Time Lower Bounds for CREW-PRAM Computation of Monotone Functions

Gianfranco Bilardi*
Abha Moitra**

TR 89-1012
May 1989

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*This research is supported in part by NSF grant MIP-02256 and by the Joint Services Electronics Program under contract F49620-87-C0044.

**The research was done at Cornell University and supported by NSF grant DCR-86-02307.

Time Lower Bounds For CREW-PRAM Computation Of Monotone Functions

Gianfranco Bilardi[1]
Department of Computer Science
Cornell University
Ithaca, New York 14853

Abha Moitra[2]
General Electric Research Laboratory
Schenectady, New York 12301

Abstract: It is shown that the time to compute a monotone boolean function depending upon n variables on a CREW-PRAM satisfies the lower bound $T = \Omega(\log l + (\log n)/l)$, where l is the size of the largest prime implicant. It is also shown that the bound is existentially tight by constructing a family of monotone functions that can be computed in $T = O(\log l + (\log n)/l)$, even by an EREW-PRAM. The same results hold if l is replaced by L, the size of the largest prime clause.

An intermediate result of independent interest is that $S(n,l)$, the size of the largest minimal vertex cover minimized over all (reduced) hypergraphs of n vertices and maximum hyperedge size l, satisfies the bounds $\Omega(n^{1/l}) \leq S(n,l) \leq O(ln^{1/l})$.

[1]Supported in part by the National Science Foundation under grant MIP-86-02256 and by the Joint Services Electronics Program under contract F49620-87-C0044.

[2]Work done while at Cornell University was supported in part by the National Science Foundation under grant DCR-86-02307

1 Introduction

Parallel Random Access Machines (PRAMs) play a central role in the study of parallel computation and in the development of parallel algorithms. A PRAM is essentially a set of processors connected to a large shared memory. To make the PRAM a precise model of computation, a number of aspects need to be specified, including conventions for simultaneous memory access, the instruction set of individual processors, whether the control is based on a single or a multiple instruction stream, the mechanism by which processors are activated, and so on. Several variants with respect to these aspects have been considered in the literature [FW 78, LPV 81, SV 81, Gol 82, BH 85, Sni 85] and various relations among these variants have been investigated [Vis 83, FRW 84, FGRW 85, Sni 85, CDR 86]. PRAM models do not capture all the aspects of computation determining algorithm performance on technologically feasible multi-processors [Sny86]. However, PRAMs can be simulated by more realistic models with a loss in performance upper bounded by a logarithmic factor, probabilistically [Upf 84, KU 86, Ran 87], and by the square of a logarithmic factor, deterministically [UW 84, AHMP 87, HB88].

The interest in designing optimal PRAM algorithms motivates the development of lower bound techniques for the PRAM model. In this work we concentrate on the computation of Boolean functions of n Boolean variables on PRAMs without simultaneous writes. Lower bounds are derived for the concurrent-read exclusive-write (CREW) variant, while upper bounds are established for the exclusive-read exclusive-write (EREW) variant of the PRAM [Sni 85]. Therefore, all bounds are valid for both models.

Our starting point is a result of [CDR 86] (see also [PY 87]) stating the following lower bound for the computation of a Boolean function $f(\mathbf{x})$ of the n-tuple of Boolean variables $\mathbf{x}$. If $f(\mathbf{y}) \neq f(\mathbf{x})$ for $\sigma(\mathbf{x})$ values of $\mathbf{y}$ that differ from $\mathbf{x}$ in exactly one variable, than the time to compute f on a CREW-PRAM is $\Omega(\log\sigma)$ where $\sigma=\max\sigma(\mathbf{x})$ is called the *critical complexity* of f. The proof of this result is subtle, but the application to specific functions is usually straightforward. For example, an $\Omega(\log n)$ lower bound is easily derived for the AND, OR, and EXCLUSIVE OR of n variables [CDR 86]. However, proving a lower bound for the quantity σ that applies to all functions in a given class is not necessarily easy.

An interesting application of the lower bound of [CRD 86] was made in [Sim 83] where it was shown that if f is a non-degenerate function of n variables, then its critical complexity is $\sigma = \Omega(\log n)$, and hence its CREW-PRAM computation time is $T = \Omega(\log\log n)$. Examples of non-degenerate functions computable by a CREW-PRAM in $T = O(\log\log n)$ time were also provided, showing that the lower bound is existentially tight.

In this paper we consider non-degenerate *monotone* functions. Since there are such functions that are computable in $O(\log\log n)$ time [Weg 85], lower bounds stronger than the general $\Omega(\log\log n)$ one can be obtained only under further assumptions. For example [Tur 84] and [Weg 85] consider the subset of monotone functions that can be interpreted as properties of monotone graphs. Here, we consider arbitrary monotone functions, but we assume that either the length l of the *largest prime implicant*, or the length L of the *largest prime clause* are known. Besides being natural parameters, l and L also determine the critical complexity of a monotone function as $\sigma = \max(l,L)$ [Weg 85]. Thus, our task is essentially that of lower bounding L given l, or vice versa. For concreteness, in the remainder of this paper we state all results in terms of l. By duality considerations on Boolean functions, it is not difficult to see that all results hold if l is replaced by L.

In Section 2, we reformulate the result of [CDR 86] for monotone functions, exploiting a natural correspondence between monotone functions and hypergraphs. We establish that the CREW-PRAM time to compute a monotone function f is $\Omega(\max(\log l, \log S(n,l)))$, where l is the size of the largest prime implicant for f, and $S(n,l)$ is defined as the minimum size of the *largest minimal vertex cover* in a hypergraph with n vertices and maximum hyperedge size equal to l.

The quantity $S(n,l)$ is investigated first in Section 3 for the more familiar setting of graphs ($l = 2$). It is shown that $S(n, 2) = \lfloor\sqrt{n}\rfloor + \lfloor n/\sqrt{n}\rfloor - 2 - \delta_n$, where δ_n is either 0 or 1 (depending on n). As a consequence of this result and that of Section 2, we have that if a monotone function f has a disjunctive normal form whose terms are either single terms or product of two terms, then the reduced conjunctive normal form for f has a clause with $\Omega(\sqrt{n})$ terms. The general, and more difficult case of hypergraphs is considered in Section 4 where it is shown that $\Omega(n^{1/l}) \le S(n,l) \le O(ln^{1/l})$. Although the exact determination of $S(n,l)$ remains an open problem of independent interest, our analysis is sufficient to obtain (existentially) tight bounds on the computation time.

