Enumerating Minimal Transversals of Geometric Hypergraphs

Khaled Elbassioni[*] Imran Rauf[†] Saurabh Ray[‡]

Abstract

We consider the problem of enumerating all minimal hitting sets of a given hypergraph $(V, \mathcal{R})$, where V is a finite set, called the *vertex set* and $\mathcal{R}$ is a set of subsets of V called the *hyperedges*. We show that, when the hypergraph admits a *balanced subdivision*, then a recursive decomposition can be used to obtain efficiently all minimal hitting sets of the hypergraph. We apply this decomposition framework to get incremental polynomial-time algorithms for finding minimal hitting sets and minimal set covers for a number of hypergraphs induced by a set of points and a set of geometric objects. The set of geometric objects includes half-spaces, hyper-rectangles and balls, in fixed dimension. A distinguishing feature of the algorithms we obtain is that they admit an efficient *global* parallel implementation, in the sense that all minimal hitting sets can be generated in *polylogarithmic* time in $|V|$, $|\mathcal{R}|$ and the total number of minimal transversals T, using a polynomial number of processors.

1 Introduction

Let $(V, \mathcal{R})$ be a hypergraph, where V is a finite set called the *vertices* and $\mathcal{R}$ is a family of subsets of V called the *hyperedges*. We say a vertex $v \in V$ *hits* hyperedge $R \in \mathcal{R}$ when $v \in R$. A subset of vertices $X \subseteq V$ is said to be a *hitting set* (or *transversal*) for $\mathcal{R}$ if every hyperedge R in $\mathcal{R}$ is hit by an element x in X. A hitting set is *minimal* if none of its proper subsets is a hitting set.

In this paper, we will consider hypergraphs induced by a set of points and certain families of geometric objects in $\mathbb{R}^d$. When the hypergraph is arbitrary, we obtain the well-known *hypergraph transversal* or *dualization* problem [1], which calls for finding all minimal hitting sets for a given hypergraph. This problem has received considerable attention in the literature (see, e.g., [2, 8, 9, 17, 22, 24]), since it is known to be polynomially or quasi-polynomially equivalent with many problems in various areas.

Clearly, the number T of minimal transversals of a hypergraph $(V, \mathcal{R})$ can be exponential in $|V|$ and $|\mathcal{R}|$, and hence one can only hope for an algorithm whose efficiency is measured in terms of the parameters $|V|$, $|\mathcal{R}|$ and T. The currently fastest known algorithm [15] for solving the hypergraph transversal problem runs in quasi-polynomial time $|V|N^{o(\log N)}$, where N is the combined input and output size $N = |\mathcal{R}|+T$. Several quasi-polynomial time algorithms with some other desirable properties also exist [4, 12, 13, 16, 25]. While it is still open whether the problem can be solved in polynomial time for arbitrary hypergraphs, polynomial time algorithms exist for several classes of hypergraphs, e.g. hypergraphs of bounded edge-size [3, 8], of bounded degree [7, 9], of bounded-edge intersections [3], of bounded conformality [3], of bounded treewidth [9], and read-once (exact) hypergraphs [11].

Recently [14], this polynomiality frontier has been extended to a number of geometric hypergraphs. In particular, polynomial-time algorithms were given for enumerating minimal hitting sets and minimal set covers, when the hyperedges are induced by hyper-rectangles (in $\mathbb{R}^d$ for fixed d), or half-planes (in $\mathbb{R}^2$). We observe that the algorithms in [14] depend on the types of ranges considered. For instance, in the case of rectangles, two different types of algorithms are used for the hitting set and set covering versions, and a substantially modified algorithm is needed for half-planes. Furthermore, it is not clear, how these algorithms can be generalized to other geometric objects, such as balls, or half-spaces in fixed dimension $d \geq 3$.

