PUBLICATIONS DE L'INSTITUT MATHÉMATIQUE
Nouvelle série, tome 64 (78), 1998, 2–8

# ON FREE BOOLEAN VECTORS

**Žarko Mijajlović**

*Communicated by Mileva Prvanović*

**Abstract.** Let $f(x_1, x_2, \dots, x_n)$ be a Boolean expression in $n$ variables $x_1, x_2, \dots, x_n$. A method for checking if the identity $f(x_1, x_2, \dots, x_n) = 1$ is valid for all boolean values of $x_1, x_2, \dots, x_n$ is proposed, based on the parallel structure of a computer k-bit processor. We give a construction of $n$ boolean vectors $b_1, b_2, \dots, b_n$ of the size $2^n$ with the following property:

*If* $f(b_1, b_2, \dots, b_n) = 1$, *then* $f(x_1, x_2, \dots, x_n)$ *is identically equal to one.*

In this case, the necessary number of computing steps for checking the identity $f(b_1, b_2, \dots, b_n) = 1$ is $2^{n-k}$, to be compared with the number of $2^n$ computing steps in the usual table-checking procedure. The complete characterization of vectors $b_1, b_2, \dots, b_n$ is given.

It is usually neglected in the design and program implementation of algorithms that even standard computer processors are capable of performing certain parallel operations. Examples of this kind include logical operations which can be performed bitwise, i.e. by use of all register bits in one processor cycle. The aim of this paper is to present an algorithm for computing values of Boolean functions using parallel computing capability of a $k$-bit register processor. The mathematical background of this analysis are free finitely generated Boolean algebras.

## 1. Finite free Boolean algebras

Let us review first some facts, terminology and notation. The set of positive integers is denoted by $N$, while $|X|$ denotes the cardinal number of a set $X$. Algebras will be denoted by bold capital letters (e.g. $\mathbf{A}$), while their domains by ordinary capital letters (e.g. $A$). If $F(x_1, x_2, \dots, x_n)$ is an algebraic expression (term) over an algebraic language $L$, $\mathbf{A}$ an algebra of $L$ and $a_1, a_2, \dots, a_n \in A$, then $F^{\mathbf{A}}(a_1, a_2, \dots, a_n)$ denotes the value of $F$ in $\mathbf{A}$ for variables $x_1, x_2, \dots, x_n$ substituted with $a_1, a_2, \dots, a_n$. If a set $X \subseteq A$ generates $\mathbf{A}$ then we write $\mathbf{A} = \langle X \rangle$. In the following we shall need the notion of a free algebra. Let $\mathcal{M}$ be an algebraic variety of algebras over a language $L$. An algebra $\mathbf{A} \in \mathcal{M}$ is said to be free for $\mathcal{M}$ over

---

*AMS Subject Classification* (1991): 03G05, 06E30, 94C10 a set of free generators $X \subseteq A$ if for every $\mathbf{B} \in \mathcal{M}$ and every map $f: X \to B$ there is a homomorphism $h: \mathbf{A} \to \mathbf{B}$ such that $f \subseteq h$. As it is well known (G. Birkhoff), every algebraic variety has a free algebra over the set $X$ as a free set of generators for an arbitrary set $X$. As an illustration we prove the following theorem.

THEOREM 1.1. *Let $\mathcal{M}$ be an algebraic variety over an algebraic language $L$, and $\mathbf{A} \in \mathcal{M}$ a free algebra over a finite free generator set $X = \{b_1, b_2, \dots, b_n\}$, where $b_1, b_2, \dots, b_n$ are distinct. Further, let*

$$u(x_1, x_2, \dots, x_n) = v(x_1, x_2, \dots, x_n)$$

*be an identity of $L$. If $u^{\mathbf{A}}(b_1, b_2, \dots, b_n) = v^{\mathbf{A}}(b_1, b_2, \dots, b_n)$, then the identity $u = v$ holds on all algebras of $\mathcal{M}$.*

