DIMACS Technical Report 2004-44

Computing Many Maximal Independent Sets for Hypergraphs in Parallel[1]

by

Endre Boros,[2] Khaled Elbassioni[2] Vladimir Gurvich[2]
Leonid Khachiyan[3]

[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.
[2]RUTCOR, Rutgers University, 640 Bartholomew Road, Piscataway NJ 08854-8003; ({boros,elbassio, gurvich}@rutcor.rutgers.edu).
[3]Department of Computer Science, Rutgers University, 110 Frelinghuysen Road, Piscataway, New Jersey 08854-8004; (leonid@cs.rutgers.edu.)

ABSTRACT

A hypergraph $\mathcal{H} \subseteq 2^V$ is called *uniformly δ-sparse* if for every nonempty subset $X \subseteq V$ of vertices, the average degree of the sub-hypergraph of $\mathcal{H}$ induced by X is at most δ. We show that there is a deterministic algorithm that, given a uniformly δ-sparse hypergraph $\mathcal{H}$, and a positive integer k, outputs k or all minimal transversals for $\mathcal{H}$ in $O(\delta \log(1+k) \text{polylog}(\delta|V|))$-time using $|V|^{O(\log \delta)} k^{O(\delta)}$ processors. Equivalently, the algorithm can be used to compute in parallel k or all maximal independent sets for $\mathcal{H}$.

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 $v \in V$. Let $\mathcal{H}^d \subseteq 2^V$ denote the set of all minimal transversals of $\mathcal{H}$:

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

A hypergraph $\mathcal{H}$ is said to be *Sperner* if no hyperedge of $\mathcal{H}$ contains another hyperedge. Clearly, for Sperner hypergraphs $\mathcal{H}^{dd} = \mathcal{H}$. We assume that a Sperner hypergraph $\mathcal{H}$ is given by the list of its hyperedges and consider the problem 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 $\mathcal{H}$.*

Note that $X \in \mathcal{H}^d$ if and only if $V \setminus X$ is a maximal independent set for $\mathcal{H}$. Problem $DUAL(\mathcal{H}, k)$ is thus equivalent to computing k or all maximal independent sets of vertices for a given hypergraph $\mathcal{H}$. Applications of $DUAL(\mathcal{H}, k)$ include combinatorics, graph theory, artificial intelligence, game theory, reliability theory, database theory, integer programming, and learning theory (see, e.g. [2, 8, 11, 15, 21, 22, 30, 31]).

In the special case of $k = 1$, $DUAL(\mathcal{H}, 1)$ calls for computing a single maximal independent set (or minimal transversal) of $\mathcal{H}$. It is known that problem $DUAL(\mathcal{H}, 1)$ can be solved efficiently in parallel for some special classes of hypergraph, e.g. for hypergraphs of bounded dimension (when $\dim(\mathcal{H}) \stackrel{\text{def}}{=} \max\{|H| : H \in \mathcal{H}\} \leq \textit{const}$), see [1, 3, 13, 20, 19, 27], for hypergraphs of bounded arboricity [18], and for linear hypergraphs [29, 32]. However in general, the existence of an efficient parallel algorithm for $DUAL(\mathcal{H}, 1)$ is an outstanding open problem [25]. For arbitrary input hypergraphs, the running time of the fastest currently known parallel algorithm [26] for $DUAL(\mathcal{H}, 1)$ grows as the square root of the size of $\mathcal{H}$.

Even though $\mathrm{DUAL}(\mathcal{H}, 1)$ can clearly be solved in polynomial time sequentially for any input hypergraph $\mathcal{H}$, the fastest currently known sequential algorithm [17] for $\mathrm{DUAL}(\mathcal{H}, k)$ is quasi-polynomial and runs in $poly(n) + m^{o(\log m)}$ time, where $n = |V|$ and $m = |\mathcal{H}| + k$. This algorithm works by iteratively solving k times the following incremental generation problem:

$NEXT(\mathcal{H}, \mathcal{X})$: *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 prove that $\mathcal{X} = \mathcal{H}^d$.*

For hypergraphs of bounded dimension, problem $\mathrm{NEXT}(\mathcal{H}, \mathcal{X})$ is NC-reducible to $\mathrm{DUAL}(\mathcal{H}', 1)$, where $\mathcal{H}'$ a partial sub-hypergraph $\mathcal{H}'$ of $\mathcal{H}$, see [5]. Since the dimension of $\mathcal{H}'$ is bounded by the dimension of $\mathcal{H}$, this yields an efficient parallel algorithm for $\mathrm{NEXT}(\mathcal{H}, \mathcal{X})$. Note, however, that when k is not bounded, an NC-algorithm for $\mathrm{NEXT}(\mathcal{H}, \mathcal{X})$ does not necessarily imply an efficient parallel algorithm for $\mathrm{DUAL}(\mathcal{H}, k)$.