In this paper, we present a simple and unified algorithm that works for both the hitting set and set covering versions, and extends to more general objects, such as balls, half-spaces, and polytopes with fixed number of facets. In fact, as we will see, all that is needed is that the hypergraph admits a certain *balanced subdivision* which can be shown to exist in several geometric instances. One more important property of the algorithms we obtain, is that they admit a *global parallel* implementation, in the sense that all minimal hitting sets can be generated in polylogarithmic time (in $|V|$, $|\mathcal{R}|$ and the total number of minimal transversals T) using a polynomial number of processors (in the PRAM model). Among all polynomially solvable classes of hypergraphs, only very few are known to exhibit this nice property, see [20, 21]. We remark that the general algorithms given in [4, 13] do *not* satisfy this global paral-

[*]Max-Planck-Institut für Informatik, Saarbrücken, Germany, elbassio@mpi-inf.mpg.de

[†]Friedrich-Schiller-Universität, Jena, Germany, imran.rauf@uni-jena.de

[‡]Max-Planck-Institut für Informatik, Saarbrücken, Germany, saurabh@mpi-inf.mpg.de

lelism property, in the sense that they only produce *each* output in parallel, that is, the time needed to produce T minimal transversals is *polylogarithmic* in $|V|, |\mathcal{R}|$, but can be *linear* (or super-linear) in T. Finding a global parallel algorithm (even with a quasi-polynomial number of processors) for the general case is an outstanding open problem, and is very useful in practice since the number of minimal transversals is typically very large.

Theorem 1 *Let $(V, \mathcal{R})$ be a hypergraph defined by a set of points $P \in \mathbb{R}^d$ and set of ranges $\mathcal{R} \subseteq \mathbb{R}^d$. If $\mathcal{R}$ is a set of half-planes, balls, or a polytopes with a fixed number of facets, in fixed dimension $d = O(1)$, then there is an algorithm that enumerates all T minimal hitting sets (resp., set covers) of $\mathcal{R}$ in parallel time* $\mathrm{polylog}(|V|, |\mathcal{R}|, T)$ *on* $\mathrm{poly}(|V|, |\mathcal{R}|, T)$ *number of processors.*

Even though the bound in Theorem 1 is stated in terms of the total number of minimal hitting sets T, we will see that our algorithm can be modified to work in an *incremental setting*, i.e, for a given integer $T' \leq T$, it finds T' minimal hitting sets (resp., set covers) of $\mathcal{R}$ in time $\mathrm{polylog}(|\mathcal{R}|, T') \cdot \tau(|V|)$ on $\mathrm{poly}(|V|, |\mathcal{R}|, T')$ number of processors, where $\tau(|V|)$ is the time needed to find a single minimal hitting set. The currently best known parallel implementation for the latter problem [19] has $\tau(|V|) = O(\sqrt{|V|})$.

Our algorithm builds on and extends the *covering decomposition* approach initially suggested in [20, 21], and used in a number of subsequent works [12, 4]. So far, the use of such decompositions has been successful for developing polynomial time dualization algorithms for limited cases, such as hypergraphs of bounded size or bounded degree. In this paper, we show that the limitations of the previous approaches can be overcome using a modified version. This allows us to obtain a large class of hypergraphs for which covering decompositions are effective.

The enumeration of minimal geometric hitting sets, as the ones described above, arises in various areas such as computational geometry, machine learning, and data mining [10]. Moreover, our efficient enumeration algorithms might be useful in developing exact algorithms, fixed-parameter tractable algorithms, and polynomial-time approximation schemes for the corresponding optimization problems (see, e.g., [18]).

2 Notation

In this paper, we will often write $\mathcal{R}$ for a hypergraph $(V, \mathcal{R})$ for notational convenience. Also, we will often refer to hypergraphs as range spaces in accordance with the terminology in the Computational Geometry literature. Accordingly, we will refer to the vertex set as the *ground set* and the hyperedges as *ranges*. Given a range spaces $\mathcal{R}$, we will denote by $\mathrm{Tr}(\mathcal{R})$ the set of all minimal hitting sets of $\mathcal{R}$. Given a range $(V, \mathcal{R})$, and a subset $V' \subseteq V$, we will denote by $\mathcal{R}|_{V'}$ the set $\{R \cap V' : R \in \mathcal{R}\}$. The hypergraph $(V', \mathcal{R}_{V'})$ is called the projection of the hypergraph $(V, \mathcal{R})$ on V'.