*Proof.* Assume $u^{\mathbf{A}}(b_1, b_2, \dots, b_n) = v^{\mathbf{A}}(b_1, b_2, \dots, b_n)$. Let $\mathbf{B} \in \mathcal{M}$ be an arbitrary algebra and $d_1, d_2, \dots, d_n \in B$. Further, let $f: X \to B$ be defined by $f(b_i) = d_i$, $i = 1, 2, \dots, n$. As $\mathbf{A}$ is free over $X$, there is a homomorphism $h: \mathbf{A} \to \mathbf{B}$ extending $f$. Then

$$\begin{aligned} u^{\mathbf{B}}(d_1, d_2, \dots, d_n) =& u^{\mathbf{B}}(hb_1, hb_2, \dots, hb_n) = hu^{\mathbf{A}}(b_1, b_2, \dots, b_n) = \\ & hv^{\mathbf{A}}(b_1, b_2, \dots, b_n) = v^{\mathbf{B}}(hb_1, hb_2, \dots, hb_n) = \\ & v^{\mathbf{B}}(d_1, d_2, \dots, d_n). \end{aligned}$$

As $d_1, d_2, \dots, d_n$ were arbitrarily selected it follows that $u = v$ holds on $\mathbf{B}$.$\diamondsuit$

Let us consider free finitely generated Boolean algebras. By a Boolean algebra, shortly BA, we mean an algebraic structure $\mathbf{B} = (B, +, \cdot, ^-, 0, 1, )$, where $B$ is the domain of $\mathbf{B}$, and $+, \cdot, ^-$ are usual Boolean operations. A two-element BA is denoted by $\mathbf{2}$, so its domain is $2 = \{0, 1\}$. The fundamental representation theorem of finite BA's states that every finite BA is isomorphic to $\mathbf{2}^n$ for some $n \in N$. If $\mathbf{B}$ is generated by $n$ elements $b_1, b_2, \dots, b_n$, i.e., $\mathbf{B} = \langle b_1, b_2, \dots, b_n \rangle$ then $|B| \leq 2^{2^n}$, so $\mathbf{B}$ is finite. From now on we shall discuss only finite BA's. We shall call a basis of a BA $\mathbf{B}$ a minimal generating set of $\mathbf{B}$. If the basis $X$ is a free generating set of $\mathbf{B}$ then we shall call $X$ a free basis, while the elements of $X$ will be called free vectors. A free Boolean algebra generated by $n$ free vectors will be denoted by $\Omega_n$. First, we shall show that $|B| = 2^{2^n}$ if and only if $\mathbf{B}$ is a free BA over a free generating $n$-element set. In the following we are going to use the following fact:

THEOREM 1.2. (R. Sikorski) *Let $\mathbf{B}$ be a BA and $X \subseteq B$. Then $\mathbf{B}$ is freely generated by $X$ if and only if:*

- $\mathbf{B} = \langle X \rangle$.
- *For any $n \in N$, $b_1, b_2, \dots, b_n \in X$ and $\alpha \in 2^n$, $b_1^{\alpha_1} b_2^{\alpha_2} \cdots b_n^{\alpha_n} \neq 0$.* $\diamondsuit$

As usual, $x^1 = x$ and $x^0 = \bar{x}$. Here are examples of some free generating sets $b_1, b_2, \dots, b_n$:

$1^0$ The Lindenbaum algebra $\mathcal{L}_P$ of the propositional calculus with $n$ propositional letters $P = \{p_1, p_2, \ldots, p_n\}$. If $\varphi$ is a propositional formula over $P$ and $[\varphi]$ its equivalence class under the logical equivalence, then we may take $b_1 = [p_1], b_2 = [p_2], \ldots, b_n = [p_n]$.

$2^0$ Let $I = [0,1]_R$ be the real interval. Then

$$b_i = \{(x_1, x_2, \ldots, x_n) \in I^n : 0 \le x_i \le 1/2\}, \quad i = 1, 2, \ldots, n$$

form a free basis of $\mathbf{B} = \langle b_1, b_2, \ldots, b_n\rangle \subseteq \mathbf{P}(I^n)$.

