

A New Algorithm for the Hypergraph Transversal Problem*

L. Khachiyan¹, E. Boros², K. Elbassioni³, and V. Gurvich²

¹ Department of Computer Science, Rutgers University, 110 Frelinghuysen Road,
Piscataway NJ 08854-8003; leonid@cs.rutgers.edu**

² RUTCOR, Rutgers University, 640 Bartholomew Road, Piscataway NJ 08854-8003;
{boros,gurvich}@rutcor.rutgers.edu

³ Max-Planck-Institut für Informatik, Saarbrücken, Germany;
elbassio@mpi-sb.mpg.de

Abstract. We consider the problem of finding all minimal transversals of a hypergraph $\mathcal{H} \subseteq 2^V$, given by an explicit list of its hyperedges. We give a new decomposition technique for solving the problem with the following advantages: (i) Global parallelism: for certain classes of hypergraphs, e.g. hypergraphs of bounded edge size, and any given integer k , the algorithm outputs k minimal transversals of \mathcal{H} in time bounded by $\text{polylog}(|V|, |\mathcal{H}|, k)$ assuming $\text{poly}(|V|, |\mathcal{H}|, k)$ number of processors. Except for the case of graphs, none of the previously known algorithms for solving the same problem exhibit this feature. (ii) Using this technique, we also obtain new results on the complexity of generating minimal transversals for new classes of hypergraphs, namely hypergraphs of bounded dual-conformality, and hypergraphs in which every edge intersects every minimal transversal in a bounded number of vertices.

1 Introduction

Let $\mathcal{H} \subseteq 2^V$ be a hypergraph 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} . Let $\mathcal{H}^d = \{\text{minimal } X \subseteq V \mid X \text{ is a transversal to } \mathcal{H}\} \subseteq 2^V$ denote the set of all (inclusion-wise) minimal transversals of \mathcal{H} . We denote further by $\mathcal{H}^c \stackrel{\text{def}}{=} \{V \setminus H \mid H \in \mathcal{H}\}$ the *complementary* hypergraph 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} . Let $\mathcal{I}(\mathcal{H}) \subseteq 2^V$ denote the family of all maximal independent sets (MIS) of \mathcal{H} . Obviously, $\mathcal{I}(\mathcal{H}) = \mathcal{H}^{dc}$. We assume that a Sperner hypergraph

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

** Our friend and co-author, Leonid Khachiyan passed away with tragic suddenness, while we were working on the final version of this paper.

\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 (or equivalently, MIS's) of \mathcal{H} .

This hypergraph transversal problem has received considerable attention in recent years [5, 7, 11–15, 23, 24] due to its generality and wide applicability in a variety of fields including artificial intelligence, reliability theory, database theory, integer programming, and learning theory (see, e.g. [6, 8, 13]). It is still open whether problem $DUAL(\mathcal{H}, k)$ can be solved in polynomial time for arbitrary hypergraphs. The fastest currently known algorithm [15] is quasi-polynomial and works by considering the following incremental generation problem:

$NEXT(\mathcal{H}, \mathcal{X})$: Given a Sperner hypergraph \mathcal{H} and a subset $\mathcal{X} \subseteq \mathcal{H}^d$, 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 [15] is $O(nN) + N^{o(\log N)}$, where $n = |V|$ and $N = |\mathcal{H}| + |\mathcal{X}|$. For several classes of hypergraphs, polynomial time algorithms exist, e.g. hypergraphs of bounded dimension (edge-size) [4, 7, 13], of bounded-degree [11, 14], of bounded-edge intersections [5], of bounded treewidth [14], and read-once (exact) hypergraphs [12].

