DIMACS Technical Report 2004-31
June 2004

A Global Parallel Algorithm for Finding All Minimal Transversals of Hypergraphs of Bounded Edge-size[1]

by

E. Boros
RUTCOR
Rutgers University
640 Bartholomew Road
Piscataway NJ 08854-8003
email: boros@rutcor.rutgers.edu

K. Elbassioni
RUTCOR
Rutgers University
640 Bartholomew Road
Piscataway NJ 08854-8003
email: elbassio@rutcor.rutgers.edu

V. Gurvich
RUTCOR
Rutgers University
640 Bartholomew Road
Piscataway NJ 08854-8003
email: gurvich@rutcor.rutgers.edu

L. Khachiyan
Department of Computer Science
Rutgers University
110 Frelinghuysen Road
Piscataway NJ 08854-8003
email: leonid@cs.rutgers.edu

[1]This research was supported in part by the National Science Foundation, grant IIS-0118635. The third author is also grateful for the partial support by DIMACS, the National Science Foundation's Center for Discrete Mathematics and Theoretical Computer Science.

DIMACS is a collaborative project of Rutgers University, Princeton University, AT&T Labs–Research, Bell Labs, NEC Laboratories America and Telcordia Technologies, as well as affiliate members Avaya Labs, HP Labs, IBM Research and Microsoft Research. DIMACS was founded as an NSF Science and Technology Center.

ABSTRACT

We give a deterministic parallel algorithm that, for any given hypergraph $\mathcal{H} \subseteq 2^V$ with maximum edge-size less than δ, outputs all minimal transversals (or equivalently all maximal independent sets) of $\mathcal{H}$ in time $O(\delta^2 \log^2 n \log m)$ using $O(n^{\delta \log \delta + 2} m^{\delta})$ processors on a CREW-PRAM, where m is the total number of minimal transversals and $n = |V|$. This algorithm can be modified so that, for any integer k, it outputs k minimal transversals of $\mathcal{H}$ in time $O(\delta^2 \log^2 n \log k + T)$ using $O(n^{\delta \log \delta + 1} k^{\delta} + k\Pi)$ processors, where T and Π are respectively the parallel time and the number of processors required to generate a single minimal transversal of $\mathcal{H}$. The latter problem is known to be in RNC for hypergraphs of bounded edge-size. This strengthens and generalizes a previously known result for graphs. We also obtain a similar result for hypergraphs whose transversal hypergraphs are δ-conformal for some constant δ.

Key Words: Bounded dimension, conformal hypergraph, dualization, global parallel algorithm, hypergraph, incremental generating, maximal independent set, minimal transversal.

1 Introduction

Let $\mathcal{H} \subseteq 2^V$ be a hypergraph (set family) on a finite vertex set V of cardinality $|V| = n$. A vertex set $X \subseteq V$ is called a *transversal* of $\mathcal{H}$ if X intersects every hyperedge of $\mathcal{H}$, and a minimal transversal, if $X \setminus \{v\}$ is not a transversal for every $v \in X$. Let $\mathcal{H}^d \subseteq 2^V$ denote the set of all minimal transversals of $\mathcal{H}$:

$$\mathcal{H}^d \stackrel{\text{def}}{=} \{X \subseteq V \mid X \text{ minimal transversal to } \mathcal{H}\}.$$

Let $\mathcal{H}^c$ be the *complementary* hypergraph

$$\mathcal{H}^c \stackrel{\text{def}}{=} \{V \setminus H \mid H \in \mathcal{H}\}$$

whose hyperedges are complementary to the hyperedges of $\mathcal{H}$. We say that a hypergraph $\mathcal{H}$ is *Sperner* if no hyperedge of $\mathcal{H}$ contains another hyperedge. Clearly, $\mathcal{H}^{cc} = \mathcal{H}$, and it is not difficult to see that for Sperner hypergraphs $\mathcal{H}^{dd} = \mathcal{H}$. A vertex set $I \subseteq V$ is called an *independent* set of $\mathcal{H}$ if I contains no hyperedge of $\mathcal{H}$, and *maximal independent* if $I \cup \{v\}$ is not independent for every $v \notin I$. Let $\mathcal{I}(\mathcal{H}) \subseteq 2^V$ denote the family of all maximal independent sets (MIS) of $\mathcal{H}$. Obviously, $I(\mathcal{H}) = \mathcal{H}^{dc}$.