In this paper, we are concerned with *global parallel algorithms* for the hypergraph transversal problem, i.e. NC-algorithms that solve problem $\mathrm{DUAL}(\mathcal{H}, k)$ in time $\mathrm{polylog}(n, |\mathcal{H}|, k)$

on poly$(n, |\mathcal{H}|, k)$ processors. When $\mathcal{H}$ is a graph, i.e. $\dim \mathcal{H} = 2$, such an algorithm was suggested in [12], where it was shown that problem DUAL$(\mathcal{H}, k)$ can be solved in $O(\log^3(nk))$ parallel time on an $O(n^2k^6)$-processor CREW-PRAM. More generally, it was shown in [7] that problem DUAL$(\mathcal{H}, k)$ remains in RNC for hypergraphs of any bounded dimension.

The goal of this paper is to obtain a global parallel dualization algorithm for another natural class of hypergraphs obtained by bounding the degrees of all vertices. In fact, we will show that DUAL$(\mathcal{H}, k) \in$ NC for the wider class of *uniformly sparse* hypergraphs, defined below.

Let us denote by $\deg_{\mathcal{H}}(v) \stackrel{\text{def}}{=} |\{H \in \mathcal{H} \ : \ v \in H\}|$ the degree of a vertex $v \in V$ in the hypergraph $\mathcal{H}$. We say that $\mathcal{H}$ is of *bounded degree* if

$$\text{max-deg}(\mathcal{H}) \stackrel{\text{def}}{=} \max_{v \in V} \deg_{\mathcal{H}}(v) \leq \delta \tag{1}$$

holds for some fixed δ. For a subset $S \subseteq V$ let $\mathcal{H}_S = \{H \in \mathcal{H} \ : \ H \subseteq S\}$ be the sub-hypergraph induced by S. Let us call $\mathcal{H}$ *uniformly δ-sparse*, if for every subset $S \subseteq V$, the *average* vertex degree of $\mathcal{H}_S$ is at most δ:

$$\text{avg-deg}(\mathcal{H}_S) \stackrel{\text{def}}{=} \frac{1}{|S|} \sum_{v \in S} \deg_{\mathcal{H}_S}(v) \leq \delta. \tag{2}$$

A polynomial-time sequential algorithm for solving problem DUAL$(\mathcal{H}, k)$ for bounded degree hypergraphs was suggested in [14]. This result was recently strengthened in [16], where a polynomial delay algorithm was obtained for the wider class of *δ-degenerate* hypergraphs, which includes uniformly δ-sparse hypergraphs. However, there were no known efficient parallel algorithms for these classes of hypergraphs until now. Our main result is the construction of such an efficient parallel procedure for uniformly δ-sparse hypergraphs:

Theorem 1 *For uniformly δ-sparse hypergraphs problem DUAL$(\mathcal{H}, k)$ can be solved in $O(\delta \log(1+k) \text{polylog}(\delta n))$-time using $n^{O(\log \delta)} k^{O(\delta)}$ processors, implying that DUAL$(\mathcal{H}, k) \in$ NC whenever δ is a fixed constant.*

The proof of Theorem 1 has two main ingredients. First, we give an NC-reduction of problem DUAL$(\mathcal{H}, k)$ for a hypergraph $\mathcal{H}$ satisfying (2) to k problems, each computing a single minimal transversal in some sub-hypergraphs of $\mathcal{H}$. (Note that all sub-hypergraphs of a uniformly δ-sparse hypergraph are also uniformly δ-sparse.)

Theorem 2 *Given any uniformly δ-sparse hypergraph $\mathcal{H} \subseteq 2^V$ on $|V| = n$ vertices, problem DUAL$(\mathcal{H}, k)$ can be solved in time $O(\delta \log n \log(\delta nk) + T)$ using $n^{O(\log \delta)} k^{O(\delta)} + k\Pi$ 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 for a sub-hypergraph of $\mathcal{H}$.*

The second ingredient in the proof of Theorem 1 is the result that a single minimal transversal of a locally δ-sparse hypergraph can be found efficiently in parallel.

