinational networks
network depth
number of levels
computational complexity
design algorithms
parallel computation

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

In this paper we present a family of techniques for the design of combinational networks whose objective is the reduction of the number of levels, subject to a constraint on the fan-in of the logic gates. We show that a boolean expression with n literals and involving the connectives AND and OR can be restructured so that the resulting network has depth at most $C_\ell \log_2 n + \delta$, where $\delta < 0.415$ and C_ℓ is 1.81, 1.38, 1.17, and 1 for maximum fan-in ℓ of 2,3,4, and 5, respectively. If we additionally require that the amount of equipment of the resulting network be bounded by a linear function of n, it is possible to bound the depth by $\log_2 n$ with a fan-in of at most 3.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

UILU-ENG 75-2248

REDUCTION OF DEPTH OF BOOLEAN NETWORKS WITH A FAN-IN CONSTRAINT

by

F. P. Preparata
D. E. Muller
A. B. Barak

This work was supported in part by the Joint Services Electronics Program (U.S. Army, U.S. Navy and U. S. Air Force) under Contract DAAB-07-72-C-0259.

REDUCTION OF DEPTH OF BOOLEAN NETWORKS WITH A FAN-IN CONSTRAINT

F. P. Preparata, D. E. Muller, and A. B. Barak

December 15, 1975

Abstract

In this paper we present a family of techniques for the design of combinational networks whose objective is the reduction of the number of levels, subject to a constraint on the fan-in of the logic gates. We show that a boolean expression with n literals and involving the connectives AND and OR can be restructured so that the resulting network has depth at most $C_\ell \log_2 n + \delta$, where $\delta < 0.415$ and C_ℓ is 1.81, 1.38, 1.17, and 1 for maximum fan-in ℓ of 2, 3, 4, and 5, respectively. If we additionally require that the amount of equipment of the resulting network be bounded by a linear function of n, it is possible to bound the depth by $2 \log_2 n$ with a fan-in of at most 3.

Index terms: boolean expressions, combinational networks, network depth, number of levels, computational complexity, design algorithms, parallel computation.

Reduction of Depth of Boolean Networks with a Fan-In Constraint#

F. P. Preparata,* D. E. Muller,** and A. B. Barak†

1. *Introduction*

In the design of digital systems it is very important to use combinational networks with small propagation delay in order to take full advantage of the speed of the logical elements. Reduction of propagation delay is achieved by designing networks with small numbers of levels, or depth, since depth and propagation delay are proportional. Obviously, any given boolean function can be realized as a two-level network, if unlimited fan-in of the logic gates is allowed. However, fan-in limitations are a very severe technological constraint and must be accounted for by any design method.

In this paper, we shall describe a family of techniques for the design of boolean networks, whose principal objective is the reduction of network depth. Specifically, we assume that a boolean function be given as an

#This work was supported in part by the Joint Services Electronics Program (U.S. Army, U.S. Navy, and U.S. Air Force) under Contract DAAB-07-72-C-0259.

*Coordinated Science Laboratory and Department of Electrical Engineering, University of Illinois at Urbana, Illinois.

**Coordinated Science Laboratory and Department of Mathematics, University of Illinois at Urbana, Illinois.

†Department of Computer Science, The Hebrew University, Jerusalem, Israel.

expression E involving the connectives AND and OR, with complements appearing only on the variables. The design methods are procedures for restructuring the given expression E into equivalent expressions whose corresponding networks have bounded depth. Different methods will be proposed depending on the maximum allowable fan-in of the logic gates used.

A main property of our techniques is that all literals of an expression E, i.e., all appearances of variables in E, are treated as distinct variables; therefore, the relevant performance parameters, such as depth and equipment, will be bounded in terms of the number of literals rather than of the number of variables. It must also be pointed out that our methods are more appropriately applicable to the restructuring of expressions where each variable appears a small number of times.

Related to this problem is the construction of design schemes of boolean networks whose number of levels is bounded by a function of the number of variables. The latter problem was studied some years ago by Spira [1] and Preparata and Muller [2], among others, and the best known result is a recent construction due to McColl and Paterson [3]. The techniques described in this paper, however, are inspired by those used for the restructuring of arithmetic expressions [4],[5],[6]. Brent *et al.* [5] were the first to suggest that their scheme for the parallel evaluation of division-free expressions could be applied to logical design, based on the analogy which makes addition and multiplication of numbers correspond to

disjunction and conjunction of boolean variables, respectively. They showed that when employing gate networks with a fan-in of two, the minimum depth of any network for the realization of the function associated with E, is no larger than C $\log_2 n$, where n is the number of literals in E and C = 2.465... . Subsequently, the constant C was reduced to 2 by Barak and Shamir [7] and to 1.81 by Preparata and Muller [8]. Some reservations were expressed in [8] about the practical significance of restricting the fan-in to the value of two, since larger values are technologically quite feasible, but it was felt that the techniques presented there could be applicable to the practical problem. It is the purpose of this paper to substantiate that expectation, by showing that increasingly smaller values of C can be achieved by letting the maximum gate fan-in grow.

In the next section we shall describe restructuring procedures for depth reduction, using various maximum fan-in values, and calculate an upper-bound to the depth achievable in each case. In Section 3 we shall describe a procedure for upper-bounding network depth subject to the additional condition that network size be a linear function of the number of literals in the original expression.