We assume that a Sperner hypergraph $\mathcal{H}$ is given by the list of its hyperedges and consider problem $DUAL(\mathcal{H}, k)$ of generating k sets in $\mathcal{H}^d$ for a given integer k:

$DUAL(\mathcal{H}, k)$: *Given a Sperner hypergraph* $\mathcal{H} \subseteq 2^V$*, and an integer* $k \in \mathbb{Z}_+$ *output* $\min\{k, |\mathcal{H}^d|\}$ *minimal transversals of* $\mathcal{H}$*.*

This problem has applications in combinatorics, graph theory, artificial intelligence, game theory, reliability theory, database theory, integer programming, and learning theory (see, e.g. [2, 6, 8, 11, 16, 17, 26, 27]). It is an open question whether problem DUAL$(\mathcal{H}, k)$ can be solved in polynomial time for arbitrary hypergraphs. The fastest currently known algorithm [12] is quasi-polynomial and works by considering the following incremental generation problem:

$DUAL(\mathcal{H}, \mathcal{X})$: *Given a Sperner hypergraph* $\mathcal{H}$ *and a collection* $\mathcal{X} \subseteq \mathcal{H}^d$ *of minimal transversals to* $\mathcal{H}$*, either find a new minimal transversal* $X \in \mathcal{H}^d \setminus \mathcal{X}$ *or show that* $\mathcal{X} = \mathcal{H}^d$*.*

The running time of the algorithm in [12] is $O(nm) + m^{o(\log m)}$, where $n = |V|$ and $m = |\mathcal{H}| + |\mathcal{X}|$. For hypergraphs of bounded dimension,

$$\dim(\mathcal{H}) \stackrel{\text{def}}{=} \max_{H \in \mathcal{H}} |H| \leq const, \tag{1}$$

it was shown in [7, 11], that problem DUAL$(\mathcal{H}, \mathcal{X})$ can be solved in polynomial time. Moreover, [5] shows that the problem can be efficiently solved in parallel, DUAL$(\mathcal{H}, \mathcal{X}) \in NC$ for $dim(\mathcal{H}) \leq 3$ and DUAL$(\mathcal{H}, \mathcal{X}) \in RNC$ for $dim(\mathcal{H}) = 4, 5, \ldots$ The latter result is based on an NC-reduction of problem DUAL$(\mathcal{H}, \mathcal{X})$ to problem DUAL$(\mathcal{H}', \emptyset)$ of finding a single minimal transversal in a partial sub-hypergraph $\mathcal{H}'$ of $\mathcal{H}$, for which NC algorithms exist,

if $\dim(\mathcal{H}) \leq 3$ (see [1, 10, 14, 15, 22]), and an RNC algorithm exists if $\dim(\mathcal{H}) = 4, 5, \ldots$ (see [3, 23]). Note, however, that the result of [5] implies only that we can solve problem DUAL$(\mathcal{H}, k)$, in parallel, in time $k \cdot \text{polylog}(n, |\mathcal{H}|, k)$ on $\text{poly}(n, |\mathcal{H}|, k)$ processors. In this paper, we are concerned with *global parallel algorithms* for the hypergraph transversal problem, i.e. algorithms that solve problem DUAL$(\mathcal{H}, k)$ in time $\text{polylog}(n, |\mathcal{H}|, k)$ on $\text{poly}(n, |\mathcal{H}|, k)$ processors. The special case of $\delta = 2$, i.e., when $\mathcal{H} = E$ for a graph $G = (V, E)$, was considered in [9], where it was shown that the problem can be solved in $O(\log^3(nk))$ parallel time on $O(n^2k^6)$ processors, on a CREW-PRAM. Here we show the following stronger and more general results.

There is a deterministic parallel algorithm that, for any hypergraph $\mathcal{H} \subseteq 2^V$ on $|V| = n$ vertices such that $\dim(\mathcal{H}) < \delta$, solves problem DUAL$(\mathcal{H}, |\mathcal{H}^d|)$ in time $O(\delta^2 \log^2 n \log |\mathcal{H}^d|)$ using $O(n^{\delta \log \delta + 2} |\mathcal{H}^d|^\delta)$ processors on a CREW-PRAM.