$3^0$ Let $A, B, C$ be circles in a Venn's diagram, i.e., "circles in the general intersecting setting". Then $\{A, B, C\}$ is a free generating set of BA $\mathbf{B} = \langle A, B, C\rangle \subseteq \mathbf{P}(K)$, where $K$ is a square to which belong $A, B, C$. According to the Theorem 1.1 every proof of a Boolean (and therefore set-theoretical) identity involving up to three variables using Venn's diagrams is mathematically correct. Compare this to the sentence "They (Venn's) diagrams are useful for verifying identities involving operations on sets, but should not be considered tools of rigorous mathematical proof" (cf. [2], p. 34).

$4^0$ Let $c_i$, $i = 0, 1, \ldots, 2^n - 1$, be the binary expansion of the integer $i$ with zeros padded to the left up to the length $n$. Let $M$ be the matrix whose columns are vectors $c_i$, and let $b_i$, $i = 1, 2, \ldots, n$, be binary vectors – rows of the matrix $M$. Then $b_1, b_2, \ldots,$ are free vectors of $\Omega_n$. In case $n = 3$, the matrix $M$ looks like

$$M = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{1.3}$$

thus, $b_1 = 00001111, b_2 = 00110011, b_3 = 01010101$.

THEOREM 1.4 *Let* $\mathbf{B}$ *be a finitely generated BA. Then* $\mathbf{B}$ *is free if and only if it is isomorphic to* $\mathbf{2}^{2^n}$.

*Proof.* Assume $\mathbf{B} = \langle b_1, b_2, \ldots, b_n\rangle$ is freely generated by $b_1, b_2, \ldots, b_n$. If $x \in B$ then $x = \Sigma_{\alpha\in\Gamma_x} b_1^{\alpha_1} b_2^{\alpha_2} \cdots b_n^{\alpha_n}$ for some $\Gamma_x \subseteq 2^n$. Let $y = \Sigma_{\alpha\in\Gamma_y} b_1^{\alpha_1} b_2^{\alpha_2} \cdots b_n^{\alpha_n}$, $\Gamma_y \subseteq 2^n$. Assume that $\Gamma_x \ne \Gamma_y$. Then $\Gamma_x \backslash \Gamma_y \ne \emptyset$, or $\Gamma_y \backslash \Gamma_x \ne \emptyset$, suppose $\Gamma_x \backslash \Gamma_y \ne \emptyset$. Let $\beta \in \Gamma_x \backslash \Gamma_y$. Then by Sikorski's condition $b_1^{\beta_1} b_2^{\beta_2} \cdots b_n^{\beta_n} \ne 0$ and $b_1^{\beta_1} b_2^{\beta_2} \cdots b_n^{\beta_n} \Sigma_{\alpha\in\Gamma_x} b_1^{\alpha_1} b_2^{\alpha_2} \cdots b_n^{\alpha_n} = b_1^{\beta_1} b_2^{\beta_2} \cdots b_n^{\beta_n} \ne 0$, so $x b_1^{\beta_1} b_2^{\beta_2} \cdots b_n^{\beta_n} \ne 0$. On the other hand, $b_1^{\beta_1} b_2^{\beta_2} \cdots b_n^{\beta_n} \Sigma_{\alpha\in\Gamma_y} b_1^{\alpha_1} b_2^{\alpha_2} \cdots b_n^{\alpha_n} = 0$, so $y b_1^{\beta_1} b_2^{\beta_2} \cdots b_n^{\beta_n} = 0$. Hence, $x \ne y$. Therefore, $\mathbf{B}$ has as many elements as there are subsets of $2^n$, so $|B| = 2^{2^n}$.

Now assume that $|B| = 2^{2^n}$. By the above, $\Omega_n$ has $2^{2^n}$ elements, so by the representation theorem for finite Boolean algebras it follows $\mathbf{B} \cong \Omega_n$. $\Diamond$

THEOREM 1.5 $\Omega_{n+1} \cong \Omega_n \times \Omega_n$.