3 Balanced Subdivisions

Given any range space $(V, \mathcal{R})$, we say that a subset $V' \subseteq V$ is *stabbed* by a range $R \in \mathcal{R}$ if there exist $x, y \in V'$ s.t. $x \in R$ and $y \notin R$. A *balanced subdivision* for a range space $(V, \mathcal{R})$ is a collection of a constant number of subsets $V_1, V_2, \ldots, V_k$ of V such that

1. For each $i \in \{1, \ldots, k\}, |V_i| \geq \epsilon |V|$ for some constant $0 < \epsilon \leq 1$.
2. For each range $R \in \mathcal{R}$, there are two disjoint subsets V_i and V_j in the collection which are not stabbed by R.

Remark 1 *In the above definition, the fact that a subset $V' \subset V$ is not stabbed by $R \in \mathcal{R}$ implies that V' is also not stabbed by $V \setminus R$. Consequently, we conclude that a balanced subdivision for any range space $(V, \mathcal{R})$ is also a balanced subdivision for the range space $(V, \mathcal{R}^c)$, where $\mathcal{R}^c$ denotes the hypergraph defined by compliments of ranges in $\mathcal{R}$, i. e., $\mathcal{R}^c = \{V \setminus R \mid R \in \mathcal{R}\}$.*

In the next section, we show that if we can compute a balanced subdivision for a range space then we can enumerate its minimal hitting sets in global parallel time.

In this section, we show that several geometrically induced range spaces admit balanced subdivisions which can be computed efficiently (in parallel). We consider range spaces induced by a set of points P and a set of geometric objects $\mathcal{H}$ in $\mathbb{R}^d$. There are two natural range spaces defined by them depending on whether we let the points or the objects form the ground set. We denote by $(P, \mathcal{H})$ the range space in which P is the ground set and each $H \in \mathcal{H}$ defines the range $H \cap P$. Similarly, we denote by $(\mathcal{H}, P)$ the range space in which the ground set is $\mathcal{H}$ and each $p \in P$ defines the range $\{H \in \mathcal{H} \mid p \in H\}$.

We now show that for any point set $P \subset \mathbb{R}^d$, a balanced subdivision exists and can be computed efficiently for both $(P, \mathcal{H})$ and $(\mathcal{H}, P)$ if $\mathcal{H}$ is a family of objects of the following kind: (i) Half-Spaces in $\mathbb{R}^d$ (ii) Polytopes with at most a constant number of facets in $\mathbb{R}^d$ and (iii) Balls in $\mathbb{R}^d$.

We will use the following results:

Theorem 2 (Fine Simplicial Partitions [23]) *Given any set P of n points in $\mathbb{R}^d$ and any parameter $1 \leq r \leq n$, there exists a partition*

$\Pi = \{P_1, P_2, \ldots, P_t\}$ of $t \leq r$ disjoint subsets of P and a set $\Delta = \{\Delta_1, \ldots, \Delta_t\}$ of simplices with the following properties: (i) $P_i \subseteq \Delta_i$ (ii) $\bigcup_i P_i = P$, (iii) $n/r \leq |P_i| \leq 2n/r$ for all $i \in \{1, \ldots, t\}$ and (iv) no half-space in $\mathbb{R}^d$ intersects more than $C_d r^{1-1/d}$ of the simplices in Δ, where C_d is a constant for any fixed d. The last property also implies that no half-space in $\mathbb{R}^d$ stabs more than $C_d r^{1-1/d}$ of the sets in Π. Further, for any $\delta > 0$, such a Π can be computed in time $O(n^{1+\delta})$. When r is bounded by a constant, Π can be computed in $O(n)$ time.

Theorem 3 (Cuttings [5]) *Given any set of n half-spaces in $\mathbb{R}^d$ and any parameter $1 \leq r \leq n$, there exists a partition of $\mathbb{R}^d$ into r simplices such that none of the simplices is stabbed by more than $C'_d n/r^{1/d}$ of the given half-spaces. Further, for any $\delta > 0$, such a partition can be computed deterministically in time $O(nr^{1-1/d})$.*

Parallel Implementation: Even though we only mention sequential running times above, such fine simplicial partitions and cuttings can be computed in polylog(n) time using poly(n) processors. For example, in the case of cuttings in any fixed dimension d, while the simplices are allowed to be arbitrary, it can be argued that they can always be chosen so that they are among a polynomial number of *canonical* simplices. In fact, it is not hard to argue that we can restrict to simplices whose corners are defined by the intersection of d of the hyperplanes defining the given set of halfspaces and indeed the construction in [5] does restrict to such simplices. The number of such simplices is at most $O(n^{d(d+1)})$. Once we have a polynomial bound on the number of canonical simplices, we can check all possible sets of t canonical simplices in parallel using a polynomial number of processors and find a simplicial partition in polylogarithmic time. A similar argument holds for simplicial partitions.