We also give an NC reduction of finding k minimal transversals of $\mathcal{H}$, when $\dim(\mathcal{H}) \leq$ *Const.*, to computing a single minimal transversal in a restricted sub-hypergraph of $\mathcal{H}$.

There is a deterministic parallel algorithm that, for any hypergraph $\mathcal{H} \subseteq 2^V$ on $|V| = n$ vertices such that $\dim(\mathcal{H}) < \delta$, and any integer $k \in \mathbb{Z}_+$, solves problem DUAL$(\mathcal{H}, k)$ in time $O(\delta^2 \log^2 n \log k + T)$ using $O(n^{\delta \log \delta + 2} k^\delta + k\Pi)$ processors on a CREW-PRAM, where T and Π are respectively the parallel time and the number of processors required to generate a single minimal transversal of $\mathcal{H}$. Furthermore, if $\delta \leq 4$, then there is a deterministic algorithm with the same running time and number of processors.

Theorem 1 extends the results of [3, 23] which show that problem DUAL$(\mathcal{H}, k) \in RNC$, for hypergraphs $\mathcal{H}$ of bounded dimension and for $k = 1$, to any integer k. Theorem 1 also extends the results of [9] where it was shown that problem DUAL$(\mathcal{H}, k) \in NC$, for hypergraphs of dimension 2 and for any integer k, to hypergraphs of any bounded dimension. We prove Theorems 1 and 1 in Sections 2 and 3. In Section 4, we examine another related special class of hypergraphs for which a similar result can be obtained. More precisely, given a hypergraph $\mathcal{H} \subseteq 2^V$, a vertex set $S \subseteq V$ is called a *sub-transversal* of $\mathcal{H}$ if $S \subseteq X$ for some minimal transversal $X \in \mathcal{H}^d$. By the above definitions, the hypergraph $\mathcal{H}^{dcd}$ is just the family of *minimal non sub-transversals* of $\mathcal{H}$. Let $\mathcal{H}$ be a hypergraph such that

$$\dim(\mathcal{H}^{dcd}) < \delta, \tag{2}$$

for some constant δ. It is not clear how to check (2), even for $\delta = 3$. However, as far as the generation of the dual hypergraph $\mathcal{H}^d$ is concerned, such a check is not needed. In fact, we shall present an efficient parallel algorithm that, for any given constant δ, either solves problem DUAL$(\mathcal{H}, k)$ or discovers that (2) is not satisfied.

There is a deterministic parallel algorithm that, given a hypergraph $\mathcal{H} \subseteq 2^V$ on $|V| = n$ vertices, and integers $k, \delta \in \mathbb{Z}_+$, either solves problem DUAL$(\mathcal{H}, k)$, or proves that $\dim(\mathcal{H}^{dcd}) \geq \delta$, on a CREW-PRAM in time $O(\delta^2 \log^2 n \log k + T)$ using $O(n^{2\delta \log \delta} k^\delta + k\Pi)$ processors on a CREW-PRAM, where T and Π are respectively the parallel time and the number of processors required to generate a single minimal transversal of a hypergraph of dimension less than δ.

A hypergraph $\mathcal{H} \subseteq 2^V$ is said to be δ-*conformal* [4] if for every vertex-set $X \subseteq V$ the following property holds: X is contained in a hyperedge of $\mathcal{H}$ whenever each subset of X of cardinality at most δ is contained in a hyperedge of $\mathcal{H}$. It is not difficult to see that a hypergraph $\mathcal{H}$ satisfies (2) if and only if its dual $\mathcal{H}^d$ is $(\delta-1)$-conformal. Note that the dualization problem for hypergraphs $\mathcal{H}$ satisfying $\dim(\mathcal{H}^d) < \delta$, for some constant δ, can be trivially solved efficiently in parallel, just by enumerating all subsets of vertices of size less than δ, and checking which of them are minimal transversals. Note further that $\dim(\mathcal{H}^c) \leq \delta$ implies that $\dim(\mathcal{H}^d) \leq \delta + 1$. Indeed, if the complement to every edge of $\mathcal{H}$ has cardinality at most δ then clearly every set of cardinality $\delta+1$ is a transversal. Substitute $\mathcal{H}^{dc}$ for $\mathcal{H}$ and $\delta+1$ for δ and obtain that if $\dim(\mathcal{H}^d) < \delta$ then $dim(\mathcal{H}^{dcd}) < \delta+1$. Thus Theorem 1 extends the parallel dualization result for hypergraphs whose duals are of bounded dimension to the wider class of hypergraphs whose duals are δ-conformal, for some constant δ.