The main result of the paper is given in Section 5: the CREW-PRAM time to compute a monotone function of n variables and with size of the largest prime implicant equal to l satisfies the bound $T = \Omega(\log l + (\log n)/l)$. Since $\log l + (\log n)/l = \Omega(\log\log n)$, we have that $T = \Omega(\log\log n)$ for any monotone function, which is of course also a corollary of the bound of [Sim 83] cited above. However if, say, $l = O(1)$, then one obtains $T = \Omega(\log n)$. It is also shown that the derived lower bound is the best that can be formulated in terms of n and l, by constructing (for each n and l) a monotone function computable in $O(\log l + (\log n)/l)$ time by an EREW-PRAM.

2 Cook, Dwork, and Reischuk's Lower Bound Reformulated for Monotone Functions

Let $y = f(x_1, x_2, \ldots, x_n)$ be a binary function of n binary variables. We shall also write $y = f(\mathbf{x})$, where $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and let $C_i\mathbf{x}$ denote the binary vector that differs from $\mathbf{x}$ exactly in the ith component. The following quantities play an important role:

$$\sigma(\mathbf{x}) = |\{i : f(C_i\mathbf{x}) \neq f(\mathbf{x})\}|,$$

$$\sigma_{01} = \max\{\sigma(\mathbf{x}) : f(\mathbf{x}) = 0\},$$

$$\sigma_{10} = \max\{\sigma(\mathbf{x}) : f(\mathbf{x}) = 1\},$$

$$\sigma = \max\{\sigma_{01}, \sigma_{10}\}.$$

The quantity σ is called the critical complexity of f. With this notation we have:

Theorem 1 [CDR 86]. The computation time of a binary function f of n binary variables on a CREW-PRAM satisfies the lower bound $T = \Omega(\log \sigma)$.

Below we investigate the properties of the critical complexity of monotone functions. Without loss of generality, we restrict our attention to positive functions. A function f is *positive* if, for any $\mathbf{x}$ such that $x_i = 0$ and $f(\mathbf{x}) = 1$, $f(C_i\mathbf{x}) = 1$. Any positive function admits a unique reduced disjunctive normal form (RDNF) containing only positive literals, and with no term containing all the factors of another term. Throughout this paper, unless the contrary is explicitly stated, we shall assume that f is truly dependent upon all of its variables, that is, for every $i = 1,2,\ldots,n$, there is an $\mathbf{x}$ such that $f(\mathbf{x}) \neq f(C_i\mathbf{x})$.

More formally, if we let $V = \{x_1, x_2, \ldots, x_n\}$ be the set of variables of f and, for $e \subseteq V$, we denote by $t_e(\mathbf{x})$ the term $\prod_{x_i \in e} x_i$, then we can write

$$f(\mathbf{x}) = \sum_{e \in E} \prod_{x_j \in e} x_j = \sum_{e \in E} t_e(\mathbf{x}), \tag{1}$$

where E is a set of subsets of V none of which is a subset of another, and each x_i belongs to some $e \in E$.

With any positive function f one can associate in a natural way an hypergraph $H = (V, E)$ whose vertices are identified with the variables and whose hyperedges correspond to the terms of the RDNF. We shall say that H is *reduced* to indicate that each vertex belongs to at least one hyperedge and that no hyperedge is contained in another one. Note that f uniquely determines H and vice versa.

In the following theorems, we study the dependence of σ not only upon the number of variables n, but also upon the size of the largest prime implicant, (which is also the largest term in the RDNF for f), denoted by $l \triangleq \max\{\, |e| : e \in E\}$.

Theorem 2 [Weg 85]. For a positive function f, $\sigma_{10} = l$.

A dual result to that of Theorem 2 is that, for a positive function $f, \sigma_{01} = L$, the size of the longest prime clause of f. Below, we develop an alternative hypergraph-theoretic characterization of L, which will prove crucial in the derivation of our results.

A set $A \subseteq V$ of vertices is called a *vertex cover* of an hypergraph $H = (V, E)$ if every hyperedge $e \in E$ contains at least one vertex of A. A vertex cover is *minimal* if none of its proper subsets is a vertex cover.

Theorem 3. For a positive function f, σ_{01} equals the size s of a largest minimal vertex cover of the associated hypergraph H.

Proof. We show first that $\sigma_{01} \geq s$. Let A be a largest minimal vertex cover of H, and let $\mathbf{a}$

$\in \{0, 1\}^n$ be such that $a_i = 0$ if and only if $x_i \in A$. We have that $f(\mathbf{a}) = 0$, since in each term there is at least one zero factor. For the same reason, if $x_i \notin A$, then $f(C_i\mathbf{a}) = 0$. We also have that, for all $x_i \in A$, $f(C_i\mathbf{a}) = 1$. Indeed, since A is minimal, for every $x_i \in A$, $A - \{x_i\}$ is not a vertex cover, that is, there exists an $e \in E$ such that $e \cap A = \{x_i\}$. For such e, $t_e(C_i\mathbf{a}) = 1$. In conclusion, $f(C_i\mathbf{a}) = 1$ if and only if $x_i \in A$, so that $\sigma_{01} \geq \sigma(\mathbf{a}) = s$.

Now we show that $\sigma_{01} \leq s$. We write $\mathbf{a} \geq \mathbf{b}$ if $a_i \geq b_i$ for $i = 1, \ldots, n$. If $\mathbf{a} \geq \mathbf{b}$ and $f(\mathbf{a}) = f(\mathbf{b}) = 0$, then $\sigma(\mathbf{a}) \geq \sigma(\mathbf{b})$. In fact, $\mathbf{a} \geq \mathbf{b}$ implies $C_i\mathbf{a} \geq C_i\mathbf{b}$ and, due to the positivity of f, $f(C_i\mathbf{a}) \geq f(C_i\mathbf{b})$, so that $f(C_i\mathbf{a}) = 1$ at least for as many i's as $f(C_i\mathbf{b}) = 1$. We can then consider a maximal $\mathbf{a}$ such that $f(\mathbf{a}) = 0$, $\sigma(\mathbf{a}) = \sigma_{01}$, and $f(\mathbf{b}) = 1$ for all $\mathbf{b} > \mathbf{a}$. Since $C_i\mathbf{a} > \mathbf{a}$ if and only if $a_i = 0$, we have that $\sigma_{01} = \sigma(\mathbf{a}) = |A|$, where $A = \{x_i : a_i = 0\}$. We claim that A is a minimal vertex cover of H.