Theorem 3 *A minimal transversal in a uniformly δ-sparse hypergraph on n vertices can be found in $O(\delta \log n \log \delta n)$ expected parallel time on $O(n^2)$ processors, and in $O(\delta \log^{O(1)}(\delta n))$ deterministic parallel time using $n^{O(1)}$ processors.*

2 A divide-and-conquer approach for solving DUAL($\mathcal{H}, k$)

Given two hypergraphs $\mathcal{H}_1, \mathcal{H}_2 \subseteq 2^V$, let us denote by $\mathcal{H}_1 \wedge \mathcal{H}_2 = \min\{H_1 \cup H_2 \ : \ H_1 \in \mathcal{H}_1 \text{ and } H_2 \in \mathcal{H}_2\}$ the conjunction of $\mathcal{H}_1$ and $\mathcal{H}_2$, where $\min \mathcal{H}$ denotes the sub-hypergraph consisting of all containment-wise minimal hyperedges of $\mathcal{H}$. Let us further call a family of subsets $S_1, ..., S_r \subseteq V$ *complete* for the hypergraph $\mathcal{H}$, if for every $H \in \mathcal{H}$, there exists an index $i \in \{1, \ldots, r\}$ such that $S_i \supseteq H$.

Proposition 1 *Let $\mathcal{H} \subseteq 2^V$ be a hypergraph and $S_1, \ldots, S_r \subseteq V$ be a complete family of subsets for $\mathcal{H}$. Then, $\mathcal{H}^d = \bigwedge_{i=1}^r \mathcal{H}_{S_i}^d$.* $\square$

Let us also recall the fact that the size of the dual of a sub-hypergraph of $\mathcal{H}$ is limited by the size of $\mathcal{H}^d$:

Proposition 2 ([28]) *For any hypergraph $\mathcal{H} \subseteq 2^V$ and any subset $S \subseteq V$, we have $|\mathcal{H}_S^d| \leq |\mathcal{H}^d|$.* $\square$

Let us also note that the conjunction of given hypergraphs can be computed efficiently in parallel. (We shall assume in what follows that we run our procedures on a CREW-PRAM.)

Proposition 3 *Given r hypergraphs $\mathcal{H}_i$, $i = 1, ..., r$ on n points such that $|\mathcal{H}_i| \leq m$ for $i = 1, ..., r$, their conjunction $\bigwedge_{i=1}^r \mathcal{H}_i$ can be computed in $O(r \log(rnm))$ time on $O(nm^{2r})$ processors.* $\square$

Putting together the above, we get the following parallel procedure for hypergraph dualization:

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

1. If $|\mathcal{H}| = 0$, then return $\emptyset$. If $\mathcal{H} = \{H\}$, then return $\{\{i\} \ : \ i \in H\}$.
2. Find a complete family of subsets $S_1, \ldots, S_q \subseteq V$ for $\mathcal{H}$.
3. Compute $\mathcal{H}_{S_i}^d \leftarrow$DUALIZE($\mathcal{H}_{S_i}, S_i$) for $i = 1, \ldots, q$.
4. Return the conjunction $\mathcal{H}^d \leftarrow \bigwedge_{i=1}^q \mathcal{H}_{S_i}$.

Proposition 1 implies that the above procedure outputs all minimal transversals of $\mathcal{H}$ correctly and in a finite number of iterations, assuming that we choose the subsets S_i in Step 2 such that $\mathcal{H}_{S_i} \subsetneq \mathcal{H}$. Of course, for the above to be efficient in parallel, we need several additional properties. Most notably, we need to be able to find a complete family S_i, $i = 1, ..., r$ for induced sub-hypergraphs $\mathcal{H}_S$ of $\mathcal{H}$ (due to the recursive nature of the above procedure), such that r is small, and such that each of the r subproblems at each level of the recursion are substantially smaller than the problem we decomposed.

For this end, let us consider a positive integer-valued, monotone set-function $f : V \mapsto \mathbb{Z}_+$ (up to Section 5, the reader may assume that $f(S) = |S|$), a positive constant $\epsilon < 1$, and let us call a family $S_1, ..., S_q \subseteq V$ of subsets (f, ϵ)-*complete* for a hypergraph $\mathcal{H} \subseteq 2^V$ if the following conditions hold:

(H1) $S_1, \ldots, S_r$ are complete for $\mathcal{H}$, and

(H2) For each index $i \in \{1, \ldots, q\}$ with $|\mathcal{H}_{S_i}| > 1$ we have $f(S_i) \leq (1-\epsilon)f(V)$.

Given $r \in \mathbb{Z}_+$ and $0 < \epsilon < 1$, let us denote by $\mathbb{H}_f(r, \epsilon)$ the family of hypergraphs $\mathcal{H} \subseteq 2^V$, such that for every subset $S \subseteq V$, there exist subsets $S_1, \ldots, S_q \subseteq S$, $q \leq r$ forming an (f, ϵ)-complete family for the induced sub-hypergraph $\mathcal{H}_S$.