2 Global parallel dualization algorithm

Let V be a finite set of cardinality $|V| = n$. Given a hypergraph $\mathcal{H} \subseteq 2^V$ and a subset $S \subseteq V$, denote by

$$\mathcal{H}_S = \{H \in \mathcal{H} \mid H \subseteq S\}$$

the sub-hypergraph of $\mathcal{H}$ induced by S. For $r \in \mathbb{Z}_+$ and $0 < \epsilon < 1$, denote by $\mathbb{H}(r, \epsilon)$ the family of hypergraphs $\mathcal{H} \subseteq 2^V$, such that for every $S \subseteq V$, there exist subsets $S_1, \ldots, S_r \subseteq S$ satisfying:

(H1) For every $H \in \mathcal{H}_S$, there exists an $i \in \{1, \ldots, r\}$ such that $S_i \supseteq H$.

(H2) $|S_i| \leq (1-\epsilon)|S|$, for each $i \in \{1, \ldots, r\}$ for which $|\mathcal{H}_{S_i}| > 1$.

[[24]] For any hypergraph $\mathcal{H} \subseteq 2^V$ and any subset $S \subseteq V$, we have $|\mathcal{H}^d_S| \leq |\mathcal{H}^d|$. Any minimal transversal $X \in \mathcal{H}^d$ can be written $X = Y \cup Z$, where $Y \in \mathcal{H}^d_S$ and Z is a minimal transversal to the sub-hypergraph $\{H \setminus S \mid H \cap Y = \emptyset\}$.

Given two hypergraphs $\mathcal{H}_1, \mathcal{H}_2 \subseteq 2^V$, denote by

$$\mathcal{H}_1 \bigwedge \mathcal{H}_2 = \{\text{minimal } H \mid H = H_1 \cup H_2 \text{ for some } H_1 \in \mathcal{H}_1 \text{ and } H_2 \in \mathcal{H}_2\}$$

the conjunction of $\mathcal{H}_1$ and $\mathcal{H}_2$.

Let $\mathcal{H} \subseteq 2^V$ be a hypergraph and $S_1, \ldots, S_r \subseteq V$ be subsets such that for every edge $H \in \mathcal{H}$ there is an $i \in \{1, \ldots r\}$ such that $H \subseteq S_i$. Then

$$\mathcal{H}^d = \bigwedge_{i=1}^{n} \mathcal{H}^d_{S_i}. \tag{3}$$

Note that $\mathcal{H} = \cup_{i=1}^n \mathcal{H}_{S_i}$ by our assumption. Thus, if $X \in \bigwedge_{i=1}^n \mathcal{H}^d_{S_i}$, then $X = H_1 \cup \ldots \cup \mathcal{H}_r$, for some $H_1 \in \mathcal{H}_{S_1}, \ldots, H_r \in \mathcal{H}_{S_r}$, and consequently, X is transversal to every edge of $\mathcal{H}$.

Conversely, if $X \in \mathcal{H}^d$, then $X \cap S_i$ is a transversal to $\mathcal{H}_{S_i}$, and therefore $X \supseteq X_1 \cup \ldots \cup X_r$, for some $X_1 \in \mathcal{H}^d_{S_1}, \ldots, X_r \in \mathcal{H}^d_{S_r}$. Now the statement follows since both sides of (3) are Sperner, by definition.

Consider the following parallel algorithm for generating all minimal transversals of a hypergraph $\mathcal{H} \in \mathbb{H}(r, \epsilon)$:

Procedure DUALIZE$(\mathcal{H}, V)$**:**

1. If $|\mathcal{H}| = 0$, then return $\emptyset$.
2. If $\mathcal{H}$ has only one hyperedge H, then
3. Return $\{\{i\} \ : \ i \in H\}$.
4. In parallel, do the following:
5. find the sets $S_1, \ldots, S_r \subseteq V$ satisfying (H1) and (H2) with $S = V$.
6. Let $\mathcal{G}_i \leftarrow$DUALIZE$(\mathcal{H}_{S_i}, S_i)$, for $i = 1, \ldots, r$.
7. Compute the conjunction $\mathcal{G} \leftarrow \bigwedge_{i=1}^{r} \mathcal{G}_i$.
8. Return $\mathcal{G}$.

