The Union of Minimal Hitting Sets: Parameterized Combinatorial Bounds and Counting[*]

Peter Damaschke and Leonid Molokov
Department of Computer Science and Engineering
Chalmers University, 41296 Göteborg, Sweden
[ptr,molokov]@chalmers.se

Abstract

A k-hitting set in a hypergraph is a set of at most k vertices that intersects all hyperedges. We study the union of all inclusion-minimal k-hitting sets in hypergraphs of rank r (where the rank is the maximum size of hyperedges). We show that this union is relevant for certain combinatorial inference problems and give worst-case bounds on its size, depending on r and k. For $r = 2$ our result is tight, and for each $r \geq 3$ we have an asymptotically optimal bound and make progress regarding the constant factor. The exact worst-case size for $r \geq 3$ remains an open problem. We also propose an algorithm for counting all k-hitting sets in hypergraphs of rank r. Its asymptotic runtime matches the best one known for the much more special problem of finding one k-hitting set. The results are used for efficient counting of k-hitting sets that contain any particular vertex.

Keywords: algorithms, parameterization, combinatorial inference, counting, hypergraph transversals

[*]The paper without Section 3 is an improved version of results presented by the first author in [10].

1 Introduction

A *hypergraph* G consists of n *vertices* and a family of h subsets of vertices called *hyperedges*. The rank r of G is the maximum number of vertices in a hyperedge. A *hitting set* is a set of vertices that intersects all hyperedges. A *k-hitting set* is a hitting set with at most k vertices. We sometimes use the term *exact-k-hitting set* if it has exactly k vertices. A hitting set is *minimal* if no proper subset of it is a hitting set as well. Carefully distinguish this from a *minimum* (cardinality) hitting set. We always use k^* for the size of a minimum hitting set in G. Hypergraphs of rank 2 are graphs, and hitting sets are called *vertex covers* there. The *degree* of a vertex is the number of hyperedges it belongs to. We will use these notations throughout the paper.

The Hitting Set problem is to find some *k-hitting set* when G, k are given. In the Hitting Set Enumeration problem, also known as Transversal Hypergraph construction, we have to list all minimal hitting sets of G. Even for limited r and k, there can exist very many minimal k-hitting sets most of which differ by a few vertices. A plain list of all transversals is therefore rather useless. In certain combinatorial inference applications (as outlined below), the main interest is which vertices appear in minimal k-hitting sets at all, and in how many. This leads to other related problems that we consider in this paper: Hitting Set Counting asks how many (exact) k-hitting sets exist in G. Finally we define the problem Counting Hitting Sets with a Specified Vertex (CHSwSV): Given G and k, compute for every vertex v of G the number of k-hitting sets that contain v.

Our interest stems from some general and fundamental inference problem appearing in, e.g., diagnostics (cf. the references in [14]): We are given a set V of possible *causes*, a set E of *effects*, a relation $R \subset V \times E$, and a set $O \subset E$ of *observed* effects. A cause or effect is either present or absent. Relation $(v, e) \in R$ means that v may cause e. We suppose no interference, i.e., causes generate effects independently. The task is to infer the set $C \subset V$ of present causes. Since each $e \in O$ must be explained by some cause, this is just an instance of the Hitting Set problem. (The hypergraph to be considered depends on whether a present cause v generates *some* or *all* effects e with $(v, e) \in R$; it is easy to figure out the details). Parameter k is some *a priori* bound on the number of present causes: $|C| \leq k$.

Following the terminology of [16], the minimal k-hitting sets form the *specific boundary of the version space* of consistent hypotheses in this inference problem. It is essential to keep all consistent hypotheses rather than only one minimal solution which might not explain the data correctly. Additional informative experiments can then discriminate between the hypotheses and finally point to a unique or most likely solution. (Cf. the general discussions of these issues in [5, 9].) The primary interest is evidence for the presence or absence of every single cause $v \in V$. A natural approach, especially if no solution is preferred *a priori*, is to count for each v the hypotheses in the version space with and without v. This yields posterior probabilities for the presence of causes. The approach can be generalized to more complicated priors (by weighting), as well as joint presence of several causes, etc.

Often, most effects e are characteristic for only few possible causes each. That is, most hyperedges have size at most r, for some small fixed integer r. In practical problem instances we may first ignore larger hyperedges: While this can make the version space larger, the hitting set problems become computationally tractable in hypergraphs of fixed small rank (see below). The temporarily ignored hyperedges can be reinserted afterwards, and the version space narrowed down in an *ad hoc* fashion for the concrete instance. Also, large hyperedges impose less powerful constraints on the family of hitting sets.

An application example is the reconstruction of protein mixtures using peptide mass fingerprinting and a database of mass spectra. The connection to hitting set and similar problems was also noticed in bioinformatics literature [2, 18]. Here the "causes" are proteins to detect, and the "effects" are peptides (or only their masses) observed in a digested mixture. A mixture is known to contain at most a certain number of proteins (think of 30-50), and large peptide masses appear in a handful candidate proteins only.

A computational problem with input size n and another input parameter k is fixed-parameter tractable (FPT) if it can be solved in $O(T(k)p(n))$ time, where T is any function (often exponential) and p some polynomial. For introductions we refer to [11, 19]. Hitting Set is unlikely to be in FPT for general hypergraphs, but is easily seen to be in FPT for hypergraphs of any fixed rank r, and so behave the corresponding enumeration and counting problems.

We presume some familiarity with the techniques of FPT algorithms, but we briefly outline some facts needed later. Any FPT problem can be reduced in polynomial time to *kernel*, that is, an instance of size depending only on parameter k but not on n. FPT algorithms are often branching algorithms (applied to a kernel), where an instance is recursively split into several instances, where parameter k is reduced by $k_1, \dots, k_s$ in s different branches. Then the "parameterized" part of the time complexity is the solution to the recurrence $T(k) = \sum_{i=1}^{s} T(k - k_i)$, namely $T(k) = x^k$, where x is the positive root of the characteristic equation $x^k = \sum_{i=1}^{s} x^{k-k_i}$. Base x is called the *branching number*. If several branching rules are applied, then the largest branching number determines the worst-case time bound.

Our contributions and related work: The main subject of this paper is *the union of all minimal k-hitting sets*, denoted $U(k)$, in a given hypergraph. We highlight some properties of $U(k)$ that we will prove, and some conclusions:

(1) All $v \notin U(k)$ appear in exactly the same number of k-hitting sets.

(2) Any $u \in U(k)$ appears in more (typically *much* more) k-hitting sets than any $v \notin U(k)$.

(3) $U(k)$ has, roughly, at most k^r vertices, regardless of n. We conjecture that the worst-case upper bound is even better by a considerable factor depending on r. (More precise technical statements are given later.)

(4) $U(k)$ can be computed in $b^k poly(n)$ time, where *poly* is some fixed polynomial, and b is smaller than r and tends to $r-1$.