Let us further modify procedure DUALIZE($\mathcal{H}, V$) by requiring in Step 2 an (f, ϵ)-complete family with $q \leq r$. The following proposition gives the running time and number of processors required by the modified DUALIZE($\mathcal{H}, V$), 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 in Step 2 the sets $S_1, \ldots, S_q \subseteq V$ satisfying properties (H1) and (H2), and to construct the induced sub-hypergraphs $\mathcal{H}_{S_i}$, $i = 1, ..., q$.

Proposition 4 *Let V be a set of n elements, $f \in \mathbb{Z}_+^V$ a monotone positive set-function, $\alpha = f(V)$, $r \in \mathbb{Z}_+$, $0 < \epsilon < 1$, and let $\mathcal{H} \in \mathbb{H}_f(r, \epsilon)$ be a hypergraph. Let further $t(\alpha, n, m)$ and $p(\alpha, n, m)$ denote respectively the time and the number of processors, required by the modified procedure DUALIZE($\mathcal{H}, V$) to output all $m \stackrel{\text{def}}{=} |\mathcal{H}^d|$ minimal transversals of the hypergraph $\mathcal{H}$. Then $t(\alpha, n, m) = O((\tau + r\log(rnm))^{\frac{\log \alpha}{\epsilon}})$ and $p(\alpha, n, m) = O((\pi + nm^{2r})r\alpha^{\frac{\log r}{\epsilon}})$.*

Proof. Step 2 requires time at most τ by definition. The computations in Step 3 can be done in parallel, requiring thus at most $t((1-\epsilon)\alpha, n, m)$ time. By Proposition 2, $|\mathcal{H}_{S_i}^d| \leq m$ for all $i = 1, \ldots, r$ and thus Step 4 can be done in $O(r \log m \log n)$ time using $O(m^{2r}n)$ processors by Proposition 3. This gives the recurrence

$$t(\alpha, n, m) = O(\tau + r\log(rnm)) + t((1-\epsilon)\alpha, n, m). \tag{3}$$

Note that $f(S) \leq 1$ implies, by (H2), that $|\mathcal{H}_{S_i}| \leq 1$ for $i = 1, \ldots, r$, since $f(X) \geq 1$ for all $X \subseteq V$ is assumed. For this base case, Step 1 requires only constant time to finish the computation. Thus (3) implies the bound on the running time. As for the number of processors, we get the recurrence

$$p(\alpha, n, m) = O(\pi + nm^{2r}) + r \cdot p((1-\epsilon)\alpha, n, m), \tag{4}$$

implying the bound stated in the proposition. $\square$

Note that, in the above procedure, all minimal transversals are generated simultaneously, at the very end, and there is no need to generate any of them individually, in the course of the computations. Thus, if the requirement is to generate only k minimal transversals of $\mathcal{H}$ (where possibly $k < |\mathcal{H}^d|$), then the above procedure may not be efficient. However, it can be modified for this purpose, 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 $\emptyset$. If $\mathcal{H} = \{H\}$, then return $\{\{i\} \ : \ i \in H\}$.
2. Find an (f, ϵ)-complete family of subsets $S_1, \ldots, S_q \subseteq V$ with $q \leq r$.
3. Compute parallel $\mathcal{G}_i \leftarrow$DUALIZE($\mathcal{H}_{S_i}, S_i, k$) for $i = 1, \ldots, q$.
4. If there is an index $i \in \{1, \ldots, q\}$ such that $|\mathcal{G}_i| = k$ then
5. Compute in parallel for each $Y \in \mathcal{G}_i$ a minimal transversal T_Y
6. of the sub-hypergraph $\mathcal{H}(Y) = \{H \setminus S_i \ : \ H \cap Y = \emptyset\} \subseteq 2^{V \setminus S_i}$,
7. and return $\mathcal{G} \leftarrow \{Y \cup T_Y \ : \ Y \in \mathcal{G}_i\}$.
8. Otherwise, return the conjunction $\mathcal{G} \leftarrow \bigwedge_{i=1}^{q} \mathcal{H}_{S_i}^d$.