Proposition 2 implies that the above procedure outputs all minimal transversals of $\mathcal{H} \in \mathbb{H}(r, \epsilon)$ correctly. The following proposition gives the running time and number of processors required by this procedure, in terms of the time $\tau = \tau(n, |\mathcal{H}|, r, \epsilon)$ and the number of processors $\pi = \pi(n, |\mathcal{H}|, r, \epsilon)$ needed to find the sets $S_1, \ldots, S_r \subseteq V$, satisfying (H1) and (H2). We shall assume in what follows that we run our procedures on a CREW-PRAM.

Let $t(n, m)$ and $p(n, m)$ be respectively the time and the number of processors, required by procedure DUALIZE$(\mathcal{H}, V)$ to output all minimal transversals of a hypergraph $\mathcal{H} \in \mathbb{H}(r, \epsilon)$ on n vertices and with $|\mathcal{H}^d| = m$. Then

$$t(n, m) = O((\tau + r \log n \log m)\frac{\log n}{\epsilon}) \text{ and } p(n, m) = O(\pi n^{\frac{\log r}{\epsilon}+1} m^r).$$

Step 5 requires time at most τ, and Step 6 requires at most $t((1-\epsilon)n, m)$. We also note by Proposition 2 that $|\mathcal{H}^d_{S_i}| \leq m$ for all $i = 1, \ldots, r$, and thus Step 7 can be done in $O(r \log m \log n)$ using $O(m^r n)$ processors. This gives the recurrence

$$t(n, m) = O(\tau + r \log m \log n) + t((1-\epsilon)n, m), \tag{4}$$

implying the bound on the running time. As for the number of processors, we get the recurrence

$$p(n, m) = O(\pi + m^r n) + r \cdot p((1-\epsilon)n, m), \tag{5}$$

implying the bound stated in the proposition.

Note that, in the above procedure, all minimal transversals are generated simultaneously, at the very end, and there is no need to generate some of them individually, in the course

of the computations. In other words, we did not need, in the above procedure, to compute a single minimal transversal in parallel. In general, the efficient parallel computation of a single minimal transversal of an arbitrary hypergraph is an outstanding open problem [20, 19], for which the fastest known parallel algorithm [21], runs in $O(\sqrt{n})$ expected parallel time on $O(n^{3/2})$ processors. For some special classes of hypergraphs, e.g. hypergraphs of bounded dimension [1, 3, 10, 15, 14, 22, 23], hypergraphs of bounded vertex-degrees [13], and linear hypergraphs [25, 28], there exist efficient parallel algorithms for finding a minimal transversal.

However, if the requirement is not to generate all minimal transversals of $\mathcal{H}$ but only k of them, then the above procedure is not appropriate in its current form, but in fact can be modified, provided that we know how to generate an individual minimal transversal efficiently in parallel.

Procedure DUALIZE$(\mathcal{H}, V, k)$**:**

1. If $|\mathcal{H}| = 0$, then return $\emptyset$.
2. If $\mathcal{H}$ has only one hyperedge H, then
3. Return $\{\{i\} \ : \ i \in H\}$.
4. In parallel, do the following:
5. find the sets $S_1, \ldots, S_r \subseteq V$ satisfying (H1) and (H2) with $S = V$.
6. Let $\mathcal{G}_i \leftarrow$DUALIZE$(\mathcal{H}_{S_i}, S_i, k)$, for $i = 1, \ldots, r$.
7. If there is an $i \in \{1, \ldots, r\}$ such that $|\mathcal{G}_i| = k$ then
8. In parallel, for each $Y \in \mathcal{G}_i$, do the following:
9. Let $\mathcal{H}_Y = \{H \setminus S_i \mid H \cap Y = \emptyset\}$.
10. Compute a minimal transversal T_Y of $\mathcal{H}_Y$.
11. Return $\mathcal{G} \leftarrow \{Y \cup T_Y \ : \ Y \in \mathcal{G}_i\}$, and stop.
12. else
13. Compute the conjunction $\mathcal{G} \leftarrow \bigwedge_{i=1}^{r} \mathcal{H}^d_{S_i}$.
14. Return $\mathcal{G}$.