(5) In the light of our cause-effect inference problem, (2) means that vertices (causes) in $U(k)$ have by far the highest probabilities for being present.

(6) Statements (1), (3), and (4) together imply that, for any fixed rank, $U(k)$ serves as a polynomial-size kernel for hitting set counting problems like CHSwSV, in the same way as kernels for FPT optimization problems.

The paper is organized as follows. In Section 2 we establish the main properties of $U(k)$. Section 3 presents a relatively simple algorithm for Hitting Set Counting and CHSwSV that works in $b^k poly(n)$ time, where *poly* is some fixed polynomial, and b is smaller than r and tends to $r-1$. This generalizes the known result for Hitting Set [20] and also shows the advantage of a counting kernel. As for the size of $U(k)$, we proved earlier [9] that $|U(k)| < (r-1)k^r + k$ and gave simple examples of hypergraphs where $|U(k)| = \Theta(k^r)$, with a tiny factor $1/r^r$. An intriguing question is: How large can the constant factor in $\Theta(k^r)$ factor actually be, in the worst case? This problem remains open (except for $r = 2$ where the optimal factor is $\frac{1}{4}$ [9]), but we make further progress: In Section 4 we "stratify" the result for $r = 2$, in that we also take the relation between k and k^* into account: We prove the tight bound $|U(k)| \leq (k - k^* + 2)k^*$. In Section 5 we improve our earlier lower and upper bounds for general r. Using some hypergraph decomposition we show $|U(k)| \leq (1 + o(1))h$ if $h = \Theta(k^r)$, where h is here the number of hyperedges in an equivalent reduced hypergraph having the same family of minimal k-hitting sets as the given hypergraph. Due to an earlier result (also in [9]) stating that $h \leq k^r$, it follows $|U(k)| \leq (1 + o(1))k^r$ if $h = \Theta(k^r)$ (and if $h = o(k^r)$ then $|U(k)|$ is smaller anyhow). Note that any further improvements on h would further reduce the $|U(k)|$ bound immediately. We also derive stronger bounds for ranks 3 and 4. Section 6 concludes the paper and points out directions for further research.

Finally we review some related work. Hitting Set is NP-complete and even unlikely to be FPT with parameter k^* [4]. However, for any fixed rank r, Hitting Set Enumeration and hence Hitting Set Counting is in FPT [9]. Enumerating and counting small vertex covers is also addressed in [12, 5, 17]. Improved algorithmic and analysis techniques for Hitting Set in hypergraphs of rank r, also in the weighted case, were given in [13, 14], however they cannot be translated into counting algorithms, as several of the reduction rules do not apply here. Exact exponential counting algorithms for combinatorial objects also appeared in other literature (e.g., [8]), however not so much in the FPT framework. A complexity-theoretic framework for parameterized counting was given in [15]. Only recently, similar ideas for parameterized counting (of, e.g., hitting sets) by kernelization have been developed independently in [22], but our bounds beat the corresponding results. New exponential but non-parameterized counting algorithms for hitting sets and related objects (2SAT and 3SAT formulae) are given in [23]. Kernels of size $O(k^{r-1})$ are obtained in [1] for Hitting Set, but not for the counting problem.

To our best knowledge, previous work on $|U(k)|$ has considered the special case $r = 2$ and $k = k^*$ only: An upper bound for the union of minimum vertex covers, in relation to the size of a minimum vertex cover and maximum matching, is given in [3]. (Note that results in [3] are formulated in terms of stable sets, i.e., complements of vertex covers.) In the same paper, NP-hardness of computing $U(k^*)$ was proved. Hardness already for $r = 2$ adds further motivation to the question of the parameterized complexity of computing $U(k)$. The bound on $|U(k^*)|$ for $r = 2$ has been further improved in [7] (among many other results).

2 A Kernel for Counting Small Hitting Sets

In this section we establish the announced principal properties of $U(k)$, the union of all minimal k-hitting sets in a hypergraph.

A vertex v in a hitting set D is called *redundant* in D, if $D \setminus \{v\}$ is still a hitting set, otherwise v is *irredundant*. Note that $v \in D$ is irredundant if and only if v is the sole vertex of D in some hyperedge, and that a hitting set is minimal if and only if it has no redundant vertices.

Lemma 1 *In any k-hitting set D, all vertices of $D \setminus U(k)$ are redundant.*

Proof. D contains some minimal hitting set D', and $D' \subseteq U(k)$. □

Recall that we are interested in the the number of k-hitting sets containing any fixed vertex. For every j with $k^* \leq j \leq k$, let $s(j)$ be the number of different exact-j-hitting sets $D \subseteq U(k)$. For any vertex $v \in U(k)$ let $s_v(j)$ be the number of different exact-j-hitting sets $D \subseteq U(k)$ such that $v \in D$.

Lemma 2 *The number of different k-hitting sets containing vertex v equals*
$\sum_{j=k^*}^{k} s_v(j) \sum_{i=0}^{k-j} \binom{n-|U(k)|}{i}$ *if $v \in U(k)$, and*
$\sum_{j=k^*}^{k} s(j) \sum_{i=0}^{k-1-j} \binom{n-1-|U(k)|}{i}$ *if $v \notin U(k)$.*

Proof. Every k-hitting set extends some minimal k-hitting set $D \subseteq U(k)$ by further vertices, unless it is minimal itself. Since $|D| \geq k^*$, any k-hitting set shares at least k^* vertices with $U(k)$. In order to count all k-hitting sets containing some vertex v we just have to consider the different hitting sets $D \subseteq U(k)$ of each cardinality j, and add all possible combinations of at most $k - j$ vertices outside $U(k)$. If $v \in U(k)$ then these D must contain v. If

$v \notin U(k)$, we have to take all $D \subseteq U(k)$ of size j, but outside $U(k)$ one free vertex less remains. $\square$

Theorem 3 *All vertices $v \notin U(k)$ belong to exactly the same number of different k-hitting sets, and this number is properly smaller than the number of different k-hitting sets containing any fixed $u \in U(k)$.*

Proof. The first assertion follows from Lemma 2, as the expression for $v \notin U(k)$ is independent of v. We may prove the second assertion by comparison of terms in Lemma 2 too, however, an exchange argument using Lemma 1 gives more structural insight:

Fix any $u \in U(k)$ and $v \notin U(k)$. Consider any k-hitting set $D \ni v$. If also $u \in D$, let $D' = D$. If $u \notin D$, let $D' = (D \setminus \{v\}) \cup \{u\}$. Since v is redundant in D by Lemma 1, D' is a k-hitting set also in the latter case. It is easy to check that all $D' \ni u$ coming from the different $D \ni v$ are different.