Since $f(\mathbf{a}) = 0$, $t_e(\mathbf{a}) = 0$ for all $e \in E$. Then, each e contains at least one variable that is set to zero in $\mathbf{a}$, which implies that A is a vertex cover. Since, for each $x_i \in A$, $f(C_i\mathbf{a}) = 1$, there must be at least one term $t_e(C_i\mathbf{a}) = 1$. All the variables in e must be set to one, implying that none of them belongs to A, except for x_i. Thus, $A - \{x_i\}$ is not a vertex cover and hence A is minimal. In conclusion, $\sigma_{01} = \sigma(\mathbf{a}) = |A| \leq s$, where the last inequality follows from the assumption that the size of a largest minimal vertex cover of H is s. □

We find it convenient to introduce the class $\mathrm{H}_{n,l}$ of the reduced hypergraphs with n vertices and maximum hyperedge size l, where $n \geq 1$ and $1 \leq l \leq n$, and to define the function

$$S(n,l) = \min \{\text{size of largest minimal vertex cover of } H : H \in \mathrm{H}_{n,l}\}. \tag{2}$$

The following result is a simple consequence of Theorems 1, 2, and 3.

Corollary 1. Let f be a positive function of n variables, and let l be the size of the largest prime implicant of f. Then, the time T to compute f on a CREW-PRAM satisfies the lower bound $T = \Omega(\max(\log l, \log S(n,l)))$.

We develop lower bounds on the function $S(n,l)$ in the next two sections.

3 Largest Minimal Vertex Cover in Graphs

We begin by investigating the size of largest minimal vertex covers for $\mathrm{H}_{n,2}$. We shall derive an almost exact expression for $S(n,2)$ by giving matching lower and upper bounds. Treating this special case will help in developing some intuition about the general case.

Theorem 4. Let $G = (V, E)$ be a graph with $n = |V|$ non isolated vertices, and let $A \subseteq V$ be a largest minimal vertex cover of G. Then $s = |A| \geq \lceil 2\sqrt{n+2} - 3 \rceil$.

Proof. For $x \in A$, let $N(x) = \{y : (x, y) \in E, y \in V - A\}$. Clearly $B = A - \{x\} \cup N(x)$ is a vertex cover of G and hence a suitable subset $A' \subseteq B$ will be a minimal vertex cover of G. Since it must be that $N(x) \subseteq A'$ to cover the edges (x, y) with $y \in N(x)$, we have that $B - A' \subseteq A$. By premise $|A'| \leq |A|$, that is

$$|A'| = |A| + |N(x)| - 1 - |B - A'| \leq |A|,$$

implying $|B - A'| + 1 \geq |N(x)|$. Defining $C(x) = (B - A') \cup \{x\}$, we have $|C(x)| \geq |N(x)|$.

We now choose $x \in A$ so as to maximize $k = |N(x)|$. Since each vertex in $V - (A \cup N(x))$ is connected to at least one vertex in $A - C(x)$, the average of $|N(z)|$ for $z \in A - C(x)$ is at least $\dfrac{|V - (A \cup N(x))|}{|A - C(x)|} \geq \dfrac{(n-s-k)}{(s-k)}$, where we have used the fact that $|C(x)| \geq |N(x)| = k$. Since $|N(z)| \leq |N(x)| = k$ for all $z \in A$, we have

$$k \geq \frac{(n-s-k)}{(s-k)}.$$

Solving for s we obtain

$$s \geq \frac{k^2-k+n}{k+1}.$$

Choosing $k=(\sqrt{n+2}-1)$ so as to minimize the right hand side of the last inequality and considering that s is an integer, we obtain $s \geq \lceil 2\sqrt{n+2}-3 \rceil$. □

Corollary 2. Let $H=(V, E) \in \mathrm{H}_{n,2}$ and let $A \subseteq V$ be a largest minimal vertex cover of H. Then, $s = |A| \geq \lceil 2\sqrt{n+2}-3 \rceil$.

Proof. Let V_1 be the set of vertices x such that $\{x\} \in E$, and let $E_1 = \{\{x\} : x \in V_1\}$. Since H is reduced, no edge in $E-E_1$ contains vertices from V_1. Therefore, we have that $A = V_1 \cup A'$, where A' is a largest minimal vertex cover of $(V-V_1,\ E-E_1)$. In conclusion, $|A| = |V_1| + |A'| \geq |V_1| + \lceil 2\sqrt{n-|V_1|+2}-3 \rceil \geq \lceil 2\sqrt{n+2}-3 \rceil$. □

For each $n \geq 2$, we define a graph $G_n=(V, E)$ of n vertices whose largest minimal vertex cover has a size almost matching the lower bound of Theorem 4. Informally, G_n is formed by $m=\lfloor \sqrt{n} \rfloor$ star-shaped graphs of approximately equal size with the centers of the stars connected as a clique. Formally, we let $p=\lceil (n-m)/m \rceil$, $V=\{1,2,\ldots,n\}$, $V_0=\{1,2,\ldots,m\}$, (the centers of the star); $V_i=\{m+(i-1)p+j : j=1,2,\ldots,p\}$, for $1 \leq i \leq m-1$, $V_m=\{m+(m-1)p+1,\ldots,n\}$, (for $i \geq 1$, V_i is the set of spokes of the i-th star); $E_0=\{(i,j) : 1 \leq i < j \leq m\}$, (the clique edges); $E_i=\{(i,j) : j \in V_i\}$, (the edges of the i-th star), for $i=1,\ldots,m$; and $E=E_0 \cup E_1 \cup \cdots \cup E_m$. Next, we compute the size of the largest minimal vertex cover of G_n.

Theorem 5. The size of a largest minimal vertex cover of G_n is $\lceil n/\lfloor \sqrt{n} \rfloor \rceil + \lfloor \sqrt{n} \rfloor - 2$.