2. *Design methods for depth reduction*

We shall now describe a family of design algorithms which can be used to restructure a given boolean expression, so that the resulting equivalent expression has as small depth as possible. The choice of algorithm will depend on the maximum allowable fan-in of the logical gates to be used in the network construction.

Let E be a boolean expression involving the connectives AND and OR, denoted respectively by the symbols "·" and "+". The *weight* of E, denoted by $|E|$, is the number of literals in E. During restructuring we shall treat the original literals of E as distinct variables. Let $c_1(E)$ denote the minimum depth of any combinational network with two outputs E' and E" so that (E' + E") is equivalent to E. Similarly, let $c_2(E)$ be the minimum depth of any combinational network with two outputs E* and E** so that E* · E** is equivalent to E. We then define $t_2(E) \triangleq \max(c_1(E), c_2(E))$. Finally, we let t(E) denote the minimum depth of any network realizing the function represented by E. The integer ℓ denotes an upper-bound to the permissible fan-in of the logic gates used.

We shall now give three design algorithms which are applicable to $\ell = 3$, $\ell = 4$, and $\ell \geq 5$, respectively. The techniques we make use of are quite general, and restructuring algorithms can be developed for larger values

of ℓ along similar lines, although very large fan-ins are not practical. Without loss of generality, an expression E is assumed to be given as a binary tree. Given any subexpression F of E, suppose we replace F in E with a *free variable* x and let G be the resulting expression. We then define the *composition* of G and F with respect to x, written $E = G \circ F$, as the expression obtained by substituting F for x in G. The expression G can be expanded around x as $G_1x + G_2$ or as $(G_3 + x) \cdot G_4$, where G_1, G_2, G_3, and G_4 are boolean expressions and $|G_j| \leq |G|$ for $j = 1,\ldots,4$. Define the sequences $L_1,\ldots,L_r$ and $T_0,\ldots,T_r$ of subexpressions of G according to the following recursive equations:

(i) $T_0 = G$;

(ii) $T_i = L_{i+1}\theta_i T_{i+1}$, with $\theta_i \in \{\cdot,+\}$ and T_{i+1} is the subexpression containing x for $i = 1,\ldots,r-1$, and $T_r = x$.

Note that the sequence $\theta_0,\ldots,\theta_{r-1}$ defines a path from the root to x in the binary tree associated with E. As we pointed out in [8], we have

$$G_1 = \prod_{\theta_i=\cdot} L_{i+1} \qquad G_3 = \sum_{\theta_i=+} L_{i+1}$$

that is, the expressions G_1 and G_2 contain no common literal. It follows that either $|G_1| \leq |G|/2$ or $|G_3| \leq |G|/2$. Without loss of generality and for simplicity of presentation, assuming that E is expressed as $A \circ (B\ \theta\ C)$ with $\theta \in \{\cdot,+\}$, in the sequel we shall designate as "+" the operation which allows us to write $E = A_1(B\ \theta\ C) + A_2$, with $|A_1| \leq |A|/2$.

Since E may be regarded as a binary tree, we shall make extensive use of a decomposition technique illustrated in the following lemma on binary trees, and originally due to Brent *et al.* [5], which we adapt to our case and state without proof.

Lemma 1. Let E be a boolean expression and q a real number in the range $1 \leq q \leq |E|$. Then E can be algorithmically expressed as $A \circ (B \; \theta \; C)$ where A, B, and C are expressions with no common literal, $(B \; \theta \; C)$ denotes one of the expressions $(B \cdot C)$ or $(B + C)$, and $|B| \leq |C| < q$ while $|B| + |C| \geq q$.

We shall now describe three design procedures, associated with Lemmas 2, 3, and 4, corresponding to the cases $\ell \geq 5$, $\ell = 4$, and $\ell = 3$, respectively. The case $\ell \geq 5$ is likely to have the highest practical value since logic gates with a fan-in of 5 are commonly available. The other algorithms are of interest for design situations with more stringent fan-in constraint, since they yield a noticeable delay improvement over the result $t(E) \leq 1.81 \log_2|E|$ which we established in the past for $\ell = 2$ [8].

Each of the following design procedures achieves in a constructive way the extensions of two inductive hypotheses, referred to in each case as P1 and P2, which respectively specify upper-bounds to $t(E)$ and $t_2(E)$. The validity of the procedures is proved in "assertions" following the individual steps.

Lemma 2. Let E be a boolean expression with $|E| \geq 1$. Then for $\ell \geq 5$ $t(E) \leq \log_2|E| + \delta_5$, where $\delta_5 = \log_2 4/3$.

Proof: We assume inductively

P1. If $|E| < n$, then $t(E) \leq \max(\log_2|E| + \delta_5, 0)$.

P2. If $|E| < n$, then $t_2(E) \leq \max(\log_2|E|, 0)$.

By constructing a few cases it is easily seen that the induction may be started with n = 4. The extension of P1 is provided by the following algorithm, where $|E| = n$.

Step 1. Using Lemma 1, decompose E as $A \circ (B \theta C)$ with $|B \theta C| \geq n/3$ and $|B| \leq |C| < n/3$.