It remains to show that k-hitting sets $C \ni u$ exist which are not among the sets D' defined above. In fact, whenever $v \notin C$ and u is irredundant in C, none of the D' equals C. Such C exist, by the following argument. Since $u \in U(k)$, some minimal k-hitting set $C \subseteq U(k)$ contains u. Note that $v \notin C$ and u is irredundant in C. $\square$

In the proof above it should be noticed that, if $|C| = j < k$, we can extend C by arbitrary combinations of $k - j$ vertices outside $U(k) \cup \{v\}$, to get *many* different k-hitting sets with u, distinct from all sets D'. Moreover, u can already belong to many different minimal k-hitting sets. That is, vertices in $U(k)$ are in general contained in *much* more k-hitting sets than vertices outside $U(k)$.

We turn to the questions of the size and computation of $U(k)$. Recall that we use h to denote the number of hyperedges. Theorem 7 in [9] can be rephrased as follows:

Theorem 4 *For any hypergraph G of rank r, and integer k, there exists a hypergraph G' of rank r such that the vertex set of G' is contained in that of G, G' has exactly the same minimal k-hitting sets as G, and all vertex degrees in G' are at most k^{r-1}. We can compute G' from G in $O(k^{r-1}h)$ time.* $\square$

It follows immediately:

Corollary 5 *For any hypergraph G of rank r, and integer k, it holds: If G has a k-hitting set then there exists a hypergraph G' of rank r such that the vertex set of G' is contained in that of G, G' has exactly the same minimal k-hitting sets as G, but G' contains at most k^r hyperedges which cover at most $(r-1)k^r + k$ vertices. Hence $|U(k)| \leq (r-1)k^r + k$ holds in G', and therefore also in G. We can compute G' from G in $O(k^{r-1}h)$ time.* $\square$

In Section 4 we will improve this "old" upper bound for $|U(k)|$ by almost a factor $r-1$. Next we show that $U(k)$ can be computed by fewer than nh applications of any FPT algorithm for Hitting Set. For example, some k-hitting can be found in $O(n+(r-1+\frac{1}{r-1})^k)$ time [20]. Based on this result we obtain:

Theorem 6 *$U(k)$ in a given hypergraph G can be computed in $O(k^{r-1}h + k^{2r}(r-1+\frac{1}{r-1})^k)$ time.*

Proof. The idea is simply to check for every vertex v whether the hypergraph without v has a $(k-1)$-hitting set. Due to Corollary 5, it suffices to do that for the $O(k^r)$ vertices v in the reduced hypergraph G' obtained in an $O(k^{r-1}h)$ time preprocessing. A trap is that we have to detect redundant vertices.

In more detail, for any vertex v and hyperedge $e \ni v$, let $G'(v,e)$ be the hypergraph obtained as follows. Remove from G' the hyperedges that contain v, and from the remaining hyperedges delete all vertices contained in e. (That is, replace each f with $f \setminus e$, and keep duplicate hyperedges.)

Now we formulate the algorithm. For every vertex v, check whether $G'(v,e)$ for some $e \ni v$ has a $(k-1)$-hitting set. If so, then take a minimal such set H and add v, which yields a minimal k-hitting set of G containing v. If the test fails for all $e \ni v$, clearly v cannot belong to any minimal k-hitting set.

To prove correctness of the first part we claim that, if H is a minimal $(k-1)$-hitting set in $G'(v,e)$, then $H \cup \{v\}$ is indeed a minimal k-hitting set in G'. This follows from three observations: (a) $H \cup \{v\}$ is obviously a k-hitting set. (b) Every vertex $w \in H$ is irredundant in $H \cup \{v\}$, because: w is irredundant in H, hence w is in some hyperedge $f \setminus e$ of $G'(v,e)$ that is free of other vertices from H. Since $e \cap H = \emptyset$, the whole f is free of other vertices from H. Finally, f does not contain v either (otherwise, $f \setminus e$ would not occur in $G(v,e)$). (c) v is irredundant in $H \cup \{v\}$ as well, because $e \cap H = \emptyset$.

To prove correctness of the second part, assume for contradiction that v is in some minimal k-hitting set $H \cup \{v\}$. Since v is irredundant, there must be some hyperedge $e \ni v$ with $e \cap H = \emptyset$. But then H is clearly a $(k-1)$-hitting set in $G'(v,e)$.

G' has $O(k^r)$ hyperedges, hence $O(k^{2r})$ pairs v, e are tested. Note that the number $O(k^r)$ of vertices in G' is dominated by $(r-1)^k$, as we consider r as a constant (although arbitrary) and k as the parameter. Now the time bound follows easily. $\Box$

Finally, the following algorithm Count solves CHSwSV, using (as "oracles") two algorithms that compute $U(k)$ and count the exact-j-hitting sets in a hypergraph. In Section 3 we will derive a time bound for Count, and hence for CHSwSV.

Count
Input: a hypergraph, an integer k.
Method:
(1) Delete all vertices outside $U(k)$ from all hyperedges.
(2) Count the exact-j-hitting sets in $U(k)$ for all $k^* \leq j \leq k$ in order to compute the $s(j)$.
(3) Do the following separately, i.e., from scratch, for each $v \in U(k)$.
(3.1) Delete the hyperedges that contain v, keeping the other vertices therein.
(3.2) Count the exact-$(j-1)$-hitting sets for all $k^* \leq j \leq k$ in order to compute the $s_v(j)$.
(3.3) Apply the suitable formula from Lemma 2.
(4) For $v \notin U(k)$, apply the suitable formula from Lemma 2 once.

Correctness is easy to establish, by the definition of $U(k)$ and Lemma 2. We stress again that the size $|U(k)|$ determines how many different values must be actually calculated.

3 Counting Small Hitting Sets Efficiently

A vertex u is said to *dominate* vertex v if all hyperedges containing u do also contain v. In algorithms for Hitting Set, dominated vertices v can be removed. In Hitting Set Counting however, this vertex domination rule does not apply, since we would lose the hitting sets containing v. Already for this reason we cannot simply extend the known algorithms to the counting problem, rather, we have to modify the rules themselves or introduce new rules.

In our approach, we first generalize the problem formulation of Hitting Set Counting. At first glance this formulation looks technical, but it will allow a simple yet powerful reduction rule. We consider hypergraphs with vertices labeled with *multiplicities*. A vertex v with multiplicity $m = m(v)$ represents m ordinary vertices that are *undistinguishable*, i.e., belong to exactly the same hyperedges. (Otherwise we call two vertices *distinguishable*.)

Accordingly, the size of a hyperedge H is now $\sum_{v \in H} m(v)$, and the rank is still the maximum size of hyperedges. Now a hitting set is defined as a vector t of nonnegative integers $t(v)$ (for each vertex v), with the following properties: $t(v) \leq m(v)$ for all v, and every edge contains a vertex v with $t(v) > 0$. The size of a hitting set t is $\sum_v t(v)$. The *multiplicity* of a hitting set t is defined as $\prod_v \binom{m(v)}{t(v)}$. The problem is to count, with multiplicities, all hitting sets of a given size k. Note that this is just the Hitting Set Counting problem rephrased.