Proposition 5 *Let V, f, r and ϵ be as above, $\alpha = f(V)$, $\mathcal{H} \in \mathbb{H}_f(r, \epsilon)$ and let k be a given integer. Then procedure DUALIZE($\mathcal{H}, V, k$) outputs k (or all if $|\mathcal{H}^d| \leq k$) minimal transversals of the hypergraph $\mathcal{H}$, in time $t(\alpha, n, k) = O((\tau + r\log(rnk))\log\alpha/\epsilon + T)$, using $p(\alpha, n, k) = O((\pi + nk^{2r})\alpha^{1+\log r/\epsilon} + k\Pi)$ processors, where T and Π are respectively the parallel time and the number of processors required to generate a single minimal transversal of a partial sub-hypergraph of $\mathcal{H}$.*

Proof. By Step 4, if there is a set S_i, for some $i \in \{1, \ldots, q \leq r\}$, such that $|\mathcal{H}_{S_i}^d| \geq k$, then the time and number of processors needed to finish the computation are T and $k\Pi$, respectively. On the other hand if, for all $i \in \{1, \ldots, q\}$, we have $|\mathcal{H}_{S_i}^d| < k$, then recurrences (3) and (4) hold with $m < k$, and we obtain the stated bounds, analogously as in the proof of Proposition 4. ☐

3 NC-dualization of uniformly sparse hypergraphs

Instead of proving Theorem 2 directly, we shall prove a more general statement in this section. For this, let $f : V \mapsto \mathbb{Z}_+$ be an integer-valued monotone positive *sub-additive* function (i.e. one for which $f(X \cup Y) \leq f(X) + f(Y)$ holds for all subsets $X, Y \subseteq V$). Assume further that $f(X)$ can be evaluated efficiently in parallel (say in $O(\log n)$ time on $O(n)$ processors), for any given subset $X \subseteq V$. Let further $\mathcal{H} \subseteq 2^V$ be a hypergraph such that

$$\frac{1}{f(S)} \sum_{H \in \mathcal{H}_S} f(H) \leq \delta \tag{5}$$

holds for every subset $S \subseteq V$ (for $f(X) = |X|$ this condition is equivalent to the uniform δ-sparseness of $\mathcal{H}$). Then, we can apply the results of the preceding section to obtain an NC-reduction of problem DUAL($\mathcal{H}, k$) to the computation of a single minimal transversal in a partial hypergraph of $\mathcal{H}$ also satisfying (5).

Theorem 4 *Given a hypergraph $\mathcal{H} \subseteq 2^V$ satisfying (5) for all $S \subseteq V$, and a positive constant $\epsilon < 1$, let $r = 1+2\delta/(1-\epsilon)$, $\alpha = f(V)$, and denote by T and Π, respectively, the parallel time and the number of processors required to generate a single minimal transversal of a partial sub-hypergraph of $\mathcal{H}$. Then, problem $DUAL(\mathcal{H}, k)$ can be solved in $O(r \log(n|\mathcal{H}|k)^2 \log \alpha/\epsilon + T)$ parallel time, using $O(n^2|\mathcal{H}|\alpha^{1+\log r/\epsilon}k^{2r} + k\Pi)$ processors.*

Proof. We shall derive this statement from Proposition 5, by showing that (5) and the sub-additivity of f implies that $\mathcal{H} \in \mathbb{H}_f(r, \epsilon)$, and that we can construct an (f, ϵ)-complete family of subsets $S_1, \ldots, S_q$ efficiently in parallel, for some $q \leq r$.

To this end, we consider an arbitrary induced sub-hypergraph $\mathcal{H}_S$ of $\mathcal{H}$, $S \subseteq V$, and first separate the "large" hyperedges of $\mathcal{H}_S$, and then divide the rest into at most r groups such that the union in each group is "small", where the threshold separating "large" from "small" is $(1-\epsilon)f(S)$. We shall see that the whole computation is essentially a parallel prefix computation, and can be done efficiently, as claimed.

More precisely, let $\mathcal{H}_0 = \{H \in \mathcal{H}_S \ : \ f(H) > (1-\epsilon)f(S)\} = \{H_1, \ldots, H_p\}$, and let us label the rest of the sets by $\mathcal{H}_S \setminus \mathcal{H}_0 = \{H_{p+1}, \ldots, H_q\}$, in an arbitrary order. Let us then choose sequentially for $j = p+1, ...$ a maximal index $i(j) \in \{p+1, ..., q\}$ such that

$$\sum_{i=i(j-1)+1}^{i(j)} f(H_i) \leq (1-\epsilon)f(S), \tag{6}$$

where $i(p) = p$ by definition. Clearly, each of these indices can be computed by a parallel prefix computation in $O(\log n \log |\mathcal{H}|)$ time on $O(n|\mathcal{H}|)$ processors. Since we separated the "large" sets we have $i(j) > i(j-1)$, and thus we must have $i(s) = q$ for some finite $s \geq p+1$. We shall show that $s \leq r$ follows from our assumptions. For this let us observe that by the maximality of the chosen indices, we have