In this paper, we consider a new decomposition method, for solving the hypergraph transversal problem, with the following advantages:

(i) *Global parallelism*: For hypergraphs of bounded dimension:

$$\dim(\mathcal{H}) \stackrel{\text{def}}{=} \max_{H \in \mathcal{H}} |H| < \delta, \quad (1)$$

for some constant $\delta \geq 2$, it was shown in [4], that problem $NEXT(\mathcal{H}, \mathcal{X})$ can be efficiently solved in parallel: $NEXT(\mathcal{H}, \mathcal{X}) \in NC$ for $\delta \leq 4$ and $NEXT(\mathcal{H}, \mathcal{X}) \in RNC$ for $\delta = 5, 6, \dots$. Note, however, that this result 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. Except for hypergraphs of dimension 2, that is for graphs [9], no *global parallel algorithms*, i.e. those that solve problem $DUAL(\mathcal{H}, k)$ in time $\text{polylog}(n, |\mathcal{H}|, k)$ on $\text{poly}(n, |\mathcal{H}|, k)$ processors, are known. As we shall see, our method allows for such global parallelism for any bounded-edge, as well as some other related classes of hypergraphs.

(ii) *Stronger results for new classes of hypergraphs*: Using the decomposition technique described in this paper, we also obtain stronger bounds on the (sequential) complexity of dualization for some classes of hypergraphs for which the only previously known result was the quasi-polynomial bound of [15]:

1. *Hypergraphs with bounded dual-conformality*: These are hypergraphs \mathcal{H} for which the dual of $\mathcal{I}(\mathcal{H})$ has bounded-edge size (see e.g. [3]):

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

for some constant $\delta \geq 2$. For instance, hypergraphs in which every two minimal transversals intersect in a bounded number of vertices, belong to this class. We show that it is possible to find all minimal transversals for such hypergraphs in polylog time using a polynomial number of processors.

2. *Hypergraphs in which every edge intersects every minimal transversal in a bounded number of vertices:*

$$|H \cap T| < \delta \text{ for every } H \in \mathcal{H} \text{ and every } T \in \mathcal{H}^d, \quad (3)$$

for some constant $\delta \geq 2$. These generalize read-once hypergraphs, in which every edge meets every minimal transversal in exactly one vertex (i.e. $\delta = 2$). Read-once hypergraphs were considered in [12], where a polynomial-delay algorithm was given. We show that, for any constant δ , we can find k minimal transversals of a hypergraph $\mathcal{H} \subseteq 2^V$ in this class in $\text{polylog}(n, |\mathcal{H}|, k) + \Delta \log |\mathcal{H}|$ time using a quasi-polynomial number of processors $\text{poly}(n, k, \Pi) \cdot |\mathcal{H}|^{O(\log |\mathcal{H}|)}$, where Δ and Π are respectively the parallel time and the number of processors required to generate a single minimal transversal of \mathcal{H} .

We describe the general decomposition method used for obtaining the above results in the next section. Direct application of this method yields our first result for hypergraphs of bounded edge-size, which will be presented in Section 3. Following that, we consider in Section 4 the class of hypergraphs of bounded dual-conformality and show how they can be dualized efficiently in parallel. Finally, a more involved application of this decomposition method gives us the claimed results about hypergraphs satisfying (3), and is presented in Section 5.

2 A Global parallel dualization algorithm

Let V be a finite set. 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 $S_1, \dots, S_r \subseteq S$ satisfying:

- (H1) For every $H \in \mathcal{H}_S$, there exists an $i \in [r] = \{1, \dots, r\}$ such that $S_i \supseteq H$.
(H2) $|S_i| \leq (1 - \epsilon)|S|$, for each $i \in [r]$ for which $|\mathcal{H}_{S_i}| > 1$.

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

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

Proposition 1. *Let $\mathcal{H} \subseteq 2^V$ be a hypergraph and $S_1, \dots, S_r \subseteq V$ be subsets such that for every $H \in \mathcal{H}$ there is an $i \in [r]$ such that $H \subseteq S_i$. Then $\mathcal{H}^d = \bigwedge_{i=1}^r \mathcal{H}_{S_i}^d$.*

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 return $\{\{i\} : i \in H\}$.
3. In parallel, do the following:
 4. find sets $S_1, \dots, S_r \subseteq V$ satisfying (H1) and (H2) with $S = V$.
 5. Let $\mathcal{G}_i \leftarrow \text{DUALIZE}(\mathcal{H}_{S_i}, S_i)$, for $i = 1, \dots, r$.
 6. Compute the conjunction $\mathcal{G} \leftarrow \bigwedge_{i=1}^r \mathcal{G}_i$.
7. Return \mathcal{G} .