*Proof.* By the previous theorem we have:

$$\Omega_{n+1} \cong \mathbf{2}^{2^{n+1}} \cong \mathbf{2}^{2^n} \times \mathbf{2}^{2^n} \cong \Omega_n \times \Omega_n. \quad \Diamond$$

There is an easy procedure for constructing in the inductive way free vectors of $\Omega_n$ which can be used in a program implementation. Suppose that $u_1, u_2, \dots, u_{n-1}$ are free vectors of $\Omega_{n-1}$. If this sequence is of length 1, then we take $u_1 = 01$. Free vectors $b_1, b_2, \dots, b_n$ of $\Omega_n$ are obtained as follows. $b_1 = 00\dots011\dots1$, i.e., $b_1$ consists of two blocks, the first one is a block of zeros of size $2^{n-1}$, while the second block is a sequence of ones also of size $2^{n-1}$. Further, $b_2 = u_1\hat{\ }u_1, b_3 = u_2\hat{\ }u_2, \dots b_n = u_{n-1}\hat{\ }u_{n-1}$. Here $u\hat{\ }v$ is a concatenation of words $u$ and $v$.

For a real number $x$, let $\lceil x \rceil$ denote the least integer greater than or equal to $x$.

COROLLARY 1.6 *The Boolean algebra $\mathbf{2}^m$ is generated by $\lceil \log_2(m) \rceil$ elements.*

*Proof.* Let $n = \lceil \log_2(m) \rceil$. Then $2^m \leq 2^{2^n}$, so BA $\mathbf{2}^m$ is a homomorphic image of BA $\mathbf{2}^{2^n}$, i.e., of $\Omega_n$. A homomorphism $h: \Omega_n \to \mathbf{2}^m$ is the projection, i.e. the mapping which cuts off the coordinates of vectors in $\mathbf{2}^{2^n}$ (which are of length $2^n$) to obtain vectors which are of length $m$. As $\Omega_n$ is generated by $n$ vectors, say by $b_1, b_2, \dots, b_n$, so is $\mathbf{2}^m$, since $\mathbf{2}^m = \langle hb_1, hb_2, \dots, hb_n \rangle$. ♢

*Example 1.7* Let us determine a generating subset of $\mathbf{2}^5$ of size $\lceil \log_2(5) \rceil = 3$. If in the matrix (1.3) the last three columns are deleted (the homomorphism-projection which "forgets" the three last coordinates), then we obtain vectors $u_1 = 00001, u_2 = 00110, u_3 = 01010$, and $\mathbf{2}^5 = \langle u_1, u_2, u_3 \rangle$. As atoms of $\mathbf{2}^m$ have exactly one bit equal to 1 in vector (binary) representation, by an obvious enumeration of atoms $a_i$, we have $u_1 = a_5, u_2 = a_3 + a_4, u_3 = a_2 + a_4$.

THEOREM 1.8 *Suppose $\Omega_n = \langle u_1, u_2, \dots, u_n \rangle$. Then $u_1, u_2, \dots, u_n$ are free vectors of $\Omega_n$.*

Let $\{b_1, b_2, \dots, b_n\}$ be a free basis of $\Omega_n$, and let the mapping $f$ be defined by $f(b_i) = u_i, i = 1, 2, \dots n$. Then $f$ extends to an endomorphism $h: \Omega_n \to \Omega_n$. As $u_1, u_2, \dots, u_n$ generate $\Omega_n$, $h$ is onto. Since $\Omega_n$ is finite, $h$ is also 1-1, therefore $h$ is an automorphism. Hence $u_1 = hb_1, u_2 = hb_2, \dots, u_n = hb_n$ are free vectors of $\Omega_n$. ♢

If $u_1, u_2, \dots, u_m \in \Omega_n$ and $m < n$ then $\langle u_1, u_2, \dots, u_m \rangle$ is a proper Boolean subalgebra of $\Omega_n$ (since it has at most $2^{2^m} < 2^{2^n}$ elements), so, free vectors of $\Omega_n$ not only form a basis of $\Omega_n$ but also a generating set with the least possible number of elements. Thus it makes sense to introduce a "dimension" function $d$ on a finite Boolean algebra $\mathbf{B}$ as $d\mathbf{B}$ = the least natural number $k$ such that $\mathbf{B}$ has a basis with $k$ elements. By the previous observation, we have $d\Omega_n = n$. In fact we can compute $d$ for every $\mathbf{2}^m$.