Assertion: $|A| = |E| - |B \theta C| \leq 2n/3$. We express E as $A_1(B \theta C) + A_2$. Since $|B| \leq |C| < n/3$ and $|A_1| \leq |A|/2 \leq n/3$, by P2 we see that $t_2(A_1), t_2(B), t_2(C) \leq \log_2(n/3) = \log_2 n - 2 + \delta_5$. Similarly $|A_2| \leq |A| \leq 2n/3$ yields, by P2, $t_2(A_2) \leq \log_2 n - 1 + \delta_5$.

Step 2. If θ is "+", write $A_1 = A_1'A_1''$, $B = B'B''$, $C = C'C''$, and $A_2 = A_2' + A_2''$ and restructure E as

$$E = A_1'A_1''B'B'' + A_1'A_1''C'C'' + A_2' + A_2''; \text{ halt.}$$

Assertion: Each of these four terms is computable with at most $\lfloor \log_2 n - 1 + \delta_5 \rfloor$ levels so that $t(E) \leq \log_2 n + \delta_5$.

Step 3. (θ is "$\cdot$"). If $|B| \leq n/4$, write $A_1 = A_1'A_1''$, $C = C'C''$, and $A_2 = A_2' + A_2''$ and restructure E as

$E = A_1'A_1''BC'C'' + A_2' + A_2''$; halt.

Assertion: $|B| \leq n/4$ implies $t(B) \leq \log_2 n - 2 + \delta_5$, by P1; thus each of the three terms is computable with at most $\lfloor \log_2 n - 1 + \delta_5 \rfloor$ levels so that $t(E) \leq \log_2 n + \delta_5$.

Step 4. (θ is "$\cdot$", $|B| > n/4$). Restructure E as $A_1B'B''C'C'' + A_2$; halt.

Assertion: $|B| > n/4$ implies that also $|C| > n/4$, whence $|A| = |E| - |B| - |C| < n - n/4 - n/4 = n/2$. Thus $|A_1| < n/4$ and $|A_2| < n/2$, and, by P1, we have $t(A_1) \leq \log_2 n - 2 + \delta_5$ and $t(A_2) \leq \log_2 n - 1 + \delta_5$. Thus, each of the terms is computable with at most $\lfloor \log_2 n - 1 + \delta_5 \rfloor$ levels, yielding $t(E) \leq \log_2 n + \delta_5$.

To extend P2 to $|E| = n$, we shall assume that the expression E is restructured as E' + E", where the choice of which operation to designate as "+" is arbitrary, a fact we shall make use of. The extension is provided by the following algorithm.

Step 1. Using Lemma 1, decompose E as $A \circ (B \theta C)$ with $|B \theta C| \geq n/2$ and $|B| \leq |C| < n/2$.

Assertion: $|A| = |E| - |B\ \theta\ C| \leq n - n/2 = n/2$. Since $|A_1| \leq |A| \leq n/2$ and also $|B| \leq |C| < n/2$, we obtain, by P2, $t_2(A_1), t_2(B), t_2(C) \leq \log_2 n - 1$.

Step 2. If θ is "+", write $A_1 = A_1'A_1''$, $C = C'C''$ and set $E' \leftarrow A_1'A_1''C'C''$, $E'' \leftarrow A_1B + A_2$; halt.

Assertion: Since $t_2(A_1), t_2(C) \leq \log_2 n - 1$, E' is computable with at most $\log_2 n$ levels. Notice that $A_1B + A_2$ is an expression obtained by replacing $(B\ \theta\ C)$ with B in the original E. Moreover, $|A_1B + A_2| = |E| - |C| \leq n - n/4 = 3n/4$, since $|C| \geq |B\ \theta\ C|/2 \geq n/4$. Thus, by P1, $t(E'') \leq \log_2 \frac{3}{4} n + \delta_5 = \log_2 \frac{3}{4}\frac{4}{3} n = \log_2 n$.

Step 3. (θ is "·"). If $|B| \leq 3n/8$, write $A_1 = A_1'A_1''$, $C = C'C''$, and set $E' \leftarrow A_1'A_1''BC'C''$, $E'' \leftarrow A_2$; halt.

Assertion: $|B| \leq 3n/8$ implies, by P1, $t(B) \leq \log_2 \frac{3}{8} n + \delta_5 = \log_2 n - 1$. Since $t_2(A), t_2(C) \leq \log_2 n - 1$, we obtain $t(E') \leq \log_2 n$. Also $|A_2| \leq |A| < n/2$ yields, by P1, $t(A_2) = t(E'') \leq \log_2 n - 1 + \delta_5 < \log_2 n$.

Step 4. (θ is "·" and $|B| > 3n/8$). Write $B = B'B''$, $C = C'C''$ and set $E' \leftarrow A_1B'B''C'C''$, $E'' \leftarrow A_2$; halt.

Assertion: $|B| > 3n/8$ implies $|C| > 3n/8$, whence $|A| = |E| - |B| - |C| < n - 3n/4 < 3n/8$. Thus, by P1, $t(A_1) \leq \log_2 n - 1$. Since $t_2(B), t_2(C) \leq \log_2 n - 1$ and $t(A_2) \leq \log_2 n$, we obtain $t(E'), t(E'') \leq \log_2 n$.

Since in all restructurings the required fan-in never exceeds 5, this concludes the proof of the lemma.||

The proofs of the two following Lemmas develops in a manner very similar to the proof of Lemma 2, except for the numerical values involved. In view of this strong similarity we shall considerably abbreviate the arguments.