Half-Spaces. Let us first consider the case when $\mathcal{H}$ is a set of half-spaces in $\mathbb{R}^d$. Given any set P with n points in $\mathbb{R}^d$, we can set r to be a large enough constant so that $C_d r^{1-1/d} \leq r-2$. Then, clearly, the collection Π given by Theorem 2 also gives us a balanced subdivision of $(P, \mathcal{H})$ with $\epsilon = 1/r$ and $k \leq r$. To get a balanced subdivision of $(\mathcal{H}, P)$, we apply Theorem 3 to the half-spaces in $\mathcal{H}$. Assuming that $|\mathcal{H}| = n$, we set r to be a large enough constant so that $C'_d n/r^{1/d} \leq n/2$. Theorem 3 gives a partition of $\mathbb{R}^d$ into r regions $R_1, \ldots, R_r$ each of which is stabbed by at most $n/2$ half-spaces in $\mathcal{H}$. Consequently, for each region R_i, we either have at least $n/4$ half-spaces of $\mathcal{H}$ each of which contains R_i or we have at least $n/4$ half-spaces of $\mathcal{H}$ none of which intersects R_i. Let $\mathcal{H}_i$ be a set of those half-spaces. Then $|\mathcal{H}_i| \geq n/4$. We arbitrarily partition each $\mathcal{H}_i$ into two disjoint sets $\mathcal{H}_i^1$ and $\mathcal{H}_i^2$ each of size at least $n/8$. These sets $\mathcal{H}_i^1$ and $\mathcal{H}_i^2$ for $i \in \{1, \ldots, r\}$ give us a balanced subdivision of $(\mathcal{H}, P)$ with $k = 2r$ and $\epsilon = 1/8$ since each point $p \in P$ lies in some region R_j and hence does not stab the disjoint sets $\mathcal{H}_j^1$ and $\mathcal{H}_j^2$.

Remark 2 *In the case of half-spaces, one may also reduce the problem of finding minimal set covers to that of finding minimal hitting sets by using* geometric *duality, which maps points in $\mathbb{R}^d$ to hyperplanes and vice versa (see e.g. [6], Chapter 8). However this method does not work for polytopes.*

Polytopes. Suppose now that $\mathcal{H}$ is a set of polytopes in $\mathbb{R}^d$, each with at most f facets. In this case the collection Π given by Theorem 2 with r being a large enough constant so that $f \cdot C_d r^{1-1/d} \leq r-2$ gives us the required balanced subdivision for $(P, \mathcal{H})$. This is because each facet of a polytope can stab at most $C_d r^{1-1/d}$ members of Π and therefore a polytope with at most f facets can stab at most $f \cdot C_d r^{1-1/d} \leq r-2$ members of Π. To get a balanced subdivision for $(\mathcal{H}, P)$, we consider for each $H \in \mathcal{H}$, the set of at most f half-spaces whose intersection forms H. Let $\mathcal{H}'$ be the set of all these half-spaces. Assuming that $|\mathcal{H}| = n$, we have that $|\mathcal{H}'| \leq fn$. We then invoke Theorem 3 for the half-spaces in $\mathcal{H}'$ with r being a large enough constant so that $C'_d(nf)/r^{1/d} \leq n/2$. The regions in the resulting partition are stabbed by at most $n/2$ half-spaces in $\mathcal{H}'$ and hence at most $n/2$ polytopes in $\mathcal{H}$. We can then construct the balanced subdivision for $(\mathcal{H}, P)$ consisting of sets $\mathcal{H}_i^1$ and $\mathcal{H}_i^2$, $i \in \{1, \ldots, r\}$, as in the last paragraph.

