Generating Minimal Redundant and Maximal Irredundant Sets in Incidence Graphs

Emanuel Castelo
Aix-Marseille Université, CNRS, LIS, 13009 Marseille, France

Jérémie Chalopin
Aix-Marseille Université, CNRS, LIS, 13009 Marseille, France

Oscar Defrain
Aix-Marseille Université, CNRS, LIS, 13009 Marseille, France

Simon Vilmin
Aix-Marseille Université, CNRS, LIS, 13009 Marseille, France

Abstract

It has been proved by Boros and Makino that there is no output-polynomial-time algorithm enumerating the minimal redundant sets or the maximal irredundant sets of a hypergraph, unless $\mathbf{P} = \mathbf{NP}$. The same question was left open for graphs, with only a few tractable cases known to date. In this paper, we focus on graph classes that capture incidence relations such as bipartite, co-bipartite, and split graphs, motivated by their strong relation with hypergraphs. Concerning maximal irredundant sets, we show that the problem on co-bipartite graphs is as hard as in general graphs and tractable in split and strongly orderable graphs, the latter being a generalization of chordal bipartite graphs. As for minimal redundant sets enumeration, we first show that the problem is intractable in split and co-bipartite graphs, answering the aforementioned open question. Then, we show that it is tractable on (C_3, C_5, C_6, C_8)-free graphs, a class of graphs incomparable to strongly orderable graphs, and which also generalizes chordal bipartite graphs. Our positive results rely on the structural properties of these graph classes and thus cannot be easily extended to bipartite graphs, for which the question remains open for both problems.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Theory of computation → Design and analysis of algorithms

Keywords and phrases Enumeration algorithms, maximal irredundant sets, minimal redundant sets, incidence graphs

Digital Object Identifier 10.4230/LIPIcs.MFCS.2026.46

Related Version *Full Version*: `https://arxiv.org/abs/2602.18362`

Funding *Emanuel Castelo*: Supported by ANR project PARADUAL (ANR-24-CE48-0610-01).
Jérémie Chalopin: Supported by ANR project MIMETIQUE (ANR-25-CE48-4089-01).
Oscar Defrain: Supported by ANR project PARADUAL (ANR-24-CE48-0610-01).
Simon Vilmin: Supported by ANR project PARADUAL (ANR-24-CE48-0610-01).

1 Introduction

Irredundant sets in hypergraphs are those sets of vertices I for which every element $x \in I$ has a private edge, i.e., an edge whose intersection with I is exactly $\{x\}$. Equivalently, they are often presented as sets of vertices that are minimal with respect to the set of hyperedges they intersect. Sets of vertices that are not irredundant are called *redundant*.

In graphs, *irredundant sets* are defined as the sets of vertices having private neighbors, that is, neighbors that are not adjacent to other vertices in the set, allowing a vertex to be its own private neighbor. *Redundant sets* are defined analogously as the subsets of vertices that are not irredundant. Redundant and irredundant sets of a graph should not be confused with

the redundant and irredundant sets of the underlying hypergraph where each edge defines a hyperedge. Rather, they correspond to the redundant and irredundant sets of the hypergraph of closed-neighborhoods of the graph, in an analogue of what is domination to transversality; we refer to Section 2 for a more in-depth presentation of these notions together with some examples.

Irredundancy in graphs was first introduced by Cockayne, Hedetniemi, and Miller in [13] for its links with independence and domination, where it was in particular noted that every maximal independent set is a minimal dominating set, which in turn is a maximal irredundant set. Due to its relation to independence and domination, irredundancy in graphs has been extensively studied [3, 7, 8, 12, 20, 27, 29]. Let us mention that computing the minimum or maximum size of a maximal irredundant set is known to be **NP**-complete [21, 28, 33].

The generalization of irredundancy to hypergraphs has been considered for algorithmic purposes in the contexts of minimal domination or transversality [1, 32], for it is known that minimal dominating sets (resp. minimal tranversals) are those dominating sets (resp. tranversals) which are irredundant.

Despite the interest for irredundant sets, little is known regarding their enumeration and that of their dual, redundant sets. Here the goal is, given a graph or a hypergraph, to list without repetition all its maximal irredundant sets or minimal redundant sets. In the following, let us denote by GraphMirr·Enum, GraphMred·Enum, HypMirr·Enum, and HypMred·Enum these problems (see Section 2 for the formal definition). Since the number of minimal redundant or maximal irredundant sets can be exponential in the size of the input (hyper)graph, evaluating the complexity of an enumeration algorithm solving either of these problems using the input size only is not a relevant efficiency measure. Instead algorithms are analyzed using *output-sensitive complexity* that estimates the running time of an algorithm in terms of both its input and output sizes. An algorithm runs in *output-polynomial time* if its execution time is polynomially bounded by the combined sizes of the input and the output. This notion can be further detailed to analyze the delay between two solutions. Namely, we say that an algorithm runs in *incremental-polynomial time* if it outputs the i-th solution after a time being polynomial in the input size plus i. If the delay between before the first output, between two consecutive outputs, and after the last output is polynomial in the input size only, then the algorithm is said to run with *polynomial delay*. We redirect the reader to [34] for further details on enumeration complexity.

Related work. We now review the state of the art concerning the four above-mentioned problems.

The complexity status of HypMirr·Enum and HypMred·Enum was first stated as an open problem by Uno during the Lorentz Workshop on "Enumeration Algorithms using Structure" held in 2015 [6]. Shortly after, Boros and Makino announced the following as a negative answer for both problems [9, 10], where the *dimension* of a hypergraph is the maximum size of its edges.

▶ **Theorem 1.1** (Reformulation of [10, Theorems 1 & 2]). *There is no output-polynomial-time algorithm for HypMirr·Enum and HypMred·Enum unless* $\mathbf{P} = \mathbf{NP}$*, even when restricted to hypergraphs of dimension at most three.*

Let us point that the original formulations in [6, 9, 10] are stated in the context of redundant and irredundant subhypergraphs,[1] which is equivalent to ours up to considering

[1] A set of hyperedges (or subhypergraph) is called *irredundant* if each of its hyperedge contains a private vertex, i.e., a vertex that is only contained in this hyperedge within the set; it is called *redundant* otherwise.

the transposed[2] hypergraph. The above statement is thus a reformulation to our context of vertex subsets, as will be the following summary of their result. In addition to Theorem 1.1, the authors in [9, 10] show that HypMred·Enum and HypMirr·Enum respectively admit polynomial and output-polynomial times algorithms in hypergraphs of bounded *degree* which is the number of edges a vertex intersects. When restricted to hypergraphs of dimension at most two, they show that HypMred·Enum can be solved in polynomial time, while HypMirr·Enum is equivalent to hypergraph dualization, arguably one of the most important open problem in enumeration for which no better than incremental-quasi-polynomial time is known [17, 19, 22]. Let us stress again the fact that the results for hypergraphs of dimension at most two do not apply to GraphMred·Enum nor GraphMirr·Enum, as (ir)redundancy in graphs correspond to (ir)redundancy in the hypergraph of closed-neighborhoods of these graphs, for which the dimension is not bounded.

Later in [14], Conte et al. have studied HypMirr·Enum under the prism of parameterized complexity and in particular derive a polynomial-delay algorithm on hypergraphs (hence graphs) of bounded degeneracy.

In [5], it is shown that GraphMirr·Enum admits an algorithm running with linear delay after polynomial-time preprocessing and polynomial space when restricted to circular-permutation and permutation graphs.

Finally, let us briefly mention that irredundant sets enumeration has triggered some interest in the context of input-sensitive enumeration [4, 24, 25], where the goal is to devise exponential time algorithms minimizing the base of the exponent.

Our contributions. In this work, we continue the research direction that was initiated in [5], as a restriction of the original question asked in [6], and wonder whether GraphMred·Enum and GraphMirr·Enum are tractable. Namely, we wish to answer the following question.

▶ **Question 1.2.** *Can GraphMred·Enum and GraphMirr·Enum be solved in output-polynomial time on general graphs?*

We note that the same question is posed as an open direction in [14].

In the light of Theorem 1.1, and driven by the intuition that the restrictions of HypMred·Enum and HypMirr·Enum to graphs should still be intractable,[3] we address Question 1.2 focusing on graph classes that capture incidence relations of hypergraphs such as bipartite, co-bipartite, and split graphs; we refer to Section 2 for their relation to hypergraphs.

First, we focus on maximal irredundant sets. On split graphs, it was already claimed by Uno in [6] that maximal irredundant sets are in bijection with minimal dominating sets, from which the following is derived using the algorithm in [31].