Merging rule: If several vertices $v_1, \ldots, v_s$ belong to exactly the same hyperedges, replace them with one new vertex v such that $m(v) := m(v_1) + \cdots + m(v_s)$.

Whenever all possible mergings are done, then any two vertices are distinguishable. It follows that vertex domination defines a partial order relation on the vertices which is strict, i.e., no equivalent vertices exist.

We say that vertex v *belongs to hitting set* t if $t(v) \geq 1$. Our branching rules will decide for certain nodes v whether v shall belong to t or not, but, in the positive case, the exact $t(v) > 0$ is fixed later, and v is only *marked*. Vertices v with $t(v) = 0$ are simply removed from the hypergraph. After each application of the merging rule, the new vertex v is marked if some of the v_i was marked. For every newly marked vertex v, we immediately remove all hyperedges containing v (but not the vertices therein!). We do not mention these obvious steps anymore in our branching rules.

We call a branching rule *exact* if every solution (hitting set) appears in exactly one of the branches. Having only exact rules ensures that we can simply sum up the numbers of solutions represented by the leaves of the final search tree. Every leaf represents a hypergraph where solutions can be counted in polynomial time, either because the hypergraph is "simple", or because all remaining vertices v are marked. In the latter case, we have to take between 1 and $m(v)$ copies of each vertex v, and a simple auxiliary algorithm counts the exact-k-hitting sets:

Lemma 7 *The number of different ways to choose, from buckets of* $b_1, b_2, b_3, \ldots$ *elements, a total of* k *elements, taking at least one element from each bucket, can be computed in* $O(k^3)$ *time (i.e., arithmetic operations).*

Proof. Clearly, there can be at most k buckets, or the result is 0. Count the number of ways of choosing exactly i elements from the first j buckets, for $i, j \leq k$. In the step from $j-1$ to j we only have to multiply the results for $j-1$ buckets with suitable binomial coefficients $\binom{b_j}{l}$, $1 \leq l \leq b_j$, and sum up the results. Details of the dynamic programming and time analysis are straightforward. □

Now we state our branching rules.

Sequential rule: Take one hyperedge and index its vertices by $v_1, \ldots, v_s$. Branch as follows. Decide on the vertex v_j with the smallest index j that shall belong to the hitting set, and remove all v_i with $i < j$.

Small-hyperedge rule: Take one hyperedge with $s \leq r-1$ vertices and apply the sequential rule to it.

It is easy to see that the sequential rule, and hence the small-hyperedge rule, is an exact branching rule. Trivially, the small-hyperedge rule has branching number $r-1$ in the worst case. We first apply the merging rule and small-hyperedge rule as long as possible. Thus we can always assume in the following that all hyperedges have exactly r vertices, and any two vertices are distinguishable.

After these precautions we are ready to adapt, basically, the Hitting Set algorithm from [20] to Hitting Set Counting.

Theorem 8 Hitting Set Counting *in hypergraphs of rank r, given as a list of h hyperedges, can be solved in $O(k^{r-1}h + rk^{r+1}(r-1+\frac{1}{r-1})^k)$ time.*

Proof. In every branching step of the algorithm, consider some hyperedge H. We denote the vertices of H by $v_1, \dots, v_r$ so that v_1 is not dominated by any other $v_j \in H$ with $j > 1$. Since any two vertices are distinguishable, such v_1 exists: It suffices to choose v_1 as any vertex in H with maximum degree. The other vertices in H are ordered arbitrarily.

A hyperedges with fewer that r vertices is called a small hyperedge. Borrowing a notation from [13, 14], let $T^i(k)$ be the number of leaves of the search tree if the parameter value (number of vertices yet to choose) is k, and at least i small hyperedges exist. We apply the sequential rule to H. Thus, for every $j \geq 2$, the jth branch where v_j is chosen produces a small hyperedge H', as v_1 is removed from some hyperedge that contains v_1 but not v_j. This gives $T^0(k) \leq T^0(k-1) + (r-1)T^1(k-1)$. Then we apply the sequential rule (small-hyperedge rule) again to H'. Thus we also get $T^1(k) \leq T^0(k-1) + (r-2)T^1(k-1)$. This system of recurrences was already analyzed in [20], the branching number is:

$$r - 1 + \frac{2}{\sqrt{(r-1)^2+4} + r - 1} < r - 1 + \frac{2}{\sqrt{(r-1)^2} + r - 1} = r - 1 + \frac{1}{r-1}.$$

Next we discuss the polynomial factor. Cardinalities of all hyperedges and degrees of all vertices are determined once in the beginning in $O(rh)$ time. All removal operations of vertices and edges, and updates of hyperedge sizes and vertex degrees, need together $O(rh)$ time on every path of the search tree.

The equivalence classes of undistinguishable vertices for the merging rule are determined in $O(rh)$ time: For this purpose we rebuild the hypergraph incrementally, inserting hyperedges one-by-one. In the initial empty hypergraph, all vertices are equivalent. When a new edge H is reinserted, it splits only equivalence classes that intersect H. Thus, we only have to put vertices of H into new equivalence classes. Using an array of vertices where names of equivalence classes are stored, this takes $O(r)$ time per hyperedge.

We need to do merging only immediately before a branching rule is applied. At any such moment we compute the equivalence classes from scratch in $O(rh)$ time. Since every branching step reduces the parameter, the merging steps take in total $O(krh)$ time, on every path of the search tree. Prior to every application of the sequential rule to a hyperedge H we identify some vertex $v_1 \in H$ with maximum degree (see above) in $O(r)$ time. In every leaf of the search tree we compute the number of hitting sets in the remaining hypergraph (with all vertices marked) from the vertex multiplicities in $O(k^3)$ time, using Lemma 7. This time is dominated by $O(krh)$.

In total, the polynomial factor is $O(krh)$, which first gives a time bound $O(krh(r-1+\frac{1}{r-1})^k)$. Finally, we use this result but avoid multiplying h with the exponential term, by kernelization. From the given hypergraph G we first compute in $O(k^{r-1}h$ time the hypergraph G' specified in Corollary 3. G' includes $U(k)$, has $O(k^r)$ hyperedges and the same minimal k-hitting sets as G. It easily follows that the exact-k-hitting sets of G are obtained from the exact-j-hitting sets of G' (for any $j \leq k$) by adding arbitrary combinations of $k-j$ vertices outside G'. Thus, for $j = 1, \ldots, k$ we count the exact-j-hyperedges in G' in $O(jrk^r(r-1+\frac{1}{r-1})^j)$ time, multiply the counts by binomial coefficients and sum up the results. Summation of all time bounds yields the asserted complexity. $\Box$