$$\sum_{i=i(j-1)+1}^{i(j)+1} f(H_i) > (1-\epsilon)f(S) \tag{7}$$

for $j = p+1, ..., s-1$. Summing up these inequalities we get

$$\begin{aligned}(s-p-2)(1-\epsilon)f(S) &< \sum_{j=p+1}^{r} \sum_{i=i(j-1)+1}^{i(j)+1} f(H_i) \\ &\leq 2\sum_{j=p+1}^{q} f(H_i) \leq 2(\delta f(S) - \sum_{j=1}^{p} f(H_i) \\ &< 2(\delta - (1-\epsilon)p)f(S),\end{aligned}$$

where the second inequality follows from the fact that each hyperedge H_i appears in the sums (7) for at most two values of j; the third inequality follows from (5); and the last inequality follows from the fact that $|H_i| > (1-\epsilon)f(S)$ for $i = 1, \ldots, p$. From the above inequality $s \leq 1 + 2\delta/(1-\epsilon)$ follows, since $f(S) > 0$.

Let us finally define $S_i = H_i$ for $i = 1, \ldots, p$, and $S_j = \bigcup_{i=i(j-1)+1}^{i(j)} H_i$ for $j = p+1, \ldots, s$. Clearly, these sets $S_1, \ldots, S_s$ satisfy (H1) by definition. They also satisfy (H2) since $f(S_j) \leq \sum_{i=i(j-1)+1}^{i(j)} f(H_i) \leq (1-\epsilon)f(S)$ by (6) and by the sub-additivity of f. □

Proof of Theorem 2. Since $|\mathcal{H}| \leq \delta n$ for any δ-sparse hypergraph on n vertices, Theorem 2 follows from Theorem 4 for $f(X) = |X|$.

4 Finding one minimal transversal in parallel

In this section we prove Theorem 3 and give an efficient parallel algorithm to solve problem DUAL$(\mathcal{H}, 1)$ for a uniformy δ-sparse hypergraph $\mathcal{H} \subseteq 2^V$. Our algorithm is an NC reduction to computing a minimal transversal in a hypergraph of bounded degree, see Section 4.3. It follows from the results of [18] that the latter computation can be carried out in NC time. Alternatively, in Sections 4.1, we consider a faster and more direct approach. Specifically, we argue that an algorithm, conjectured by Beame and Luby [3] to solve problem $DUAL(\mathcal{H}, 1)$ efficiently in parallel for *any* input hypergraph, does solve this problem in RNC time for hypergraphs of bounded degree (and also for a related more general class of hypergraphs described in Remark 1 in Section 5). To finish the proof of Theorem 3, we then discuss in Section 4.2 a standard derandomization of the algorithm presented in Section 1.

4.1 Solving DUAL$(\mathcal{H}, 1)$ for hypergraphs of bounded degree

Consider the following problem of reducing a given transversal for $\mathcal{H}$ to a minimal one:
$TRANS(\mathcal{H}, V, T)$: *Given $\mathcal{H} \subseteq 2^V$, and a transversal $T \subseteq V$ of $\mathcal{H}$, find a minimal transversal $X \subseteq T$ of $\mathcal{H}$.*

Clearly, problem DUAL$(\mathcal{H}, 1)$ is identical to $TRANS(\mathcal{H}, V, V)$. For convenience, in this subsection we do not assume $\mathcal{H}$ to necessarily be Sperner. To solve problem TRANS$(\mathcal{H}, V, T)$, we can use the following randomized method:

Procedure MINT$(\mathcal{H}, V, T)$:

1. Repeat until $\forall v \in T\ \exists H \in \mathcal{H}$ such that $H \cap T = \{v\}$:
2. Let $\pi \in \mathbb{S}_n$ be a random permutation of the vertices of V.
3. $\forall\ H \in \mathcal{H}$, in parallel, let $\mu(H) \leftarrow \min\{\pi(v) \ : \ v \in H \cap T\}$.
4. $T \leftarrow \{\mu(H) \ : \ H \in \mathcal{H}\}$.

Lemma 1 *For any hypergraph $\mathcal{H} \subseteq 2^V$ satisfying (1), procedure MINT$(\mathcal{H}, V, T)$ finds a minimal transversal of $\mathcal{H}$ in $O(\delta \log^2 n)$ expected parallel time using $O(n^2)$ processors.*

Proof.