▶ **Theorem 1.3** ([6])**.** *There is a bijection between maximal irredundant sets and minimal dominating sets in split graphs. Consequently, GraphMirr·Enum can be solved with linear delay and space on this class.*

As for co-bipartite graphs, we obtain the following negative result, which shows the problem to be challenging in this class.

▶ **Theorem 1.4.** *There is an output-polynomial-time algorithm solving GraphMirr·Enum on general graphs if and only if there is one solving the same problem on co-bipartite graphs.*

[2] The *transposed hypergraph* is obtained by considering each edge as a vertex, and each vertex as an edge, while keeping their mutual incidences. In terms of its incidence bipartite graph, it amounts to "swap" its two sides. In particular, the dimension of the hypergraph becomes the degree of its transposed hypergraph, and its degree becomes its dimension.

[3] In what would be an analogue of transversals to dominating sets [31].

To obtain Theorem 1.4 we consider the hypergraph of closed neighborhoods of the input graph. We then use the symmetry of its corresponding co-bipartite incidence graph to argue that, besides a polynomial number of solutions of bounded size, the problem amounts to generating all maximal irredundant sets of the input graph twice, and thus obtain the aforementioned result.

Finally, we obtain the following tractable result on strongly orderable graphs, which generalize chordal bipartite graphs [16].

▶ **Theorem 1.5.** *There is a polynomial-delay and polynomial-space algorithm solving* Graph Mirr·Enum *on strongly orderable graphs.*

We devise the algorithm of Theorem 1.5 by using ordered generation in the trace of the closed-neighborhood hypergraph. Our method is based on earlier work of [14], where the algorithm aims to break any redundancies obtained from adding a new vertex into the set by using structural properties of an elimination ordering.

We turn to minimal redundant sets and obtain the following definite answer to Question 1.2 for GraphMred·Enum.

▶ **Theorem 1.6.** *There is no output-polynomial-time algorithm for* GraphMred·Enum *unless* $\mathbf{P} = \mathbf{NP}$*, even when restricted to split and co-bipartite graphs.*

Both results are obtained by reducing from HypMred·Enum and arguing that, except for a polynomial number of minimal redundant sets, the remaining solutions of the constructed instance of GraphMred·Enum are in an one-to-one correspondence with the solutions of the hypergraph. To argue the existence of no more than a polynomial number of solutions to discard, we exploit the structure of split graphs and, for the case of co-bipartite graphs, the fact that HypMred·Enum remains intractable when the hypergraph has dimension equals three.

On the other hand, we obtain the following positive results. Let us note that (C_3, C_5, C_6, C_8)-free graphs contain chordal bipartite graphs, which are $(C_3, C_{\geq 5})$-free [26].

▶ **Theorem 1.7.** *There is a polynomial-delay algorithm that solves* GraphMred·Enum *on* (C_3, C_5, C_6, C_8)*-free graphs, and an incremental quasi-polynomial-time algorithm that solves* GraphMred·Enum *on* (C_3, C_5, C_6)*-free graphs.*

These algorithms rely on a characterization of the minimal redundant sets of (C_3, C_5, C_6)-free graphs. In particular, we show that the number of redundant vertices in a minimal redundant set is bounded for these classes of graphs and that, except for the case of a unique redundant vertex, the set either has bounded size or corresponds to the closed neighborhood of some vertex in the set. The remaining case where the set has a single redundant vertex is then reduced to instances of red-blue domination for which algorithms running in the specified times are known on these classes.

Unfortunately, in our quest of characterizing the complexity of GraphMirr·Enum and GraphMred·Enum in graph classes that capture incidence relations, we did not manage to solve the case of bipartite graphs. The graph classes appearing in Theorem 1.5 and in the first part of Theorem 1.7 are actually motivated by the fact that they generalize chordal bipartite graphs [16, 26], a well-studied subcase of bipartite graphs that has also served as a first step in other enumeration contexts [11, 24]. These graph classes arises when trying to generalize as much as possible the class on which our techniques for chordal bipartite graphs applied. The case of bipartite graphs, on the other hand, is left open, as we did not succeed in applying our techniques to this context.

Organization. The rest of the paper is organized as follows. We introduce concepts in Section 2. We focus on maximal irredundant sets in Section 3. We then turn on minimal redundant sets in Section 4. Lastly, future directions are discussed in Section 5. Due to space constraints, all proofs are omitted and the reader is referred to the full version of this manuscript for more details.

2 Preliminaries

In this paper, we consider undirected finite simple graphs. We assume the reader is familiar with standard terminology from graph and hypergraph theories, and refer to [2, 15] for the notations not defined below. Given a vertex v and an integer d, by $N^d(v)$ and $N^d[v]$ we respectively mean the set of vertices lying at distance exactly d and at most d from v.

Strongly orderable graphs. Given a graph G, a pair of vertices x and y of G are *comparable* if either $N(x) \setminus \{y\} \subseteq N(y) \setminus \{x\}$ or $N(y) \setminus \{x\} \subseteq N(x) \setminus \{y\}$. A vertex x is *quasi-simple* if the vertices in $N(x)$ are pairwise comparable. In Figure 1, we give an example of a graph admitting a quasi-simple vertex. Lastly, a *quasi-simple elimination ordering* is an ordering $v_1, \ldots, v_n$ of $V(G)$ such that for every $i \in [n]$ it follows that v_i is quasi-simple in the graph $G[\{v_1, \ldots, v_i\}]$ induced by the i first vertices. Graphs that admit a quasi-simple elimination ordering are precisely the *strongly orderable graphs*, which were introduced as a generalization of both chordal bipartite and strongly chordal graphs in [16].

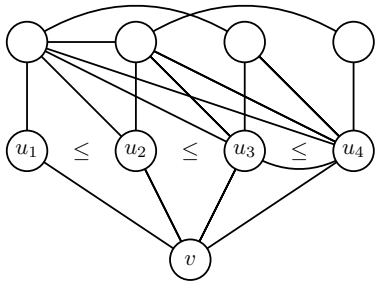

Figure 1 A graph G in which the vertex v is quasi-simple. Within its neighborhood, we have $u_1 \leq u_2 \leq u_3 \leq u_4$ where $\leq$ indicates comparability.

Hypergraphs. Let $\mathcal{H}$ be a hypergraph. Then, the vertices and edges of $\mathcal{H}$ are denoted by $V(\mathcal{H})$ and $\mathcal{E}(\mathcal{H})$, respectively. We call $\mathsf{inc}_{\mathcal{H}}(x) \coloneqq \{E \in \mathcal{E}(\mathcal{H}) \mid x \in E\}$ the set of hyperedges containing x. The subscript is omitted when the hypergraph is clear from context. If G is a graph, then its *closed-neighborhood hypergraph*, denoted $\mathcal{N}(G)$, is defined on the same vertex set and hyperedge set $\{N[x] \mid x \in V(G)\}$.

Let $E_1, \ldots, E_m$ be an ordering of $\mathcal{E}(\mathcal{H})$. The *transposed hypergraph* of $\mathcal{H}$ is the hypergraph $\mathcal{H}^t$ with $V(\mathcal{H}^t) \coloneqq \{e_1, \ldots, e_m\}$ and $\mathcal{E}(\mathcal{H}^t) \coloneqq \{F_1, \ldots, F_n\}$ where $F_i = \{e_j \in V(\mathcal{H}^t) \mid v_i \in E_j\}$. Moreover, given a set $S \subseteq V$, the *trace* of $\mathcal{H}$ with respect to S is the hypergraph $\mathcal{H}_S$ where $V(\mathcal{H}_S) = S$ and $\mathcal{E}(\mathcal{H}_S) \coloneqq \{E \cap S \mid E \in \mathcal{E}(\mathcal{H}),\ E \cap S \neq \emptyset\}$.

Incidence graphs. Let $\mathcal{H}$ be a hypergraph with $V(\mathcal{H}) = \{v_1, \ldots, v_n\}$ and $\mathcal{E}(\mathcal{H}) = \{E_1, \ldots, E_m\}$. The *incidence bipartite graph* of $\mathcal{H}$ is the bipartite graph $B(\mathcal{H})$ with parts $V = V(\mathcal{H})$ and $U \coloneqq \{e_1, \ldots, e_m\}$, and edge set $\{v_i e_j \mid v_i \in E_j\}$. Similarly, the *incidence co-bipartite graph* of $\mathcal{H}$ is the co-bipartite graph $C(\mathcal{H})$ obtained from $B(\mathcal{H})$ by turning V and U into

cliques. Finally, we can define two *incidence split graphs* for $\mathcal{H}$ depending on which of V and U is the independent set. We define $S_1(\mathcal{H})$ to be the split graph obtained from $B(\mathcal{H})$ by making U a clique, and $S_2(\mathcal{H})$ to be the split graph where V is the clique. We give in Figure 2 an example illustrating all four definitions.