It might be helpful to repeat the overall structure of the branching procedure, without details. Initially it is called for the given G and $l := k$, and finally the results returned by all leaves are added.

```
(1) If there are still hyperedges:
(1.1) If $l = 0$ then return 0 (no solution exists).
(1.2) Find and merge all undistinguishable vertices.
(1.3) If a small hyperedge exists, then take it as $H$, else take the first hyperedge in the list as $H$.
(1.4) Apply the sequential rule to $H$. Branch.
(1.5) In every branch, remove all hyperedges containing the newly chosen vertex, and let $l := l - 1$.
(2) If no hyperedges are left, return the number of $k$-hitting sets computed as in Lemma 7.
```

Together with Theorem 6 this also yields a time bound for algorithm Count in Section 2. Remember that we need to run a counting algorithm k times for only $|U(k)|+1$ distinct vertices v, and the binomial coefficients in Lemma 2 can be precomputed. Thus we have:

Corollary 9 CHSwSV *is solvable in* $O(k^{r-1}h + (k^{2r} + r|U(k)|k^{r+2})(r-1+\frac{1}{r-1})^k)$ *time.* $\Box$

4 The Union of Minimal Vertex Covers of Bounded Size

For a subset X of vertices in a graph, $N(X)$ denotes the set of all vertices with a neighbor in X. If X is independent then $N(X) \cap X = \emptyset$. The following simple lemma holds for any minimal (not necessarily minimum!) vertex cover.

Lemma 10 *Let C be a fixed minimal vertex cover. Let D be any other minimal vertex cover, and $I := C \setminus D$. Then $D = (C \setminus I) \cup N(I)$. Consequently, D is uniquely determined by I.*

Proof. For any edge $e = uv$ incident to some $u \in I$, vertex cover D must contain v. It follows $(C \setminus I) \cup N(I) \subseteq D$. But since the left-hand side is a vertex cover and D is a minimal vertex cover, the inclusion is actually a set equation. $\Box$

It follows $|U(k)| \leq (k+1)k^*$: Take some vertex cover C with k^* vertices, and observe that $|N(v)| \leq k$ for each $v \in C$ that appears in some $I = C \setminus D$, where D is a minimal vertex cover with $|D| \leq k$. Below we will improve this size bound, but already now we can nicely limit the complexity of computing $U(k)$. The currently best time bound for finding a minimum vertex cover is taken from [6].

Corollary 11 *$U(k)$ in a graph with m edges can be computed in time $O(km + k^4 1.2738^k)$ or $O(k^* m 2^{k^*})$.*

Proof. The first bound comes from the method in Theorem 6. For the second bound we use a simple consequence of Lemma 10: There exist at most 2^{k^*} mimimal vertex covers. By trying all sets $I \subseteq C$ we can determine them in polynomial time each, and then form their union. $\Box$

Depending on the relation of k^* and k, either of the two algorithms can be the faster one. Similarly, we can use Lemma 10 for enumeration: In [9] we computed a repetition-free concise description (suitably defined) of all minimal k-vertex covers in $O^*(1.74^k)$ time. (A more "dirty" description that tolerates redundant vertex covers is obtained much easier in $O(1.62^k)$ time.) But we can do better if $2^{k^*} < 1.74^k$, that is, if $k > 1.25k^*$: Compute some minimum vertex cover C, and test for all independent sets $I \subseteq C$ whether $(C \setminus I) \cup N(I)$ is a minimal k-vertex cover. (For every I, the time for this test is polynomial in k, as it suffices to consider vertices of degree at most k.) However, for $k < 1.25k^*$ the concise enumeration is still more efficient.

The rest of this section deals with the size $|U(k)|$. The following result absorbs Theorem 3 from [9] as a special case.

Theorem 12 *$|U(k)| \leq (k - k^* + 2)k^*$ in graphs, and this bound is tight.*

Proof. To establish the lower bound, consider the disjoint union of k^* stars, each with a central vertex connected to $x+1$ leaves, where $x = k - k^*$. The centers build a minimum vertex cover, any $k^* - 1$ centers together with the $k - k^* + 1$ leaves of the remaining star build a minimal vertex cover, hence $U(k)$ is the whole graph, which has $(x+2)k^* = (k - k^* + 2)k^*$ vertices.

We are going to prove the upper bound. Let C be some fixed vertex cover of size k^*. By Lemma 10, any other minimal vertex cover D (of any size) has the form $D = (C \setminus I) \cup N(I)$. Since $I = C \setminus D$ is in the complement of a vertex cover, I is an independent set, hence $I \cap N(I) = \emptyset$. Conversely, each independent set $I \subseteq C$ yields a vertex cover $D = (C \setminus I) \cup N(I)$. Since C has minimum size, $|N(I) \setminus C| \geq |I|$ holds for every independent set $I \subseteq C$. For making $|D| \leq k = k^* + x$ true, it must be $|N(I) \setminus C| \leq |I| + x$.

Due to these necessary conditions, we call an independent set $I \subseteq C$ a *replacement set* if $|N(I) \setminus C| \leq |I| + x$ *and* $(C \setminus I) \cup N(I)$ is actually a minimal vertex cover, in particular, no vertex from $C \setminus I$ can be removed without uncovering some edge. Now it suffices to prove the following

Claim: The union of the $N(I) \setminus C$ of all replacement sets I has at most $(x+1)k^*$ vertices.

Let $I_1, I_2, I_3, \ldots, I_l$ be a non-extendible sequence of replacement sets such that $I_{t+1} \not\subseteq \bigcup_{j=1}^{t} I_j$ for each t, $0 < t < l$. Here, non-extendible means that no further replacement set fulfilling the condition can be added. It suffices to prove the Claim for replacement sets in this sequence, as the $N(I) \setminus C$ for further replacement sets I cannot contribute more vertices to the union. Define $\Delta_t := \bigcup_{j=1}^{t} N(I_j) \setminus C$. We shall prove that $|\Delta_t| \leq |\bigcup_{j=1}^{t} I_j| + xt$. Since our sequence can consist of at most k^* replacement sets, this would imply the Claim and finish the proof.

We apply induction on t. Induction base $t = 1$ is true by the definition of replacement sets. Suppose that our induction hypothesis holds for some t. The induction step has to show $|\Delta_t \cup (N(I_{t+1}) \setminus C)| \leq |(\bigcup_{j=1}^{t} I_j) \cup I_{t+1}| + x(t+1)$.