Proof. Let A be a minimal vertex cover of G_n. Then, $|V_0 - A| \leq 1$. In fact, if two vertices of V_0 were not in A, the edge joining them would not be covered. Let us consider two cases.

(1) $|V_0 - A| = 0$. Clearly $V_0 \subseteq A$. We claim that $A = V_0$. Indeed, every edge of E has at least one vertex in V_0, so that if $|A - V_0| > 0$, then A is not minimal, against the hypothesis.

(2) $|V_0 - A| = 1$. Let $V_0 - A = \{i\}$. Then, $V_i \subseteq A$, or otherwise some edge in E_i would be uncovered. Since every edge of E has at least one vertex in $V_0 - \{i\} \cup V_i$ and A is minimal, it must be $A = V_0 - \{i\} \cup V_i$.

From the above analysis, it follows that the minimal vertex covers of G_n are V_0 and $V_0 - \{i\} \cup V_i$ for $i = 1,2,...,m$. A largest minimal vertex cover is then $V_0 - \{1\} \cup V_1$ whose size is $m - 1 + p = \lfloor \sqrt{n} \rfloor + \lceil n / \lfloor \sqrt{n} \rfloor \rceil - 2$. □

From Theorem 5 and Corollary 2, with some manipulations, we obtain the tight bound $S(n, 2) = \lfloor \sqrt{n} \rfloor + \lceil n / \lfloor \sqrt{n} \rfloor \rceil - 2 - \delta_n$, where $0 \leq \delta_n \leq 1$.

4 Largest Minimal Vertex Cover in Hypergraphs

We shall derive lower and upper bounds for $S(n,l)$. These bounds will be non-trivial roughly for $l < \log n$. The range $l \geq \log n$ is less interesting for our purposes since, as we shall see, in this range the bound on computation time of Corollary 1 is determined by the term $\log l$. The upper bound on $S(n,l)$ gives an indication on the quality of the lower bound. Moreover, the hypergraphs on which the upper bound is based will be crucial in the definition of boolean functions whose computation time matches the lower bound to be derived in Section 5.

4.1 Lower Bound

The main result of this section is a lower bound to the size of the largest minimal vertex cover of a hypergraph H in terms of the number of vertices that belong to hyperedges of a given size. To formulate the result precisely we introduce, for $i = 1,2,...,l$, the set $E_i \triangleq \{e \in E : |e| = i\}$ of the hyperedges of size i, and the set $V_i \triangleq \{x \in V : \exists\, e \in E_i\, (x \in e)\}$ of the vertices that belong to hyperedges of size i. We observe that $E_1, E_2,...,E_l$ partition E and that $V_1 \cup V_2 \cdots \cup V_l = V$, assuming no isolated vertices. We have:

Theorem 6. Let $H = (V,E) \in \mathrm{H}_{n,l}$. Let $E_1,...,E_l$ and $V_1,...,V_l$ defined as above, and let $n_i \triangleq |V_i|$. If $A \subseteq V$ is a largest minimal vertex cover of H, then, for each $i = 1,2,...,l$, $s \triangleq |A| \geq c\ n_i^{1/i}$, where $c = \ln(3/2)$.

In the proof of Theorem 6 we shall use two lemmas, which contain some of the key ideas. Given a hypergraph $H = (V,E)$, a vertex cover $A \subseteq V$ of H, and a set $X \subseteq A$, let $H(X) = (V(X), E(X))$ be the hypergraph formed by taking all the hyperedges of H that are not covered by $A - X$, and by eliminating from them any vertex of X. Formally, $E(X) \triangleq \{e - A : e \in E,\ e \cap A \subseteq X\}$ and $V(X) \triangleq \{x : \exists e \in E(X)\, (x \in e)\}$. When A is a largest minimal vertex cover of H, hypergraph $H(X)$ has the following property.

Lemma 1. Let $H = (V,E)$ be a hypergraph and $A \subseteq V$ a largest minimal vertex cover of H. If no hyperedge $e \in E$ is completely contained in $X \subseteq A$, then the size of any minimal vertex cover B of $H(X)$ satisfies the bound $|B| \leq |A|$.

Proof. All hyperedges in E have some vertex in A. Moreover, since no hyperedge is completely included in X, if $e \in E$ has no vertex in $A - X$, then $e - X$, and hence e, has some vertex in

B. Therefore, $(A-X) \cup B$ is a vertex cover of H and a suitable $A' \subseteq (A-X) \cup B$ is a minimal vertex cover of H. Since B is minimal in $H(X)$ and no hyperedge covered by B is covered by $A-X$, we have that $B \subseteq A'$. In conclusion, $|B| \leq |A'| \leq |A|$, where the last inequality follows from A being a largest minimal vertex cover of H. □

Observing that if $H \in \mathrm{H}_{n,l}$ then $H(X) \in \mathrm{H}_{n,q}$, with $q < l$, we can see how Lemma 1 can be used in an inductive proof of Theorem 6, as long as X can be chosen so that $H(X)$ contains a sufficient number of vertices in hyperedges of a given size. For this purpose, we shall need the following classification of hyperedges and vertices of H.

For $j = 1,2,\ldots,i-1$, we let $E_{ij} \triangleq \{e : e \in E_i, \ |e \cap A| = j\}$, $V_{ij} \triangleq \{x : x \in V-A, \ \exists e \cup E_{ij} \ (x \in e)\}$, and $n_{ij} \triangleq |V_{ij}|$. Clearly, $V_{i1} \cup V_{i2} \cup \cdots \cup V_{i,i-1} = V_i - A$, so that $n_{i1} + n_{i2} + \cdots + n_{i,i-1} \geq n_i - s$.

Next, we let $n_{ij}(X) \triangleq |V_{ij}(X)|$, where $V_{ij}(X) \triangleq \{x : x \in V-A, \ \exists e \in E_{ij}(e \cap A = X, \ x \in e)\}$. (Since $e \in E_{ij}$ implies $|e \cap A| = j$, the set $V_{ij}(X)$ is empty whenever $|X| \neq j$.) We observe that $V_{ij}(X)$ is a subset of the vertices of $H(X)$ that belong to hyperedges in $E(X)$ of size $i-j$. This fact motivates the interest in the following lower bound for $n_{ij}(X)$ in terms of n_{ij}.