Procedure DUALIZE$(\mathcal{H}, V, k)$ outputs k (or all if $|\mathcal{H}^d| \leq k$) minimal transversals of a hypergraph $\mathcal{H} \in \mathbb{H}(r, \epsilon)$ on n vertices, in time $t(n, k) = O((\tau + r \log n \log k) \log n/\epsilon + T)$, using $p(n, k) = O(\pi n^{\log r/\epsilon + 1} k^r + k\Pi)$ processors, where T and Π are respectively the parallel time and the number of processors required to generate a single minimal transversal of $\mathcal{H}$. By Step 4, if there is a set S_i, for some $i \in \{1, \ldots, r\}$, such that $|\mathcal{H}^d_{S_i}| \geq k$, then the time and number or processors needed to finish the computation are T and $k\Pi$, respectively. On the other hand if, for all $i \in \{1, \ldots, r\}$, we have $|\mathcal{H}^d_{S_i}| < k$, then recurrences (4) and (5) hold with $m < k$, and we get the bounds stated in the proposition.

3 Proofs of Theorems 1 and 1

In this section we consider hypergraphs of bounded dimension. Let $\mathcal{H} \subseteq 2^V$ be a hypergraph such that $\dim(\mathcal{H}) < \delta$. To apply the results of the preceding section to solve problem DUAL$(\mathcal{H}, k)$, we let $r = \delta$ and $\epsilon = 1/\delta$. Then $\mathcal{H} \in \mathbb{H}(r, \epsilon)$. Indeed, given any set $S \subseteq V$, we partition S into r equal parts $W_1, \ldots, W_r$, and let $S_i = S \setminus W_i$, for $i = 1, \ldots, r$. The fact that any hyperedge $H \in \mathcal{H}_S$ has size at most δ implies that H cannot intersect all sets $W_1, \ldots, W_r$, i.e. H must be contained in at least one of the sets $S_1, \ldots, S_r$. This implies that (H1) holds for $\mathcal{H}$. (H2) follows form the fact that $|S_i| = \frac{(\delta-1)|S|}{\delta}$, for $i = 1, \ldots, r$. Note also that, for constant δ, the sets $S_1, \ldots, S_r$ can be computed in constant time using $O(n)$ processors.

Thus Theorems 1 and 1 follow from Propositions 2 and 2 respectively.

4 Proof of Theorem 1

In this section, we consider dual δ-conformal hypergraphs. Let $\mathcal{H} \subseteq 2^V$ be a hypergraph such that (2) is satisfied for some constant δ. Below, we shall present an algorithm that, will keep generating in parallel minimal transversals of $\mathcal{H}$, and halt only when either all or at least k of such transversals have been generated, or when the algorithm discovers that condition (2) is not satisfied.

For a subset $S \subseteq V$, and a vertex $v \in S$, let $\mathcal{H}_v(S) = \{H \in \mathcal{H} \mid H \cap S = \{v\}\}$ denote the family of all hyperedges of the hypergraph $\mathcal{H}$ whose intersection with S is exactly v. Let further $\mathcal{H}_0(S) = \{H \in \mathcal{H} \mid H \cap S = \emptyset\}$ denote the partial hypergraph consisting of the hyperedges of $\mathcal{H}$ disjoint from S. A selection of $|S|$ hyperedges $\{H_v \in \mathcal{H}_v(S) \mid v \in S\}$ is called *covering* if there exists a hyperedge $H \in \mathcal{H}_0(S)$ such that $H \subseteq \bigcup_{v \in S} H_v$. The next proposition states that a non-empty set S is a sub-transversal of $\mathcal{H}$ if and only if there exists a non-covering selection for S.

[cf. [7]] Let $S \subseteq V$ be a non-empty vertex set in a hypergraph $\mathcal{H} \in 2^V$.

i) If S is a sub-transversal for $\mathcal{H}$ then there exists a non-covering selection $\{H_v \in \mathcal{H}_v(S) \mid v \in S\}$ for S.

ii) Given a non-covering selection $\{A_v \in \mathcal{A}_v(S) \mid v \in S\}$ for S, we can extend S to a minimal transversal of $\mathcal{H}$ by solving problem DUAL$(\mathcal{H}', 1)$ for the induced partial hypergraph

$$\mathcal{H}' = \{H \cap U \mid H \in \mathcal{H}_0(S)\} \subseteq 2^U,$$

where

$$U = V \setminus \bigcup_{v \in S} H_v.$$

In general, finding a non-covering selection for S (or equivalently, testing if S is a sub-transversal) is NP-hard if the cardinality of S is not bounded. In fact, this is so even for

$\dim(\mathcal{H}) = 2$, that is for graphs (see [5]). However, if the size of S is bounded by a constant then there are only polynomially many selections $\{H_v \in \mathcal{H}_v(S) \mid v \in S\}$ for S. All of these selections, including the non-covering ones, can be enumerated efficiently in parallel.