Since $I' := (\bigcup_{j=1}^{t} I_j) \cap I_{t+1}$ is contained in a replacement set, I' is an independent set, thus $|N(I') \setminus C| \geq |I'|$. Furthermore, note that for any vertex sets $A, B \subseteq C$ in a graph the trivial relation $N(A \cap B) \setminus C \subseteq N(A) \cap N(B) \setminus C$ holds. In particular, $N(I') \setminus C \subseteq \Delta_t \cap N(I_{t+1}) \setminus C$. For the cardinalities we get

$$\left|(\bigcup_{j=1}^{t} I_j) \cap I_{t+1}\right| = |I'| \leq |N(I') \setminus C| \leq |\Delta_t \cap N(I_{t+1}) \setminus C|.$$

Since $|\Delta_t| \leq |\bigcup_{j=1}^{t} I_j| + xt$ by the induction hypothesis for t, and $|N(I_{t+1}) \setminus C| \leq |I_{t+1}| + x$ (replacement set), the induction hypothesis for $t+1$ follows:

$$|\Delta_t \cup (N(I_{t+1}) \setminus C)| = |\Delta_t| + |N(I_{t+1}) \setminus C| - |\Delta_t \cap (N(I_{t+1}) \setminus C)|$$

$$\leq |\bigcup_{j=1}^{t} I_j| + xt + |I_{t+1}| + x - \left|(\bigcup_{j=1}^{t} I_j) \cap I_{t+1}\right| = \left|(\bigcup_{j=1}^{t} I_j) \cup I_{t+1}\right| + x(t+1).$$

□

5 The Union of Minimal Hitting Sets of Bounded Size in Hypergraphs of Bounded Rank

In this purely combinatorial section we will much improve the constant factor $r-1$ from Corollary 3, for any fixed r: Our final result is $|U(k)| \leq (1+o(1))k^r$, where $o(1)$ is meant to tend to 0 as k grows. Our proof is independent of earlier results, as we will actually show $|U(k)| \leq (1+o(1))h$ if $h = \Theta(k^r)$. Any smaller bound, say $h \leq ak^r$ ($a < 1$ constant), on the number of hyperedges in an "equivalent reduced" hypergraph G' would therefore immediately improve $|U(k)|$ further. However, we easily see that $h \leq k^r$ in Corollary 3 is a tight bound: Take r disjoint sets of k vertices and choose one vertex from each set, in all possible ways. This hypergraph has obviously k^r hyperedges. On the other hand, it has only kr vertices, each of degree k^{r-1}. There might be a chance for smaller bounds on h provided that $|U(k)| = \Theta(k^r)$, but this question must be left for further research. The worst example for $|U(k)|$ we could come up with is:

Proposition 13 *There exist hypergraphs where* $|U(k)| \approx \frac{(r-1)^{r-1}}{r!r^{r-1}}k^r \approx \frac{1}{er!}k^r$.

Proof. Take a set C of roughly $\frac{r-1}{r}k$ vertices and create $\frac{1}{r}k$ hyperedges for every $D \subset C$ with $|D| = r-1$, by adding $\frac{1}{r}k$ different single vertices to D. In fact, each vertex of this hypergraph of rank r is in some minimal k-hitting set. $\Box$

Note that almost all vertices in these examples in Proposition 13 have degree 1, and hence $|U(k)| \approx h$. For proving that $|U(k)| \leq (1+o(1))h$ if $h = \Theta(k^r)$, we need a few technical preparations. The proof of the following lemma is straightforward.

Lemma 14 *Let H be a minimal hitting set in a hypergraph G. Consider a partitioning of the family of hyperedges of G into s subfamilies. There exist sets H_i such that $H = H_1 \cup \ldots \cup H_s$, and H_i is a minimal hitting set of the ith subfamily, for $i = 1, \ldots, s$.* $\Box$

We will use the decomposition lemma as follows to bound $|U(k)|$. By definition, $U(k)$ is the union of minimal k-hitting sets. By Lemma 14, each of them is a union of minimal k-hitting sets of subhypergraphs. Hence, it suffices to bound the $|U(k)|$ there and to add the results. Since Lemma 14 holds for arbitrary decompositions, we can be flexible and decompose the given hypergraph so that the $|U(k)|$ in the subhypergraphs are small enough and easy to bound. Below we apply the principle in a particular way, using the next lemma, but further research may find more clever decompositions leading to improved results.

Lemma 15 *In any hypergraph with h hyperedges one can color the vertices black and white so that at most h vertices belong to unicolored hyperedges.*

Proof. We color the vertices independently black and white with probability 1/2. For any hyperedge let X be the random variable with $X = t$ if the hyperedge is unicolored, and $X = 0$ else. For $|X| = t$ we clearly have $E[X] = t/2^{t-1} \leq 1$. Thus, by linearity of expectation, the expected number of vertices in all unicolored hyperedges is at most h, hence there exists a particular coloring with the desired property. □

Now we have the necessary tools to prove the announced result, where the $o(1)$ term refers to the asymptotics for growing k. Notice that this bound clearly improves on our previous one [9] by a factor $r - 1$.

Theorem 16 *In hypergraphs of rank r with $h = \Theta(k^r)$ hyperedges, it holds that $|U(k)| \leq (1 + o(1))h$. Consequently, $|U(k)| \leq (1 + o(1))k^r$ in any hypergraph with rank r.*

Proof. Given a hypergraph G and an integer k, let d be some integer threshold that we specify later. We call a vertex of degree smaller than d a *d-thin* vertex, otherwise it is *d-fat*. From G we construct a *diminished hypergraph* G' as follows. Using Lemma 16, we color the d-thin vertices in G black and white so that at most h of the d-thin vertices belong to unicolored hyperedges. Then we delete the black d-thin vertices to obtain G'. Finally we remove emptied hyperedges and identical copies of hyperedges.

We call a hyperedge *good* if it has lost at least one d-thin vertex from G but also retained at least one in G'. Otherwise the hyperedge is called *bad*. We apply Lemma 14 to the decomposition of G into good and bad hyperedges. Even simpler, we do not attempt to estimate the union of minimal k-hitting sets of the bad hyperedges. We only notice that at most h of the d-thin vertices of G belong to bad hyperedges (by construction), and at most rh/d vertices in the whole G are d-fat (since G has h hyperedges and rank r). It remains to bound the number of d-thin vertices that belong to minimal k-hitting sets of the good hyperedges.

Let G_0 and G'_0 be the hypergraph of all good hyperedges before and after diminishing, respectively. Thus G'_0 has rank $r - 1$. Consider any minimal k-hitting set H of G_0. We construct some minimal hitting set H' of G'_0 as follows. We start from $H' := H$. Every d-thin vertex $v \in H$ that has been deleted may leave fewer than d hyperedges of G'_0 without any vertex from H'. We replace every such v with other d-thin vertices, adding at most one from each "abandoned" hyperedge to H'. Such d-thin vertices exist, since G'_0 consists entirely of good hyperedges. Currently H' has fewer than dk vertices. However, adding new vertices can have made other vertices of H' redundant, so that H' is not minimal. Next, we successively remove redundant vertices arbitrarily, until H' is a minimal dk-hitting set of G'_0. From $|U(k)| = O(k^r)$ it follows that all such H' are contained in a fixed set of size $O((dk)^{r-1})$.