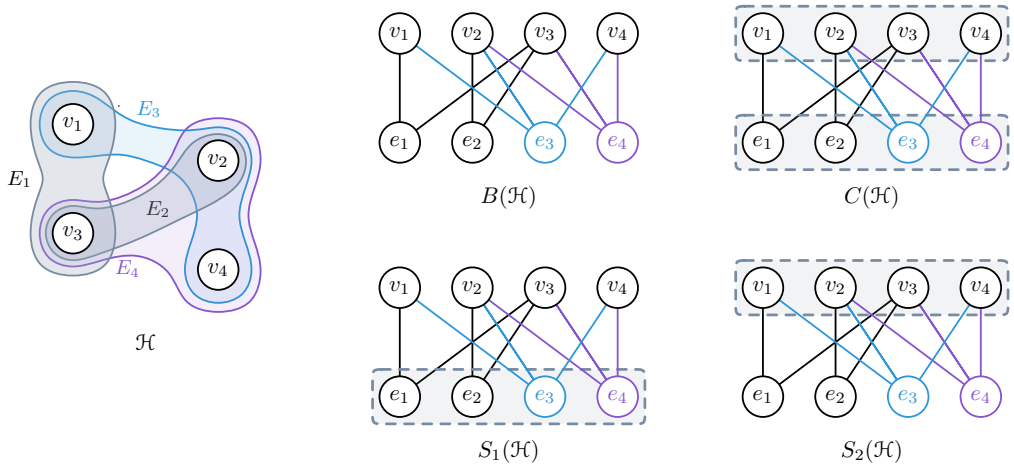

Figure 2 An hypergraph $\mathcal{H}$ and its incidence graphs $B(\mathcal{H})$, $C(\mathcal{H})$, $S_1(\mathcal{H})$ and $S_2(\mathcal{H})$. In graphs, cliques are indicated by dotted grey zones.

Domination. Let G be a graph. We say that a subset of vertices D is a *dominating set* of G if $V(G) = N[D]$. Given a bipartition (A, B) of the vertices, we say that a subset $D \subseteq A$ *dominates* B if $B \subseteq N(D)$. In this latter context, these sets are usually referred to as *red-blue dominating sets*, where elements of A are assumed to be colored red, and those in B to be blue. Generating every minimal red-blue dominating set is known to be solvable with polynomial delay and space when restricted to chordal bipartite graphs [11, 23]. It can also be solved with polynomial delay in (C_6, C_8)-free bipartite graphs [30] as long as red and blue vertices belong to distinct parts of the graph. In general, the problem is equivalent to hypergraph dualization and can thus be solved in incremental-quasi-polynomial time; see e.g. [19, 23].

(Ir)redundancy. Let $\mathcal{H}$ be a hypergraph, $I \subseteq V(\mathcal{H})$, and $x \in I$. We call *private edges of x with respect to I* the edges in the set $\mathsf{priv}_{\mathcal{H}}(x, I) := \{E \in \mathcal{E}(\mathcal{H}) \mid E \cap I = \{x\}\}$. If $\mathcal{H}$ is clear from the context, we drop it from this notation. We say that I is an *irredundant set* of $\mathcal{H}$ if every element x of I has a private edge, and that it is a *redundant set* otherwise.

Observe that, given $\mathcal{H}$, one can check in polynomial time whether a set I of vertices is redundant or irredundant by computing $\mathsf{priv}(x, I)$ for each $x \in I$. We denote by $\mathsf{MaxIrr}(\mathcal{H})$ the family of inclusion-wise maximal irredundant sets of $\mathcal{H}$, and by $\mathsf{MinRed}(\mathcal{H})$ its family of inclusion-wise minimal redundant sets. The two problems HypMirr·Enum and HypMred·Enum are respectively defined as producing, given $\mathcal{H}$, the sets $\mathsf{MaxIrr}(\mathcal{H})$ and $\mathsf{MinRed}(\mathcal{H})$.

We recall that these problems were shown intractable even on hypergraphs of dimension at most three [9, 10] and refer to the introduction for the state of the art on these problems.

In this paper, we are interested in the analogues of redundancy and irredundancy in graphs, that we recall now. Let G be a graph, $I \subseteq V(G)$, and $x \in I$. We call *private neighbors* of x the elements of the set $\mathsf{priv}_G(x, I) := \{y \in N[x] \mid N[y] \cap I = \{x\}\}$. Note that a vertex may be self-private. Analogously, as for hypergraphs, a subset I of vertices is

called *irredundant* if $\mathsf{priv}_G(x, I) \neq \emptyset$ for all $x \in I$, and *redundant* otherwise. We denote by $\mathsf{MaxIrr}(G)$ and $\mathsf{MinRed}(G)$ the set of maximal irredundant and minimal redundant sets of G. To avoid any confusion between graphs and hypergraphs, we will only refer to hypergraphs using calligraphic letters. The two problems GraphMirr·Enum and GraphMred·Enum are respectively defined as producing, given G, the sets $\mathsf{MaxIrr}(G)$ and $\mathsf{MinRed}(G)$.

As stated in the introduction, it is easily seen that $\mathsf{MaxIrr}(G)$ and $\mathsf{MinRed}(G)$ coincide with the maximal irredundant and minimal redundant sets of $\mathcal{N}(G)$. They should, however, not be confused with the maximal irredundant and minimal redundant of G seen as a hypergraph of dimension 2.

3 Maximal irredundant sets

In this section we characterize the complexity of GraphMirr·Enum in strongly orderable graphs, a generalization of chordal bipartite graphs. Moreover, we show that the existence of an output-polynomial-time algorithm solving GraphMirr·Enum on co-bipartite graphs implies solving the same problem on general graphs.

3.1 Strongly orderable graphs

We give an algorithm that generates all maximal irredundant sets of strongly orderable graphs with polynomial delay and space. Our algorithm is based on the *ordered generation* framework (also known as the *sequential method*) that has been successfully applied to various instances admitting elimination orders [1, 11, 18]. Our algorithm, however, differs from the strategy used in the aforementioned works by considering traces of the hypergraph (instead of its induced subhypergraphs) in the decomposition, following what has been done in [14] for irredundant sets.

Let G be a graph. In the remaining, let us fix an arbitrary ordering $v_1, \dots, v_n$ of the vertex set of G, and let $V_i \coloneqq \{v_1, \dots, v_i\}$ denote the set of its i first vertices.

Let $\mathcal{H} \coloneqq \mathcal{N}(G)$ be the closed-neighborhood hypergraph of G, and $\mathcal{H}_i \coloneqq \mathcal{H}_{V_i}$ be a shorthand for the trace of $\mathcal{H}$ on V_i. For each $I \subseteq V_i$ and $x \in I$, by $\mathsf{priv}_i(x, I)$ we mean the set of private edges of x with respect to I in $\mathcal{H}_i$, and by $\mathsf{inc}_i(x)$ we mean the collection of hyperedges of $\mathcal{H}_i$ that contain x. Note that $\mathsf{MaxIrr}(\mathcal{H}_1) = \{\{v_1\}\}$. This is due to $N[v_1]$ being a hyperedge in $\mathcal{H}$, hence the singleton $\{v_1\}$ is the only hyperedge of $\mathcal{H}_1$.

The goal of the algorithm will be to construct $\mathsf{MaxIrr}(\mathcal{H}_i)$ for i ranging from 1 to n, from which we only output $\mathsf{MaxIrr}(\mathcal{H}_n) = \mathsf{MaxIrr}(\mathcal{H})$. We will call *partial solutions* the maximal irredundant sets of $\mathcal{H}_i$ for $i < n$, and *solutions* those of $\mathcal{H}_n$. However, and to guarantee polynomial delay between consecutive outputs, we will not construct the sets $\mathsf{MaxIrr}(\mathcal{H}_i)$ one after the other. Rather, we will define a tree over the set of solutions and partial solutions that will be traversed by our algorithm.

To define this tree, we start with an easy observation whose proof is omitted; see also [14]. The intuition is that whenever a vertex v_j has a private edge in $\mathcal{H}_i$ for some $i > j$, then it keeps that private edge in any subhypergraph $\mathcal{H}_k$, where $j \leq k \leq i$.