It is clear that for any input transversal T, the procedure computes correctly a minimal sub-transversal of T. We claim further that the expected number of iterations is $O(\delta \log n)$. Indeed, let T_k be the sub-transversal of the input transversal computed after k iterations, and let $T_k^* = \{v \in T_k : H \cap T_k = \{v\} \text{ for some } H \in \mathcal{H}\}$. Then for any $u \in T_k \setminus T_k^*$, if the hyperedges of $\mathcal{H}$ containing u are $H_1, \ldots, H_{\deg(u)}$, then $|H_i \cap T_k| \geq 2$ for all $i = 1, \ldots, deg(u)$. Let us imagine that we pick arbitrarily a vertex $u_i \neq u$ in each set $H_i \cap T_k$, and let $X(u) = \{u_i : i = 1, \ldots, deg(u)\}$. Clearly, if $\pi(u_i) < \pi(u)$ for all $i = 1, \ldots, \deg(u)$, then u will surely be deleted from T_k at iteration k. Thus, the expected number of vertices deleted from T_k can be bound from below by

$$\mathbb{E}[T_k \setminus T_{k+1}] \geq \sum_{u \in T_k \setminus T_k^*} \frac{1}{|X(u)|} \geq \frac{|T_k \setminus T_k^*|}{\deg(u)} \geq \frac{|T_k| - |T_k^*|}{\delta}. \tag{8}$$

Since $T_k^* \subset T_{k+1}^*$ for all k, and the algorithm halts as soon as $T_k = T_k^*$, the expected number of iterations is $O(\delta \log n)$. Each iteration of the above procedure can be implemented to run in $O(\log(\delta n))$ time on $n|\mathcal{H}|/\log n \leq \delta n^2/\log n = O(n^2)$ processors, which implies the lemma. $\square$

4.2 Derandomization

The algorithm of Section 4.1 can de derandomized by using in Step 2 of procedure $\mathrm{MINT}(\mathcal{H}, V, T)$ *approximately min-wise independent permutations*, i.e. a set $\mathcal{F} \subseteq \mathbb{S}_n$ of permutations such that for each subset X of vertices, $\min(\pi(X))$ for $\pi \in \mathcal{F}$ is distributed nearly uniformly on X (see e.g., [10]). By deterministically selecting a permutation from such a set $\mathcal{F}$ which results in the best possible reduction in the size of T_k, we can still guarantee that the number of iteration stays bounded by $O(\delta \log n)$. It is known that there exists an NC-constructible explicit set $\mathcal{F}$ of approximately min-wise independent permutations of size $|\mathcal{F}| = n^{O(1)}$ (see e.g., [10, 23]).

Lemma 2 *For any hypergraph $\mathcal{H} \subseteq 2^V$ satisfying (1), a minimal transversal of $\mathcal{H}$ can be found in $O(\delta \log^{O(1)}(\delta n))$ expected parallel time using $n^{O(1)}$ processors.*

4.3 Solving DUAL$(\mathcal{H}, 1)$ for uniformly sparse hypergraphs

The following procedure recursively reduces problem $\mathrm{DUAL}(\mathcal{H}, 1)$ for a non-empty uniformy δ-sparse hypergraph $\mathcal{H}$ to $\log n$ similar subproblems for hypergraphs of degree at most 2δ:

PROCEDURE DUAL($\mathcal{H}, 1$):

1. $S \leftarrow \{v \in V \;:\; \deg_{\mathcal{H}}(v) > 2\delta\}$.
2. If $S = \emptyset$, use procedure TRANS($\mathcal{H}, V, V$) to compute a minimal transversal of $\mathcal{H}$.
3. Recursively compute a minimal transversal X of $\mathcal{H}_S$.
4. $\mathcal{H}(X) \leftarrow \{H \cap (V \setminus S) \mid H \cap X = \emptyset\}$.
5. $Y \leftarrow TRANS(\mathcal{H}(X), V \setminus S, V \setminus S)$.
6. Return $X \cup Y$.

Note that since uniform δ-sparseness is a hereditary property, the hypergraph H_S in step 3 is uniformy δ-sparse, while the hypergraph $\mathcal{H}$ is step 2 and the hypergraph $\mathcal{H}(X)$ is step 4 have degree at most 2δ. Note also that if $\mathcal{H}$ is Sperner, then so is $\mathcal{H}_S$, and that by (2), the number of vertices of $\mathcal{H}_S$ is at most $n/2$. Therefore the number of recursive calls in step 3 is bounded by $\log n$, which implies Theorem 3.

5 Concluding remarks

It can be shown that Lemma 1 holds for the class of hypergraphs $\mathcal{H} \subseteq 2^V$ satisfying the condition that for every subset S of V,

$$\frac{1}{|S|} \sum_{v \in S} \deg_{\mathcal{H}^S}(v) \leq \delta, \tag{9}$$

where $H^S \stackrel{\text{def}}{=} \min\{H \cap S \;:\; H \in \mathcal{H},\; H \cap S \neq \emptyset\}$. Clearly, any hypergraph $\mathcal{H}$ with maximum degree δ satisfies (9).