Proposition 2 ([21]). *For any hypergraph $\mathcal{H} \subseteq 2^V$ and any $S \subseteq V$, we have $|\mathcal{H}_S^d| \leq |\mathcal{H}^d|$.*

Proposition 1 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 $S_1, \dots, S_r \subseteq V$, satisfying (H1) and (H2). We shall assume in what follows that we run our procedures on a CREW-PRAM.

Proposition 3. *Let $t(n, m, M)$ and $p(n, m, M)$ be respectively the time and the number of processors, required by procedure $DUALIZE(\mathcal{H}, V)$ to output all minimal transversals of a hypergraph $\mathcal{H} \in \mathbb{H}(r, \epsilon)$ of size $m = |\mathcal{H}|$ on n vertices and with $|\mathcal{H}^d| = M$. Then $t(n, m, M) = O((\tau + \log(n(m + M^r)))\frac{\log n}{\epsilon})$ and $p(n, m, M) = O((\pi + (m + M^{2r})n)n^{\frac{\log r}{\epsilon}})$.*

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 [18]. For some special classes of hypergraphs, e.g. hypergraphs of bounded dimension [1, 2, 10, 20], hypergraphs of bounded vertex-degrees [16], and linear hypergraphs [22], 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 return $\{\{i\} : i \in H\}$.
3. In parallel, do the following:
 4. find sets $S_1, \dots, S_r \subseteq V$ satisfying (H1) and (H2) with $S = V$.
 5. Let $\mathcal{G}_i \leftarrow DUALIZE(\mathcal{H}_{S_i}, S_i, k)$, for $i = 1, \dots, r$.
 6. If there is an $i \in \{1, \dots, r\}$ such that $|\mathcal{G}_i| = k$ then
 7. In parallel, for each $Y \in \mathcal{G}_i$, do the following:
 8. Let $\mathcal{H}[Y] = \{H \setminus S_i \mid H \cap Y = \emptyset\}$.
 9. Compute a minimal transversal T_Y of $\mathcal{H}[Y]$.
 10. Return $\mathcal{G} \leftarrow \{Y \cup T_Y : Y \in \mathcal{G}_i\}$, and stop.
 11. else
 12. Compute the conjunction $\mathcal{G} \leftarrow \bigwedge_{i=1}^r \mathcal{H}_{S_i}^d$.
 13. Return $\min\{k, |\mathcal{G}|\}$ elements of \mathcal{G} .

Notation. In the rest of the paper, we use n, m respectively to denote the number of vertices $|V|$ and number of edges $|\mathcal{H}|$ of the input hypergraph $\mathcal{H} \subseteq 2^V$.

We denote respectively further by Δ and Π the parallel time and the number of processors required to generate a single minimal transversal for a hypergraph \mathcal{H} belonging to the class under consideration.

Proposition 4. *DUALIZE(\mathcal{H}, V, k) outputs $\min\{k, |\mathcal{H}^d|\}$ minimal transversals of a hypergraph $\mathcal{H} \in \mathbb{H}(r, \epsilon)$ in time $t(n, m, k) = O((\tau + \log(n(m + k^r))) \log n / \epsilon + \Delta)$, using $p(n, m, k) = O((\pi + (m + k^{2r})n)^{\log r / \epsilon} + k\Pi)$ processors.*

Remark. More generally, we may consider a positive integer-valued, monotone set-function $f : V \mapsto \mathbb{Z}_+$, and replace condition (H2) above by

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

Then one can show similarly that the depth of the recursion tree required by procedure DUALIZE(\mathcal{H}, V, k) is $O(\log f(V)/\epsilon)$. In Sections 3 and 4, we use $f(S) = |S|$. In Section 5, we use $f(S) = |\mathcal{H}_S|$.