Lemma 3. Let α be the largest root of the equation $z^3 = z^2 + 2z - 1$ and let E be a boolean expression with $|E| \geq 1$. Then, for $\ell = 4$,

$$t(E) \leq \log\left(\frac{\alpha^2}{3}|E|\right)/\log \alpha \triangleq C_4 \log_2|E| + \delta_4$$

(since $\alpha = 1.8019$, we have $C_4 = 1.177...$ and $\delta_4 = 0.134...$).

Proof: We assume inductively

P1. If $|E| < n$, then $t(E) \leq C_4 \max(\log_2|E| + \delta_4, 0)$.

P2. If $|E| < n$, then $t_2(E) \leq C_4 \max(\log_2(\alpha - 1)|E| + \delta_4, 0)$.

It is easily seen that the induction may be started with $n = 4$. The extension of P1 is given by the following algorithm, where $|E| = n$.

Step 1. Using Lemma 1, decompose E as $A \circ (B \theta C)$ with

$$|B \theta C| > (1 - 1/\alpha(\alpha - 1))n \text{ and } |B| \leq |C| < (1 - 1/\alpha(\alpha - 1))n.$$

Assertion: $|A| = |E| - |B \theta C| \leq n\alpha^{-1}/(\alpha - 1)$, whence, by P2, $t_2(A_2) \leq C_4 \log_2 n - 1 + \delta_4$. Also, $|A_1| \leq |A|/2 \leq n\alpha^{-1}/2(\alpha - 1) < n\alpha^{-2}/(\alpha - 1)$

yields, by P2, $t_2(A_1) \le C_4\log_2 n - 2 + \delta_4$. Notice that $|B| \le |C| < n(1 - 1/\alpha(\alpha - 1)) = n\alpha^{-2}$, by the defining equation of α, whence, by P1, $t(B), t(C) \le C_4\log_2 n - 2 + \delta_4$.

Step 2. If θ is "+", write $A_1 = A_1'A_1''$, $A_2 = A_2' + A_2''$ and restructure E as $E = A_1'A_1''B + A_1'A_1''C + A_2' + A_2''$; halt.

Assertion: Each of the four terms requires at most $\lfloor C_4\log_2 n - 1 + \delta_4 \rfloor$ levels, whence $t(E) \le C_4\log_2 n + \delta_4$.

Step 3. (θ is "$\cdot$"). Write $A_1 = A_1'A_1''$, $A_2 = A_2' + A_2''$ and set $E = A_1'A_1''BC + A_2' + A_2''$; halt.

Assertion: Each of the three terms requires at most $C_4\log_2 n - 1 + \delta_4$ levels, whence $t(E) \le C_4\log_2 n$. The extension of P1 is completed.

The following algorithm extends P2 to $|E| = n$. Here again we seek a restructuring of the form $E = E' + E''$.

Step 1. Using Lemma 1, decompose E as $A \circ (B\ \theta\ C)$ with $|B\ \theta\ C| \ge n\alpha^{-1}(\alpha - 1)$ and $|B| \le |C| < n\alpha^{-1}(\alpha - 1)$.

Assertion: $|A| \le n - n\alpha^{-1}(\alpha - 1) = n\alpha^{-1}$, whence $t_2(A_1) \le C_4\log_2(\alpha - 1)n - 1 + \delta_4$ and $t(A_2) \le C_4\log_2 n\alpha^{-1} + \delta_4 < C_4\log_2(\alpha - 1)n + \delta_4$. Also $t(B), t(C) \le C_4\log_2 n\alpha^{-1}(\alpha - 1) + \delta_4 = C_4\log_2(\alpha - 1)n - 1 + \delta_4$.

Step 2. If θ is "+", write $A_1 = A_1'A_1''$ and set $E' \leftarrow A_1'A_1''C$, $E'' \leftarrow A_1B + A_2$; halt.

Assertion: Since $t_2(A_1), t(C) \leq C_4\log_2(\alpha - 1)n - 1 + \delta_4$, then $t(E') \leq C_4\log_2(\alpha - 1)n + \delta_4$. Considering now the expression $A_1B + A_2$, since $|C| \geq |B \ \theta \ C|/2 \geq n\alpha^{-1}(\alpha - 1)/2$, we have $|A_1B + A_2| \leq n - n\alpha^{-1}(\alpha - 1)/2 < n(\alpha - 1)$ (by the defining equation of α) and $t(E'') \leq C_4\log_2(\alpha - 1)n + \delta_4$.

Step 3. (θ is "·"). Write $A_1 = A_1'A_1''$, and set $E' \leftarrow A_1'A_1''BC$, $E'' \leftarrow A_2$; halt.

Assertion: $t_2(A_1), t(B), t(C) \leq C_4\log_2(\alpha - 1)n - 1 + \delta_4$ and $t(A_2) \leq C_4\log_2(\alpha - 1)n + \delta_4$ yield the desired result. This completes the extension of P2.

To conclude the proof of the lemma, notice that in no restructuring a fan-in larger than 4 has been used. ||

Lemma 4. Let β be the positive root of the equation $z^4 = 2z^2 + 2$ and let E be a boolean expression with $|E| \geq 1$. Then for $\ell = 3$,

$$t(E) \leq \log|E|/\log \beta \triangleq C_3\log_2|E|$$

(since $\beta = 1.6528...$, we have $C_3 = 1.379...$).