THEOREM 1.9 $d\mathbf{2}^m = \lceil \log_2(m) \rceil$.

*Proof.* By Corollary 1.6, Boolean algebra $\mathbf{2}^m$ has a generating set having $\lceil \log_2(m) \rceil$ elements. If $u_1, u_2, \dots, u_k \in \mathbf{2}^m$ and $k < \lceil \log_2(m) \rceil$, then the subalgebra $\langle u_1, u_2, \dots, u_k \rangle \subseteq \mathbf{2}^m$ has at most $2^{2^k} < 2^m$ elements so it is a proper subalgebra of $\mathbf{2}^m$. Thus $\mathbf{2}^m$ does not have a generating subset with fewer elements than $\lceil \log_2(m) \rceil$. ♢

Since atoms of $\mathbf{2}^m$ make a partition of 1, and under every non-zero element of $\mathbf{2}^m$ there is an atom, it follows that the set of atoms $A \subseteq 2^m$ is a basis of $\mathbf{2}^m$ with the maximal number of elements. Hence, if $S \subseteq 2^m$ is any basis of $2^m$, then

$$\lceil \log_2(m) \rceil \leq |S| \leq m \tag{1.10}$$

Now, we shall describe all free base of $\Omega_n$.

THEOREM 1.11 *Let $b_1, b_2, \ldots, b_n$ be free vectors of $\Omega_n$ as defined in Example $4^0$, $b_i = (b_{i1}, b_{i2}, \ldots, b_{i2^n})$, $i = 1, 2, \ldots, n$. Let $M = ||b_{ij}||_{n \times 2^n}$ be a matrix whose rows are formed by coordinates of vectors $b_i$ (see the Example $4^0$). Let $p$ be any permutation of columns of matrix $M$, and let us denote the matrix so obtained by $p * M$. Then*

1. *The rows of $p * M$ form a free basis of $\Omega_n$.*
2. *Every free basis of $\Omega_n$ is obtained as row-vectors of $p*M$ for some permutation $p$.*

*Proof.* To every column $\alpha$ of the matrix $M$ corresponds an atom

$$a_\alpha = b_1^{\alpha_1} b_2^{\alpha_2} \cdots b_n^{\alpha_n}$$

of $\Omega_n$. Thus the permutation $p$ permutes in fact atoms of $\Omega_n$, so it defines an automorphism $h_p$ of $\Omega_n$. Then the rows of $p * M$ are images of free vectors $b_1, b_2, \ldots, b_n$ under $h_p$, hence $h_p b_1, h_p b_2, \ldots, h_p b_n$ also form a free basis of $\Omega_n$.

Now, suppose that $u_1, u_2, \ldots, u_n$ is any free basis of $\Omega_n$. Let the mapping $f$ be defined by $f(b_i) = u_i, i = 1, 2, \ldots, n$. As $u_1, u_2, \ldots, u_n$ generate $\Omega_n$, $f$ extends to an automorphism $h$ of $\Omega_n$ (see the proof of Theorem 1.8). Therefore $h$ permutes atoms of $\Omega_n$, and $h(b_i) = u_i$, hence $u_i$ are rows of the matrix $p * M$, where $p$ is the restriction of $h$ to the set of atoms of $\Omega_n$. $\diamondsuit$

By the above consideration, we see that the number of free basis of $\Omega_n$ is equal to $|\mathrm{Aut}\Omega_n| = (2^n)!$.