3 Hypergraphs of bounded dimension

Let $\mathcal{H} \subseteq 2^V$ be a hypergraph on V , satisfying (1). As mentioned in the introduction, an efficient parallel algorithm exists [4] for solving problem NEXT(\mathcal{H}, \mathcal{X}) for such hypergraphs. This is based on an NC-reduction of problem NEXT(\mathcal{H}, \mathcal{X}) to problem DUAL($\mathcal{H}', 1$) of finding a single minimal transversal in a partial sub-hypergraph \mathcal{H}' of \mathcal{H} , for which NC algorithms exist, if $\dim(\mathcal{H}) \leq 3$ ([1, 10]), and an RNC algorithm exists if $\dim(\mathcal{H}) = 4, 5, \dots$ ([2, 20]). In particular, these reductions do not yield a global parallel algorithm. The case of graphs, $\dim(\mathcal{H}) \leq 2$, was considered in [9], where it was shown that problem DUAL(\mathcal{H}, k) 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.

Theorem 1. *There is a deterministic parallel algorithm that, for any hypergraph $\mathcal{H} \subseteq 2^V$ such that $\dim(\mathcal{H}) < \delta$, solves problem DUAL($\mathcal{H}, |\mathcal{H}^d|$) in time $O(\delta^2 \log n \log(n|\mathcal{H}^d|))$ using $O(n^{\delta \log \delta + 1}(m + |\mathcal{H}^d|^{2\delta}))$ processors.*

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

Theorem 2. *There is a deterministic parallel algorithm that, for any $\mathcal{H} \subseteq 2^V$ such that $\dim(\mathcal{H}) < \delta$, and any integer $k \in \mathbb{Z}_+$, solves problem DUAL(\mathcal{H}, k) in time $O(\delta^2 \log n \log(nk) + \Delta)$ using $O(n^{\delta \log \delta + 1}(m + k^{2\delta}) + k\Pi)$ processors.*

Theorem 2 extends the results of [2, 20] 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 2 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.

Proofs of Theorems 1 and 2. We apply the results of the preceding section to solve problem $\text{DUAL}(\mathcal{H}, k)$, by setting $r = \delta$ and $\epsilon = 1/\delta$. Then $\mathcal{H} \in \mathbb{H}(r, \epsilon)$. Indeed, given any set $S \subseteq V$, we partition S into r (almost) equal parts W_1, \dots, W_r , and let $S_i = S \setminus W_i$, for $i = 1, \dots, r$. The fact that any hyperedge $H \in \mathcal{H}_S$ has size at most $\delta - 1$ implies that H cannot intersect all sets W_1, \dots, W_r , i.e. H must be contained in at least one of the sets S_1, \dots, S_r . This implies that (H1) holds for \mathcal{H} . (H2) follows from the fact that $|S_i|$ is roughly $\frac{(\delta-1)|S|}{\delta}$, for $i = 1, \dots, r$. Thus Theorems 1 and 2 follow from Propositions 3 and 4 respectively. \square

4 Hypergraphs of bounded dual-conformality

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} . For a subset $S \subseteq V$, and a vertex $v \in S$, let $\mathcal{H}_v(S) = \{H \in \mathcal{H} \mid H \cap S = \{v\}\}$ and let $\mathcal{H}_0(S) = \{H \in \mathcal{H} \mid H \cap S = \emptyset\}$. 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 .

Proposition 5 (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 $\text{DUAL}(\mathcal{H}', 1)$ 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 [4]). 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.

Corollary 1. *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 $O(\log(nm))$ parallel time using $O(nm^\delta)$ processors.*

Let \mathcal{H} be a hypergraph such that (2) is satisfied 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 present below an efficient parallel algorithm that, for any given constant δ , either solves problem $\text{DUAL}(\mathcal{H}, k)$ or discovers that (2) is not satisfied.