▶ **Proposition 3.1.** *Let $i \in \{2, \dots, n\}$ and $I \in \mathsf{MaxIrr}(\mathcal{H}_i)$. Then $I \setminus \{v_i\}$ is an irredundant set of $\mathcal{H}_{i-1}$.*

Let $i \in \{2, \dots, n\}$ and $I \in \mathsf{MaxIrr}(\mathcal{H}_i)$. The *parent of I with respect to i*, denoted $\mathsf{parent}(I, i)$, is defined as the set obtained by the following procedure. First, if v_i belongs to I, we remove it. Then, while there exists a vertex $x \in V_{i-1} \setminus I$ such that $I \cup \{x\}$ is irredundant in $\mathcal{H}_{i-1}$, we add the smallest such vertex x into I. Note that the obtained set is

uniquely defined. By Proposition 3.1, it is an irredundant set of $\mathcal{H}_{i-1}$ after the first step, and it is extended into a maximal irredundant set of $\mathcal{H}_{i-1}$ at the end of the procedure. By construction we thus get $\mathsf{parent}(I,i) \in \mathsf{MaxIrr}(\mathcal{H}_{i-1})$ and $\mathsf{parent}(I,i) = I$ whenever $v_i \notin I$.

Conversely, given $i \in [n-1]$ and $I^\star \in \mathsf{MaxIrr}(\mathcal{H}_i)$, we define the *children of $I^\star$ with respect to i* as $\mathsf{children}(I^\star, i) \coloneqq \{I \in \mathsf{MaxIrr}(\mathcal{H}_{i+1}) \mid \mathsf{parent}(I, i+1) = I^\star\}$.

The following shows that any partial solution has at least one child, and that it has precisely one child if it can be extended into an irredundant set of $\mathcal{H}_{i+1}$ by adding v_{i+1}.

▶ **Lemma 3.2.** *Let $i \in [n-1]$ and $I^\star \in \mathsf{MaxIrr}(\mathcal{H}_i)$. Then, either:*
- *$I^\star \in \mathsf{MaxIrr}(\mathcal{H}_{i+1})$, in which case $I^\star \in \mathsf{children}(I^\star, i)$; or*
- *$I^\star \cup \{v_{i+1}\} \in \mathsf{MaxIrr}(\mathcal{H}_{i+1})$, in which case $\mathsf{children}(I^\star, i) = \{I^\star \cup \{v_{i+1}\}\}$.*

Note that the parent relation defines a tree $\mathcal{T}$ on the set of nodes $\{(I,i) \mid I \in \mathsf{MaxIrr}(\mathcal{H}_i), 1 \leq i \leq n\}$, whose root is $(\{v_1\}, 1)$, and where there is an edge between two nodes $(I^\star, i)$ and $(I, i+1)$ if $I \in \mathsf{children}(I^\star, i)$. Moreover, by Lemma 3.2, every node (I,i) corresponding to a partial solution I has a child. Hence the set of leaves of this tree is precisely the family $\{(I,n) \mid I \in \mathsf{MaxIrr}(\mathcal{H})\}$ we wish to enumerate.

In the following, we show that it is sufficient to be able to list the children with polynomial delay and space to derive a polynomial delay and space algorithm for GraphMirr·Enum.

▶ **Proposition 3.3.** *Let $f, s \colon \mathbb{N} \to \mathbb{N}$ be two functions. Suppose that there is an algorithm that, for any $i \in [n-1]$ and $I^\star \in \mathsf{MaxIrr}(\mathcal{H}_i)$, generates $\mathsf{children}(I^\star, i)$ with $f(n)$ delay and $s(n)$ space. Then there is an algorithm that generates $\mathsf{MaxIrr}(\mathcal{H})$ with $O(n \cdot f(n))$ delay and $O(n \cdot s(n))$ space.*

We now focus on generating $\mathsf{children}(I^\star, i)$ for fixed $i \in [n-1]$ and $I^\star \in \mathsf{MaxIrr}(\mathcal{H}_i)$ such that $I^\star \cup \{v_{i+1}\}$ is redundant. Recall by Lemma 3.2 that $I^\star$ is one of these children, and we may thus focus on those containing v_{i+1}, which we shall call *non-trivial children*.

Our strategy is to find all sets of the form $Z \coloneqq (I^\star \setminus X) \cup \{v_{i+1}\}$, where $X \subseteq I^\star$, that are maximal irredundant sets of $\mathcal{H}_{i+1}$. We call each such set Z an *extension of $I^\star$ to $i+1$*. Note that these extensions do not define supersets of $I^\star$, as they are obtained by adding v_{i+1} and removing elements of $I^\star$. As another remark, note that possibly not all extensions of $I^\star$ to $i+1$ are children of $I^\star$ with respect to i. However, all non-trivial children I of $I^\star$ are extensions of $I^\star$ to $i+1$: by definition, if $I^\star = \mathsf{parent}(I)$ then $I^\star$ is of the form $(I \setminus \{v_{i+1}\}) \cup X$ for some $X \subseteq V_i$, and so $I = (I^\star \setminus X) \cup \{v_{i+1}\}$ with $X \subseteq I^\star$.

Let us now assume that G is strongly orderable and that $v_1, \ldots, v_n$ is a quasi-simple elimination ordering of its vertices. In the following, by $N_i(x)$ we mean the neighborhood of x intersected with V_i.

To compute all extensions efficiently, we will rely on a characterization of their intersection with the set of vertices that become redundant when adding v_{i+1} to $I^\star$. More formally, let $R(I^\star, i) \coloneqq \{x \in I^\star \mid \mathsf{priv}_{i+1}(x, I^\star) \subseteq \mathsf{inc}_{i+1}(v_{i+1})\}$ denote such a set. Note that the vertices in $R(I^\star, i)$ lie at a distance of at most 2 from v_{i+1} in $G[V_{i+1}]$. An upper bound on the number of elements in $R(I^\star, i)$ found in an extension of $I^\star$ is given in the following lemma.

▶ **Lemma 3.4.** *Let Z be an extension of $I^\star$ to $i+1$. Then:*
- *$|R(I^\star, i) \cap N_{i+1}(v_{i+1}) \cap Z| \leq 1$; and*
- *$|R(I^\star, i) \cap N^2_{i+1}(v_{i+1})| \leq 1$.*

Let us point out that Z cannot be removed from the above statement. In fact, the cardinality of $R(I^\star, i) \cap N_{i+1}(v_{i+1})$ is unbounded in general. This can be seen by considering the situation where $N_{i+1}(v_{i+1})$ is an independent set whose elements are self-private with respect to $I^\star$.

Additionally, let us point that the set $X \subseteq I^\star$ such that $Z = (I^\star \setminus X) \cup \{v_{i+1}\}$ is an extension may contain vertices that are not in $R(I^\star, i)$. Indeed, removing vertices from $I^\star \setminus R(I^\star, i)$ may provide private edges to the remaining vertices in $R(I^\star, i)$.

However, from Lemma 3.4 we know that at most two vertices from $R(I^\star, i)$ may be kept in an extension, from which we derive the following.

▶ **Lemma 3.5.** *The number of extensions of $I^\star$ to $i+1$ is bounded by $O(n^3)$, and these extensions can be computed in polynomial time in n.*

We get that generating the children of a partial solution can be done with polynomial delay and space, by enumerating all extensions.

▶ **Lemma 3.6.** *There is an algorithm that generates* $\mathsf{children}(I^\star, i)$ *in polynomial time.*

We conclude to Theorem 1.5 as a corollary of Proposition 3.3 and Lemma 3.6.

3.2 Co-bipartite

To show that GraphMirr·Enum on co-bipartite graphs is as hard as GraphMirr·Enum on arbitrary graphs, we consider the co-bipartite incidence graph $C(\mathcal{N}(G))$ of the closed-neighborhood hypergraph $\mathcal{N}(G)$ of a given graph G; see Figure 3 for an illustration.

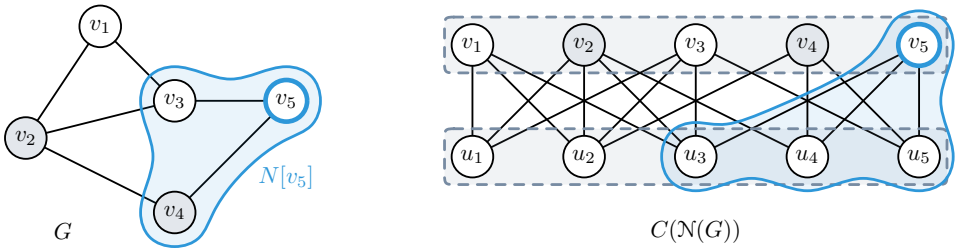

Figure 3 The reduction of Theorem 1.4. On the left a graph G where shaded vertices indicate a maximal irredundant set. On the right the incidence co-bipartite graph $C(\mathcal{N}(G))$ of $\mathcal{N}(G)$. Grey dotted zones indicate the clique bipartition and the shaded vertices form the maximal irredundant set $C(\mathcal{N}(G))$ induced by the maximal irredundant set in pictured in G. The vertices boxed in blue illustrate how $N[v_5]$ (in G) is encoded in $C(\mathcal{N}(G))$.