## 2. Computation of Boolean formulas by use of free vectors

Let $f\colon \Omega_n \to 2$ be a Boolean function. Representing $f$ by its Boolean values, we may consider that $f = F^{\Omega_n}(b_1, b_2, \ldots, b_n)$, where $F$ is a Boolean expression and $b_1, b_2, \ldots, b_n$ form a free basis of $\Omega_n$. Thus,

- If we want to check that $f$ is identically equal to 1, it suffices to compute $F^{\Omega_n}(b_1, b_2, \ldots, b_n)$ and check if the Boolean values consist of the sequence of 1's. Observe that in the computation, the expression $F^{\Omega_n}(b_1, b_2, \ldots, b_n)$ is evaluated only once. Also, we can obtain this fact by use of the Theorem 1.1.
- On the other hand, the sequence of binary values of $F^{\Omega_n}(b_1, b_2, \ldots, b_n)$ is exactly the sequence of values of the function $f$. If $M = [b_1, b_2, \ldots, b_n]$ is the matrix whose rows are vectors $b_1, b_2, \ldots, b_n$, and $\alpha$ is the $i$th column of

$M$ (so $0 \leq \alpha \leq 2^n - 1$ if $\alpha$ is considered as a binary number), then the $i$th member of the sequence $F^{\Omega_n}(b_1, b_2, \ldots, b_n)$ is exactly $f(\alpha)$. Therefore, $F^{\Omega_n}(b_1, b_2, \ldots, b_n)$ represents $f$ as a vector of the size $2^n$. It should be observed that the DNF (disjunctive normal form) of $F$ is read out from the binary representation of $f$ immediately.

Now we shall consider an algorithm for fast checking whether $f$ is identically equal to 1. Suppose that we have at our disposal a $2^k$-bit processor (for example $2^k = 2^5 = 32$), and that we want to check whether $f(x_1, x_2, \ldots, x_n)$ (for example $n = 30$) is identically equal to 1. According to the previous discussion, it suffices to see whether $f(b_1, b_2, \ldots, b_n) = 1$, where $b_1, b_2, \ldots, b_n$ is a free basis of $\Omega_n$. For the case $2^n \leq 2^k$ one should put $b_1, b_2, \ldots, b_n$ as binary sequences into $2^k$-bit registers and compute $f(b_1, b_2, \ldots, b_n)$. It means that in contemporary computer processors, due to the fact that logical operations are preformed bitwise, i.e. in parallel on the vectors of length $2^k$, the evaluation of $f$ can be done in the time interval of length $t_b = \lambda\mu$, where $\lambda$ is the number of the nodes in the evaluation tree (the expression tree) of the corresponding Boolean expression $F$, and $\mu$ is the processor clock cycle which we may take to be about $5 \cdot 10^{-9}$sec. In the standard table-checking method this time interval would be of the length $t_s = 2^k t_b$, as it is performed on single bits, not on whole vectors. Thus, the proposed method speeds up the evaluation of $f$ by $2^k$ times.

The algorithm works well on functions with small number of variables. Now, let us consider the case when the length of free vectors (i.e. when $f$ have large number of arguments) is greater then the length of processor registers; this is the case when $2^n > 2^k$. In this case each vector $b_i$ should be divided simultaneously into blocks each of size $2^k$. Observe that there are $2^{n-k}$ such blocks. Thus we may consider that $b_j$ consists of blocks $b_i^j, 1 \leq i \leq 2^{n-k}$, each block having $2^k$ bits. Then, in order to to find out whether 0 appears in the binary expansion of $f(b_1, b_2, \ldots, b_n)$, $\alpha = f(b_i^1, b_i^2, \ldots, b_i^n), 1 \leq i \leq 2^{n-k}$, is computed and for each $i$ it is checked whether 0 appears in the register containing the value $\alpha$.

If $2^r$ processors are at our disposal, we may speed up the procedure by dividing the evaluation of $f$ into $2^r$ parallel tasks, as the computation of $f$ on a single block obviously does not depend on the values of $f$ on other blocks. Then the necessary time of the whole computing process would take about $t_s/2^{k+r}$.

If for some reason the security of the computation is needed, as in cryptography, we can permute the bits of $b_i$, or the blocks $b_j^i$ of the division of $b_i$. More precisely, the columns of the matrix $M = [b_1, b_2, \ldots, b_n]$ are permuted (cf. Theorem 1.11), and then we apply $f$ on the rows (blocks) of the matrix obtained in that manner.