In order to bound $|U(k)|$ in G_0 we also need to count the vertices v in all difference sets $H \setminus H'$. Every such v was removed from H' since either (1) v did not belong to G_0' or (2) v became redundant. In both cases we can assign to v some d-thin vertex $w \in H'$ such that some hyperedge of G_0 contains both v and w. This is easily seen: In case (1), v was not redundant in H, since H was a minimal hitting set of G_0. That means, after the deletion step at least one hyperedge incident with v was without vertices of H'. We had to insert another d-thin w from this hyperedge in H', and at least one such w had to stay in the final H'. In case (2) we argue similarly. Since $v \in H$ was not redundant in H, it became redundant only later when some d-thin vertices w had been added to H', in some hyperedge that was previously hit by v only. At least one such w had to stay in the final H', otherwise H' would no longer hit this hyperedge.

Since the vertices w are d-thin and belong to sets H', they are all contained in a fixed set of size $O((dk)^{r-1})$ as shown above. Furthermore, fewer than rd vertices v can be assigned to each w, as they have to be in some hyperedge with w. It follows that all $H \setminus H'$ are contained in a fixed set of size $O(rd(dk)^{r-1})$.

Summing up the bounds we get $|U(k)| \leq h + rh/d + O(rd^r k^r)/k$ in G. For $h = \Theta(k^r)$ this yields $|U(k)| \leq (1+r/d+O(rd^r)/k)h$. Setting the free parameter d depending on k such that $d \to \infty$ but d grows slower than $k^{1/r}$, we can make the last two terms $o(1)$ simultaneously. This gives $|U(k)| \leq (1 + o(1))h$ as k grows.

The explicit bound $|U(k)| \leq (1 + o(1))k^r$ follows from this relation and Corollary 3: Consider a reduced hypergraph having $h \leq k^r$ hyperedges and the same $U(k)$. If $h > \frac{1}{r}k^r = \Theta(k^r)$, we apply the previous result. If h is smaller, we just apply the trivial relation $|U(k)| \leq rh$. □

Let us define $f(r)$ to be the smallest factor with $|U(k)| \leq f(r)k^r + o(k^r)$. Trivially we have $f(1) = 1$, and in [9] we got $f(2) = \frac{1}{4}$. Theorem 16 says $f(r) \leq 1$ for each r. In the remainder of this section we prove better upper bounds for $r = 3$ and $r = 4$, by using Lemma 14 in a different way.

Let H be a fixed minimum hitting set in a hypergraph G of rank r. Unlike case $r = 2$, we call $I \subseteq H$ a replacement set if there exists a minimal hitting set H' with $I = H \setminus H'$ and $|H'| \leq k$. Let I be any fixed replacement set. We define a hypergraph $G(I)$ whose hyperedges are the sets $e \setminus I$, for all hyperedges e in G with $\emptyset \neq e \cap H \subseteq I$. Vertices in a hypergraph are, without loss of generality, the vertices contained in its hyperedges.

We decompose $G(I)$ into several hypergraphs G_J as follows. G_J with rank $r - j$ ($j = |J|$) consists of those hyperedges $e \setminus J$ of $G(I)$ with $e \cap H = J$. (Since G_J is the same for each $I \supseteq J$, we can omit subscript I.)

Lemma 17 *For any fixed I, any minimal hitting set H' with $H \setminus H' = I$ contains vertices only from H and from minimal hitting sets of the G_J, $J \subseteq I$, $0 < |J| < r$.*

Proof. If H' with $H \setminus H' = I$ is a minimal hitting set in G, then H' is the union of $H \setminus I$ and some minimal hitting set of $G(I)$. Since every hyperedge of $G(I)$ belongs to some G_J, Lemma 14 yields the assertion. □

Theorem 18 *$|U(k)| \leq \frac{1}{4}k^*(k^2 + (k^*)^2) + o(k^3)$ in hypergraphs of rank 3.*

Proof. Let H be any minimum hitting set H, thus $k^* = |H|$. For each $v \in H$ let I_v be some maximum replacement set with $v \in I_v$, and $x_v = |I_v|$. (We can assume that I_v exists, since a vertex of H in no replacement set belongs to every minimal hitting set, and putting these vertices aside we get a reduced instance with smaller k and k^*.) We define $x = \max_v x_v$. For each v we distinguish $x_v - 1$ two-vertex sets J with $v \in J \subseteq I_v$. By Lemma 17, all vertices of $U(k)$ are in H or in minimal hitting sets of the G_J, $1 \leq |J| \leq 2$.

The G_J with $|J| = 1$ contribute together at most $\frac{1}{4}\sum_{v\in H}(k-k^*+x_v)^2+o(k^3)$ vertices to $U(k)$. This is true because the G_J have rank 2, at most $k - k^* + x_v$ vertices outside H are in every minimal hitting set of G_J, $J = \{v\}$, and $f(2) = \frac{1}{4}$.

The G_J with $|J| = 2$ have rank 1. Thus, every vertex in G_J, $|J| = 2$, $J \subseteq I$ (any replacement set) must be in every hitting set that extends $H \setminus I$, limiting the *total* number of vertices in all these G_J, $J \subseteq I$, to $k - k^* + |I|$.

We apply this observation in two ways: All G_J of distinguished sets J, $|J| = 2$, contribute together at most $k^*(k - k^* + x) \leq k^*k$ vertices to $U(k)$. Each of the remaining G_J with $|J| = 2$, these are fewer than $\frac{1}{2}((k^*)^2 - \sum_{v\in H} x_v)$ pairs, contributes at most $k - k^* + x$ vertices to $U(k)$. Altogether we obtain

$$|U(k)| \leq \frac{1}{4}\sum_{v\in H}(k - k^* + x_v)^2 + \frac{1}{2}((k^*)^2 - \sum_{v\in H} x_v)(k - k^* + x) + o(k^3).$$

After rewriting $(k^*)^2 = \sum_{v\in H} k^*$, algebraic manipulation easily yields

$$|U(k)| \leq \frac{1}{4}k^*(k^2 - (k^*)^2 + 2k^*x)) + \frac{1}{4}\sum_{v\in H} x_v(x_v - 2x) + o(k^3).$$

Since the middle term is negative, and $x \leq k^*$, we get the claimed result. $\Box$

We believe that this is not yet optimal. Note especially that the optimal bound for $r = 2$ due to Theorem 12 is linear in $k - k^*$ for any fixed k^*. An intriguing question is whether a similar bound with factor $k - k^*$ in the main term holds also for $r = 3$ (whereas the result in Theorem 18 is always cubic). This question is interesting for limits k close to k^*.