Using the fact that $C(\mathcal{N}(G))$ is symmetric, we prove that listing the maximal irredundant sets of $C(\mathcal{N}(G))$ amounts to list each maximal irredundant set of G twice, plus a polynomial number of other solutions of size 2 spread across the parts of $C(\mathcal{N}(G))$. We refer to the full version for the details. This proves Theorem 1.4.

▶ **Theorem 1.4.** *There is an output-polynomial-time algorithm solving GraphMirr·Enum on general graphs if and only if there is one solving the same problem on co-bipartite graphs.*

4 Minimal redundant sets

In this section, we investigate the complexity of GraphMred·Enum. We propose a polynomial-delay algorithm for the class of (C_3, C_5, C_6, C_8)-free graphs. Recall that this class contains chordal bipartite graphs, which are precisely $(C_3, C_{\geq 5})$-free graphs [26]. Then, we prove that the problem is intractable for co-bipartite and split graphs.

4.1 Graphs excluding small cycles

In this section, we characterize the minimal redundant sets of graphs excluding small cycles. Our characterization relies on the number of redundant vertices in a minimal redundant set and involves minimal red-blue dominating sets within a ball of radius 2. This yields a polynomial-delay algorithm for GraphMred·Enum on (C_3, C_5, C_6, C_8)-free graphs – which generalize chordal bipartite graphs – relying on existing results from the literature on red-blue domination, and an output quasi-polynomial-time algorithm on (C_3, C_5, C_6)-free graphs.

The remainder of the section is organized as follows. We start by stating preliminary properties in general graphs in Section 4.1.1, and then turn to graphs with no small cycles. We prove properties for these graphs in Section 4.1.2, provide the aforementioned characterization of minimal redundant sets in Section 4.1.3, and describe the algorithm in Section 4.1.4.

4.1.1 Preliminary properties in graphs

If R is a redundant set, we denote by $\mathsf{red}(R) \coloneqq \{x \in R \mid \mathsf{priv}(x, R) = \emptyset\}$ the set of its redundant vertices. The next lemma states that in a minimal redundant set R with redundant vertex x, any vertex other than x prevents x from having a dedicated private neighbor.

▶ **Lemma 4.1.** *Let G be a graph and $R \in \mathsf{MinRed}(G)$. Then for any two distinct vertices $x, y \in R$ such that $x \in \mathsf{red}(R)$ there exists a vertex $z \in \mathsf{priv}(x, R \setminus \{y\}) \cap \mathsf{priv}(y, R \setminus \{x\})$.*

The minimality of the redundant sets also implies the following.

▶ **Proposition 4.2.** *Let G be a graph and $R \in \mathsf{MinRed}(G)$ be a minimal redundant set. Then $R \subseteq N^2[x]$ for every $x \in \mathsf{red}(R)$.*

4.1.2 Properties in graphs excluding small cycles

▶ **Lemma 4.3.** *Let G be a (C_3, C_5)-free graph. Then for every vertex $x \in V(G)$, the graph induced by $N^2[x]$ is bipartite.*

In the following, we show that a minimal redundant set containing a redundant vertex x and one of its neighbor y such that $N(y) \neq \{x\}$ may contain other elements in $N(y)$, or in $N(x)$, but not in both sets at the same time, while it should intersect exactly one if y is redundant. This is illustrated in Figure 4.

▶ **Lemma 4.4.** *Let G be a (C_3, C_5)-free graph, $R \in \mathsf{MinRed}(G)$ be a minimal redundant set, and $x \in \mathsf{red}(R)$ be a redundant vertex. If $y \in R$ is adjacent to x and $N(y) \neq \{x\}$, then at most one of the following holds:*

- $R \cap (N(y) \setminus \{x\}) \neq \emptyset$; *or*
- $R \cap (N(x) \setminus \{y\}) \neq \emptyset$.

Moreover, if $y \in \mathsf{red}(R)$ then exactly one of the items is satisfied.

▶ **Lemma 4.5.** *Let G be a (C_3, C_5)-free graph, $R \in \mathsf{MinRed}(G)$ be a minimal redundant set, and suppose there exists an edge xy such that $x, y \in \mathsf{red}(R)$. Then either $R = N[y]$ or $R = N[x]$.*

We continue by proving properties on (C_3, C_5, C_6)-free graphs.

▶ **Lemma 4.6.** *Let G be a (C_3, C_5, C_6)-free graph, $R \in \mathsf{MinRed}(G)$ a minimal redundant set, $x \in \mathsf{red}(R)$ a redundant vertex, and z_1, z_2 be two distinct vertices in $R \cap N^2(x)$. Then $(N(z_1) \cap N(z_2)) \setminus N(x) = \emptyset$.*

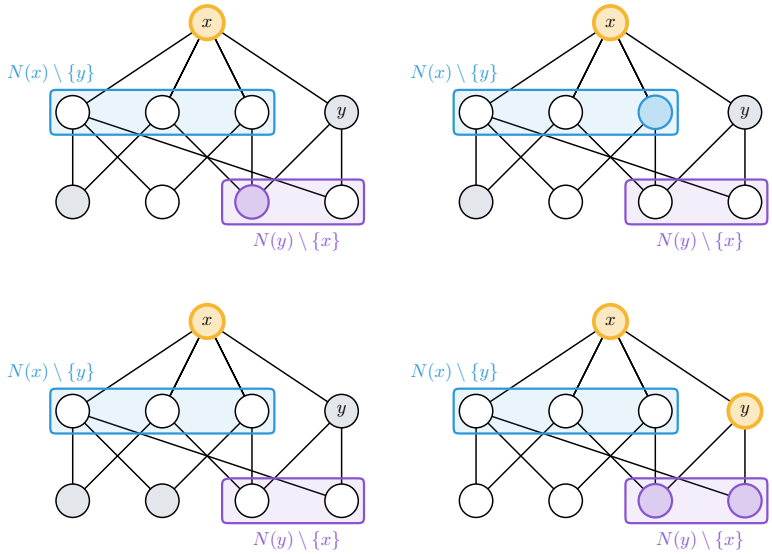

Figure 4 The situations of Lemma 4.4. In each case, a minimal redundant set is indicated by shaded vertices with (yellow) bold vertices being redundant. The top-left, top-right, and bottom-left cases illustrate that R intersects at most one of $N(x) \setminus \{y\}$ or $N(y) \setminus \{x\}$. The bottom-right graph illustrates one of the two cases of Lemma 4.5, where y is also redundant: $R = N[y]$. The other case, not pictured here, is $R = N[x]$.

▶ **Lemma 4.7.** *Let G be a (C_3, C_5, C_6)-free graph. If $R \in \mathsf{MinRed}(G)$ and $x \in \mathsf{red}(R)$, then:*
- $|\mathsf{red}(R) \cap N(x)| \leq 2$*; and*
- $|\mathsf{red}(R) \cap N^2(x)| \leq 1$.

From Lemma 4.7 we conclude that if G is a (C_3, C_5, C_6)-free graph, then the maximum number of redundant vertices a minimal redundant set contains is 4. This bound can be further improved by analyzing what happens when there exists a redundant vertex at distance two from another redundant vertex.

▶ **Lemma 4.8.** *Let G be a (C_3, C_5, C_6)-free graph. If $R \in \mathsf{MinRed}(G)$ such that $x \in \mathsf{red}(R)$ and $|\mathsf{red}(R) \cap N^2(x)| = 1$, then $|R \cap N(x)| = 1$, and so $|R| = 3$.*

We now conclude the following.

▶ **Corollary 4.9.** *If G is (C_3, C_5, C_6)-free and $R \in \mathsf{MinRed}(G)$, then R contains at most 3 redundant vertices.*

The bound in Corollary 4.9 is tight, as witnessed by 3 consecutive vertices in a cycle of length 4.

We end this section with a last property that will be used in the characterization of minimal redundant sets.