Here is a real situation where the algorithm can be applied. Suppose that we have two Boolean expressions $F(x_1, x_2, \ldots, x_n)$ and $G(x_1, x_2, \ldots, x_n)$ that should define the same Boolean function $f$. Of course, we would like to know if $F$ and $G$ really define the same function, as $f$ is intended to be firmware implemented on a certain chip. This problem may arise when $G$ is obtained from $F$ by some optimization process, so that $F$ describes the intended function $f$ of the chip, while

$G$ is the actual implementation of $f$ on the chip. In order to show that the chip will work properly, it is necessary to show that $F$ and $G$ represent the same function $f$. There are algorithms that are used for this purpose, as the Quine's BDP (binary decision procedure). In our approach we would take $H = \overline{F \triangle G}$, where $\triangle$ stands for the Boolean operation XOR ($F \triangle G = \bar{F}G + f\bar{G}$). Then, check by the proposed procedure if $H$ is identically equal to 1. If $n \leq 30$, this can be done on a computer with one 32-bit processor in real time. But if parallel computers with processor fields, or if specialized chip ("logical chip") with registers of appropriate size, say with $2^{24}$ bits, is designed, then the number of arguments of $f$ can be raised up to 50.

## 3. Remark

In the search for zero in the sequence of binary values of $F^{\Omega_n}(b_1, b_2, \dots, b_n)$, the reading of bits of a $k$-bit register for large $k$ might be time consuming. In fact, this problem is equally hard to the computing problem of evaluating $L\mathbf{x}$, where $L$ is an operator defined on the Boolean algebra $\mathbf{B} = \mathbf{2}^k$, $\mathbf{x} \in B$, which satisfies the following two axioms: $x < 1 \Rightarrow Lx = 0$, $L1 = 1$. If the dual operator $M$ is defined by $Mx = \bar{L}\bar{x}$, then $L$ and $M$ behave as modal S5 necessity and possibility operators, respectively. In fact, finite structures $(\mathbf{B}, M, L)$ are characteristic algebras for S5 modal logic, i.e. modal propositional sentences true on such structures are exactly the theorems of S5 logic. There is a simple implementation of the operator $M$ by a switching circuit by rows of AND gates described by the following recurrent formulas for $\mathbf{x} \in 2^k$, $k = 2^m$, $\mathbf{x} = (x_1, x_2, \dots, x_n)$:

$$
\begin{aligned}
&x_1^1 = x_1 x_2,\ \ x_2^1 = x_3 x_4, \dots \qquad \dots \qquad \dots,\ \ x_{2^{m-1}}^1 = x_{2^m - 1} x_{2^m} \\
&x_1^2 = x_1^1 x_2^1,\ \ x_2^2 = x_3^1 x_4^1, \dots,\ \ x_{2^{m-2}}^2 = x_{2^{m-1}-1}^1 x_{2^{m-1}}^1 \\
&\vdots \\
&x_1^m = x_1^{m-1} x_2^{m-1}.
\end{aligned}
$$

Then $M\mathbf{x} = x_1^m$. Observe that the depth of this circuit is $m = \log_2 k$. In a similar way, a Boolean circuit for $L\mathbf{x}$ can be constructed, but with OR gates. Therefore, the "logic processor" mentioned above should contain an extra bit $e$ for each register. The bit $e$ would carry out the status $M\mathbf{x}$ of the register containing the value $\mathbf{x}$. An extra property of such a device would be the capability of computing values of modal S5 formulas.

## References


[1] P. M. Cohen, *Universal Algebra*, D. Reidel, Dordrecht, 1981.

[2] G.E. Hughes, M.J. Cresswell, *An Introduction to Modal Logic*, Methuen & Co, London, 1968.

[3] E. Mendelson, *Boolean Algebra and Switching Circuits*, Schaums's Outline Series, McGraw Hil, New York, 1970.

[4] R. Sikorski, *Boolean Algebras*, Springer-Verlag, Berlin, 1969.

[5] R. Zacks, *From Chips to Systems*, Sybex, Berkely, 1981.


Matematički fakultet, Studentski trg 16, 11001 Beograd, p.p. 550, Yugoslavia

(Received 01 04 1998; Revised 20 11 1998)