Lemma 2. Let $H = (V,E) \in \mathrm{H}_{n,l}$ and let $A \subseteq V$ be a largest minimal vertex cover of H of size $s \triangleq |A|$. Then, for each $i = 1,2,\ldots,l$ and $j = 1,2,\ldots,i-1$, there exists an $X \subseteq A$, with $|X| = j$, such that $n_{ij}(X) \geq n_{ij} / \binom{s}{j}$. Moreover, no hyperedge $e \in E$ is completely contained in X.

Proof. If $n_{ij} = 0$, then let X be the empty set. Otherwise, let X be chosen so as to maximize $n_{ij}(Y)$, among the sets $Y \subseteq A$ of size $|Y| = j$. We first establish that

$$\sum_{Y \subseteq A, \ |Y| = j} n_{ij}(Y) \geq n_{ij} \ .$$

Indeed, each $x \in V_{ij}$ belongs to some $V_{ij}(Y)$ because, by definition of V_{ij}, $x \in e$ for some e with $|e| = i$ and $|e \cap A| = j$. Then, $x \in V_{ij}(e \cap A)$. Since there are $\begin{bmatrix} s \\ j \end{bmatrix}$ terms in the above summation, the average of $n_{ij}(Y)$ is at least $n_{ij} / \begin{bmatrix} s \\ j \end{bmatrix}$. Clearly, $n_{ij}(X)$ is no smaller than this average.

It remains to show that for no $e \in E$ we have $e \subseteq X$. This is obvious if $n_{ij} = 0$. If $n_{ij} > 0$, then $n_{ij}(X) \geq 1$, since $n_{ij}(X) \geq n_{ij} / \begin{bmatrix} s \\ j \end{bmatrix}$, and $n_{ij}(X)$ is an integer. By definition of $V_{ij}(X)$, there is an $e' \in E$ such that $e' \cap A = X$, hence $X \subseteq e'$. Thus, $e \subseteq X$ would imply $e \subseteq e'$, a contradiction since H is reduced. $\square$

Proof of Theorem 6. We proceed by induction on l. The base case, $l = 1$, is obvious since the only vertex cover of a hypergraph $H \in \mathrm{H}_{n,1}$ is the set of all vertices V, so that $s = n = n_1 > cn_1$.

We now assume the theorem true for $\mathrm{H}_{n,1}, \ldots, \mathrm{H}_{n,l-1}$ and prove it for $\mathrm{H}_{n,l}$. We let $H = (V,E) \in \mathrm{H}_{n,l}$ and let A be a largest minimal vertex cover of size s. For each $i = 1,2,\ldots,l$ and $j = 1,2,\ldots,i-1$, we let $X \subseteq A$ to be chosen as in Lemma 2 so that $n_{ij}(X) \geq n_{ij} / \begin{bmatrix} s \\ j \end{bmatrix}$. As already observed, $H(X) \in \mathrm{H}_{n,q}$, with $q < l$, and contains at least $n_{ij}(X)$ vertices that belong to hyperedges of size $i-j$. Thus, by the inductive hypothesis, the largest minimal vertex cover B of $H(X)$ has size $|B| \geq c\,(n_{ij}(X))^{1/(i-j)} \geq c\,(n_{ij} / \begin{bmatrix} s \\ j \end{bmatrix})^{1/(i-j)}$. Moreover, from Lemma 1, $s = |A| \geq |B|$. Summarizing, we have that, for $i = 1,2,\ldots,l$ and $j = 1,2,\ldots,i-1$, $s \geq c\,(n_{ij} / \begin{bmatrix} s \\ j \end{bmatrix})^{1/(i-j)}$. Straightforward manipulations and use of the inequality $\begin{bmatrix} s \\ j \end{bmatrix} \leq s^j / j!$ yield

$(s/c)^i\,(c^j/j!) \geq n_{ij}$. Summing over $j=1,2,\ldots,i-1$, using the inequality $\sum_{j=1}^{i-1}(c^j/j!) < \sum_{j=1}^{\infty}(c^j/j!) = e^c-1$, and recalling that $n_{i1}+\ldots+n_{i,i-1} \geq n_i - s$ we obtain $(s/c)^i\,(e^c-1) \geq n_i - s$. Since $c = \ln(3/2)$, $e^c - 1 = 1/2$, and the latter bound can be rewritten as $(s/c)^i \geq 2(n_i - s)$. Since we can assume $s \leq n_i/2$, (otherwise $s > n_i/2 > cn_i^{1/i}$ and the theorem is true anyway), we have that $2(n_i - s) \geq n_i$. Thus $(s/c)^i \geq n_i$, from which the thesis follows immediately. □

As a consequence of Theorem 6 we have the following lower bound on $S(n,l)$.

Theorem 7. Let $c = \ln(3/2)$. Then, $S(n,l) \geq (c/2)n^{1/l}$.

Proof. If $l \geq \log n/\log(2/c)$, then $(c/2)n^{1/l} \leq 1$, and the theorem holds trivially. It remains to consider the case when $l < \log n/\log(2/c)$. From the definition of $S(n,l)$ (Section 2) and Theorem 6, it follows that $S(n,l) \geq \max\{cn_i^{1/i} : i = 1,2,\ldots,l\}$, where $n_i = |V_i|$. As already observed, $V_1 \cup V_2 \cup \cdots \cup V_l = V$, hence $n_1 + n_2 + \cdots + n_l \geq n$. Then, if we let

$$z \triangleq \min_{n_1+\ldots+n_l \geq n} \max\{n_i^{1/i} : i = 1,2,\ldots,l\} \ ,$$

where the n_i's are allowed to range over the reals, we clearly have $S(n,l) \geq cz$. It is easy to show that z is obtained when $n_i = n_l^{i/l}$, and $\sum_{i=1}^{l} n_i = n$. In this case, $n_i = z^i$, and $z + z^2 + \ldots + z^l = n$. It is not possible that $z \leq 2$, as it would imply $2 + 2^2 + \ldots 2^l \geq n$, which is impossible for $l < \log n/log(2/c)$. For $z > 2$, we have $2z^l > z + z^2 + \ldots + z^l = n$, hence $z > n^{1/l}/2^{1/l} \geq n^{1/l}/2$. In conclusion, $S(n,l) \geq cz > (c/2)n^{1/l}$. □

4.2 Upper Bound

For each n, l and m, with $l-1 \le m < n$, we define a hypergraph $H_{n,l,m} \in \mathrm{H}_{n,l}$ for which we can evaluate the size of a largest minimal vertex cover. A suitable choice of m as a function of n and l will yield an upper bound on $S(n,l)$. The constructions of this section generalize those of Section 3.