▶ **Lemma 4.10.** *Let G be a (C_3, C_5, C_6)-free graph. If $R \in \mathsf{MinRed}(G)$ such that $x \in \mathsf{red}(R)$ and $|\mathsf{red}(R) \cap N(x)| = 2$, then $|R| = 3$.*

4.1.3 Characterization of redundant sets in graphs with no small cycles

In the remainder of this section, we assume G to be a (C_3, C_5, C_6)-free graph. By Corollary 4.9 we know that the number of redundant vertices in a minimal redundant set of G is at most 3. In the following, we characterize the minimal redundant sets depending on whether their number of redundant vertices is 3, 2, or 1.

Let us recall that any minimal redundant set R contains a redundant vertex x, and that by Lemma 4.7, it contains at most two additional redundant vertices at distance 1 from x, and at most one at distance 2. We derive the following as a corollary of Lemmas 4.8 and 4.10.

▶ **Corollary 4.11.** *If $R \in \mathsf{MinRed}(G)$ contains 3 redundant vertices, then $|R| = 3$.*

For minimal redundant sets containing two redundant vertices, we derive the following as a corollary of Lemmas 4.5 and 4.8.

▶ **Corollary 4.12.** *If $R \in \mathsf{MinRed}(G)$ contains 2 redundant vertices, then either:*
- *$R = N[x]$ for some $x \in \mathsf{red}(R)$; or*
- *$|R| = 3$.*

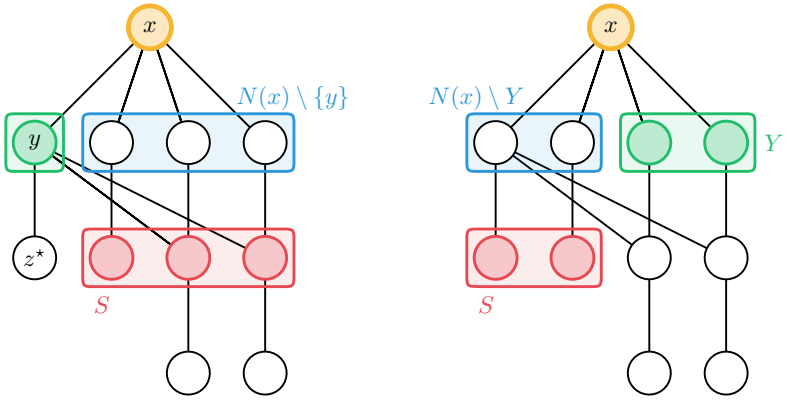

Figure 5 Illustration of Lemma 4.13 on the left, and Lemma 4.14 on the right. Shaded vertices indicate minimal redundant sets, and x (in bold yellow) is the redundant vertex in each case. On the left, y (boxed in green) has a private vertex as well as each vertex of S adjacent to y (the vertices of S are boxed in red) and S minimally dominates $N(x) \setminus \{y\}$ (boxed in blue). On the right, Y has at least two elements, S contains no vertex adjacent to Y and minimally dominates $N(x) \setminus Y$.

We are now left with the characterization of minimal redundant sets containing exactly one redundant vertex which we call x. We distinguish two cases depending on whether they intersect the neighborhood of x on one or more vertices. These lemmas are better understood together with Figure 5.

▶ **Lemma 4.13.** *Let $x \in V(G)$ and $y \in N(x)$. The following are equivalent:*

(a) *$R \in \mathsf{MinRed}(G)$ where $\mathsf{red}(R) = \{x\}$ and $N(x) \cap R = \{y\}$.*

(b) *$R = S \cup \{x, y\}$ where:*
- **i)** *$S \subseteq N^2(x)$;*
- **ii)** *The vertex y has a private neighbor with respect to $S \cup \{x\}$;*
- **iii)** *For every $z \in S$ satisfying $N(z) \subseteq N(x)$ it follows that $z \notin N(y)$; and*
- **iv)** *S minimally dominates $N(x) \setminus \{y\}$;*

▶ **Lemma 4.14.** *Let $x \in V(G)$ and $Y \subseteq N(x)$ such that $|Y| \geq 2$. The following are equivalent:*

(a) *$R \in \mathsf{MinRed}(G)$ where $\mathsf{red}(R) = \{x\}$ and $N(x) \cap R = Y$.*

(b) *$R = S \cup Y \cup \{x\}$ such that:*
- **i)** *$S \subseteq N^2(x)$;*
- **ii)** *$S \cap N(Y) = \emptyset$;*
- **iii)** *Each vertex $y \in Y$ has a private neighbor with respect to $Y \cup \{x\}$;*
- **iv)** *S minimally dominates $N(x) \setminus Y$.*

4.1.4 Algorithm for graphs with no small cycles

We now propose an algorithm for listing the minimal redundant sets of a graph with no small cycles. It runs with polynomial-delay for (C_3, C_5, C_6, C_8)-free graphs and incremental quasi-polynomial time for (C_3, C_5, C_6)-graphs.

The algorithm follows the breakdown of solutions based on redundant vertices we developed so far. First, following Corollaries 4.11 and 4.12, it enumerates all minimal redundant sets with 2 or 3 redundant vertices by running through all possible closed neighborhoods and triplets of vertices, and checking for each of these sets if it is a minimal redundant, all of which in polynomial time. Then it lists, for each vertex x, the minimal redundant sets whose unique redundant vertex is x. According to Lemma 4.13 and Lemma 4.14, these can be further partitioned with respect to the number of neighbors of x in the redundant sets. We detail our approaches for both cases below.

Let us first consider the computation of all minimal redundant sets containing exactly one neighbor of x. The algorithm first ranges over $N(x)$ and for each choice $y \in N(x)$, selects a neighbor $z^\star$ of y that will be kept as a private neighbor of y. For simplicity, let us put $Z_y := \{z \mid z \in N(y),\ N(z) \subseteq N(x)\}$. Then, if any, we list the minimal dominating sets $S \subseteq N^2(x) \setminus (\{z^\star\} \cup Z_y)$ of $N(x) \setminus \{y\}$. According to Lemma 4.13, the sets $R = S \cup \{x, y\}$ obtained that way are precisely the minimal redundant sets such that $\mathsf{red}(R) = \{x\}$, $N(x) \cap R = \{y\}$. Also, note that the sets S are the solutions to an instance of red-blue domination where we intend to minimally dominate $N(x) \setminus \{y\}$ (blue) with vertices from $N^2(x) \setminus (\{z^\star\} \cup Z_y)$ (red).

As for the complexity, the pairs $\{y, z^\star\}$ can be listed in polynomial time, much as the set $N^2(x) \setminus (\{z^\star\} \cup Z_y)$ for each pair. However, given some y, different $z^\star$'s can produce repetitions, as these are chosen to be excluded from the resulting solutions and hence can lead to same sets S. A given solution may thus be repeated $O(n)$ times.

In general graphs red-blue domination can be solved in incremental quasi-polynomial time [22, 23]. When applying the algorithm for red-blue domination on each $z^\star$ successively, the delay before the next solution thus remains bounded by the number of solutions already obtained times a polynomial factor (the number of repetitions). We obtain:

▶ **Proposition 4.15.** *Let G be a (C_3, C_5, C_6)-free graph and let $x \in V(G)$. There is an incremental-quasi-polynomial time algorithm that lists all minimal redundant sets R where $\mathsf{red}(R) = \{x\}$ and $R \cap N(x) = \{y\}$ for some $y \in N(x)$.*

In (C_6, C_8)-free bipartite graphs, red-blue domination can be solved with polynomial delay [30] when red and blue vertices belong to distinct parts. However, our approach does not directly preserve delay due to repetitions. Yet, we can still preserve polynomial delay overall. To do so, instead of outputting a previously unseen solution, we store it into a queue, a member of which is popped and printed every $O(n) \cdot f(n)$ time, where $f(n)$ is the delay for red-blue domination. Storing solutions into the queue and keeping a structure to track repetitions requires however exponential space. We get:

▶ **Proposition 4.16.** *Let G be a (C_3, C_5, C_6, C_8)-free graph and let $x \in V(G)$. There is a polynomial delay algorithm that lists all minimal redundant sets R where $\mathsf{red}(R) = \{x\}$ and $R \cap N(x) = \{y\}$ for some $y \in N(x)$.*

We now turn our attention to minimal redundant sets containing at least two neighbors of x, corresponding to Lemma 4.14. Two steps are needed. The first step consists in identifying subsets of $N(x)$ that can be extended into a minimal redundant sets. Given $Y \subseteq N(x)$, $|Y| \geq 2$, we say that Y is *extendable* with respect to x if there exists a minimal

redundant set R with $\mathsf{red}(R) = \{x\}$ and $R \cap N(x) = Y$. Below, we characterize extendable sets and show that they constitute an independence set system (i.e., a family of sets containing the empty set and being closed by subset), disregarding singletons and the empty set.