Proof: We assume inductively:

P1. If $|E| < n$, then $t(E) \leq C_3 \log_2 |E|$,

P2. If $|E| < n$, then $t_2(E) \leq C_3 \log_2 (\beta |E|/2)$.

In this case it is seen that the induction can be started with $n = 5$. The extension of P1 is provided by the following algorithm, where $|E| = n$.

Step 1. Using Lemma 1, decompose E as $A \circ (B \; \theta \; C)$ with

$$|B \; \theta \; C| \geq n(1 - 2\beta^{-2}) \text{ and } |B| \leq |C| < n(1 - 2\beta^{-2}).$$

Assertion: $|A| = n - n(1 - 2\beta^{-2}) = 2n\beta^{-2}$, whence $|A_1| \leq n\beta^{-2}$ and $|A_2| \leq 2n\beta^{-2}$. Thus $t(A_1) \leq C_3 \log_2 n - 2$ and $t_2(A_2) \leq C_3 \log_2 \frac{\beta}{2} 2n\beta^{-2} = C_3 \log_2 n - 1$. Also, since $(1 - 2\beta^{-2}) = 2\beta^{-4} < \beta^{-2}$, we have $t(B), t(C) \leq C_3 \log_2 n - 2$ and $t_2(C) \leq C_3 \log_2 \frac{\beta}{2} 2n\beta^{-4} = C_3 \log_2 n - 3$.

Step 2. If θ is "$\cdot$", write $A_2 = A_2' + A_2''$ and set $E = A_1 BC + A_2' + A_2''$; halt.

Assertion: Each of these three terms is computable with at most $\lfloor C_3 \log_2 n - 1 \rfloor$ levels whence $t(E) \leq C_3 \log_2 n$.

Step 3. (θ is "+"). If $|B| \leq \beta^{-3} n$, then write $C = C' + C''$, $A_2 = A_2' + A_2''$ and set $E = A_1(B + C' + C'') + A_2' + A_2''$; halt.

Assertion: $|B| \leq \beta^{-3} n$ implies $t(B) \leq C_3 \log_2 n - 3$, and, since $t_2(C) \leq C_3 \log_2 n - 3$ we obtain $t(B + C' + C'') \leq C_3 \log_2 n - 2$. This and the upper-bounds to $t(A_1)$ and $t_2(A_2)$ yield $t(E) \leq C_3 \log_2 n$.

Step 4. (θ is "+" and $|B| > \beta^{-3}n$). Set $E = A_1B + A_1C + A_2$ and halt.

Assertion: $|B| > \beta^{-3}n$ implies $|C| > \beta^{-3}n$ and $|A| < n - 2\beta^{-3}n < n\beta^{-1}$, by the defining equation of β, whence $t(A_2) \leq C_3\log_2 n - 1$. Also $|A_1| \leq |A|/2 < n\beta^{-1}/2 < n\beta^{-2}$ yields $t(A_2) \leq C_3\log_2 n - 2$. Therefore each of the terms requires no more than $C_3\log_2 n - 1$ levels, so that $t(E) \leq C_3\log_2 n$. This completes the extension of P1.

The following algorithm extends P2 to $|E| = n$.

Step 1. Using Lemma 1, decompose E as $A \circ (B\ \theta\ C)$ with

$$|B\ \theta\ C| \geq n/2 \text{ and } |B| \leq |C| < n/2.$$

Assertion: $|A| \leq n - n/2 = n/2$, so if $A \circ (B\ \theta\ C)$ is written as $A_1(B\ \theta\ C) + A_2$ then $t(A_1), t(A_2), t(B), t(C) \leq C_3\log_2\frac{\beta}{2}\, n\beta^{-1} = C_3\log_2\frac{\beta}{2}\, n - 1$.

Step 2. If $|C| \leq n\beta^{-1}/2$, set $E' \leftarrow A_1(B\ \theta\ C)$ and $E'' \leftarrow A_2$; halt.

Assertion: $|C| \leq n\beta^{-1}/2$ implies $|B| < n\beta^{-1}/2$, whence $t(B), t(C) \leq C_3\log_2\frac{\beta}{2}\, n - 2$, so that $t(B\ \theta\ C) \leq C_3\log_2 \frac{\beta}{2}\, n - 1$ and $t(E') \leq C_3\log_2 \frac{\beta}{2}\, n$.

Step 3. ($|C| > n\beta^{-1}/2$). If θ is "$\cdot$", set $E' \leftarrow A_1BC$ and $E'' \leftarrow A_2$; halt.

Assertion: $t(A_1), t(B), t(C) \leq C_3\log_2 \frac{\beta}{2}\, n - 1$ yield $t(E') \leq C_3\log_2 \frac{\beta}{2}\, n$.

Step 4. ($|C| > n\beta^{-1}/2$, θ is "+"). Set $E' \leftarrow A_1C$, $E'' \leftarrow A_1B + A_2$; halt.