We begin by introducing the *(p,l)–star*, a hypergraph with $p+l-1$ vertices and p hyperedges of size l. A subset of $l-1$ distinguished vertices, called the center of the star, is part of every hyperedge. Each of the remaining p vertices, called the spokes, is part of exactly one hyperedge.

Informally, $H_{n,l,m}$ can be described as follows. There is a kernel of m vertices where each subset of l vertices forms a hyperedge. Each of the $k = \begin{pmatrix} m \\ l-1 \end{pmatrix}$ subsets of vertices of the kernel of size $l-1$ is the center of either a (p,l)-star or a $(p-1,l)$-star whose spokes are vertices not in the kernel, where $p = \lceil (n-m)/k \rceil$. Different stars have no common spokes.

Formally, we let $V_0 = \{1,2,\ldots,m\}$, (the vertices in the kernel), and $E_0 = \{e : e \subseteq V_0,\ |e| = l\}$, (the hyperedges within the kernel). Letting $h \triangleq n-m-k(p-1)$ (so that $0 < h \le k$ and $hp + (k-h)(p-1) = n-m$), the set of the spokes of the i-th star is defined as $V_i = \{m + (i-1)p + j : j = 1,2,\ldots,p\}$ for $1 \le i \le h$ and $V_i = \{m + hp + (i-h-1)(p-1) + j : j = 1,2,\ldots p-1\}$ for $h+1 \le i \le k$. It is convenient to introduce the notation r_i for the i-th smallest positive integer with exactly $l-1$ bits equal to 1 in its binary representation, and to denote by $r_i(j)$ the coefficient of 2^j in the binary expansion of integer r_i. Then, for $1 \le i \le k$, we define $C(r_i) = \{j+1 : 0 \le j < m,\ \ r_i(j) = 1\}$, (the center of the i-th star), and $E_i = \{C(r_i) \cup \{x\} : x \in V_i\}$, (the hyperedges of the i-th star). Finally, we let $H_{n,m,l} = (V,E)$, where $V \triangleq V_0 \cup V_1 \cup \cdots \cup V_k = \{1,2,\ldots,n\}$ and $E \triangleq E_0 \cup E_1 \cup \cdots \cup E_k$.

Theorem 8. If A is a largest minimal vertex cover of $H_{n,l,m}$, then $|A| = m-(l-1) + \lceil (n-m) / \binom{m}{l-1} \rceil$.

Proof. Let A be a minimal vertex cover of $H_{n,l,m}$. Then, $|V_0 - A| < l$. In fact, if l vertices of V_0 were not in A, the hyperedge joining them would not be covered. Let us consider two cases.

(1) $|V_0 - A| < l-1$. First, we claim that $A \subseteq V_0$. Indeed, every hyperedge of E has at least $l-1$ vertices in V_0 and hence at least one in $A \cap V_0$. Thus, $A \cap V_0$ is a vertex cover and if A were different from $A \cap V_0$, it would not be minimal. Next, we claim that $|A| = m-(l-1) + 1$. Indeed, it is easily seen that every subset of V_0 of size larger than $m-(l-1)$ is a vertex cover. Therefore, any subset of V_0 of size larger than $m-(l-1)+1$ properly contains a vertex cover and is not minimal.

(2) $|V_0 - A| = l-1$. For some i, with $1 \le i \le \binom{m}{l-1}$, we have $V_0 - A = C(r_i)$. We claim that $A = (V_0 - C(r_i)) \cup V_i$. In fact, all the hyperedges have at least one vertex in $V_0 - C(r_i)$, except for those in E_i. To cover the latter, each vertex of V_i must be in A. In conclusion, the size of A will be $|A| = m-(l-1) + |V_i| \le m-(l-1) + \lceil (n-m) / \binom{m}{l-1} \rceil$, where equality holds for $1 \le i \le h$, with $h = n-m-(p-1)\binom{m}{l-1}$, as in the definition of $H_{n,l,m}$.

The thesis is finally obtained by comparing the size of the minimal vertex covers in cases (1) and (2) and observing that, since $n > m$, $\lceil (n-m) / \binom{m}{l-1} \rceil \ge 1$. □

Choosing $m = l-2 + l\lceil n^{1/l} \rceil$ and applying Theorem 8 yields (after several manipulations) $|A| < l\lceil n^{1/l}\rceil + n^{1/l} \leq (l+1)\lceil n^{1/l}\rceil$. From this and Theorem 7 we conclude:

Theorem 9. $\Omega(n^{1/l}) \leq S(n,l) \leq O(ln^{1/l})$.

Although a more accurate analysis of $S(n,l)$ would be of independent interest, the bounds of Theorem 9 are sufficient for our purposes. In fact, they imply that $\max(\log l, \log S(n,l)) = \Theta(\log l + (\log n)/l)$, and hence provide an asymptotically tight bound on the quantity that lower bounds computation time according to Corollary 1.

5 Time Lower Bounds

We shall combine the results of Sections 2 and 4 to obtain a lower bound on the computation time of a positive function on a CREW-PRAM. The lower bound will be expressed in terms of n, the number of variables, and l, the size of the largest prime implicant of the function. As a simple corollary, we shall also obtain a lower bound formulated solely in terms of n, which is of the same order as that of [Sim 83]. We shall also show that the lower bounds obtained are existentially tight, in the sense that, for each n and l, there exists a positive function whose computation time is within a constant factor of the lower bound.

Theorem 10. Let f be a positive function of n variables, and let l be the size of the largest prime implicant of f. Then, the time to compute f on a CREW-PRAM satisfies the lower bound $T = \Omega(\max(\log l, (\log n)/l))$.

Proof. This follows from Corollary 1 and the lower bound for $S(n,l)$ of Theorem 7. □

Theorem 11. The time to compute a positive function of n variables on a CREW-PRAM satisfies the lower bound $T = \Omega(\log \log n)$.