Theorem 3. *There is a deterministic parallel algorithm that, given $\mathcal{H} \subseteq 2^V$, $k \in \mathbb{Z}_+$, and a constant integer δ , either solves problem $DUAL(\mathcal{H}, k)$, or proves that $\dim(\mathcal{H}^{dcd}) \geq \delta$, in time $O(\log n \log(nmk) + \Delta)$ using $O((nmk)^{2\delta \log \delta} + k\Pi)$ processors, where Δ and Π are respectively the parallel time and the number of processors required to generate a single minimal transversal in a hypergraph of dimension less than δ .*

A hypergraph $\mathcal{H} \subseteq 2^V$ is said to be δ -conformal [3] 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 $\dim(\mathcal{H}^d) \leq \delta$ implies that $\dim(\mathcal{H}^{dcd}) \leq \delta + 1$. Thus Theorem 3 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 .

Proof of Theorem 3. We 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. The algorithm proceeds in the following two steps:

Step 1. Generate the hypergraph $\mathcal{F} \subseteq 2^V$, whose hyperedges are defined as follows: $\mathcal{F} = \{S \subseteq V : |S| < \delta \text{ and } S \text{ is a minimal non sub-transversal of } \mathcal{H}\}$. For a constant δ , the hypergraph \mathcal{F} can be generated in $O(\log(nm))$ parallel time using $O((nm)^\delta)$ processors by Corollary 1.

Step 2. Note that $\dim(\mathcal{F}) < \delta$. Thus Theorem 2 implies that problem $DUAL(\mathcal{F}, k)$ can be solved efficiently in global RNC time. However, we may not need to generate 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

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

Now Theorem 3 becomes a consequence of Theorem 2, applied to the hypergraph \mathcal{F} constructed in Step 1 above. \square

5 Hypergraphs generalizing read-once hypergraphs

Let V be a finite set, $\delta \geq 2$ be a positive constant, and $\mathcal{H} \subseteq 2^V$ be a hypergraph satisfying (3). Note that testing a hypergraph \mathcal{H} for (3) can be done efficiently in parallel. Indeed, \mathcal{H} satisfies (3) if and only if for every edge $H \in \mathcal{H}$ and every subset $X \subseteq H$ of size $|X| = \delta$, X is not a sub-transversal to \mathcal{H} .

If $\delta = 2$, any hypergraph satisfying (3) is read-once (exact). For such hypergraphs, the following corollary is almost immediate from Theorem 1.

Corollary 2. *Let $\mathcal{H} \subseteq 2^V$ be a read-once hypergraph. Then for any integer k Problem $DUAL(\mathcal{H}, k)$ can be solved deterministically in $\text{polylog}(|V|, |\mathcal{H}|, k)$ parallel time using $\text{poly}(|V|, |\mathcal{H}|, k)$ number of processors.*

It is not known whether a similar criterion can be used to reduce the dualization of hypergraphs satisfying (3) to that of hypergraphs of bounded dimension. Nevertheless, we use our decomposition approach to prove the following result.

Theorem 4. *For any hypergraph $\mathcal{H} \subseteq 2^V$ satisfying (3), Computing k elements of \mathcal{H}^d can be done in $O((\delta \log(nmk) + \Delta)\delta \log m)$ parallel time using $O(((nmk^2)^\delta + k\Pi)m^{2\delta^2 \log m})$ processors.*

Proof. Let \mathcal{H} be a hypergraph satisfying (3). For a subset $S \subseteq V$, let $\mathcal{H}(S) = \{H \in \mathcal{H} : H \cap S \neq \emptyset\}$, $\mathcal{H}^S = \{\text{minimal}(H \cap S) \mid H \in \mathcal{H}, H \cap S \neq \emptyset\}$ and $\mathcal{H}_S = \{H \in \mathcal{H} \mid H \subseteq S\}$. Note that if \mathcal{H} satisfies (3) then so does \mathcal{H}_S for any $S \subseteq V$, while \mathcal{H}^S satisfies (3) if S is a transversal to \mathcal{H} . Denote by $\deg(v) = \deg_{\mathcal{H}}(v)$ the number of hyperedges of \mathcal{H} containing $v \in V$, and let $0 < \epsilon_1 < \epsilon_2 < 1/(\delta - 1)$ be positive constants. Let $\epsilon = \max\{1 - \epsilon_1, (\delta - 1)\epsilon_2, 1 - (\epsilon_2 - \epsilon_1)\} \in (0, 1)$. To generate k minimal transversals of \mathcal{H} , we use the following procedure:

Step 1. Set $Z \leftarrow \cup\{H \in \mathcal{H} : |H| = 1\}$, $V \leftarrow V \setminus Z$, and $\mathcal{H} \leftarrow \mathcal{H}_V$.