Assertion: From $t(A_1), t(C) \leq C_3 \log_2 \frac{\beta}{2} n - 1$ we have $t(E') \leq C_3 \log_2 \frac{\beta}{2} n$. With regard to the expression $A_1B + A_2$, since $|C| > n\beta^{-1}/2$, we have $|A_1B + A_2| = |E| - |C| \leq n - n\beta^{-1}/2 < n\beta/2$, whence $t(E'') \leq C_3 \log_2 \frac{\beta}{2} n$. This completes the extension of P2.

To complete the proof of the lemma, we notice that in no restructuring has a fan-in larger than 3 been used.||

The three preceding results and the result of [8] yield the following theorem.

Theorem 1 Let E be a boolean expression. Then, depending upon the maximum allowed fan-in ℓ of the logic gates used, the boolean function described by E is realizable by a logical network requiring no more than $C_\ell \log_2 |E| + \delta_\ell$ levels where the constant C_ℓ is 1.81, 1.38, 1.17, and 1, and the constant δ_ℓ is 0, 0, 0.134, and 0.415 for $\ell = 2$, $\ell = 3$, $\ell = 4$, and $\ell \geq 5$, respectively.

It seems unlikely that a significantly smaller constant C_ℓ can be obtained for any practical values of ℓ. Nevertheless, it can be shown theoretically that C_ℓ can be made to approach zero as ℓ approaches infinity. To see this, let ε be any positive number less than 1. We choose the fan-in ℓ to be the least integer no smaller than $3^{2^{2/\varepsilon}}$ and

$\delta_\ell = 2 - \varepsilon\log_2 3$, and show that $t(E) \leq \varepsilon\log_2|E| + \delta_\ell$ using this fan-in. As before, assume inductively the result holds when $|E| < n$ for some given integer n. This can certainly be verified initially when $n = 4$. Next, take $|E| = n$ and use Lemma 1 with $q = n2^{-2/\varepsilon}$ to decompose E as $A \circ (B\ \theta\ C)$ where $|B\ \theta\ C| \geq n2^{-2/\varepsilon}$ and $|B| \leq |C| < n2^{-2/\varepsilon}$. Writing E as $A_1BC + A_2$ if θ is "$\cdot$" or as $A_1B + A_1C + A_2$ if θ is "+", we can further decompose A_1 and A_2 if either or both have weights as great as $n2^{-2/\varepsilon}$. For these decompositions, we use the same q in Lemma 1 as before. Continuing in this way, we see that it is possible to write E as a sum of products in which each factor has weight less than $n2^{-2/\varepsilon}$. We check that the number of products and also the number of terms in each product can be no greater than $3^{2^{2/\varepsilon}} \leq \ell$, so the products can be formed simultaneously in one level and the sum in a second level. Hence, $t(E) \leq \varepsilon\log_2 n2^{-2/\varepsilon} + \delta_\ell + 2 = \varepsilon\log_2 n + \delta_\ell$ and the inductive hypothesis is justified.

Before closing this section, we consider the determination of the upper-bound to the amount of equipment required by the design methods outlined above. This determination can be carried out by techniques which have been described elsewhere[5,6] and will only be sketched here. As a measure of the amount of equipment we may consider either the number of gates or the number of gate inputs; it is easily seen, however, that the bounds we would obtain for these two measures have the same rate of growth as a function of $|E|$. Referring to any of the algorithm pairs associated with Lemmas 2, 3, and 4, we assume inductively that, for a given integer n

and $|E| < n$, the equipment required to realize E is at most $k_1|E|^{\xi}$ or $k_2|E|^{\xi}$, depending upon whether E is restructured as one or as two expressions, with constants $k_1 > 0$, $k_2 > 0$, $\xi > 1$. We then take $|E| = n$, and, for each of the restructuring forms presented in the algorithms, we obtain an inequality involving k_1, k_2, and ξ. For example, if E is restructured as $A_1'A_1''BC'C'' + A_2' + A_2''$ (Step 3 of the algorithm for P1 in Lemma 2) we have the inequality

$$k_2|A_1|^{\xi} + k_2|A_2|^{\xi} + k_1|B|^{\xi} + k_2|C|^{\xi} \leq k_1|E|^{\xi}$$

where the left side must be maximized in the domain of $|A_1|$, $|A_2|$, $|B|$, and $|C|$, treated as real variables. The exponent ξ may then be chosen as the least one for which all the inequalities produced by an algorithm pair are satisfied. For example, for the case of $\ell \geq 5$, an upper-bound to the amount of equipment is $O(|E|^{1.55})$.

In general, since in all cases separate subnetworks appear to be necessary for realizing A_1 and A_2, the resulting overlap yields a bound in the equipment which is superlinear in $|E|$. In the next section we shall present a design algorithm which yields networks whose equipment is guaranteed to be $O(|E|)$, but whose delay is greater than that for the unrestricted case.

3. A design method yielding equipment $O(|E|)$

Let E be a boolean expression, and let $d_1(E)$ denote the minimum depth of any combinational network with two outputs E' and E", so that (E' + E") is equivalent to E, and with no more than $6|E| - 6$ gate inputs. Similarly, let $d_2(E)$ denote the minimum depth of any network with two outputs E* and E**, so that E* · E** is equivalent to E, and with no more than $6|E| - 6$ gate inputs. We then define $\tau_2(E) \triangleq \max(d_1(E), d_2(E))$.