Proof. From Theorem 10, if $l > \log n / \log\log n$, then $T = \Omega(\log l) = \Omega(\log\log n)$ and, if $l \leq \log n / \log\log n$, then $T = \Omega((\log n)/l) = \Omega(\log\log n)$. □

Using the correspondence between positive functions and reduced hypergraphs introduced in Section 2, we define the function $f_{n,l,m}$ associated with the hypergraph $H_{n,l,m}$ constructed in Section 4.2. (Throughout the remainder of this section, we shall use the notation and the terminology related to $H_{n,l,m}$ introduced in Section 4.2.) It is convenient to express $f_{n,l,m}$ as the sum of $k+1 = \left\lceil \begin{matrix} m \\ l-1 \end{matrix} \right\rceil + 1$ functions, one corresponding to the hyperedges in the kernel, and the others corresponding to the hyperedges of the k stars:

$$f_{n,l,m}(\mathbf{x}) = \sum_{i=0}^{k} f_{n,l,m}^{(i)}(\mathbf{x}), \tag{6}$$

$$f_{n,l,m}^{(i)}(\mathbf{x}) = \sum_{e \in E_i} t_e(\mathbf{x}). \tag{7}$$

The following lemma states a useful property of $f_{n,l,m}$.

Lemma 3. Let $\mathbf{x} \in \{0, 1\}^n$, and let $q \triangleq |\{j \in V_0 : x_j = 1\}|$ be the number of variables associated with vertices of the kernel of $H_{n,l,m}$ that have value 1 in $\mathbf{x}$. If $q > l-1$, then $f_{n,l,m}(\mathbf{x}) = 1$. If $q < l-1$, then $f_{n,l,m}(\mathbf{x}) = 0$. If $q = l-1$, then $f_{n,l,m}(\mathbf{x}) = \sum_{j \in V_i} x_j$, where i is the unique integer such that $C(r_i) = \{j \in V_0 : x_j = 1\}$.

Proof. If $q > l-1$, there are at least l kernel variables equal to one. Since each l-tuple of kernel variables forms a term in $f_{n,l,m}^{(0)}$, we have that $f_{n,l,m}^{(0)}(\mathbf{x}) = 1$, hence that $f_{n,l,m}(\mathbf{x}) = 1$.

If $q < l-1$, we argue that all terms of $f_{n,m,l}$ are null and hence $f_{n,m,l}(\mathbf{x}) = 0$. Indeed, since each $e \in E$ contains at least $l-1$ kernel vertices, the corresponding term $t_e(\mathbf{x})$ contains at least $l-1$ kernel variables, at least one of which has value 0 in $\mathbf{x}$.

Finally, if $q = l-1$, let $C(r_i) = \{j \in V_0 : x_j = 1\}$. Clearly, all the terms containing a variable x_j with $j \in (V_0 - C(r_i))$ have value 0. The remaining terms are exactly those corresponding to hyperedges in E_i. Since the kernel variables in these terms have value one, each term simplifies to just the spoke variable. $\square$

The next result provides us with an expression for the computation of the index i of set $C(r_i)$ in the previous lemma.

Lemma 4. Let $r_i = 2^{j_\lambda} + 2^{j_{\lambda-1}} + ... + 2^{j_1}$ be the i-th smallest integer with exactly λ bits equal to one in its binary representation. Then,

$$i = \sum_{h=1}^{\lambda} \binom{j_h}{h} + 1. \tag{8}$$

Proof. We proceed by induction on λ. The base case, $\lambda = 1$, is easily verified. In general, for an integer $z < r_i$ with exactly λ bits equal to one, we can distinguish two cases.

(a) $z < 2^{j_\lambda}$. In this case, z has λ bits equal to one in any of the j_λ least significant positions. Thus, there are $\binom{j_\lambda}{\lambda}$ such z's.

(b) $2^{j_\lambda} < z < r_i$. Then, $z - 2^{j_\lambda}$ has $\lambda - 1$ bits equal to one, and is smaller than $r_i - 2^{j_\lambda} = 2^{j_{\lambda-1}} + ... + 2^{j_1}$. By the inductive assumption, there are $\sum_{h=1}^{\lambda-1} \binom{j_h}{h}$ such z's.

Combining the contributions of cases (a) and (b), we obtain the number of z's smaller than r_i. Adding 1 to account for r_i itself, we obtain (8). $\square$

We now outline an EREW-PRAM algorithm for the computation of $f_{n,l,m}$. We assume that, initially, the input variable x_i is stored in memory cell i, for $i = 1,2,...,n$. The output will be stored in cell 0.

First, $q \triangleq |\{j \in V_0 : x_j = 1\}|$ is computed. Recalling (from Section 4.2) that $V_0 = \{1,2,...,m\}$, q is the integer sum of $x_1, x_2, ..., x_m$, and can be obtained in $O(\log m)$ time by standard methods, using m processors.

Second, a distinguished processor, say P_1, examines q. Based on Lemma 3, if $q > l-1$ then P_1 writes 1 in memory cell 0, and if $q < l-1$ then P writes 0 in memory cell 0. In the remaining case, $q = l-1$, the computation proceeds as follows to determine $\sum_{j \in V_i} x_j$.

The quantities $(1!, 2!, ..., m!)$ are obtained as the prefixes of the sequence $(1, 2, ..., m)$ by standard parallel prefix techniques [LF 80, KRS 85], in $O(\log m)$ time, by processors $P_1, P_2, ..., P_m$. Similarly, the same processors compute the quantities $(y_1, y_2, ..., y_m)$ where y_j is defined as the integer sum of $x_1, x_2, ..., x_j$. With additional constant work, processor P_j computes the binomial coefficient $\binom{j}{y_j}$, and sets a variable z_j to $\binom{j}{y_j}$ if $x_j = 1$, and to zero otherwise. Finally, using Lemma 4, i can be obtained as $i = z_1 + z_2 + ... + z_m + 1$, again in $O(\log m)$ time by m processors.

At this point a processor, say P_1, computes the value $v = \min\{j : j \in V_i\} - 1$ using a conceptually straightforward procedure based on the definition of V_i (Section 4.2). Then, the value of v is broadcast to $P_1, P_2, ..., P_p$. As in Section 4.2, $p = \lceil (n-m) / \binom{m}{l-1} \rceil$, so that $p-1 \le |V_i| \le p$. The broadcast takes $O(\log p)$ time.