Balls. Finally assume that $\mathcal{H}$ is a set of balls in $\mathbb{R}^d$. There is a standard *lifting* (veronese map) which maps each point in $\mathbb{R}^d$ to a point in $\mathbb{R}^{d+1}$ and each ball in $\mathbb{R}^d$ to a half-space in $\mathbb{R}^{d+1}$ so that the incidence relations among them are preserved. Since balanced subdivisions exist for half-spaces in $\mathbb{R}^{d+1}$, we can conclude that balanced subdivisions exist for balls in $\mathbb{R}^d$ as well (for both $(P, \mathcal{H})$ and $(\mathcal{H}, P)$). The same trick can be applied to other algebraically defined sets like ellipses, parabolas etc.

Since we invoke Theorems 2 and 3 with r being a constant, the collection in Theorem 2 and the partition in Theorem 3 can be computed in time $O(n)$. It follows that balanced subdivisions for the above range spaces can be computed in $O(n+m)$ time where n is the size of the ground set and m is the number of ranges.

4 The Enumeration Algorithm

Given a range space $(V, \mathcal{R})$, a divide-and-conquer algorithm to enumerate all minimal hitting sets of $\mathcal{R}$ is presented in Figure 1. If $|V|$ is at most some fixed constant μ, all minimal hitting sets of $\mathcal{R}$ can be enumerated by just enumerating all subsets of V and outputting those which form a minimal hitting set of $\mathcal{R}$. We assume the existence of a procedure Enumerate-Small for the enumeration of minimal hitting sets in these trivial cases.

Algorithm 1 Procedure Enumerate($V, \mathcal{R}$):

```
Input: A finite range space $(V, \mathcal{R})$
Output: The set of all minimal hitting sets of $\mathcal{R}$
 1: if $|V| \leq \mu$ then
 2:   return Enumerate-Small$(V, \mathcal{R})$
 3: end if
 4: Type1-Set:=$\emptyset$
 5: Compute a balanced subdivision $V_1, \ldots, V_\lambda$ of $(V, \mathcal{R})$
 6: for $i = 1, \ldots, \lambda$ do
 7:   Type1-Set := Type1-Set $\cup$ Enumerate$(V \setminus V_i, \mathcal{R}|_{V \setminus V_i})$
 8: end for
 9: Type1-Set := Remove-Duplicates(Type1-Set)
10: Type2-Set := $\emptyset$
11: for $i = 1, \ldots, \lambda$ do
12:   $\mathcal{X}_i$ := Enumerate$(V \setminus V_i, \mathcal{R}_i)$
13: end for
14: for each $(M_1, \ldots, M_\lambda) \in \mathcal{X}_1 \times \ldots \times \mathcal{X}_\lambda$ do
15:   $M := \bigcup_i M_i$
16:   if $M$ is a type 2 minimal hitting set of $\mathcal{R}$ then
17:     Type2-Set := Type2-Set $\cup \{M\}$
18:   end if
19: end for
20: Type2-Set := Remove-Duplicates(Type2-Set)
21: return Type1-Set $\cup$ Type2-Set
```

When $|V| > \mu$, we assume the existence of a balanced subdivision $\Pi = (V_1, \ldots, V_\lambda)$, where λ is a constant and for each $i \in \{1, \ldots, \lambda\}$, $|V_i| \leq \epsilon|V|$ where $0 < \epsilon < 1$ is another constant. We classify the minimal hitting sets of $\mathcal{R}$ into two types. Type 1 minimal hitting sets are those that have an empty intersection with one of the V_is. The remaining minimal hitting sets which contain elements from each V_i are of type 2. Since each V_i contains a constant fraction of the elements in V, type 1 hitting sets are easily enumerated recursively. This is done in line 7 of Figure 1. Enumerating Type 2 minimal hitting sets require more work.

Let us first observe that any minimal hitting set M of $\mathcal{R}$ and for any $v \in M$, there is always some range $R \in \mathcal{R}$ which *requires* v, i.e., $R \cap M = \{v\}$. We call such a range a *certificate* range for v in M. Clearly, M is also a minimal hitting set for the set of certificate ranges of its elements.

Let M be any type 2 minimal hitting set and let $R \in \mathcal{R}$ be any range that has a nonempty intersection with each of the V_is. Since Π is a balanced subdivision, there are at least two sets V_j and V_k which are not stabbed by R. Since R has a nonempty intersection with both of them, it must contain both the sets as subsets. Now, since M contains an element from each V_i, R contains at least two elements of M implying that R cannot be a certificate range for any element of M. This means that for the purpose of enumerating type 2 minimal hitting sets, we can discard all ranges which have a non-empty intersection with each of the V_is. Let $\mathcal{R}_i = \{R \in \mathcal{R} : R \cap V_i = \emptyset\}$ and let $\tilde{\mathcal{R}} = \bigcup_i \mathcal{R}_i$.