Step 2. If $|\mathcal{H}| = 0$, then return $\{Z\}$. If \mathcal{H} has only one hyperedge H , then return $\{Z \cup \{i\} : i \in H\}$.

Step 3. Let $L = \{v \in V : \deg(v) \geq \epsilon_1 |\mathcal{H}|\}$. If $V \setminus L$ is a transversal to \mathcal{H} , then we proceed with Steps 3.1, 3.2, and 3.3 below, otherwise, we go to Step 4.

Step 3.1. Let $T = \{v_1, \dots, v_t\} \subseteq V \setminus L$ be a minimal transversal to \mathcal{H} . Let $T_1 \cup T_2 \cup \dots \cup T_\delta = T$ be a partition of T computed by the following procedure: (a) find indices $i_1, \dots, i_{\delta-1} \in [t]$ such that, for $j = 1, \dots, \delta - 1$,

$$|\mathcal{H}(\{v_{i_{j-1}+1}, \dots, v_{i_j}\})| \leq \epsilon_2 |\mathcal{H}| \text{ and } |\mathcal{H}(\{v_{i_{j-1}+1}, \dots, v_{i_{j+1}}\})| > \epsilon_2 |\mathcal{H}|,$$

where $i_0 = 0$ and $i_\delta = t$, by definition (b) set $T_j \leftarrow \{v_{i_{j-1}+1}, \dots, v_{i_j}\}$ and $S_j \leftarrow V \setminus T_j$, for $j = 1, \dots, \delta$. Note that for every $H \in \mathcal{H}$, since $|H \cap T| \leq \delta - 1$, we have $H \subseteq S_j$ for some $j \in [\delta]$. Note also that $|\mathcal{H}_{S_j}| < (1 - (\epsilon_2 - \epsilon_1))|\mathcal{H}|$, for $j = 1, \dots, \delta$, and $|\mathcal{H}_{S_\delta}| \leq (\delta - 1)\epsilon_2 |\mathcal{H}|$.

Step 3.2. Let recursively (and in parallel) $\mathcal{G}_j \leftarrow DUAL(\mathcal{H}_{S_j}, k)$, for $j = 1, \dots, \delta$.

Step 3.3. If $|\mathcal{G}_j| = k$, for some $j \in [\delta]$, then compute (in parallel) for each $Y \in \mathcal{G}_j$ a minimal transversal T_Y of the sub-hypergraph $\mathcal{H}[Y] = \{H \setminus S_j : H \cap Y = \emptyset\} \subseteq 2^{V \setminus V_j}$. Note that $\mathcal{H}[Y]$ also satisfies (3). Return $\mathcal{G} \leftarrow \{Y \cup T_Y \cup Z : Y \in \mathcal{G}_j\}$.

Step 3.4. Otherwise, return k elements of the conjunction $\mathcal{G} \leftarrow \mathcal{G}_1 \wedge \dots \wedge \mathcal{G}_\delta \wedge \{Z\}$.

Step 4. If there are distinct vertices $v_1, \dots, v_\delta \in L$ such that no edge of \mathcal{H} contains $\{v_1, \dots, v_\delta\}$, then proceed with Steps 4.1 and 4.2 below, otherwise go to Step 5.

Step 4.1. Let $S_j \leftarrow V \setminus \{v_j\}$, for $j = 1, \dots, \delta$. Then $\mathcal{H} = \bigcup_{j=1}^\delta \mathcal{H}_{S_j}$, and $|\mathcal{H}_{S_j}| \leq (1 - \epsilon_1)|\mathcal{H}| \leq \epsilon |\mathcal{H}|$, for $j = 1, \dots, \delta$.

Step 4.2. Compute recursively $\mathcal{G}_j \leftarrow \text{DUAL}(\mathcal{H}_{S_j}, k)$, for $j = 1, \dots, \delta$, and continue as in Steps 3.3 and 3.4 above.