For any fixed δ there is an algorithm which, given a hypergraph $\mathcal{H} \subseteq 2^V$ and a set S of less than δ vertices, determines whether S is a sub-transversal to $\mathcal{H}$ and if so finds a non-covering selection $\{H_v \in \mathcal{H}_v(S) \mid v \in S\}$, in $O(\log \delta \log |V|)$ parallel time using $O((|V||\mathcal{H}|)^\delta)$ processors.

Let δ be a given constant. Given a hypergraph $\mathcal{H}$ satisfying (2) and any integer k, the algorithm proceeds in the following two steps:

Step 1. Generate the hypergraph $\mathcal{F} \subseteq 2^V$, whose hyperedges are defined as follows:

$$\mathcal{F} = \{S \subseteq V \ : \ |S| < \delta \text{ and } S \text{ is a minimal non sub-transversal of } \mathcal{H}\}. \tag{6}$$

For a constant δ, the hypergraph $\mathcal{F}$ can be generated in $O(\log |V|)$ parallel time using $O(|V|^{2\delta}|\mathcal{H}|)^\delta)$ processors by Corollary 4.

Step 2. Note that $\dim(\mathcal{F}) < \delta$. Thus Theorem 1 implies that problem DUAL$(\mathcal{F}, k)$ can be solved efficiently in global RNC time. However, we do not need to generate always all the hyperedges of $\mathcal{F}^d$. We stop generation when either an edge $X \in \mathcal{F}^d$ is generated such that $V \setminus X$ is not a minimal transversal of $\mathcal{H}$, or when k edges of $\mathcal{F}^d$ have been generated, whichever happens sooner.

To verify that the above procedure indeed generates k transversals of $\mathcal{H}^d$ in global RNC time if (2) is satisfied, notice the equivalences

$$\dim(\mathcal{H}^{dcd}) \leq \delta \iff \mathcal{F} = \mathcal{H}^{dcd} \iff \mathcal{F}^{dc} \subseteq \mathcal{H}^d.$$

The first equivalence is clear from the definition of $\mathcal{F}$ since a subset $S \subseteq V$ belongs to $\mathcal{H}^{dcd}$ if and only if S is is a minimal non sub-transversal of $\mathcal{H}$. To see the second equivalence, suppose that $\mathcal{F}^{dc} \subseteq \mathcal{H}^d$, and take an $X \in \mathcal{H}^{dcd}$. Then $\mathcal{F}^{dc} \subseteq \mathcal{H}^d$ implies that X contains a subset $X' \in \mathcal{F}$, while $\mathcal{F} \subseteq \mathcal{H}^{dcd}$ implies that $X' \in \mathcal{H}^{dcd}$. But since $\mathcal{H}^{dcd}$ is Sperner, we get $X = X' \in \mathcal{F}$.

Now Theorem 1 becomes a consequence of Proposition 2, applied to the hypergraph $\mathcal{F}$ constructed in Step 1 above.

References

[1] N. Alon, L. Babai, A. Itai, A fast randomized parallel algorithm for the maximal independent set problem, J. Algorithms 7 (1986) 567-583

[2] M. Anthony and N. Biggs, *Computational Learning Theory*, Cambridge University Press, 1992.

[3] P. Beame, M. Luby, Parallel search for maximal independence given minimal dependence, Proceedings of the First SODA Conference (1990) 212-218.

[4] C. Berge, *Hypergraphs*, North Holland Mathematical Library, Vol. 445, Elsevier-North Holland, Amestrdam, 1989.

[5] E. Boros, K. Elbassioni, V. Gurvich, and L. Khachiyan, An efficient incremental algorithm for generating all maximal independent sets in hypergraphs of bounded dimension, *Parallel Processing Letters*, **10** (2000), pp. 253–266.