Let M be any type 2 minimal hitting set of $\mathcal{R}$. Since $\tilde{\mathcal{R}}$ contains all certificate ranges of M, M is also a minimal hitting set for $\tilde{\mathcal{R}}$. Also, since the ranges in $\mathcal{R}_i$ do not contain any element of V_i, $M \setminus V_i$ is a hitting set (not necessarily minimal) for $\mathcal{R}_i$ and therefore contains some $M_i \subseteq M \setminus V_i$ which is a minimal hitting set for $\mathcal{R}_i$.

Notice that each element of M appears in at least one of the M_is. This is because each $v \in M$ has a certificate range R which belongs to some $\mathcal{R}_i$ implying that $v \in M_i$. In other words, $M = \bigcup_i M_i$. This suggests the following algorithm for enumerating the minimal hitting sets of type 2. For each $i \in \{1, \ldots, \lambda\}$, recursively compute the set $\mathcal{X}_i$ of all minimal hitting sets of $(V \setminus V_i, \mathcal{R}_i)$. Then, try all possible ways of picking one minimal hitting set $M_i \in \mathcal{X}_i$ from each $\mathcal{X}_i$ and output $M = \bigcup_i M_i$ if it is a type 2 minimal hitting set for $\mathcal{R}$. This way we surely enumerate all type 2 minimal hitting sets. Now, we need to bound the number of combinations we try. We do it by proving an upper bound on each $|\mathcal{X}_i|$.

Lemma 4 *Let T be the number of minimal hitting sets of $\mathcal{R}$. Then, $|\mathcal{X}_i| \leq T$.*

Proof. We show that each $N \in \mathcal{X}_i$ can be extended to $N' = N \cup S$ for some $S \subseteq V_i$ so that N' is a minimal hitting set of $\mathcal{R}$. The lemma then follows since each distinct $N \in \mathcal{X}_i$ is extended to a distinct minimal hitting set N' of $\mathcal{R}$. Given any $N \in \mathcal{X}_i$, let $\bar{\mathcal{R}}$ be the set of ranges in $\mathcal{R}$ that are not hit by N. Clearly, each range in $\bar{\mathcal{R}}$ has a non-empty intersection with V_i (otherwise it would be in $\mathcal{R}_i$ and thus would be hit by N). Therefore, there exists a set $S \subseteq V_i$ which is a minimal hitting set for $\bar{\mathcal{R}}$. Now, $N' = N \cup S$ is certainly a hitting set of $\mathcal{R}$. Furthermore, it is also minimal since each element of N' has a certificate range. The certificate ranges for each element v in N are also certificate ranges for v in N' since these ranges belong to $\mathcal{R}_i$ and hence do not contain any elements of S. Also, the certificate ranges for each v in S are certificate ranges for v in N', since these ranges belong to $\bar{\mathcal{R}}$ and hence do not intersect N. □

It follows from the above lemma that the number of combinations of M_i's we need to try is at most T^λ. After we find all type 1 minimal hitting sets we run a procedure called Remove-Duplicates to remove any duplicates we may have generated. Similarly, after we find all type 2 minimal hitting sets, we run Remove-Duplicates to remove any duplicates. This ensures that in the end we do not output any duplicates.

We now do an analysis of the running time of the algorithm. In the analysis, we treat the number of ranges $m = |\mathcal{R}|$ and the number T of the number of minimal hitting sets of $\mathcal{R}$ as constants. We denote by $t(n)$, the running time of the procedure Enumerate on a hypergraph $(V, \mathcal{R})$ where $|V| = n$. The recursive calls in Line 7 of Algorithm 1 for enumerating the type 1 minimal hitting sets take time $\lambda t((1-\epsilon)n)$. Similarly, the total time spent in Line 12 is $\lambda t((1-\epsilon)n)$. The loop starting on Line 14 is executed at most T^λ times. In each iteration, checking whether M is a type 2 minimal hitting set of $\mathcal{R}$ takes $O(mn)$ time. Hence the total time spent in the loop is $O(mnT^\lambda)$. Since there are at most T distinct minimal hitting sets of $\mathcal{R}$, when we reach Line 9, Type1-Set has at most λT minimal hitting sets. Each of these have to be tested against a set of at most T distinct minimal hitting sets to see if it has already been reported. Therefore, this takes $O(\lambda T^2 n)$ time assuming that it takes $O(n)$ to check if two minimal hitting sets are the same. Similarly, when we reach Line 20, the size of Type2-Set is at most T^λ and each of the hitting sets in it is compared against a set of at most T minimal hitting sets to see if it has been reported before. This takes $O(T^{\lambda+1}n)$ time. We therefore have the following recursion:

$$t(n) \leq 2\lambda t((1-\epsilon)n) + \lambda n T^2 + mnT^\lambda + nT^{\lambda+1} + \tau,$$

where τ is the time required to find a balanced subdivision. Using the fact that $t(n)$ is a constant when n is smaller than some constant μ, we see that $t(n) = O((\tau + nT^{\lambda+1} + nmT^\lambda) \cdot n^{\frac{\log \lambda}{\log 1/(1-\epsilon)}})$. We therefore have the following theorem.

Theorem 5 *Procedure Enumerate$(V, \mathcal{R})$ finds all minimal hitting sets of a range space $(V, \mathcal{R})$ which admits a balanced subdivision $V_1, \ldots, V_\lambda$ with each $|V_i| \geq \epsilon|V|$, whenever $|V|$ is larger than a fixed constant μ, in time $O((\tau + nT^{\lambda+1} + nmT^\lambda) \cdot n^{\frac{\log \lambda}{\log 1/(1-\epsilon)}})$, where $n = |V|$, $m = |\mathcal{R}|$, T is the number of minimal hitting sets of $\mathcal{R}$ and τ is the time required to compute a balanced subdivision.*

Remark 3 *The way the above algorithm is described gives an output polynomial algorithm for generating* $\mathrm{Tr}(\mathcal{R})$. *Using techniques from [20], we can modify the algorithm to become incremental polynomial, that is, for every $M' \leq M$ the algorithm outputs M' transversals in time polynomial in n, m and M'.*

Parallel Implementation of the Algorithm: Algorithm 1 can be parallelized in an obvious way. Each of the For loops can be executed in parallel, i.e., all the iterations are done in parallel. Using $\mathrm{poly}(n, m, T)$ processors, each of the other steps can be executed in $\mathrm{polylog}(n, m, T)$ time. If we denote by $t^{\|}(n)$ the running time of such a parallel algorithm, again treating m and T as constants, we get the following recurrence: $t^{\|}(n) = t^{\|}((1-\epsilon)n) + \mathrm{polylog}(n, m, T)$. We therefore have that $t^{\|}(n)$ is in $\mathrm{polylog}(n, m, T)$. It can be checked that the total number of processors required is only $\mathrm{poly}\,(n, m, T)$.

Using the techniques in [20], we can also get an incremental version of this, i.e., for any $M' \leq M$, the running time depends polylogarithmically on M', provided that there is an efficient parallel algorithm for finding a single minimal transversal of the input hypergraph $\mathcal{R}$. The existence of the latter algorithm for general hypergraphs, and in particular for range spaces, is an outstanding open question (see e.g. [19]). The currently best known parallel implementation for the later problem is due to Karp, Upfal, and Wigderson [19] who gave a randomized algorithm which makes only $O(\sqrt{n})$ parallel oracle calls on $O(n^{3/2})$ processors to compute a maximal independent set (complement of a minimal transversal, in the case of explicitly given hypegraphs) of an independence system given by an oracle on n vertices.

The running time of the algorithm described above is super-linear in the output size and hence not ideal for some applications. Similarly, previous algorithms for generating minimal hitting sets of half-planes [14] suffer from the same short-coming. In some simple cases, however, there exist algorithms which produce output with polynomial delay i.e. the time spent between enumerating two minimal hitting sets is polynomial in the size of the input and does not depend on the size of the output. Such algorithms clearly have a running time linear in the size of the output. We state the following result without proof.