Step 5. Assume now that $V \setminus L$ is not a transversal to \mathcal{H} , and that for every subset $L' \subseteq L$ of δ distinct vertices, there is an edge $H \in \mathcal{H}$ such that $H \supseteq L'$. Then for every minimal transversal $T \in \mathcal{H}^d$, we have $1 \leq |T \cap L| \leq \delta - 1$ by (3).

In this case, we proceed as follows. Let $\mathcal{S} \stackrel{\text{def}}{=} \{S \subseteq L \mid \exists T \in \mathcal{H}^d : T \cap L = S\}$, and for $S \in \mathcal{S}$, denote by V_S the set $V \setminus (L \setminus S)$. Elements of \mathcal{S} can be identified as follows: a non-empty set $S \subseteq L$ of size at most $\delta - 1$ is in \mathcal{S} if and only if (i) V_S is a transversal to \mathcal{H} , and (ii) S is a sub-transversal of the hypergraph \mathcal{H}^{V_S} . For $S \in \mathcal{S}$, let $\mathcal{T}(S) = \{T \in \mathcal{H}^d \mid T \cap L = S\}$. Then \mathcal{H}^d is the disjoint union

$$\mathcal{H}^d = \left(\bigcup_{S \in \mathcal{S}} \mathcal{T}(S) \right) \wedge \{Z\}. \quad (4)$$

For each $S \in \mathcal{S}$, we find $\mathcal{T}(S)$ by applying the sub-transversal criterion (Proposition 5). More precisely, for $v \in S$, let $\mathcal{H}_v(S) = \{H \in \mathcal{H}^{V_S} \mid H \cap S = \{v\}\}$ and $\mathcal{H}_0(S) = \{H \in \mathcal{H}^{V_S} \mid H \cap S = \emptyset\}$. Let \mathbb{F}_S be the family of non-covering selections of \mathcal{H}^{V_S} with respect to S , i.e. collections of $|S|$ hyperedges $\{H_v \in \mathcal{H}_v(S) \mid v \in S\}$ for which there exists no hyperedge $H \in \mathcal{H}_0(S)$ with $H \subseteq \bigcup_{v \in S} H_v$. For every $\mathcal{F} = \{H_v \in \mathcal{H}_v(S) \mid v \in S\} \in \mathbb{F}_S$, denote by $V_{\mathcal{F}} = V \setminus \left(\bigcup_{H \in \mathcal{F}} H \cup L \right)$. Then it is easy to see that

$$\mathcal{T}(S) = \bigcup_{\mathcal{F} \in \mathbb{F}_S} (\mathcal{H}_0(S)^{V_{\mathcal{F}}})^d \wedge \{S\}. \quad (5)$$

Note that, for all $\mathcal{F} \in \mathbb{F}_S$ and $S \in \mathcal{S}$, we have $|\mathcal{H}_0(S)^{V_{\mathcal{F}}}| \leq (1 - \epsilon_1)|\mathcal{H}|$ since none of the edges intersecting S belongs to $\mathcal{H}_0(S)$. Using (4) and (5), we compute k elements of \mathcal{H}^d by recursively finding $\mathcal{G}_{\mathcal{F}} \leftarrow \text{DUAL}(\mathcal{H}_0(S)^{V_{\mathcal{F}}}, k)$ for every $\mathcal{F} \in \mathbb{F}_S$ and $S \in \mathcal{S}$. We stop either when $|\mathcal{G}_{\mathcal{F}}| = k$ for some $\mathcal{F} \in \mathbb{F}_S$ and some $S \in \mathcal{S}$, or when all families $(\mathcal{H}_0(S)^{V_{\mathcal{F}}})^d$ have been generated, for all $\mathcal{F} \in \mathbb{F}_S$ and $S \in \mathcal{S}$, in which case we use (4) and (5) to compute \mathcal{H}^d .

This completes our procedure for finding \mathcal{H}^d . The following proposition gives the parallel running time $t(n, m, k)$ and the number of processors $p(n, m, k)$ required to generate k minimal transversals, in terms of n , m , and k .