▶ **Lemma 4.17.** *Let $x \in V(G)$ be a vertex and $Y \subseteq N(x)$ be a set such that $|Y| \geq 2$. Then, the set Y is extendable with respect to x if and only if the two conditions hold:*
(a) *Each $y \in Y$ has a private neighbor with respect to $Y \cup \{x\}$; and*
(b) *The set $N^2(x) \setminus N(Y)$ dominates $N(x) \setminus Y$.*

▶ **Lemma 4.18.** *Let $x \in V(G)$ be a vertex. If $Y^\star \subseteq N(x)$ is extendable with respect to x, then every subset $Y \subsetneq Y^\star$ satisfying $|Y| \geq 2$ is also extendable with respect to x.*

Let Y be an extendable set. Following Lemmas 4.14 and 4.17, the minimal redundant sets R such that $\mathsf{red}(R) = \{x\}$ and $R \cap N(x) = Y$ are precisely those of the form $S \cup Y \cup \{x\}$ where Y is extendable and $S \subseteq N^2(x) \setminus N(Y)$ minimally dominates $N^2(x) \setminus N(Y)$. Again, this can be framed as red-blue domination: we need to minimally dominate $N(x) \setminus Y$ (blue) with vertices from $N^2(x) \setminus N(Y)$ (red). Therefore, the algorithm for this part lists all extendable sets Y using an algorithm to enumerate all members of an independence set system, discarding singletons and the empty set, and for each such Y solves the corresponding instance of red-blue domination. Note that, as long as the algorithm for red-blue domination does not produce repetitions, this algorithm will not either, as extendable Y's partition the solutions to be enumerated.

As for complexity, observe first that whether a set is extendable can be tested in polynomial time. Given that the members of an independence set system can be enumerated with polynomial delay provided they can be recognized in polynomial time (see, e.g., [31] for an example), we derive the following propositions, mimicking Propositions 4.15 and 4.16:

▶ **Proposition 4.19.** *Let G be a (C_3, C_5, C_6)-free graph and let $x \in V(G)$. There is an incremental-quasi-polynomial time algorithm that lists all minimal redundant sets R where $\mathsf{red}(R) = \{x\}$ and $R \cap N(x) = Y$ for some $Y \subseteq N(x)$ and $|Y| \geq 2$.*

▶ **Proposition 4.20.** *Let G be a (C_3, C_5, C_6, C_8)-free graph and let $x \in V(G)$. There is a polynomial-delay algorithm that lists all minimal redundant sets R where $\mathsf{red}(R) = \{x\}$ and $R \cap N(x) = Y$ for some $Y \subseteq N(x)$ and $|Y| \geq 2$.*

Combining the above propositions we obtain Theorem 1.7. Let us observe that a solution of the form $N[x]$ might be obtained twice: at preprocessing or in the algorithm used for finding all minimal redundant sets where x is the unique redundant vertex. Since the repetition is unique, this, however, can be handled without impact on the delay by simply not outputting the solution the second time it is produced.

4.2 Co-bipartite and split graphs

To show that, unless $\mathbf{P} = \mathbf{NP}$, there is no output-polynomial-time algorithm solving GraphMred·Enum on co-bipartite graphs, our reduction relies on earlier results of Boros and Makino [10], stating that HypMred·Enum is intractable for hypergraphs of dimension 3. Nevertheless, the problem can be solved in polynomial time for hypergraphs of maximum degree 3 due to every solution having size at most 4.

We consider a hypergraph $\mathcal{H}$ of dimension 3, whose transposed hypergraph $\mathcal{H}^t$ has maximum degree at most 3. By looking at the co-bipartite incidence graph $C(\mathcal{H})$ of $\mathcal{H}$, we prove that listing the members of $\mathsf{MinRed}(C(\mathcal{H}))$ amounts to enumerating $\mathsf{MinRed}(\mathcal{H})$, together with a polynomial number of solutions of size at most 4 – including $\mathsf{MinRed}(\mathcal{H}^t)$ and solutions spread across the parts of $C(\mathcal{H})$.

A similar argument can be used for the case of split graphs, instead constructing the split incidence graph of the input hypergraph. This yields the following.

▶ **Theorem 1.6.** *There is no output-polynomial-time algorithm for* GraphMred·Enum *unless* $\mathbf{P} = \mathbf{NP}$*, even when restricted to split and co-bipartite graphs.*

Note that our reduction preserves the delay as there is only a polynomial number of solutions to discard.

5 Perspectives

In this paper, we studied the problems GraphMirr·Enum and GraphMred·Enum in graph classes capturing incidence relations such as co-bipartite, split and bipartite graphs. In particular, we proved the hardness of GraphMred·Enum on split and co-bipartite graphs, while for GraphMirr·Enum we showed that the problem is as hard in general graphs as in co-bipartite graphs. Question 1.2 remains thus open for GraphMirr·Enum.

Concerning tractability, in addition to the case of GraphMirr·Enum admitting a polynomial-delay algorithm on split graphs, originally claimed by Uno in [6], we showed that both problems admit polynomial-delay algorithms on generalizations of chordal bipartite graphs. The case of bipartite graphs is left open, which leads to the following question:

▶ **Question 5.1.** *What is the complexity of* GraphMirr·Enum *and* GraphMred·Enum *in bipartite graphs?*

Towards this question, we observe that the sequential method we applied to solve Graph Mirr·Enum in strongly orderable graphs cannot be used straightforwardly in the bipartite case. Namely, we prove the following.

▶ **Theorem 5.2.** *Unless* $\mathbf{P} = \mathbf{NP}$*, there is no output-polynomial time algorithm which, given a bipartite graph* G*, an ordering* $v_1, \dots, v_n$ *of its vertices, an integer* $i \in [n-1]$*, and* $I^\star \in \mathsf{MaxIrr}(\mathcal{H}_i)$ *where* $\mathcal{H} = \mathcal{N}(G)$*, enumerates* $\mathsf{children}(I^\star, i)$*.*

As for GraphMred·Enum, the strategy we used in graphs without small cycles relied on two crucial facts: (1) a minimal redundant set has a constant number of redundant vertices (at most 3), and (2) the minimal redundant sets containing a single redundant vertex and a unique neighbor of this vertex can be enumerated efficiently. In bipartite graphs these two facts do not hold anymore.

For redundant sets with unbounded number of redundant vertices, let us consider the following bipartite graph G. We start from vertices $v_1, \dots, v_n$ and create, for each $i \neq j$ a new vertex u_{ij} adjacent to v_i and v_j. Then, we add a vertex y adjacent to each v_i and a vertex z adjacent to y. This complete the description of G. The set $R \coloneqq \{y, v_1, \dots, v_n\}$ is a minimal redundant set of G where $\mathsf{red}(R) = \{v_1, \dots, v_n\}$ is indeed of unbounded size. We illustrate this construction in Figure 6.

For the task of enumerating the minimal redundant sets with a single redundant vertex x and containing a unique neighbor y of x, i.e., the case of Lemma 4.13, we show in the next theorem that it becomes intractable in bipartite graphs. The hardness comes from the fact that, when selecting a subset of $N(y)$ to be part of the redundant set, one must guarantee that each chosen vertex has a private neighbor outside $N^2(x)$ all the while being useful to dominate $N(x) \setminus \{y\}$, a problem that does not arise in bipartite graphs with no induced C_6.

▶ **Theorem 5.3.** *Let* G *be a bipartite graph and* $x, y \in V(G)$ *be a pair of adjacent vertices. Then, unless* $\mathbf{P} = \mathbf{NP}$*, there is no output-polynomial time algorithm to enumerate the minimal redundant sets* R *of* G *that satisfy* $\mathsf{red}(R) = \{x\}$ *and* $R \cap N(x) = \{y\}$*.*

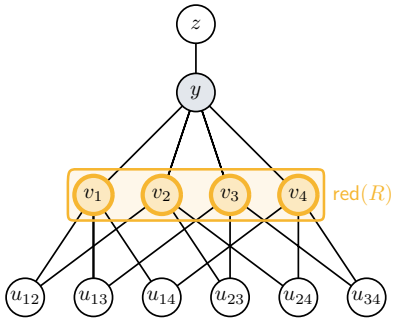

Figure 6 A bipartite graph with a minimal redundant set (shaded vertices) having an unbounded number of redundant vertices (boxed in yellow). The vertex z shows that a minimal redundant set with an unbounded number of redundant vertices does not need to be the closed-neighborhood of a vertex.