[6] E. Boros, K. Elbassioni, V. Gurvich, L. Khachiyan and K.Makino, Dual-bounded generating problems: All minimal integer solutions for a monotone system of linear inequalities, *SIAM Journal on Computing*, **31** (5) (2002) pp. 1624–1643.

[7] E. Boros, V. Gurvich, and P.L. Hammer, Dual subimplicants of positive Boolean functions, *Optimization Methods and Software*, 10 (1998) 147-156.

[8] C. J. Colbourn, *The combinatorics of network reliability*, Oxford University Press, 1987.

[9] E. Dahlhaus and M. Karpinski, A fast parallel algorithm for computing all maximal cliques in a graph and the related problems, *Proc. 1st Scandinavian Workshop on Algorithm Theory (SWAT)* Halmstad, Sweden, July 5-8, 1988, LNCS 318, pp. 139-144.

[10] E. Dahlhaus, M. Karpinski, P. Kelsen, An efficient parallel algorithm for computing a maximal independent set in a hypergraph of dimension 3, *Inf. Process. Lett.* **42**(6), pp. 309-313 (1992).

[11] T. Eiter and G. Gottlob, Identifying the minimal transversals of a hypergraph and related problems, *SIAM Journal on Computing*, 24 (1995), pp. 1278–1304.

[12] M. L. Fredman and L. Khachiyan, On the complexity of dualization of monotone disjunctive normal forms. *J. Algorithms*, 21 (1996) 618-628.

[13] O. Garrido, P. Kelsen and A. Lingas, A simple NC-algorithm for a maximal independent set in a hypergraph of polylog arboricity, *Information Processing Letters* 58(2) (1996), pp. 55–58.

[14] M. Goldberg and T. Spencer, Constructing a maximal independent set in parallel, SIAM J. Disc. Math. 2 (1989) 322–328.

[15] M. Goldberg and T. Spencer, A new parallel algorithm for the maximal independent set problem, SIAM J. Computing 18 (1989) 419-427.

[16] V. Gurvich, To theory of multistep games, *USSR Comput. Math. and Math Phys.* **13-6** (1973) 1485-1500.

[17] V. Gurvich, Nash-solvability of games in pure strategies, *USSR Comput. Math and Math. Phys.* **15-2** (1975) 357-371.

[18] D. S. Johnson, M. Yannakakis and C. H. Papadimitriou, On generating all maximal independent sets, *Information Processing Letters*, 27 (1988) 119-123.

[19] R. Karp and V. Ramachandran, Parallel algorithms for shared memory machines, in Handbook of Theoretical Computer Science, J. van Leeuwen, ed., North Holland (1990) 869-941.

[20] R. Karp and A. Wigderson, A fast parallel algorithm for the maximal independent set problem, JACM 32 (1985) 762-773.

[21] R. Karp, E. Upfal, and A. Wigderson, The complexity of parallel search, *Journal of Computer and System Science*, 36 (1988) 225-253.

[22] P. Kelsen, An efficient parallel algorithm for finding an MIS in hypergraphs of dimension 3, Manuscript, Dept. of Computer Sciences, Univ. of Texas, Austin, TX, Jan. 1990.

[23] P. Kelsen, On the parallel complexity of computing a maximal independent set in a hypergraph, Proceedings of the 24-th Anual ACM STOC Conference (1992).

[24] E. Lawler, J. K. Lenstra and A. H. G. Rinnooy Kan, Generating all maximal independent sets: NP-hardness and polynomial-time algorithms, *SIAM Journal on Computing*, 9 (1980) 558-565.

[25] T. Luczak and E. Szymanska, A parallel randomized algorithm for finding a maximal independent set in a linear hypergraph, *J. Algorithms*, v.25 (1997) n.2, pp. 311–320.

[26] K. G. Ramamurthy, *Coherent Structures and Simple Games*, Kluwer Academic Publishers, 1990.

[27] R. C. Read, Every one a winner, or how to avoid isomorphism when cataloging combinatorial configurations, *Annals of Discrete Mathematics* 2 (1978) 107-120.

[28] E. Szymanska, Derandomization of a Parallel MIS Algorithm in a Linear Hypergraph, in *Proc. 4th International Workshop on Randomization and Approximation Techniques in Computer Science*, pp. 39–52, 2000.