Furthermore, as mentioned in Section 3, the reduction of Theorem 2 is applicable to a wider class of hypergraphs than those satisfying (2).

Let $w : V \mapsto \mathbb{Z}_+$ be non-negative weights assigned to the vertices of a hypergraph $\mathcal{H} \subseteq 2^V$. The linear function $w(X) = \sum_{v \in X} w(v)$ is clearly sub-additive. It follows then from Theorem 4 that we can find all minimal transversals of hypergraphs $\mathcal{H}$, satisfying

$$\sum_{H \in \mathcal{H}_S} w(H) \leq \delta w(S), \tag{10}$$

for every subset $S \subseteq V$ and for some constant δ, in time polylog$(w(V), |\mathcal{H}|)$ using poly$(w(V), |\mathcal{H}|, |\mathcal{H}^d|)$ number of processors. Furthermore, it is easy to see that the algorithm of Section 4 can be extended to compute a minimal transversal of any hypergraph satisfying (10) in time

$O(\log n \log w(V))$ using $poly(n)$ processors. It is interesting to note that the family of hypergraphs satisfying (10) is exactly the family of δ-degenerate hypergraphs, introduced in [16], and can equivalently be defined as those hypergraphs for which

$$\text{min-deg}(\mathcal{H}_S) \stackrel{\text{def}}{=} \min_{v \in V} \deg_{\mathcal{H}_S}(v) \leq \delta \tag{11}$$

holds for a given constant δ for all subsets $S \subseteq V$. Unfortunately, in the above construction one needs exponentially large weights to represent certain δ-degenerate hypergraphs. In fact the parallel complexity of DUAL$(\mathcal{H}, k)$ is an open problem for the class of δ-degenerate hypergraphs, even in case of $k = 1$.

Other examples, when Theorem 4 can be applied include hypergraphs $\mathcal{H}$ satisfying

$$\sum_{H \in \mathcal{H}_S} f_{\mathcal{H}_S}(H) \leq \delta |\mathcal{H}_S|, \tag{12}$$

for every $S \subseteq V$, where δ is a fixed constant, and where $f_{\mathcal{H}}(X) = |\{H \in \mathcal{H} \;:\; H \cap X \neq \emptyset\}|$, for $X \subseteq V$; or hypergraphs satisfying

$$\sum_{H, H' \in \mathcal{H}_S} |H \cap H'| \leq \delta \sum_{H \in \mathcal{H}_S} |H|, \tag{13}$$

for every $S \subseteq V$, where δ is again a constant and $f_{\mathcal{H}}(X) = \sum_{H \in \mathcal{H}} |H \cap X|$, for $X \subseteq V$.

Acknowledgement: The authors thank P. Kelsen for pointing out that the method applied in [18] can also be used to show that DUAL$(\mathcal{H}, 1) \in NC$ for bounded degree hypergraphs.

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 and L. Khachiyan, Generating Maximal Independent Sets for Hypergraphs with Bounded Edge-Intersections, in *the Proceedings of the 6th Latin American Theoretical Informatics Conference (LATIN 2004)* (Martin Farach-Colton, ed.), *LNCS* 2976 , pp. 488–498.

[7] E. Boros, K. Elbassioni, V. Gurvich and L. Khachiyan, A Global Parallel Algorithm for Finding All Minimal Transversals of Hypergraphs of Bounded Edge-size, DIMACS Technical Report 2004-31, Rutgers University.

[8] 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.

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

[10] A. Broder, M. Charikar and M. Mitzenmacher, A derandomization using min-wise independent permutations, *RANDOM'98* (M. Luby, J. Rolim, M. Serna, eds.) (1998) pp. 15-24.

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

[12] E. Dahlhaus, 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.

[13] 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).

[14] C. Domingo, N. Mishra and L. Pitt, Efficient read-restricted monotone CNF/DNF dualization by learning with membership queries, *Machine learning* 37 (1999) 89-110.

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

[16] T. Eiter, G. Gottlob and K. Makino, New results on monotone dualization and generating hypergraph transversals, in *the 34th Annual ACM Symposium on Theory of Computing (STOC)*, 2002.

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

[18] O. Garrido, P. Kelsen, 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.

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

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

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

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

[23] Piotr Indyk: A Small Approximately Min-Wise Independent Family of Hash Functions. *J. Algorithms* **38**(1) (2001) pp. 84-90.

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

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

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

[27] 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).

[28] 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.

[29] T. Luczak , 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.

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

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

[32] 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.