In an analogous manner, let G be a boolean expression with a free variable x so that we may restructure G into the form $G_1x + G_2$. Then define $\tau(G)$ as the minimum number such that a network can be constructed which simultaneously realizes the functions G_1 and G_2 using no more than $\tau(G) - 1$ and $\tau(G)$ levels, respectively, and with no more than $6|G| - 6$ gate inputs.

We can now prove the following lemma:

Lemma 5. Let E and G be boolean expressions and let G contain a free variable x. Then for fan-in $\ell = 3$ we have

(i) $\tau_2(E) \leq \max(2\log_2|E| - 1, 0)$

(ii) $\tau(G) \leq 2 \log_2|G| + 1$.

Proof: Propositions (i) and (ii) are seen to hold when $|E| < 4$ and $|G| < 4$. We formulate the following inductive hypotheses, starting with $n = 4$.

P1. If $|E| < n$, then $\tau_2(E) \leq \max(2 \log_2|E| - 1, 0)$.

P2. If $|G| < n$, then $\tau(G) \leq 2 \log_2|G| + 1$.

To extend P1 to $|E| = n$, we shall assume that E is realized as $E' + E''$ and leave free the designation of which operation is "+". We shall then use the following algorithm:

Step 1. Using Lemma 1, decompose E as $A \circ (B \theta C)$ with $|B \theta C| \geq n/2$, $|B| \leq |C| < n/2$.

Assertion: $|A| = |E| - |B \theta C| \leq n/2$. The expression A can be restructured as $A_1 x + A_2$. Then, by P2, $\tau(A) \leq 2 \log n - 1$. Also $|B| \leq |C| < n/2$ imply, by P1, $\tau_2(B), \tau_2(C) \leq 2 \log_2 n - 3$.

Step 2. If $|B| < n(\sqrt{2})^{-3}$, write $B = B'B''$, $C = C' \theta C''$, and set $E' \leftarrow A_1((B' B'') \theta C' \theta C'')$, $E'' \leftarrow A_2$; halt.

Assertion: $|B| < n(\sqrt{2})^{-3}$ implies, by P1, $\tau_2(B) \leq 2 \log_2 n - 4$. Since $\tau_2(C) \leq 2 \log_2 n - 3$, $(B'B'') \theta C' \theta C''$ has depth no more than $2 \log_2 n - 2$, so that E' has depth no more than $2 \log_2 n - 1$; also, the depth of E'' is at most $2 \log_2 n - 1$. Let $Q(E',E'')$ be the number of gate inputs of the network realizing E' and E'', with analogous definitions of $Q(B',B'')$, $Q(C',C'')$, and $Q(A_1,A_2)$. Since at most 7 gate inputs are required to combine A_1, A_2, B', B'', C', and C'' into E' and E'', and by the inductive

assumptions P1 and P2, $Q(A_1,A_2) \leq 6|A| - 6$, $Q(B',B'') \leq 6|B| - 6$, and $Q(C',C'') \leq 6|C| - 6$, we have

$$\begin{aligned} Q(E',E'') &= Q(A_1,A_2) + Q(B',B'') + Q(C',C'') + 7 \\ &\leq 6|A| - 6 + 6|B| - 6 + 6|C| - 6 + 7 = 6(|A| + |B| + |C|) - 18 + 7 \\ &= 6|E| - 11 < 6|E| - 6. \end{aligned}$$

In order to abbreviate subsequent analogous arguments, we notice that all that is needed to prove the upper-bound on the number of gate inputs is that no more than 12 gate inputs are used in the restructuring combination.

Step 3. ($|B| \geq n(\sqrt{2})^{-3}$). Write $B = B'B''$, and $C = C'C''$ and set $E'' \leftarrow A_2$. If θ is "+" set $E' \leftarrow A_1C'C'' + A_1B'B''$, else set $E' \leftarrow (A_1C'C'')B'B''$; halt.

Assertion: $|B| \geq n(\sqrt{2})^{-3}$ implies $|C| \geq n(\sqrt{2})^{-3}$, whence $|A| = |E| - |B| - |C| \leq n(1 - 2(\sqrt{2})^{-3}) < n(\sqrt{2})^{-3}$. Thus, by P2, $\tau(A) \leq 2\log_2 n - 2$. It follows that $A_1C'C''$ and $A_1B'B''$ both have depth at most $2\log_2 n - 2$; similarly $(A_1C'C'')B'B''$ has depth at most $2\log_2 n - 1$, whence in all cases E' has depth at most $2\log_2 n - 1$. We also note that $E'' = A_2$ has depth at most $2\log_2 n - 2$, and that at most 8 gate inputs are needed for the restrucuring combination, thus completing the extension of P1.

To extend P2 to $|G| = n$ we need a simple lemma on binary trees, also due to Brent [4], which we state without proof:

Let G be a boolean expression with a free variable x and let q be a real number in the range $1 \leq q \leq |G|$. Then G can be algorithmically expressed as $A \circ (B \theta C)$ so that $|B \theta C| \geq q$, C contains the free variable x and $|C| < q$.

We can now constructively extend P2.

Step 1. Using Lemma 6, decompose G as $A \circ (B \theta C)$ with $|B \theta C| \geq n/2$, x in C, and $|C| < n/2$.