Theorem 6 *Let P be a set of n points and $\mathcal{R}$ be a set of m half-planes in $\mathbb{R}^2$. Then all minimal hitting sets of the range spaces $(P, \mathcal{R})$ and $(\mathcal{R}, P)$ can be generated in* $\mathrm{poly}(n, m) \cdot k$ *time where k is the size of the output.*

References

[1] C. Berge. *Hypergraphs.* Elsevier-North Holand, Amsterdam, 1989.

[2] J. C. Bioch and T. Ibaraki. Complexity of identification and dualization of positive boolean functions. *Information and Computation*, 123(1):50–63, 1995.

[3] E. Boros, K. Elbassioni, V. Gurvich, and L. Khachiyan. Generating maximal independent sets for hypergraphs with bounded edge-intersections. In *LATIN '04*, pages 488–498, 2004.

[4] E. Boros and K. Makino. A fast and simple parallel algorithm for the monotone duality problem. In *ICALP '09, to appear*, 2009.

[5] Bernard Chazelle. Cutting hyperplanes for divide-and-conquer. *Discrete & Computational Geometry*, 9:145–158, 1993.

[6] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry, Algorithms and Applications*. SpringerVerlag, Amsterdam, 1997.

[7] C. Domingo, N. Mishra, and L. Pitt. Efficient read-restricted monotone cnf/dnf dualization by learning with membership queries. *Machine Learning*, 37(1):89–110, 1999.

[8] T. Eiter and G. Gottlob. Identifying the minimal transversals of a hypergraph and related problems. *SIAM J. Computing*, 24(6):1278–1304, 1995.

[9] T. Eiter, G. Gottlob, and K. Makino. New results on monotone dualization and generating hypergraph transversals. *SIAM J. Computing*, 32(2):514–537, 2003.

[10] T. Eiter, K. Makino, and G. Gottlob. Computational aspects of monotone dualization: A brief survey. *Discrete Applied Mathematics*, 156(11):2035–2049, 2008.

[11] Thomas Eiter. Exact transversal hypergraphs and application to Boolean μ-functions. *Journal of Symbolic Computation*, 17(3):215–225, 1994.

[12] K. Elbassioni. On the complexity of the multiplication method for monotone CNF/DNF dualization. In *ESA '06*, pages 340–351, 2006.

[13] K. Elbassioni. On the complexity of monotone dualization and generating minimal hypergraph transversals. *Discrete Applied Mathematics*, 156(11):2109–2123, 2008.

[14] K. Elbassioni, K. Makino, and I. Rauf. Output-sensitive algorithms for enumerating minimal transversals for some geometric hypergraphs. In *ESA*, 2009.

[15] M. L. Fredman and L. Khachiyan. On the complexity of dualization of monotone disjunctive normal forms. *Journal of Algorithms*, 21:618–628, 1996.

[16] D. R. Gaur and R. Krishnamurti. Average case self-duality of monotone boolean functions. In *Canadian AI '04*, pages 322–338, 2004.

[17] G. Gottlob. Hypergraph transversals. In *FoIKS '04: Proc. of the 3rd International Symposium on Foundations of Information and Knowledge Systems*, pages 1–5, 2004.

[18] F. Hüffner, R. Niedermeier, and S. Wernicke. Techniques for practical fixed-parameter algorithms. *Comput. J.*, 51(1):7–25, 2008.

[19] R. M. Karp, E. Upfal, and A. Wigderson. The complexity of parallel search. *Journal of Computer and System Sciences*, 36(2):225–253, 1988.

[20] L. Khachiyan, E. Boros, K. Elbassioni, and V. Gurvich. A global parallel algorithm for the hypergraph transversal problem. *Information Processing Letters*, 101(4):148–155, 2007.

[21] L. Khachiyan, E. Boros, V. Gurvich, and K. Elbassioni. Computing many maximal independent sets for hypergraphs in parallel. *Parallel Processing Letters*, 17(2):141–152, 2007.

[22] L. Lovász. Combinatorial optimization: some problems and trends. DIMACS Technical Report 92-53, Rutgers University, 1992.

[23] Jirí Matousek. Efficient partition trees. *Discrete & Computational Geometry*, 8:315–334, 1992.

[24] C. Papadimitriou. NP-completeness: A retrospective. In *ICALP '97*, 1997.

[25] H. Tamaki. Space-efficient enumeration of minimal transversals of a hypergraph. Technical Report IPSJ-AL 75, 2000.