References

1 Valentin Bartier, Oscar Defrain, and Fionn Mc Inerney. Hypergraph dualization with FPT-delay parameterized by the degeneracy and dimension. In *International Workshop on Combinatorial Algorithms*, pages 111–125. Springer, 2024. doi:10.1007/978-3-031-63021-7_9.

2 Claude Berge. *Hypergraphs: combinatorics of finite sets*, volume 45. Elsevier, 1984.

3 Alan A. Bertossi and Alessandro Gori. Total domination and irredundance in weighted interval graphs. *SIAM Journal on Discrete Mathematics*, 1(3):317–327, 1988. doi:10.1137/0401032.

4 Daniel Binkele-Raible, Ljiljana Brankovic, Marek Cygan, Henning Fernau, Joachim Kneis, Dieter Kratsch, Alexander Langer, Mathieu Liedloff, Marcin Pilipczuk, Peter Rossmanith, and Jakub Onufry Wojtaszczyk. Breaking the 2^n-barrier for irredundance: Two lines of attack. *Journal of Discrete Algorithms*, 9(3):214–230, 2011. Selected papers from the 7th International Conference on Algorithms and Complexity (CIAC 2010). doi:10.1016/j.jda.2011.03.002.

5 Sarah Blind, Nadia Creignou, and Frédéric Olive. Locally definable vertex set properties are efficiently enumerable. *Discrete Applied Mathematics*, 303:186–202, 2021. doi:10.1016/J.DAM.2020.05.037.

6 Hans Bodlaender, Endre Boros, Pinar Heggernes, and Dieter Kratsch. *Open Problems of the Lorentz Workshop, "Enumeration Algorithms using Structure"*. Number UU-CS-2015-016 in Technical Report Series. UU BETA ICS Departement Informatica, 2015.

7 Béla Bollobás and Ernest J. Cockayne. Graph-theoretic parameters concerning domination, independence, and irredundance. *Journal of Graph Theory*, 3(3):241–249, 1979. doi:10.1002/JGT.3190030306.

8 Béla Bollobás and Ernest J. Cockayne. The irredundance number and maximum degree of a graph. *Discrete Mathematics*, 49(2):197–199, 1984. doi:10.1016/0012-365X(84)90118-3.

9 Endre Boros and Kazuhisa Makino. Generating maximal irredundant and minimal redundant subfamilies of a given hypergraph. In *WEPA 2016 & Boolean Seminar 2017*, 2016. URL: http://clp.mff.cuni.cz/booleanseminar/booleanseminar2017/presentations/endre.pdf.

10 Endre Boros and Kazuhisa Makino. Generating minimal redundant and maximal irredundant subhypergraphs. *Discrete Applied Mathematics*, 358:217–229, 2024. doi:10.1016/J.DAM.2024.07.006.

11 Emanuel Castelo, Oscar Defrain, and Guilherme C. M. Gomes. Enumerating minimal dominating sets and variants in chordal bipartite graphs. In Pat Morin and Eunjin Oh, editors, *19th International Symposium on Algorithms and Data Structures (WADS 2025)*, volume 349 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 15:1–15:15, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.WADS.2025.15.

12 Ernest J. Cockayne, Odile Favaron, C. Payan, and Andrew G. Thomason. Contributions to the theory of domination, independence and irredundance in graphs. *Discrete Mathematics*, 33(3):249–258, 1981. doi:10.1016/0012-365X(81)90268-5.

13 Ernest J. Cockayne, Stephen T. Hedetniemi, and D. J. Miller. Properties of hereditary hypergraphs and middle graphs. *Canadian Mathematical Bulletin*, 21(4):461–468, 1978.

14 Alessio Conte, Mamadou Moustapha Kanté, Andrea Marino, and Takeaki Uno. Maximal irredundant set enumeration in bounded-degeneracy and bounded-degree hypergraphs. In *International Workshop on Combinatorial Algorithms*, pages 148–159. Springer, 2019. doi:10.1007/978-3-030-25005-8_13.

15 R. Diestel. *Graph Theory*, volume 173 of *Graduate Texts in Mathematics*. Springer, 2005.

16 Feodor F. Dragan. Strongly orderable graphs a common generalization of strongly chordal and chordal bipartite graphs. *Discrete Applied Mathematics*, 99(1-3):427–442, 2000. doi:10.1016/S0166-218X(99)00149-3.

17 Thomas Eiter and Georg Gottlob. Identifying the minimal transversals of a hypergraph and related problems. *SIAM Journal on Computing*, 24(6):1278–1304, 1995. doi:10.1137/S0097539793250299.

18 Thomas Eiter, Georg Gottlob, and Kazuhisa Makino. New results on monotone dualization and generating hypergraph transversals. *SIAM Journal on Computing*, 32(2):514–537, 2003. doi:10.1137/S009753970240639X.

19 Thomas Eiter, Kazuhisa Makino, and Georg Gottlob. Computational aspects of monotone dualization: A brief survey. *Discrete Applied Mathematics*, 156(11):2035–2049, 2008. doi:10.1016/J.DAM.2007.04.017.

20 Odile Favaron. Stability, domination and irredundance in a graph. *Journal of Graph Theory*, 10(4):429–438, 1986.

21 Michael Fellows, Gerd Fricke, Stephen Hedetniemi, and David Jacobs. The private neighbor cube. *SIAM Journal on Discrete Mathematics*, 7(1):41–47, 1994. doi:10.1137/S0895480191199026.

22 Michael L. Fredman and Leonid Khachiyan. On the complexity of dualization of monotone disjunctive normal forms. *Journal of Algorithms*, 21(3):618–628, 1996. doi:10.1006/JAGM.1996.0062.

23 Petr A. Golovach, Pinar Heggernes, Mamadou Moustapha Kanté, Dieter Kratsch, and Yngve Villanger. Enumerating minimal dominating sets in chordal bipartite graphs. *Discrete Applied Mathematics*, 199:30–36, 2016. doi:10.1016/J.DAM.2014.12.010.

24 Petr A. Golovach, Dieter Kratsch, Mathieu Liedloff, and Mohamed Yosri Sayadi. Enumeration and maximum number of maximal irredundant sets for chordal graphs. *Discrete Applied Mathematics*, 265:69–85, 2019. doi:10.1016/J.DAM.2019.03.018.

25 Petr A. Golovach, Dieter Kratsch, and Mohamed Yosri Sayadi. Enumeration of maximal irredundant sets for claw-free graphs. *Theoretical Computer Science*, 754:3–15, 2019. doi:10.1016/J.TCS.2018.02.014.

26 Martin Charles Golumbic and Clinton F. Goss. Perfect elimination and chordal bipartite graphs. *Journal of Graph Theory*, 2(2):155–163, 1978. doi:10.1002/JGT.3190020209.

27 Martin Charles Golumbic and Renu C. Laskar. Irredundancy in circular arc graphs. *Discrete Applied Mathematics*, 44(1-3):79–89, 1993. doi:10.1016/0166-218X(93)90223-B.

28 S. T. Hedetniemi, R. Laskar, and John Pfaff. Irredundance in graphs: A survey. Combinatorics, graph theory and computing, Proc. 16th Southeast. Conf., Boca Raton/Fla. 1985, Congr. Numerantium 48, 183-193 (1985)., 1985.

29 Michael S. Jacobson and Ken Peters. Chordal graphs and upper irredundance, upper domination and independence. *Discrete Mathematics*, 86(1-3):59–69, 1990. doi:10.1016/0012-365X(90)90349-M.

30 Mamadou M. Kanté, Kaveh Khoshkhah, and Mozhgan Pourmoradnasseri. Enumerating Minimal Transversals of Hypergraphs without Small Holes. In *43rd International Symposium on Mathematical Foundations of Computer Science (MFCS 2018)*, volume 117 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 55:1–55:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.MFCS.2018.55.

31 Mamadou Moustapha Kanté, Vincent Limouzy, Arnaud Mary, and Lhouari Nourine. On the enumeration of minimal dominating sets and related notions. *SIAM Journal on Discrete Mathematics*, 28(4):1916–1929, 2014. doi:10.1137/120862612.

32 Mamadou Moustapha Kanté, Vincent Limouzy, Arnaud Mary, Lhouari Nourine, and Takeaki Uno. Polynomial delay algorithm for listing minimal edge dominating sets in graphs. In *Workshop on Algorithms and Data Structures*, pages 446–457. Springer, 2015. doi:10.1007/978-3-319-21840-3_37.

33 R. Laskar and J. Pfaff. Domination and irredundance in graphs. In *Tech. Rept.*, volume 434. Department Mathematical Sciences, Clemson University, 1983.

34 Yann Strozecki. Enumeration complexity. *Bulletin of EATCS*, 3(129), 2019.