Finally, for $j = 1,2,..., |V_i|$, processor P_j fetches x_{v+j} and then participates to a computation that yields $\sum_{1 \le j \le |V_i|} x_{v+j} = \sum_{j \in V_i} x_j$, which is the value of $f_{m,n,l}$ in the present case. Again, $O(\log p)$ time is sufficient for these operations. We conclude:

Theorem 12. The positive function $f_{n,l,m}$ defined in this section can be computed by an EREW-PRAM in time $T = O(\log m + \log p)$, with $O(n+p)$ processors.

We now consider the class of functions $g_{n,l}$ obtained from $f_{n,l,m}$ as when $m = l - 2 + l\lceil n^{1/l} \rceil$, and obtain the following upper bound.

Theorem 13. The positive function $g_{n,l}$ can be computed by a EREW-PRAM in time $T = O(\log l + (\log n)/l)$ with $O(l\, n^{1/l})$ processors.

Proof. Since $m = l - 2 + l\lceil n^{1/l} \rceil$, clearly $\log m = O(\log l + (\log n)/l)$. Since $p = \left\lceil (n-m)/\binom{m}{l-1} \right\rceil$ and, as seen in the proof of Theorem 9, $\binom{m}{l-1} > n^{(l-1)/l}$, we have $p = O(n^{1/l})$, hence $\log p = O((\log n)/l)$. These bounds on m and p, used in Theorem 12, give the desired results. □

Since $g_{n,l}$ is a positive function of m variables and maximum term length l, the upper bound on T given by Theorem 13 matches with the lower bound given by Theorem 10. Similarly, for $l = \log n / \log\log n$, the time to compute $g_{n,l}$ is $O(\log\log n)$, matching the lower bound of Theorem 11.

6 Conclusions

We have proved time lower bounds for the CREW-PRAM computation of monotone functions. An important role in our techniques is played by the correspondence between monotone functions and hypergraphs. It would be of interest to extend the approach to arbitrary boolean functions.

References

[AHMP 87] H. Alt, T. Hagerup, K. Melhorn and F.P. Preparata, Simulation of idealized parallel computers on more realistic ones, *SIAM J. Comput.*, 16 (1987), pp.808-835.

[BH 85] A. Borodin, and J.E. Hopcroft, Routing, merging, and sorting on parallel models of computation, *J. of Computer and System Sciences, 30* (1985), pp. 130-145.

[CDR 86] S. Cook, C. Dwork and R. Reischuk, Upper and lower time bounds for parallel random access machines without simultaneous writes, *SIAM J. Comput., 15* (1986), pp. 87-97.

[FHRW 85] F.E. Fich, F. M. auf der Heide, P. Ragde, and A. Wigderson, One, two, three... infinity: lower bounds for parallel computation, *Proc. 17th Annual ACM Symposium on Theory of Computing*, 1985, pp. 48-58.

[FRW 84] F.E. Rich, P.L. Ragde, and A. Wigderson, Relations between concurrent - write models of parallel computation, *Proc. 3rd Annual ACM Symposium on Principles of Distributed Computing,* 1984, pp. 179-189.

[FW 78] S. Fortune, and J. Wyllie, Parallelism in random access machines, *Proc. 10th ACM Symposium on Theory of Computing*, 1978, pp. 114-118.

[Gol 82] L.M. Goldschlager, A universal interconnection pattern for parallel computers, *J. Assoc. Comput. Mach., 29* (1982), pp. 1073-1086.

[HB 88] K. Herley and G. Bilardi, Deterministic simulations of PRAMs on bounded degree networks, *Proceedings of the 26th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, Illinois, September 1988.

[KRS 85] C.P. Kruskal, L. Rudolph, and M. Snir, The power of parallel prefix, *IEEE Trans. on Computers, C-34* (1985), pp. 965-968.

[KU 86] A. Karlin, and E. Upfal, Parallel hashing - an efficient implementation of shared memory, *Proc. 18th Annual ACM Symposium on Theory of Computing*, 1986, pp. 160-168.

[LF 80] R.E. Ladner, and M.J. Fischer, Parallel prefix computation, *J. Assoc. Comput. Mach.*, 27 (1980), pp. 831-838.

[LPV 81] G. Lev, N. Pippenger, and L.G. Valiant, A fast parallel algorithm for routing in permutation networks, *IEEE Trans. on computers, C-30* (1981), pp. 93-100.

[PU 87] I.Parberry and P.Y. Yan, Improved upper and lower time bounds for parallel random access machines without simultaneous writes, *The Pensylvania State University*, TR CS-87-30, October 1987.

[Ran 87] A.G. Ranade, How to emulate shared memory, *Proc. 28th Annual IEEE Symposium on Foundations of Computer Science*, 1987, pp. 185-194.

[Sim 83] H.U. Simon, A tight $\Omega(\log\log n)$-bound on the time for parallel RAMs to compute nondegenerate boolean functions, *Proceedings of 1983 FCT*, Lecture Notes in Computer Science 158, pp.439-444.

[Sni 85] M. Snir, On parallel searching, *SIAM J. Comput.*, 14 (1985), pp. 688-708.

[Sny 86] L. Snyder, Type architectures, shared memory and the corollary of modest potential, Annual Review of Computer Science, 1(1986), pp. 289-317.

[SV 81] Y. Shiloach, and U. Vishkin, Finding the maximum, merging and sorting in a parallel computation model, *J. of Algorithms, 2* (1981), pp. 88-102.

[Tur 84] G. Turan, The critical complexity of graph properties, *Information Processing Letters, 18* (1984), pp. 151-153.

[Upf 84] E. Upfal, A probabilistic relation between desirable and feasible models of parallel computation, *Proc. of the 16th Annual ACM Symposium on Theory of Computing*, 1984, pp. 258-265.

[UW 87] E. Upfal and A. Wigderson, How to share memory in a distributed system, *J. of Association of Computing Machinery, 34* (1987), pp. 116-127.

[Vis 83] U. Vishkin, Implementation of simultaneous memory access in models that forbid it, *J. Algorithms, 4* (1983), pp. 45-50.

[Weg 85] I. Wegener, The critical complexity of all (monotone) boolean functions and monotone graph properties, *Information and Control, 67* (1985), pp. 212-222.