Assertion: $|A| = |G| - |B \theta C| \leq n/2$. Thus, by P2, $\tau(A) \leq 2 \log_2 n - 1$. Restructuring C as $C_1 x + C_2$, by hypothesis P2 we also have $\tau(C) \leq 2 \log_2 n - 1$. However, we can only bound $|B|$ as $|B| < n$, whence, by P1, $\tau_2(B) \leq 2 \log_2 n - 1$.

Step 2. If θ is "+", write B as B'B", set $G_1 \leftarrow A_1 C_1$, $G_2 \leftarrow A_1 C_2 + A_1 B'B'' + A_2$; halt.

Assertion: Since both A_1 and C_1 have depth at most $2 \log_2 n - 2$, G_1 has depth at most $2 \log_2 n - 1$. As to G_2, each of the three terms in its expression has depth at most $2 \log_2 n$, whence G_2 has depth $2 \log_2 n + 1$. Since only 10 gate inputs are needed by the restructuring combination we have $\tau(G) \leq 2 \log_2 n + 1$.

Step 3. (θ is "$\cdot$"), If $|B| \leq n(\sqrt{2})^{-1}$, write $B = B'B''$ and set

$G_1 \leftarrow (A_1C_1)B'B''$, $G_2 \leftarrow A_1C_2(B'B'') + A_2$; halt.

Assertion: $|B| \leq n(\sqrt{2})^{-1}$ implies, by P1, $\tau_2(B) \leq 2\log_2 n - 2$. Since, by the assertion on Step 2, A_1C_1 has depth no more than $2\log_2 n - 1$, G_1 has depth bounded by $2\log_2 n$. We also recognize that G_2 has depth at most $2\log_2 n + 1$, and since exactly 12 gate inputs are used for the restructuring combination, we conclude that $\tau(G) \leq 2\log_2 n + 1$.

Step 4. (θ is "$\cdot$", $|B| > n(\sqrt{2})^{-1}$). Write $B = B'B''$ and set

$G_1 \leftarrow (A_1C_1)B'B''$, $G_2 \leftarrow (A_1C_2)B'B'' + A_2$; halt.

Assertion: $|B| > n(\sqrt{2})^{-1}$ implies $|C| < |G| - |B| < n(1 - (\sqrt{2})^{-1}) < n(\sqrt{2})^{-3}$, i.e., by P2, $\tau(C) \leq 2\log_2 n - 2$. From this and $\tau(A) \leq 2\log_2 n - 1$, $\tau_2(B) \leq 2\log_2 n - 1$ we conclude that G_1 and G_2 have depth at most $2\log_2 n$ and $2\log_2 n + 1$, respectively. Since 12 gate inputs are used for the restructuring combination, we have proved $\tau(G) \leq 2\log n + 1$, thereby completing the extension of P2 and the proof of the lemma.||

We now have the following theorem:

Theorem 2. A boolean expression E can be realized by a network having at most $2\log_2|E|$ levels and $6|E| - 6$ gate inputs, and using a fan-in of at most 3.

Proof: Since, by the extension of P1 in Lemma 5, $\tau_2(E) \leq 2 \log_2|E| - 1$, and one more level is needed to realize the expression in $E' + E''$, the theorem follows.||

A related results was obtained by Brent [9], who showed that for fan-in of 2 the coefficient of $\log_2|E|$ need not be larger than 3 if equipment linear in $|E|$ is desired. Thus, a sizable penalty in depth appears to be necessary when the fan-in restriction is tightened. An interesting open question is whether by relaxing the fan-in restriction to 4, 5, or more it is possible to obtain a coefficient significantly lower than 2 while maintaining equipment linear in $|E|$.

4. *References*

1. P. M. Spira, "On the Time Necessary to Compute Switching Functions," IEEE Transactions on Computers, Vol. C-20, Jan 1971, pp. 104-105.

2. F. P. Preparata and D. E. Muller, "On the Delay Required to Realize Boolean Functions," IEEE Transactions on Computers, Vol. C-20, April 1971, pp. 459-461.

3. W. F. McColl and M. S. Paterson, "The Depth of all Boolean Functions," Tech. Rep. Dept. of Computer Science, University of Warwick, Coventry, England, Aug. 1975.

4. R. P. Brent, "The Parallel Evaluation of General Arithmetic Expressions," Journal of the ACM, Vol. 19, No. 2, April 1974, pp. 201-206.

5. R. P. Brent, D. J. Kuck, and K. Maruyama, "The Parallel Evaluation of Arithmetic Expressions Without Division," IEEE Transactions on Computers, Vol. C-22, No. 5, pp. 532,534.

6. D. E. Muller and F. P. Preparata, "Restructuring of Arithmetic Expressions for Parallel Evaluation," to appear in the Journal of the ACM (Also available as Report R-676, Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana, Illinois, April 1975).

7. A. Barak and E. Shamir, "On the Parallel Evaluation of Boolean Expressions," Report No. 17, Department of Computer Science, The Hebrew University of Jerusalem, Jerusalem, Israel, March 1975.

8. F. P. Preparata and D. E. Muller, "Efficient Parallel Evaluation of Boolean Expressions," to appear in the IEEE Transactions on Computers.

9. R. P. Brent, "The parallel evaluation of arithmetic expressions in logarithmic time," in Complexity of Sequential and Parallel Numerical Algorithms, Academic Press, New York, 1973.