Proposition 6. $t(n, m, k) = O((\delta \log(nmk) + \Delta) \log m / \log \frac{1}{\epsilon})$ and $p(n, m, k) = O(((nmk^2)^\delta + k\Pi)m^{\delta \log m / \log \frac{1}{\epsilon}})$.

Setting $\epsilon = 1 - 1/(2\delta)$ in Proposition 6 completes the proof of Theorem 4. \square

References

1. N. Alon, L. Babai, A. Itai, A fast randomized parallel algorithm for the maximal independent set problem, *J. Algorithms* 7 (1986) pp. 567–583.
2. P. Beame, M. Luby, Parallel search for maximal independence given minimal dependence, in *Proc. of the First SODA Conference (1990)*, pp. 212–218.

3. C. Berge, *Hypergraphs*, North Holland Mathematical Library, Vol. 445, 1989.
4. 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.
5. E. Boros, K. Elbassioni, V. Gurvich and L. Khachiyan, Generating Maximal Independent Sets for Hypergraphs with Bounded Edge-Intersections, in *Proc. 6th Latin American Theoretical Informatics Conference 2004, LNCS 2976*, pp. 488–498.
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 J. Comput.*, 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) pp. 147–156.
8. C. J. Colbourn, *The combinatorics of network reliability*, Oxford Univ. 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)*, Sweden, July 5-8, 1988, LNCS 318, pp. 139–144.
10. E. Dahlhaus, M. Karpinski and P. Kelsen, An efficient parallel algorithm for computing a maximal independent set in a hypergraph of dimension 3, *Inf. Process. Lett.* 42(6) (1992), pp. 309–313.
11. C. Domingo, N. Mishra and L. Pitt, Efficient read-restricted monotone CNF/DNF dualization by learning with membership queries, *Machine learning* 37 (1999) pp. 89–110.
12. T. Eiter, Exact Transversal Hypergraphs and Application to Boolean μ -Functions, *J. Symb. Comput.* 17(3) (1994), pp. 215–225.
13. T. Eiter and G. Gottlob, Identifying the minimal transversals of a hypergraph and related problems, *SIAM J. Comput.*, 24 (1995), pp. 1278–1304.
14. T. Eiter, G. Gottlob and K. Makino, New results on monotone dualization and generating hypergraph transversals, in *Proc. 34-th Annual ACM STOC Conf.*, 2002, pp. 14–22.
15. M. L. Fredman and L. Khachiyan, On the complexity of dualization of monotone disjunctive normal forms. *J. Algorithms*, 21 (1996) pp. 618–628.
16. 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.
17. D. S. Johnson, M. Yannakakis and C. H. Papadimitriou, On generating all maximal independent sets, *Information Processing Letters*, 27 (1988), pp. 119–123.
18. R. Karp and A. Wigderson, A fast parallel algorithm for the maximal independent set problem, *JACM* 32 (1985) pp. 762–773.
19. R. Karp, E. Upfal, and A. Wigderson, The complexity of parallel search, *Journal of Computer and System Science*, 36 (1988) pp. 225–253.
20. P. Kelsen, On the parallel complexity of computing a maximal independent set in a hypergraph, *Proc. 24-th Annual ACM STOC Conf.* (1992).
21. E. Lawler, J. K. Lenstra and A. H. G. Rinnooy Kan, Generating all maximal independent sets: NP-hardness and polynomial-time algorithms, *SIAM J. Comput.*, 9 (1980) pp. 558–565.
22. T. Luczak and E. Szymanska, A parallel randomized algorithm for finding a maximal independent set in a linear hypergraph, *J. Algorithms* 25(2) (1997), 311–320.
23. K. Makino, Efficient Dualization of $O(\log n)$ -Term Monotone Disjunctive Normal Forms, *Discrete Applied Mathematics*, 126 (2003) pp. 305–312.
24. H. Tamaki, Space-efficient enumeration of minimal transversals of a hypergraph, *IPSJ-AL* 75 (2000), pp. 29–36.