Corollary 19 *Let $f(r)$ be the smallest factor with $|U(k)| \leq f(r)k^r + o(k^r)$. Then $f(3) \leq \frac{1}{2}$ and $f(4) \leq \frac{19}{24}$.*

Proof. Consider $r = 3$. For any fixed k, our bound from Theorem 18 is maximized when $k^* = k$, and then it becomes $\frac{1}{2}k^3 + o(k^3)$, hence $f(3) \leq \frac{1}{2}$. Next, Lemma 17 implies

$$|U(k)| \leq k + \sum_{j=1}^{r-1}\binom{k}{j}f(r-j)k^{r-j} \leq k + \sum_{j=1}^{r-1}\frac{f(r-j)}{j!}k^r.$$

Neglect of the lower-order term k gives the recursion $f(r) \leq \sum_{j=1}^{r-1} \frac{f(r-j)}{j!}$. For $r = 4$ it gives the mentioned upper bound. (Unfortunately, the recursive formula grows exponentially in r. Already for $r = 5$ the result exceeds 1.) $\square$

6 Conclusions

The union $U(k)$ of minimal k-hitting sets in a hypergraph is useful in combinatorial inference, if sets of non-interfering causes shall be concluded from observed binary data. In a certain sense, $U(k)$ contains the most likely hypotheses. We have $|U(k)| = O(k^r)$ in hypergraphs of any constant rank r, but the factor depending on r is open. We have significantly improved our previous upper bounds [9], using some intricate hypergraph decompositions. We believe that the techniques introduced here are more powerful than what the current results exhibit, so that further research should be able to narrow the gap between $\frac{1}{er!}k^r$ and $(1 + o(1))k^r$. In particular, bounds on the number of hyperedges in equivalent reduced hypergraphs would immediately improve the bounds on $|U(k)|$ further. Possible ideas for improvements are to delete certain hyperedges without changing $U(k)$, or to use refined decompositions.

$U(k)$ is also a problem kernel for several hitting set counting and enumeration problems. It may be possible to make the kernel even smaller than $U(k)$, if the "influence" of certain vertices on the counts can be figured out in polynomial time.

We presented a relatively simple Hitting Set Counting algorithm with branching number below $r - 1 + \frac{1}{r-1}$. With much more elaborated branching rules and combinatorics we have recently brought the branching number even closer to $r - 1$ and got worthwhile improvements for ranks $r \geq 6$, but this would go beyond the scope of this paper. The currently fastest counting algorithm for $r = 2$, with branching number about 1.38, uses tree decomposition [17], and it seems that iterative compression can lift this result to higher r, but from some r on our approach becomes superior. It would be interesting to combine the various techniques to get even better branching numbers for small ranks r.

Inferring hitting sets is a special case of logical abduction, when all clauses in the theory consist of positive literals only. A complexity classification of abduction problems was given in [21] (cf. the paper for further background), including hardness results for most of the other special cases. This motivates parameterization by the model size, and similar methods as here may be fruitful.

Acknowledgments

This work has been initiated and partially supported by the *Combinatorial Search Algorithms in Bioinformatics* group at the University of Bielefeld, led by Ferdinando Cicalese, through his Sofja Kovalevskaja Award 2004 from the

Alexander von Humboldt Foundation. Partial support also came from the Swedish Research Council (Vetenskapsrådet), grant no. 2007-6437, "Combinatorial inference algorithms – parameterization and clustering". Technical comments of the anonymous referees helped to streamline the presentation.

References

[1] F.N. Abu-Khzam. Kernelization algorithms for d-hitting set problems, 10th Int. Workshop on Algorithms and Data Structures WADS 2007, *LNCS 4619*, 434-445

[2] P. Alves, R.J. Arnold, M.V. Novotny, P. Radivojac, J.P. Reilly, H. Tang. Advancement in protein inference from shotgun proteomics using peptide detectability, *12th Pacific Symposium on Biocomputing 2007*, 409-422

[3] E. Boros, M.C. Golumbic, V.E. Levit. On the number of vertices belonging to all maximum stable sets of a graph, *Discrete Applied Mathematics* 124 (2002), 17-25

[4] J. Chen, X. Huang, I.A. Kanj, G. Xia. Strong computational lower bounds via parameterized complexity, *Journal of Computer and System Sciences* 72 (2006), 1346-1367

[5] J. Chen, I.A. Kanj, J. Meng, G. Xia, F. Zhang. On the effective enumerability of NP problems, 2nd Int. Workshop on Parameterized and Exact Computation IWPEC 2006, *LNCS* 4169, 215-226

[6] J. Chen, I.A. Kanj, G. Xia. Simplicity is beauty: Improved upper bounds for vertex cover, Technical Report, 2008

[7] M. Chlebik, J. Chlebikova. Crown reductions for the minimum weighted vertex cover problem, *Discrete Applied Mathematics* 156 (2008), 292-312

[8] V. Dahllöf, P. Jonsson and M. Wahlström. Counting models for 2SAT and 3SAT formulae, *Theoretical Computer Science* 332 (2005), 265-291

[9] P. Damaschke. Parameterized enumeration, transversals, and imperfect phylogeny reconstruction, *Theoretical Computer Science* 351 (2006), 337-350, special issue of selected papers from *IWPEC 2004*

[10] P. Damaschke. The union of minimal hitting sets: Parameterized combinatorial bounds and counting, *24th Symposium on Theoretical Aspects of Computer Science STACS 2007*, *LNCS* 4393, 332-343

[11] R.G. Downey, M.R. Fellows. *Parameterized Complexity*, Springer, 1999

[12] H. Fernau. On parameterized enumeration, *8th Int. Computing and Combinatorics Conference COCOON 2002*, *LNCS* 2387, 564-573

[13] H. Fernau. A top-down approach to search-trees: Improved algorithmics for 3-hitting set, ECCC Report 073 (2004)

[14] H. Fernau. Parameterized algorithms for hitting set: The weighted case, *6th Italian Conference on Algorithms and Complexity CIAC 2006*, *LNCS* 3998, 332-343

[15] J. Flum, M. Grohe. The parameterized complexity of counting problems, *SIAM Journal on Computing* 33 (2004), 892-922

[16] T. Mitchell. *Machine Learning*, McGraw-Hill 1997

[17] D. Mölle, S. Richter, P. Rossmanith. Enumerate and expand: Improved algorithms for connected vertex cover and tree cover, *Theory of Computing Systems* 43 (2008), 234-253

[18] A.I. Nesvizhskii, R. Aebersold. Interpretation of shotgun proteomic data: The protein inference problem, *Molecular and Cellular Proteomics* 4 (2005), 1419-1440

[19] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*, Oxford Lecture Series in Mathematics and Its Applications, Oxford University Press 2006

[20] R. Niedermeier, P. Rossmanith. An efficient fixed-parameter algorithm for 3-hitting set, *Journal of Discrete Algorithms* 1 (2003), 89-102

[21] G. Nordh, B. Zanuttini. Propositional abduction is almost always hard, *19th Int. Joint Conference on Artificial Intelligence IJCAI 2005*, 534-539

[22] M. Thurley. Tractability and intractability of parameterized counting problems, Master's thesis, Institute for Computer Science, Humboldt-Univ. Berlin 2006, also in: *Master-/Diploma Theses Series of ECCC*

[23] M. Wahlström. Algorithms, measures, and upper bounds for satisfiability and related problems, PhD Thesis 1079, Linköping Studies in Science and Technology (2007)